# **ESP32 Learning Kit**

keyestudio WiKi

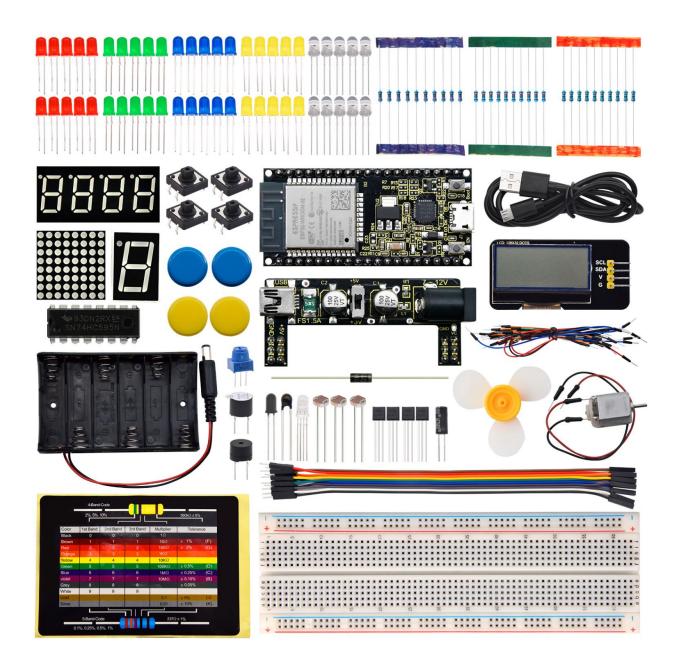
Dec 06, 2023

# **KEYESTUDIO ESP32 LEARNING KIT BASIC EDITION**

1	1.Description	3
2	2.Kit list	5
3	3.Keyestudio ESP32 Core board	7
4	Getting started with Arduino         4.1       Windows System         4.2       Arduino MacOS	<b>11</b> 11 34
5	Arduino Tutorial5.1Download Arduino code and library files5.2Project 01: Hello World5.3Project 02: Turn On LED5.4Project 03LED Flashing5.5Project 04: Breathing Led5.6Project 05Traffic Lights5.7Project 06: RGB LED5.8Project 07: Flowing Water Light5.9Project 081-Digit Digital Tube5.10Project 094-Digit Digital Tube5.11Project 1094-Digit Digital Tube5.12Project 1174HC595N Control 8 LEDs5.13Project 12Active Buzzer5.14Project 13Passive Buzzer5.15Project 14: Mini Table Lamp5.16Project 16: I2C 128×32 LCD5.17Project 16: I2C 128×32 LCD5.18Project 19Flame Alarm5.20Project 19Flame Alarm5.21Project 128wiFi Station Mode5.22Project 22Bluetooth5.23Project 22Bluetooth5.24Project 23WiFi Station Mode5.25Project 25WiFi Station+AP Mode	123 128 133 138 143 143 148 154 160 168 179
6	Getting started with Python	195
7	Python Tutorial	239

	7.1	Download Python code files	239
	7.2	Development Environment Configuration	239
	7.3	Project 01: Hello World	239
	7.4	Project 02: Turn On LED	242
	7.5	Project 03LED Flashing	257
	7.6	Project 04: Breathing Led	
	7.7	Project 05Traffic Lights	
	7.8	Project 06: RGB LED	
	7.9	Project 07: Flowing Water Light	
	7.10	Project 081-Digit Digital Tube	
	7.11	Project 094-digit Digital Tube	
	7.12	Project 108×8 Dot-matrix Display	
	7.12	Project 1174HC595N Control 8 LEDs	
	7.14	·J.	
	7.15	Project 13Passive Buzzer	
	7.16	Project 14: Mini Table Lamp	
	7.17	Project 15Tilt And LED	
	7.18	Project 16: I2C 128×32 LCD	
	7.19	5	
	7.20	3 0 0	
	7.21	Project 19Flame Alarm	355
	7.22	Project 20Night Lamp	365
	7.23	Project 21Temperature Instrument	373
	7.24	Project 22WiFi Station Mode	384
	7.25	Project 23WiFi AP Mode	388
	7.26	Project 24WiFi Station+AP Mode	
		5	
8	Getti	ing started with C language(Raspberry Pi)	397
8	Getti 8.1	ing started with C language(Raspberry Pi) Install the Raspberry Pi OS System	
8			397
8	8.1	Install the Raspberry Pi OS System	397 430
	8.1 8.2 8.3	Install the Raspberry Pi OS System         Preparation of C language control basic hardware:         Import the Arduino C library	397 430 451
	8.1 8.2 8.3 C lan	Install the Raspberry Pi OS System	<ul><li>397</li><li>430</li><li>451</li><li>457</li></ul>
	<ul> <li>8.1</li> <li>8.2</li> <li>8.3</li> <li>C lan</li> <li>9.1</li> </ul>	Install the Raspberry Pi OS System	<ul> <li>397</li> <li>430</li> <li>451</li> <li>457</li> <li>457</li> </ul>
	8.1 8.2 8.3 C lan	Install the Raspberry Pi OS System	<ul> <li>397</li> <li>430</li> <li>451</li> <li>457</li> <li>457</li> </ul>
	<ul> <li>8.1</li> <li>8.2</li> <li>8.3</li> <li>C lan</li> <li>9.1</li> </ul>	Install the Raspberry Pi OS System	<ul> <li>397</li> <li>430</li> <li>451</li> <li>457</li> <li>457</li> <li>462</li> </ul>
	<ul> <li>8.1</li> <li>8.2</li> <li>8.3</li> <li>C lan</li> <li>9.1</li> <li>9.2</li> </ul>	Install the Raspberry Pi OS System	<ul> <li>397</li> <li>430</li> <li>451</li> <li>457</li> <li>457</li> <li>462</li> <li>474</li> </ul>
	<ul> <li>8.1</li> <li>8.2</li> <li>8.3</li> <li>C lan</li> <li>9.1</li> <li>9.2</li> <li>9.3</li> </ul>	Install the Raspberry Pi OS System         Preparation of C language control basic hardware:         Import the Arduino C library         nguage (Raspberry Pi) Tutorial         Project 01: Hello World         Project 02: Turn On LED         Project 03LED Flashing	397 430 451 <b>457</b> 457 462 474 480
	8.1 8.2 8.3 <b>C lan</b> 9.1 9.2 9.3 9.4	Install the Raspberry Pi OS System         Preparation of C language control basic hardware:         Import the Arduino C library         nguage (Raspberry Pi) Tutorial         Project 01: Hello World         Project 02: Turn On LED         Project 03LED Flashing         Project 04: Breathing Led         Project 05Traffic Lights	397 430 451 <b>457</b> 457 462 474 480
	8.1 8.2 8.3 <b>C lan</b> 9.1 9.2 9.3 9.4 9.5	Install the Raspberry Pi OS System         Preparation of C language control basic hardware:         Import the Arduino C library         nguage (Raspberry Pi) Tutorial         Project 01: Hello World         Project 02: Turn On LED         Project 03LED Flashing         Project 04: Breathing Led         Project 05Traffic Lights         Project 06: RGB LED	397 430 451 <b>457</b> 457 462 474 480 488
	8.1 8.2 8.3 <b>C lan</b> 9.1 9.2 9.3 9.4 9.5 9.6 9.7	Install the Raspberry Pi OS System         Preparation of C language control basic hardware:         Import the Arduino C library         nguage (Raspberry Pi) Tutorial         Project 01: Hello World         Project 02: Turn On LED         Project 03LED Flashing         Project 04: Breathing Led         Project 05Traffic Lights         Project 06: RGB LED         Project 07: Flowing Water Light	<ul> <li>397</li> <li>430</li> <li>451</li> <li>457</li> <li>457</li> <li>462</li> <li>474</li> <li>480</li> <li>488</li> <li>493</li> </ul>
	8.1 8.2 8.3 <b>C lan</b> 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8	Install the Raspberry Pi OS System         Preparation of C language control basic hardware:         Import the Arduino C library         nguage (Raspberry Pi) Tutorial         Project 01: Hello World         Project 02: Turn On LED         Project 03LED Flashing         Project 04: Breathing Led         Project 05Traffic Lights         Project 07: Flowing Water Light         Project 081-Digit Digital Tube	397 430 451 <b>457</b> 457 462 474 480 488 493 497 499
	8.1 8.2 8.3 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9	Install the Raspberry Pi OS System         Preparation of C language control basic hardware:         Import the Arduino C library         nguage (Raspberry Pi) Tutorial         Project 01: Hello World         Project 02: Turn On LED         Project 03LED Flashing         Project 04: Breathing Led         Project 05Traffic Lights         Project 06: RGB LED         Project 07: Flowing Water Light         Project 081-Digit Digital Tube	397 430 451 <b>457</b> 462 474 480 488 493 497 499 506
	8.1 8.2 8.3 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9 9.10	Install the Raspberry Pi OS System	397 430 451 <b>457</b> 462 474 480 488 493 497 499 506 511
	8.1 8.2 8.3 <b>C lan</b> 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9 9.10 9.11	Install the Raspberry Pi OS SystemPreparation of C language control basic hardware:Import the Arduino C librarynguage (Raspberry Pi) TutorialProject 01: Hello WorldProject 02: Turn On LEDProject 03LED FlashingProject 04: Breathing LedProject 05Traffic LightsProject 07: Flowing Water LightProject 081-Digit Digital TubeProject 094-Digit Digital TubeProject 108×8 Dot-matrix DisplayProject 1174HC595N Control 8 LEDs	397 430 451 <b>457</b> 462 474 480 488 493 497 499 506 511 517
	8.1 8.2 8.3 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9 9.10 9.11 9.12	Install the Raspberry Pi OS SystemPreparation of C language control basic hardware:Import the Arduino C librarynguage (Raspberry Pi) TutorialProject 01: Hello WorldProject 02: Turn On LEDProject 03LED FlashingProject 04: Breathing LedProject 05Traffic LightsProject 06: RGB LEDProject 07: Flowing Water LightProject 081-Digit Digital TubeProject 108×8 Dot-matrix DisplayProject 1174HC595N Control 8 LEDsProject 12Active Buzzer	397 430 451 <b>457</b> 457 462 474 480 488 493 497 499 506 511 517 520
	8.1 8.2 8.3 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9 9.10 9.11 9.12 9.13	Install the Raspberry Pi OS SystemPreparation of C language control basic hardware:Import the Arduino C librarynguage (Raspberry Pi) TutorialProject 01: Hello WorldProject 02: Turn On LEDProject 03LED FlashingProject 04: Breathing LedProject 05Traffic LightsProject 07: Flowing Water LightProject 081-Digit Digital TubeProject 094-Digit Digital TubeProject 108×8 Dot-matrix DisplayProject 1174HC595N Control 8 LEDsProject 13Passive Buzzer	397 430 451 <b>457</b> 457 462 474 480 488 493 497 506 511 517 520 524
	8.1 8.2 8.3 <b>C lan</b> 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9 9.10 9.11 9.12 9.13 9.14	Install the Raspberry Pi OS SystemPreparation of C language control basic hardware:Import the Arduino C librarynguage (Raspberry Pi) TutorialProject 01: Hello WorldProject 02: Turn On LEDProject 03LED FlashingProject 04: Breathing LedProject 05Traffic LightsProject 06: RGB LEDProject 07: Flowing Water LightProject 081-Digit Digital TubeProject 108×8 Dot-matrix DisplayProject 12Active BuzzerProject 13Passive BuzzerProject 14: Mini Table Lamp	397 430 451 <b>457</b> 462 474 480 488 493 497 506 511 517 520 524 527
	8.1 8.2 8.3 <b>C lan</b> 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9 9.10 9.11 9.12 9.13 9.14 9.15	Install the Raspberry Pi OS System	397 430 451 457 462 474 480 488 493 497 506 511 517 520 524 527 532
	8.1 8.2 8.3 <b>C lan</b> 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9 9.10 9.11 9.12 9.13 9.14 9.15 9.16	Install the Raspberry Pi OS System	397 430 451 457 462 474 480 488 493 497 499 506 511 517 520 524 527 532 536
	8.1 8.2 8.3 <b>C lan</b> 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9 9.10 9.11 9.12 9.13 9.14 9.15 9.16 9.17	Install the Raspberry Pi OS System	397 430 451 457 462 474 480 488 493 497 499 506 511 517 520 524 527 532 536 540
9	8.1 8.2 8.3 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9 9.10 9.11 9.12 9.13 9.14 9.15 9.16 9.17 9.18	Install the Raspberry Pi OS System	397 430 451 457 462 474 480 488 493 497 499 506 511 517 520 524 527 532 536 540 544
	8.1 8.2 8.3 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9 9.10 9.11 9.12 9.13 9.14 9.15 9.16 9.17 9.18 9.19	Install the Raspberry Pi OS System	397 430 451 <b>457</b> 462 474 480 488 493 497 506 511 517 520 524 527 532 536 540 544 550
	8.1 8.2 8.3 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9 9.10 9.11 9.12 9.13 9.14 9.15 9.16 9.17 9.18	Install the Raspberry Pi OS System	397 430 451 457 462 474 480 488 493 497 499 506 511 517 520 524 527 532 536 540 544 550 556

9.22	Project 22Bluetooth	567
	Project 23WiFi Station Mode	
9.24	Project 24WiFi AP Mode	583
9.25	Project 25WiFi Station+AP Mode	588



## **1.DESCRIPTION**

Do you want to learn about programming?

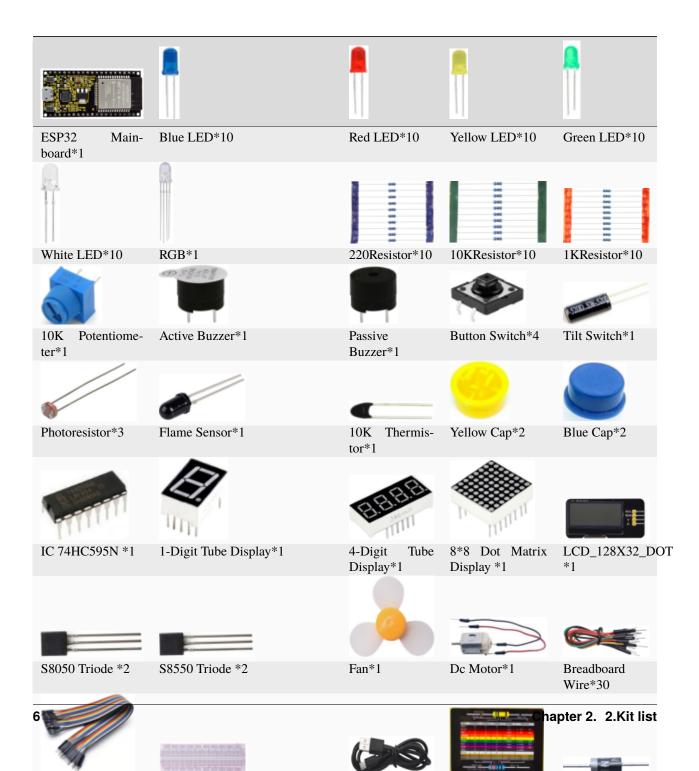
As long as you're passionate about science and dare to explore new things, this kit is surely the best choice for you. The Keyestudio ESP32 Learning Kit Basic Edition mainly contains some common electronic components/sensors/modules, a ESP32 mainboard and bread wires are also included.

As many as 74 project tutorials are provided, which contain detailed wiring diagrams, components knowledge, and fascinating project code. Each project is produced using Thonny for Windows, Arduino IDE for Windows, and Arduino IDE for Raspberry Pi. It's easy to get started.

You can create numerous fascinating DIY experiments with one controller (ESP32), various of sensors/modules and electronics. These courses can give you a deeper understanding of programming methods, logic, electronic circuits and the Linux operating system (Raspberry Pi).

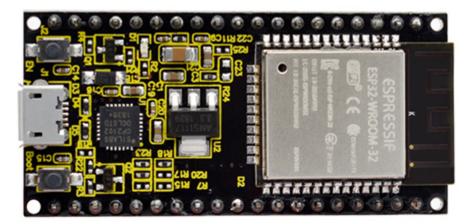
### TWO

## 2.KIT LIST



THREE

# **3.KEYESTUDIO ESP32 CORE BOARD**



#### Introduction

Keyestudio ESP32 Core board is a Mini development board based on the ESP-WROOM-32 module. The board has brought out most I/O ports to pin headers of 2.54mm pitch. These provide an easy way of connecting peripherals according to your own needs.

When it comes to developing and debugging with the development board, the both side standard pin headers can make your operation more simple and handy.

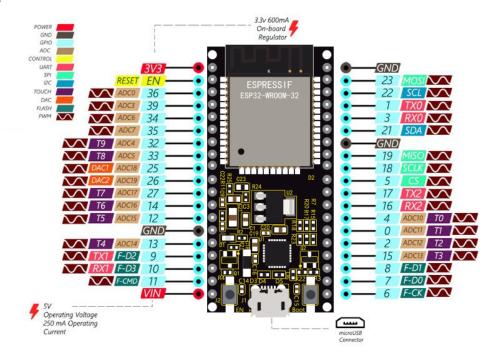
The ESP-WROOM-32 module is the industry's leading integrated WiFi + Bluetooth solution with less than 10 external components. It integrates antenna switches, RF balun, power amplifiers, low noise amplifiers, filters as well as power management modules. At the same time, it also integrates TSMC's low-power 40nm technology, power performance and RFperformance, making it safe, reliable and easy to expand to a variety of applications.

Specifications

- Microcontroller: ESP-WROOM-32 Module
- USB to serial port chip: CP2102-GMR
- Working voltage: DC 5V
- Working current80mAAverage
- Current supply500mAMinimum
- Working temperature range : -40°C ~ +85°C
- WiFi modeStation/SoftAP/SoftAP+Station/P2P
- WiFi protocol802.11b/g/n/e/i802.11nspeed up to 150 Mbps
- WiFi frequency range2.4 GHz ~ 2.5 GHz

- Bluetooth protocolconform to Bluetooth v4.2 BR/EDR and BLE Standard
- Dimensions55\*26\*13mm
- Weight9.3g

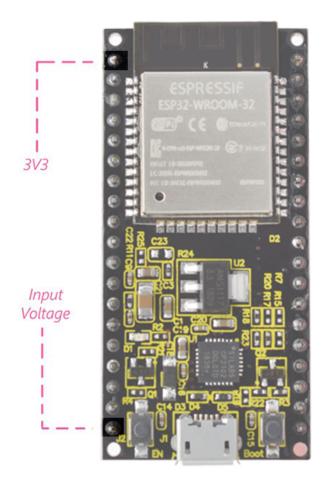
#### Pin out



ESP32 has fewer pins than commonly used processors, but it doesn't have any problems reusing multiple functions on pins.

**Warning**: The pin voltage level of the ESP32 is 3.3V. If you want to connect the ESP32 to another device with an operating voltage of 5V, you should use a level converter to convert the voltage level.

**Power Pins:** The module has two power pins +5V and 3.3V. You can use these two pins to power other devices and modules.



GND Pins: The module has three grounded pins.

**Enable pin (EN) :** This pin is used to enable and disable modules. The pin enables module at high level and disables module at low level.

**Input/Output pins (GPIO) :** You can use 32 GPIO pins to communicate with LEDs, switches and other input/output devices. You can also pull these pins up or down internally.

**Note:** Though GPIO6 to GPIO11 pins (SCK/CLK, SDO/SD0, SDI/SD1, SHD/SD2, SWP/SD3 and SCS/CMD pins) are used for SPI communication for the internal module, which are not recommended.

**ADC:** You can use the 16 ADC pins on this module to convert analog voltages (the output of some sensors) into digital voltages. Some of these converters are connected to internal amplifiers and which are capable of measuring small voltages with high accuracy.

DAC: ESP32 module has two A/D converters with 8-bit precision.

**Touch pad:** There are 10 pins on the ESP32 module that are sensitive to capacitance changes. You can attach these pins to certain PCB's pads and use them as touch switches.

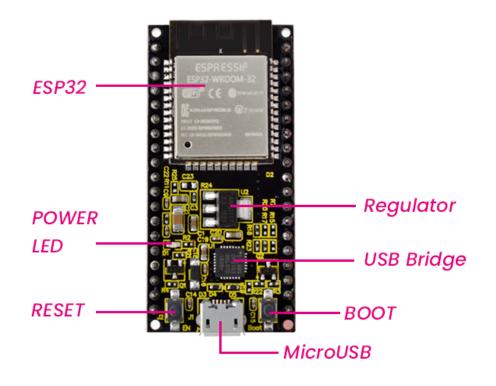
**SPI:** There are two SPI interfaces on the module, which can be used to connect the display screen, SD/microSD memory card module as well as external flash memory, etc.

I2C: SDA and SCL pins are used for I2C communication.

**Serial Communication (UART) :** There are two UART serial interfaces on this module, which can be used to transfer up to 5Mbps of information between two devices . The UARTO also has CTS and RTS control functions.

**PWM:** Almost all ESP32 input/output pins can be used for PWM(pulse-width modulation). Using these pins can control the motors, LED lights and color changes for some other sensorsfor example: color sensor, etc.

#### Components



FOUR

# **GETTING STARTED WITH ARDUINO**

Development Environment Configuration

Click on the link to enter the development environment setup tutorial *Development Environment Configuration-Windows* 

# 4.1 Windows System



### 4.1.1 Download and install Arduino software

1First, enter arduino's official website: https://www.arduino.cc/, and click "SOFTWARE" to enter the download page. As shown in the figure below





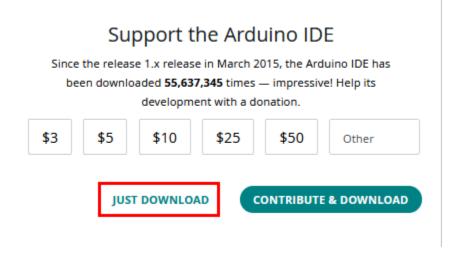
# Downloads



(2) Then, select and download the corresponding installer for your operating system. If you are a Windows user, please select "Windows Installer" to download to install the driver correctly.



Choose to click the **Windows Win7 and newer** to download Arduino 1.8.16 version installer, which requires manual installation. But when click the **Windows ZIP File**, the Arduino 1.8.16 zip file will be downloaded directly, just unzip it to complete the installation.



In general, you can click JUST DOWNLOAD to download it.

(3) After the Arduino IDE is downloaded, continue the installation. When you receive the warning from the operating system, please allow the driver installation by clicking **I** Agree first, and then click **Next** after selecting the components to install.

💿 Arduino Setup: License Agreement	_		$\times$
Please review the license agreement befor accept all terms of the agreement, click I A		ino. If you	
GNU LESSER GENERAL PUBLIC LICENSE			^
Version 3, 29 June 2007			
Copyright (C) 2007 Free Software Foundation, Inc. < <u>http://fsf.org/</u> >			
Everyone is permitted to copy and distribute verb document, but changing it is not allowed.	Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.		
This version of the GNU Lesser General Public Lice and conditions of version 3 of the GNU General Pu			
by the additional permissions listed below.	blic License, sup	piementeu	· 🗸
Cancel Nullsoft Install System v3.0		I Agr	ee
1			
o Arduino Setup: Installation Options			×
<ul> <li>Arduino Setup: Installation Options</li> <li>Check the components you want to install you don't want to install. Click Next to con</li> </ul>	 and uncheck the tinue.	componer	
Check the components you want to install you don't want to install. Click Next to con Select components to install:	tinue. uino software	componer	
Check the components you want to install you don't want to install. Click Next to con Select components to install:	tinue. uino software driver art Menu shortcu		
Check the components you want to install you don't want to install. Click Next to con Select components to install:	tinue. uino software driver art Menu shortcu sktop shortcut		
Check the components you want to install you don't want to install. Click Next to con Select components to install:	tinue. uino software driver art Menu shortcu sktop shortcut		

4Select the installation directory (we recommend keeping the default directory), and then click Install.

🤝 Arduino Setup: Installation Folder	_		$\times$
Setup will install Arduino in the following for folder, click Browse and select another for installation.			
Destination Folder			
C:\Users\Administrator\Desktop\Arduino		Browse	
Space required: 543.4MB			
Space available: 14.7GB			
Cancel Nullsoft Install System v3.0	< Back	Inst	all

5Select Install if the following screen appears.

Windows Security				
Would you like to install this device software?				
Name: Arduino USB Driver Publisher: Arduino LLC				
Always trust software from "Arduino LLC". Install Don't Install				
You should only install driver software from publishers you trust. How can I decide which device software is safe to install?				

This process extracts and installs all the necessary files to properly execute the Arduino software (IDE).

💿 Arduino Setup: Installing	_		$\times$
Extract: iox128a1u.h			
Show details			
Cancel Nullsoft Install System v3.0	< Back	Clos	se

After installation is complete, an Arduino Software shortcut will be generated in the desktop.

💿 Arduino Setup: Completed	_		
Completed			
Show details			
Cancel Nullsoft Install System v3.0	< Back	Clo	se 📗

### 4.1.2 Install a driver on Windows

NoteIf you have installed the driver, just skip it

Before using the ESP32 board, you must install a driver, otherwise it can not communicate with the computer. Unlike the USB series chip (ATMEGA8U2) of the Arduino UNO R3, the ESP32 board is used the CP2102 chip USB series chip and USB type C interface.

The driver of the CP2102 chip is included in 1.8.0 version and newer version of Arduino IDE. Usually, you connect the board to the computer and wait for Windows to begin its driver installation process. After a few moments, the process will succeed.

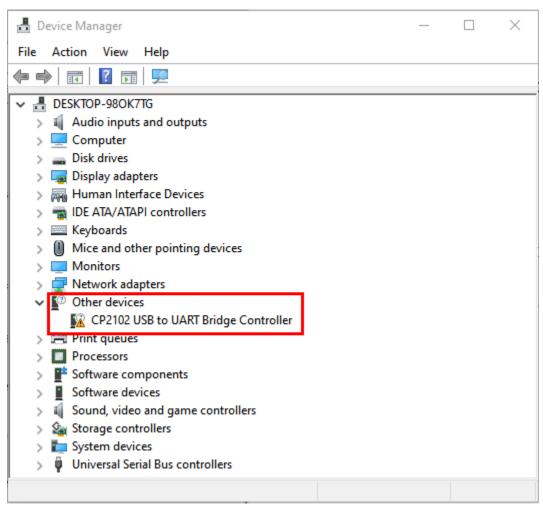
Note:

<sup>1.</sup> Please make sure that your IDE is updated to 1.8.0 or newer version

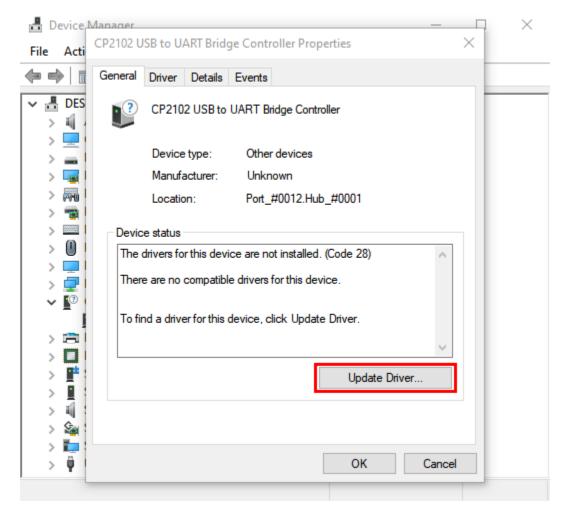
2. If the version of Arduino IDE you download is below 1.8, you should download the driver of CP2102 and install it manually.

Link to download the driver of CP2102: CP2102-Driver-File-Windows

If the driver installation process fail, you need to install the driver manually. Open device Manager for your computer and right-click"the computer" $\rightarrow$ click"Properties" $\rightarrow$ Click"Device Manager". Look under Ports (COM & LPT) or other device, a yellow exclamation mark means that the CP2102 driver installation failed.



It shows that the driver for CP2102 fails to be installed successfully if there is a yellow mark. Doubleclick CP2102 USB to UART Bridge Controller, and then click "Update drive..." to update the driver.

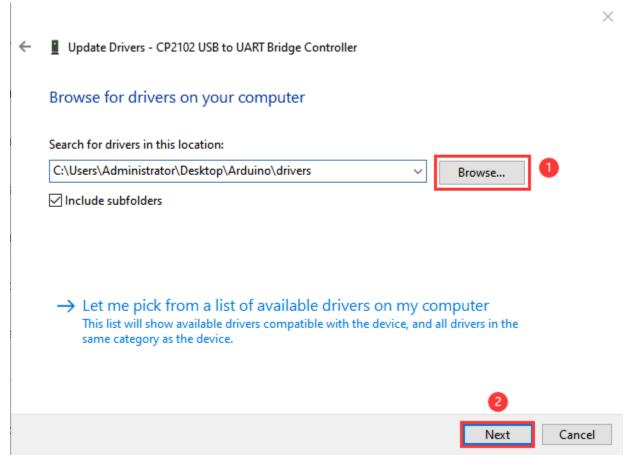


Click "Browse my computer for drivers" to find the Arduino software we installed or downloaded.

How	v do you want to search for drivers?		
→	Search automatically for drivers Windows will search your computer for the best your device.	available driver and install it on	
$\rightarrow$	Browse my computer for drivers Locate and install a driver manually.		

There is a **drivers** folder in Arduino software installed package Arduino ), open the driver folder and you can see the driver of CP210X series chips.

Click"Browse...", then find the drivers folder, or you could enter"driver" to search in rectangular box, then click"Next",



After a while, the driver is installed successfully.

 $\times$ 

Update Drivers - Silicon Labs CP210x USB to UART Bridge (COM3)

Windows has successfully updated your drivers

Windows has finished installing the drivers for this device:



Silicon Labs CP210x USB to UART Bridge



Open the computer device Manager again, you can see that the CP2102 driver has been successfully installed, and find the yellow exclamation mark disappear.

🖶 Device Manager	_		$\times$	
File Action View Help				
V 🛔 DESKTOP-980K7TG				
> 🐗 Audio inputs and outputs				
> 💻 Computer				
> 👝 Disk drives				
> 🏣 Display adapters				
> 🛺 Human Interface Devices				
> 📹 IDE ATA/ATAPI controllers				
> 🔤 Keyboards				
> II Mice and other pointing devices				
> 💻 Monitors				
> 🚽 Network adapters				
V 🛱 Ports (COM & LPT)				
Silicon Labs CP210x USB to UART Bridge (COM3)				
> 🖻 Print queues				
> Processors				
> P Software components				
Software devices				
> 4 Sound, video and game controllers				
> 🍇 Storage controllers				
> 🏣 System devices				
> 🏺 Universal Serial Bus controllers				

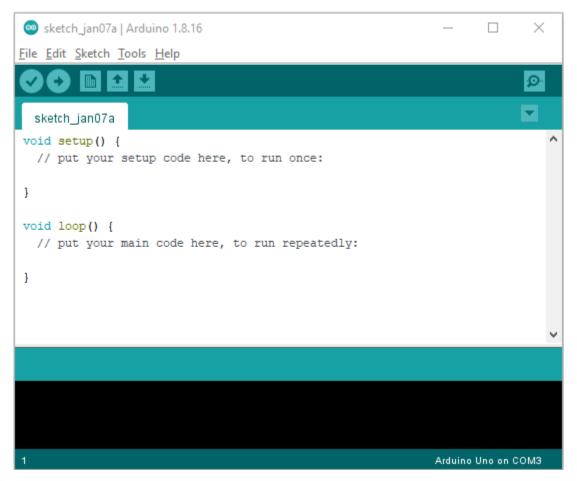
### 4.1.3 Install the ESP32 on Arduino IDE

The installation process for ESP32 is almost the same as that for ESP8266. If you are to install ESP32 on an Arduino IDE, follow these steps

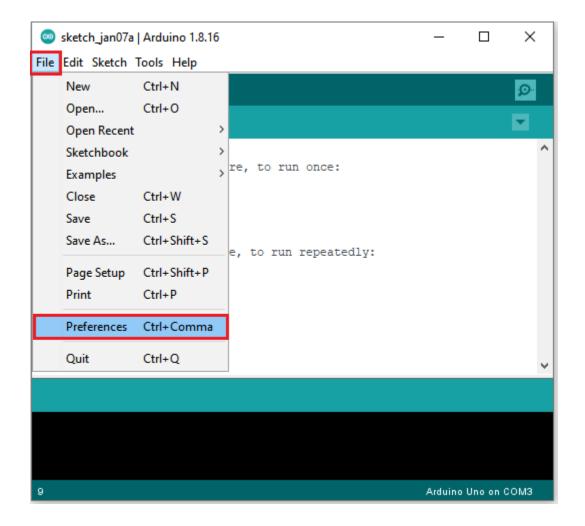
Noteyou need to download Arduino IDE 1.8.5 or advanced version to install the ESP32.



1) Click the icon Arduino to open the Arduino IDE



2Click"File"  $\rightarrow$  \*\*"Preferences"\*\*copy the website addresshttps://dl.espressif.com/dl/package\_esp32\_index.json in the "Additional Boards Manager URLs:"and then click" OK" to save the address.



Preferences	×				
Settings Network					
Sketchbook location:					
C:\Users\Administrator\Docu	ments\Arduino Browse				
Editor language:	System Default $\sim$ (requires restart of Arduino)				
Editor font size:	12				
Interface scale:	Automatic 100 💼 🕷 (requires restart of Arduino)				
Theme:	Default theme $\checkmark$ (requires restart of Arduino)				
Show verbose output during:	compilation upload				
Compiler warnings:	None 🗸				
🗌 Display line numbers	Enable Code Folding				
✓ Verify code after upload	Use external editor				
🗹 Check for updates on sta	rtup 🗹 Save when verifying or uploading				
🗌 Use accessibility featur	Use accessibility features				
Additional Boards Manager URLs: https://dl.espressif.com/dl/package_esp32_index.json 1					
More preferences can be edited directly in the file					
C:\Users\Administrator\AppDa	C:\Users\Administrator\AppData\Local\Arduino15\preferences.txt				
(edit only when Arduino is r	not running) 2				
	OK Cancel				

3First click "Tools"  $\rightarrow$  "Board:" and click "Boards Manager..." to enter "Boards Manager", enter "ESP32" in the boxafter "ALL", then select the latest version to install, the installation package is not large, click "Install" to Install the plug-in, as shown in the figure below.

🐵 Boards Manager 🚺	×
Type All V ESP32	
esp32 by Espressif Systems Boards included in this package: ESP32 Dev Module, WEMOS LoLin32, WEMOS D1 MINI ESP32. <u>More Info</u>	2 1.0.6 v Install
	Close

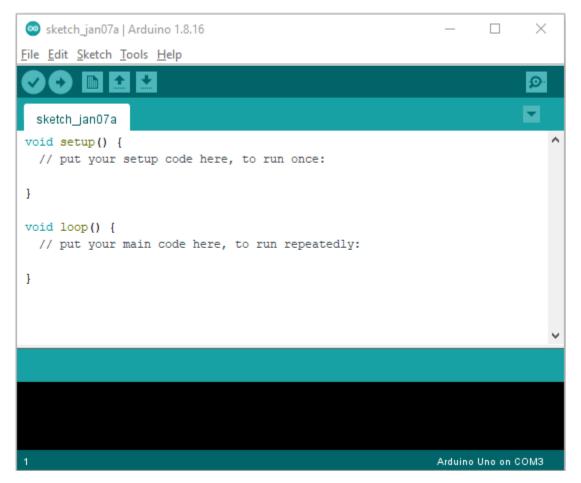
🥯 Boards Manager	$\times$
Type All ~ ESP32	
esp32 by Espressif Systems Boards included in this package: ESP32 Dev Module, WEMOS LoLin32, WEMOS D1 MINI ESP32. <u>More Info</u>	^
Installing	
Downloading boards definitions. Downloaded 17,637kb of 51,126kb.	<ul><li>✓</li></ul>

4. After successful installation, click"Close"to Close the page

### 4.1.4 Arduino IDE Setting:



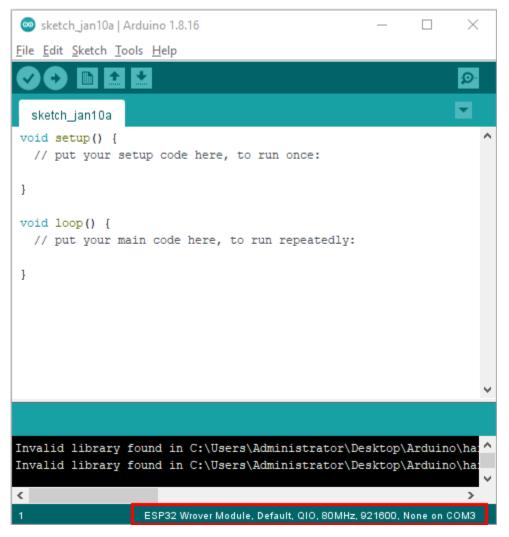
1Click the icon Arduino to open the Arduino IDE.



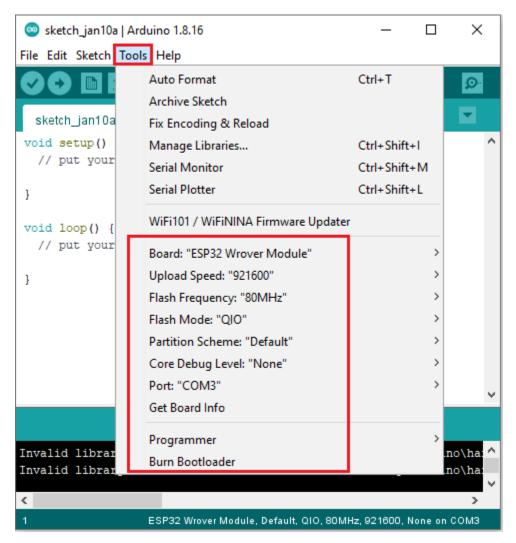
2When downloading the code to the board, you must select the correct name of Arduino board that matches the board connected to your computer, then click"**Tools**" $\rightarrow$ "**Board:**". As shown below ;

(Note: we use the ESP32 board in this tutorial; therefore, we select ESP32 Arduino\*\*)\*\*

🥯 sketch_jan10a   A	rduino 1.8.16	- 0	×					
File Edit Sketch Too	ols Help							
sketch_jan10a     Archive Sketch_jan10a       void setup()     Manage Lib       // put your     Serial Monit	Auto Format Archive Sketch Fix Encoding & Reload Manage Libraries Serial Monitor	Ctrl+T Ctrl+Shift+I Ctrl+Shift+M						
<pre>} void loop() {    // put your </pre>	Serial Plotter WiFi101 / WiFiNINA Firmwa	Ctrl+Shift+L re Updater				TTGO LoRa32-OLED V1 XinaBox CW02 SparkFun ESP32 Thing		
}	Board: "Arduino Uno" Port: "COM3" Get Board Info	2	Boards Manager Arduino AVR Boards Arduino Mbed OS RP2040 Boards		ds >	u-blox NINA-W10 series (ESP32) Widora AIR Electronic SweetPeas - ESP320		
	Programmer: "AVRISP mkll" Burn Bootloader		ESP32	Arduino	3	Nano32 LOLIN D32 LOLIN D32 PRO		
			~			WEMOS LOLIN32 Dongsen Tech Pocket 32 "WeMos" WiFi&Bluetooth Battery		
						ESPea32 Noduino Quantum		
1		Arduino Uno on	сомз			Node32s Hornbill FSP32 Dev		

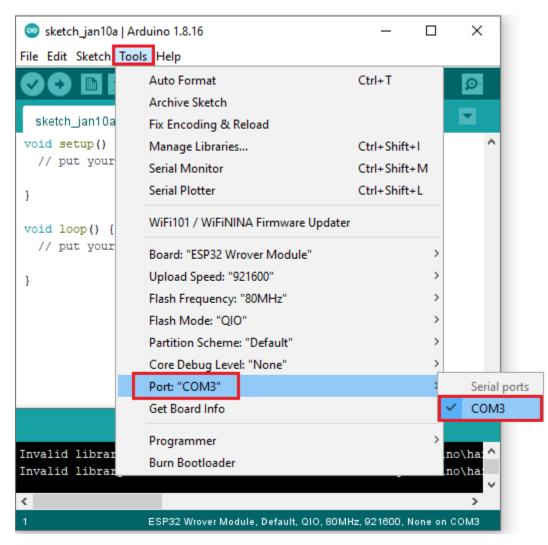


Set the board type as follows:

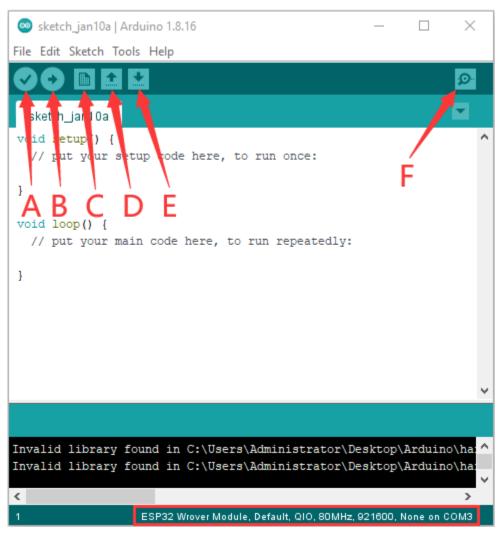


Then select the correct COM port (the corresponding COM port can be seen after the driver is installed successfully).

ᡖ Device Manager	_	$\times$
File Action View Help		
V 🗄 DESKTOP-980K7TG		
> 🐗 Audio inputs and outputs		
> 💻 Computer		
> 🚘 Disk drives		
> 🏣 Display adapters		
> 🛺 Human Interface Devices		
> 🦷 IDE ATA/ATAPI controllers		
> 🔤 Keyboards		
> III Mice and other pointing devices		
> 💻 Monitors		
> 🚽 Network adapters		
✓		
Silicon Labs CP210x USB to UART Bridge (COM3)		
> 💼 Print queues		
> Processors		
> P Software components		
Software devices		
> 4 Sound, video and game controllers		
> 🍇 Storage controllers		
> 🏣 System devices		
> 🏺 Universal Serial Bus controllers		



Before a code was uploaded to the ESP32 mainboard, we have to demonstrate the functionality of each symbol that appeared in the Arduino IDE toolbar.



- A- Used to verify whether there is any compiling mistakes or not.
- B- Used to upload the sketch to your Arduino board.
- C- Used to create shortcut window of a new sketch.
- D- Used to directly open an example sketch.
- E- Used to save the sketch.
- F- Used to send the serial data received from board to the serial monitor.

# 4.2 Arduino MacOS

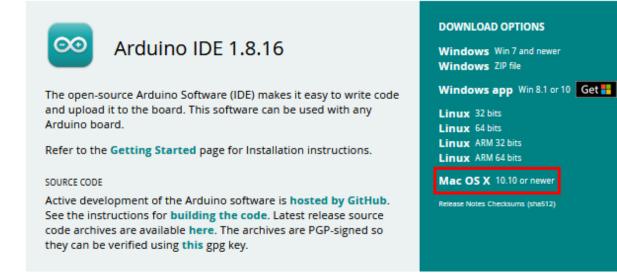
## 4.2.1 Development Environment Configuration-Mac OS



Click on the link to enter the development environment setup tutorial Development Environment Configuration-MacOS

## 4.2.2 Download Arduino IDE:

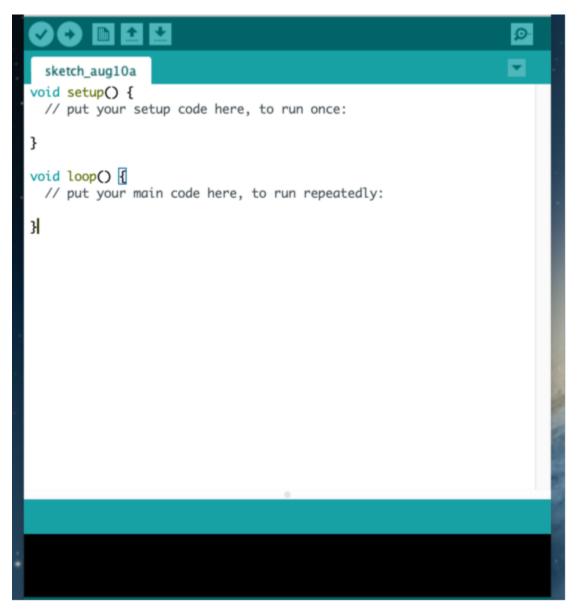
# Downloads



## 4.2.3 How to install the CP2102 driver

(Note: If you haven't installed the driver installed, please do the following.)

(1) Connect the ESP32 motherboard to your MacOS computer using a USB cable and open the Arduino IDE.



Click Tools→Board: ESP32 Dev Module and /dev/cu.usbserial-0001.

<b>É Arduino</b> File Edit Sketch	Tools Help		0		拼	*	ŝ
• • •	Auto Format						
	Archive Sketch						
	Fix Encoding & Reload						
sketch_jan19a	Manage Libraries						
<pre>void setup() {</pre>	Serial Monitor	<sub>ት</sub> ж M	1				
// put your setup code here, to ru	Serial Plotter						
}	WiFi101 / WiFiNINA Firmware Updater						
<pre>void loop() {     // put your main code here, to run</pre>	ArduBlock						
}	Board: "ESP32 Dev Module"	>	•				
	Upload Speed: "921600"	>	•				
	CPU Frequency: "240MHz (WiFi/BT)"	>	•				
	Flash Frequency: "80MHz"	>	•				
	Flash Mode: "QIO"	>	•				
	Flash Size: "4MB (32Mb)"	>	•				
	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"	>	•				
	Core Debug Level: "None"	>					
	PSRAM: "Disabled"	>				_	
	Port						
	Get Board Info		/de	ev/cu.Bluet	ooth-Inc	oming-Po	rt
	Programmer	>	/d	ev/cu.usbs	erial-000	)1	
	Burn Bootloader						

Click to upload the test code

Note: If the the upload fails, follow the steps below to install the CP2102 driver. Perform step (2) to (16).

2Download link for CP2102CP2102-Driver-File-MAC.zip

3Download MacOS version

#### Download for WinCE

Platform	Software	Release Notes
WinCE 6.0 (2.1)	Download VCP (276 KB)	Download WinCE 6.0 Revision History
WinCE 5.0 (2.1)	Download VCP (271 KB)	Download WinCE 5.0 Revision History

## Download for Macintosh OSX (v5.3.5)

Platform	Software	Release Notes
📫 Mac OSX	Download VCP (832 KB)	Download Mac VCP Revision History

#### Download for Linux

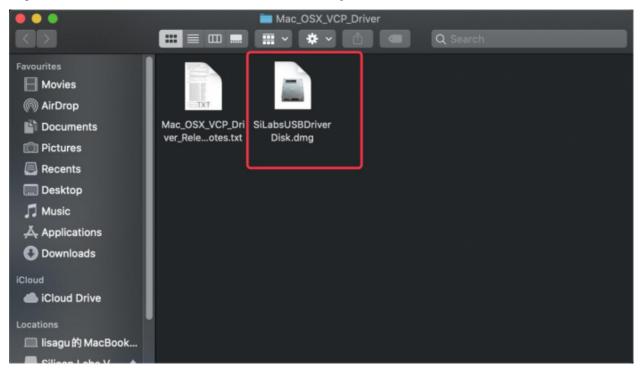
Platform Software		Release Notes
∆ Linux 3.x.x and 4.x.x	Download VCP (10.0 KB)	Download Linux 3.x.x and 4.x.x VCP Revision History
🛕 Linux 2.6.x	Download VCP (10.2 KB)	Download Linux 2.6.x VCP Revision History

\*Note: The Linux 3.x.x and 4.x.x version of the driver is maintained in the current Linux 3.x.x and 4.x.x tree at www.kernel.org.

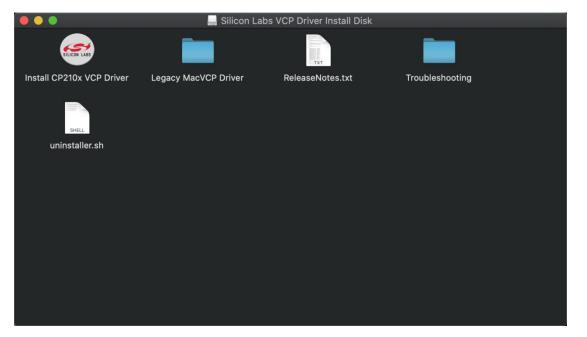
4Unzip the downloaded package



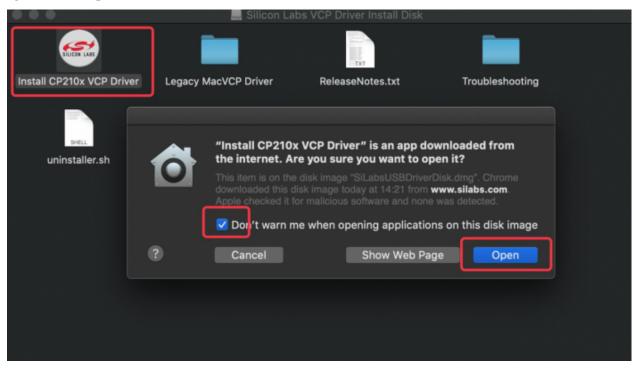
5Open the folder and double-click"SiLabsUSBDriverDisk.dmg"file



Then you can see the following file



Double-click\*\*\*'Install CP210x VCP Driver"tap\*\*\*'Don't warn me when opening application on this disk image"and click"Open"



7Click"Continue"

	Install CP210x VCP Driver
<ul> <li>Introduction</li> <li>License</li> <li>Info</li> <li>Install</li> </ul>	You will be guided through the steps necessary to install the virtual COM port driver for the Silicon Labs CP210x USB to UART Bridge.
Guidance	
Rebuild Cache	
• Summary	Back Continue

#### 8Click"Agree"then tap"Continue"

0 0	Install CP210x VCP Driver
license agreement.	are you must agree to the terms of this software ue or Disagree to cancel the installation.
Read License	Disagree Agree TO
<ul> <li>Install</li> <li>Guidance</li> <li>Rebuild Cache</li> <li>Summary</li> </ul>	AGREEMENT CONSTITUTES YOUR AND (IF APPLICABLE) YOUR COMPANY'S ASSENT TO AND ACCEPTANCE OF THIS END-USER LICENSE AGREEMENT (THE "LICENSE" OR "AGREEMENT"). IF YOU DO NOT AGREE WITH ALL OF THE TERMS, YOU MUST NOT USE THIS PRODUCT. WRITTEN APPROVAL IS NOT A PREREQUISITE TO THE VALIDITY OR ENFORCEABILITY OF THIS AGREEMENT, AND NO SOLICITATION OF SUCH WRITTEN APPROVAL BY OR ON BEHALF OF SILICON LABORATORIES, INC. ("SILICON LABS") SHALL BE CONSTRUED AS AN INFERENCE TO THE CONTRARY. IF THESE TERMS ARE CONSIDERED AN OFFER BY SILICON LABS, ACCEPTANCE IS EXPRESSLY LIMITED TO THESE TERMS.
SILICON LABS	LICENSE AND WARRANTY: The Licensed Product and the embedded Software which is made the subject of this License is either the property of SILICONULABS or a third party from whom Back Continue

9Click"Continue"then input your user password

	Install CP210x VCP Driver
<ul><li>Introduction</li><li>License</li></ul>	To Be Installed: Version 5.3.5 Currently Installed: None
Info	Version 5.3.5 will be installed in /Library/Extensions/.
Install	
Guidance	
Rebuild Cache	
Summary	
SILICON LABS	You will be prompted to enter your password. Back Continue
helper	ur password to allow this. me:
	Cancel Install Helper

10. Select"Select Open Security Preferences"

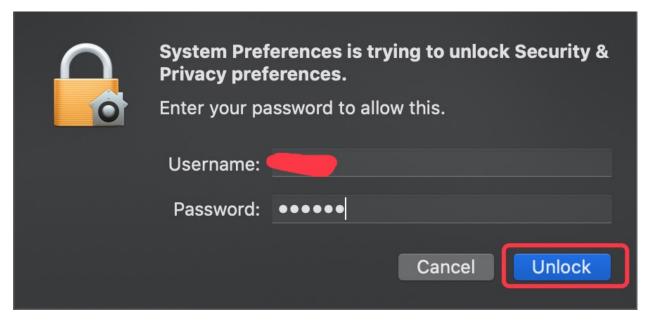
## System Extension Blocked

A program tried to load new system extension(s) signed by "Silicon Laboratories Inc", which will be incompatible with a future version of macOS. If you want to enable these extensions, open Security & Privacy System Preferences.

Open Security Preferences OK

11Click on security lock and enter your user password to authorize.

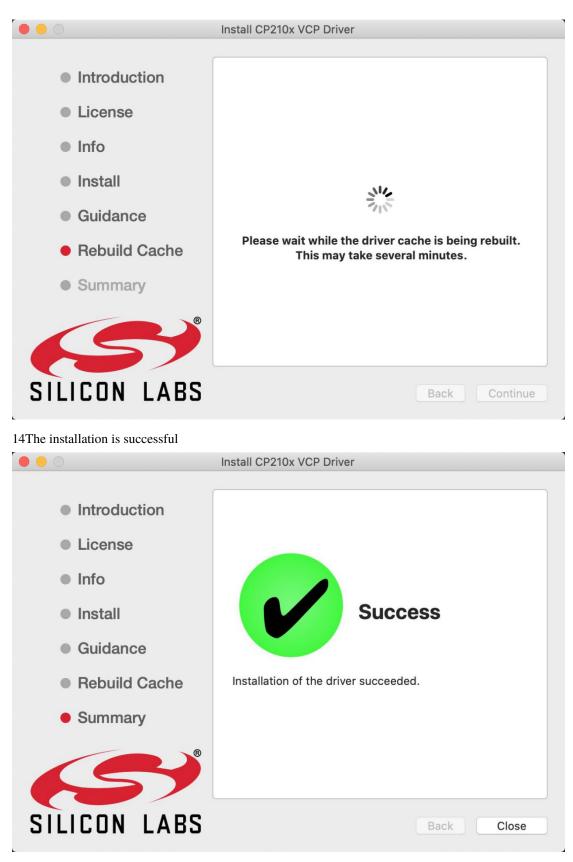
•••		Security & Privacy	Q Search	
	General	FileVault Firewall	Privacy	
	A login password has been se	t for this user Change	Password	
		the screen is locked	or screen saver begins et Lock Message	
	☑ Disable automatic logir			
	Allow apps downloaded from:			
	App Store App Store and identified	ed developers		
	System software from develog blocked from loading.	per "Silicon Laboratories II	nc" was Allow	
CI	ick the lock to make changes.			



11) When you see that the lock is opened, click "Allow".

•••		Security & Privacy	Q Search
	General	FileVault Firewall Privacy	
	A login password has been set Require password 1 ha Show a message when the Disable automatic login	our is locked Set Lock Mess	
	Allow apps downloaded from: App Store App Store and identified Anywhere	d developers	
	System software from develope blocked from loading.	er "Silicon Laboratories Inc" was	Allow
	ick the lock to prevent further ch	nanges.	Advanced ?

Return to the installation interface and wait for the installation as prompted.



15Open arduinoIDEclick"Tools" and tap"ESP32 Dev Module" and"/dev/cu.usbserial-0001".

🗯 Arduino File Edit Sketch	Tools Help	👃 @		× #	≵ ⊡	ŝ
• • •	Auto Format					
	Archive Sketch					
	Fix Encoding & Reload					
sketch_jan19a	Manage Libraries					
<pre>void setup() {     // put your setup code here, to ru</pre>	Serial Monitor					
77 put your setup code here, to ru	Serial Plotter					
}	WiFi101 / WiFiNINA Firmware Updater					
<pre>void loop() {     // put your main code here, to run</pre>	ArduBlock					
}	Board: "ESP32 Dev Module"					
	Upload Speed: "921600"					
	CPU Frequency: "240MHz (WiFi/BT)"					
	Flash Frequency: "80MHz"					
	Flash Mode: "QIO"					
	Flash Size: "4MB (32Mb)"					
	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"					
	Core Debug Level: "None"					
	PSRAM: "Disabled"			_		
	Port					
	Get Board Info		/dev/cu.Bl	uetooth-In	coming-Po	rt
	Programmer		/dev/cu.us	bserial-00	01	
	Burn Bootloader					

16Click to upload the program, and you can see the program burned successfully

#### CHAPTER

FIVE

# **ARDUINO TUTORIAL**

# 5.1 Download Arduino code and library files

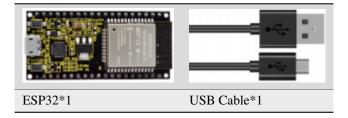
Click on the link to download Arduino code files and library files: Arduino-Codes-and-Libraries

# 5.2 Project 01: Hello World

## 5.2.1 1.Introduction

For ESP32 beginners, we'll start with some simple things. In this project, you just need an ESP32 mainboard, a USB cable and a computer to complete "Hello World!" Project. It is not only a communication test for ESP32 mainboard and computer, but also a primary project for ESP32.

#### 5.2.2 2.Components



## 5.2.3 3.Wiring

In this project, we use a USB cable to connect the ESP32 to the computer.

#### 5.2.4 4.Test Code

```
/*
* Filename : Hello World
* Description : Enter the letter R, and the serial port displays"Hello World".
* Auther :http//www.keyestudio.com
*/
char val;// defines variable "val"
void setup()
{
Serial.begin(115200);// sets baudrate to 115200
}
void loop()
{
 if (Serial.available() > 0) {
  val=Serial.read();// reads symbols assigns to "val"
  if(val=='R')// checks input for the letter "R"
  { // if so,
   Serial.println("Hello World!");// shows "Hello World !".
  }
 }
}
```

Before uploading the project code to ESP32click"Tools"→"Board" and select"ESP32 Wrover Module".

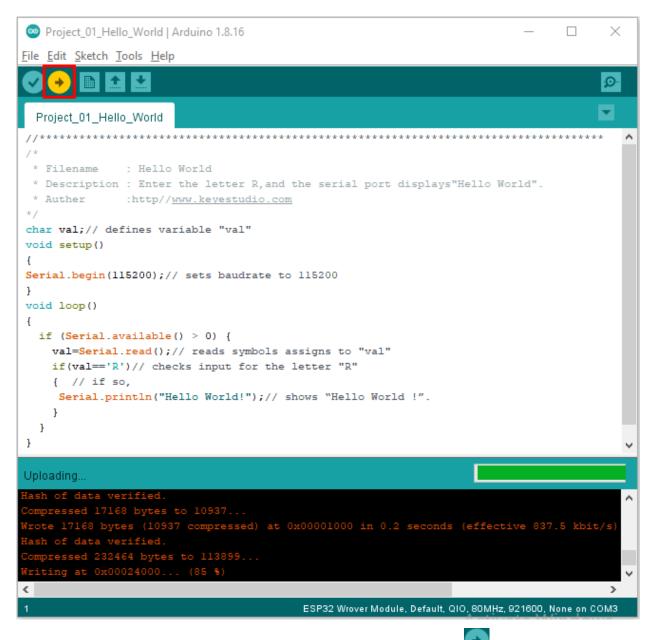
	Auto Format	Ctrl+T		ø	
	Archive Sketch			~	
roject_01_He	Fix Encoding & Reload				Δ
*****	Manage Libraries	Ctrl+Shift+I	*****	**** /	ESP32 Dev Module
	Serial Monitor	Ctrl+Shift+M			<ul> <li>ESP32 Wrover Module</li> </ul>
ilename escriptio	Serial Plotter	Ctrl+Shift+L	lays"Hello World".		ESP32 Pico Kit
ather	Scharriotter	Guittoninette	Tays Herro Worrd .		TTGO LoRa32-OLED V1
	WiFi101 / WiFiNINA Firmware Upd	ater			XinaBox CW02
<pre>val;// d setup()</pre>	Board: "ESP32 Wrover Module"	2	Boards Manager	_	SparkFun ESP32 Thing
	Upload Speed: "921600"	3	Arduino AVR Boards		u-blox NINA-W10 series (ESP32)
al.begin(	Flash Frequency: "80MHz"	,	Arduino Mbed OS RP204	0 Boards	Widora AIR
loop()	Flash Mode: "QIO"	,	ESP32 Arduino	o Dourus -	Electronic SweetPeas - ESP320
	Partition Scheme: "Default"		ESP8266 Boards (2.5.0)		Nano32
(Serial.	Core Debug Level: "None"	[	ESPOZOO BOdius (2.3.0)	1	LOLIN D32
val=Seria if(val=='	Port: "COM3"				LOLIN D32 PRO
{ // if					WEMOS LOLIN32
Serial.p	Get Board Info				Dongsen Tech Pocket 32
}	Programmer	>		`	"WeMos" WiFi&Bluetooth Batter
	Burn Bootloader				ESPea32
id library	found in C:\Users\Administrat	or\Desktop\Ardu	ino\hardware\espressif\	esp32\1:	Noduino Quantum
lid library	found in C:\Users\Administrat	or\Desktop\Ardu	ino\hardware\espressif\	esp32\1:	Node32s
				`	Hornbill ESP32 Dev
				>	Hornbill ESP32 Minima
	ESI	P32 Wrover Module, De	fault, QIO, 80MHz, 921600, None	on COM3	5. 0. 4. 50000

Select the serial port.

Project_01_Hello_	World   Arduino 1.8.16		- 🗆 X	
File Edit Sketch Too	ls Help			
	Auto Format	Ctrl+T	<u>o</u>	
	Archive Sketch			
Project_01_He	Fix Encoding & Reload			
//*******	Manage Libraries	Ctrl+Shift+I	*****	^
/* * Filename	Serial Monitor	Ctrl+Shift+M		
* Descriptio	Serial Plotter	Ctrl+Shift+L	lays"Hello World".	
* Auther */	WiFi101 / WiFiNINA Firmware Updater			
char val;// d void setup()	Board: "ESP32 Wrover Module"	>		
{	Upload Speed: "921600"	>		
Serial.begin(	Flash Frequency: "80MHz"	>		
void loop()	Flash Mode: "QIO"	>		
{ if (Serial.	Partition Scheme: "Default"	>		
val=Seria	Core Debug Level: "None"	>		
if(val=='	Port: "COM3"	3	Serial ports	
{ // if Serial.p	Get Board Info		✓ СОМЗ	
}	Programmer	>		×
	Burn Bootloader			
Invalid library f	ound in C:\Users\Administrator	\Desktop\Ardu:	ino\hardware\espressif\esp32\l:	~
Invalid library f	ound in C:\Users\Administrator	\Desktop\Ardu	ino\hardware\espressif\esp32\l:	
<			>	~
1	ESP32	Wrover Module, De	fault, QIO, 80MHz, 921600, None on COM3	

**Note:** For macOS users, if the uploading fails, please set the baud rate to 115200 before clicking.

Project_01_Hello_V	Vorld   Arduino 1.8.16				_	-		×
File Edit Sketch Tool	s Help							
	Auto Format	Ctrl+T						Ø
	Archive Sketch							
Project_01_He	Fix Encoding & Reload							
//*****	Manage Libraries	Ctrl+Shift+I	*****	****	*****	****	*****	rse 🖌
/* * Filename	Serial Monitor	Ctrl+Shift+M						
* Descriptio	Serial Plotter	Ctrl+Shift+L	lays"H	ello	World	۰.		
* Auther			-					
*/ char val;// d	WiFi101 / WiFiNINA Firmware Updater		-					
void setup()	Board: "ESP32 Wrover Module"	3	>					
{	Upload Speed: "921600"	:	• 921	600				
<pre>Serial.begin( }</pre>	Flash Frequency: "80MHz"	3	115	5200				
void loop()	Flash Mode: "QIO"	:	256	5000				
{	Partition Scheme: "Default"	2	230	0400				
if (Serial. val=Seria	Core Debug Level: "None"	2	512	2000				
if(val=='	Port: "COM3"	3	>		]			
{ // if	Get Board Info							
Serial.p }	_		-					
	Programmer	:	> 					-
	Burn Bootloader							
1	ESP32	Wrover Module, D	efault, QIO	, вомн	lz, 9216	00, N	one on C	омз
Click to download	the code to ESP32.							



**Note:** If uploading the code fails, you can press the Boot button on ESP32 after click and release the Boot button after the percentage of uploading progress appears, as shown below:





#### The code is uploaded successfully

🐵 Project_01_Hello_World   Arduino 1.8.16 – 🗆 🗙
<u>File Edit Sketch Tools H</u> elp
Project_01_Hello_World
//************************************
* Filename : Hello World
<pre>* Description : Enter the letter R, and the serial port displays"Hello World". * Auther :http//www.keyestudio.com */</pre>
char val;// defines variable "val"
<pre>void setup()</pre>
{
<pre>Serial.begin(115200);// sets baudrate to 115200 }</pre>
void loop()
{
if (Serial.available() > 0) {
<pre>val=Serial.read();// reads symbols assigns to "val"</pre>
<pre>if(val=='R')// checks input for the letter "R"</pre>
{ // if so,
<pre>Serial.println("Hello World!");// shows "Hello World !".</pre>
Done uploading.
Invalid library found in C:\Users\Administrator\Desktop\Arduino\hardware\espressif\esp32\1:
Invalid library found in C:\Users\Administrator\Desktop\Arduino\hardware\espressif\esp32\1
<
1 ESP32 Wrover Module, Default, QIO, 80MHz, 921600, None on COM3

#### 5.2.5 5.Test Result

After the code is uploaded successfully, power up with a USB cable and click the icon to enter the serial monitor. Set baud rate to 115200 and type "R" in the text box. Click "Send", and the serial monitor will display "Hello World!".

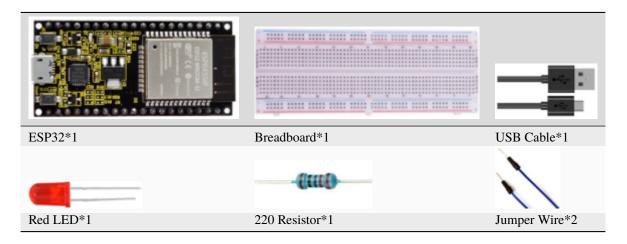


# 5.3 Project 02: Turn On LED

## 5.3.1 1.Introduction

In this project, we will show you how to light up the LED. We use the ESP32's digital pin to turn on the LED so that the LED is lit up.

#### 5.3.2 2.Components

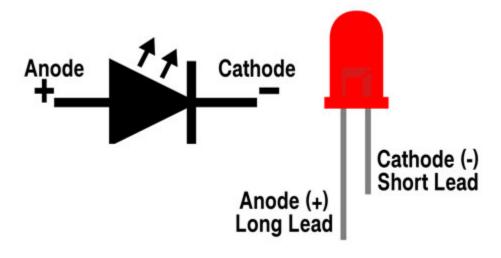


#### 5.3.3 3.Component Knowledge

1LED:

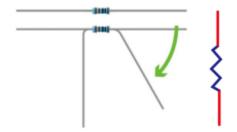


The LED is a semiconductor known as "light-emitting diode", which is an electronic device made from semiconducting materials(silicon, selenium, germanium, etc.). It has an anode and a cathode, the short lead is cathode, which connects to GND, the long lead is anode, which connects to 3.3V or 5V.



#### 2Five-band resistor

A resistor is an electronic component in a circuit that restricts or regulates the flow current to flow. On the left is the appearance of the resistor and on the right is the symbol for the resistance in the circuit. Its unit is(). 1 m = 1000 k1k = 1000.



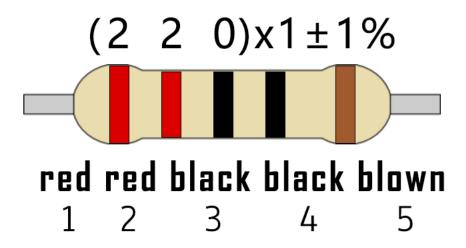
We can use resistors to protect sensitive components, such as LED. The strength of the resistance is marked on the body of the resistor with an electronic color code. Each color code represents a number, and you can refer to it in a resistance card.

- -Color 1 1st Digit.
- -Color 2 2nd Digit.
- -Color 3 3rd Digit.
- -Color 4 Multiplier.
- -Color 5 Tolerance.

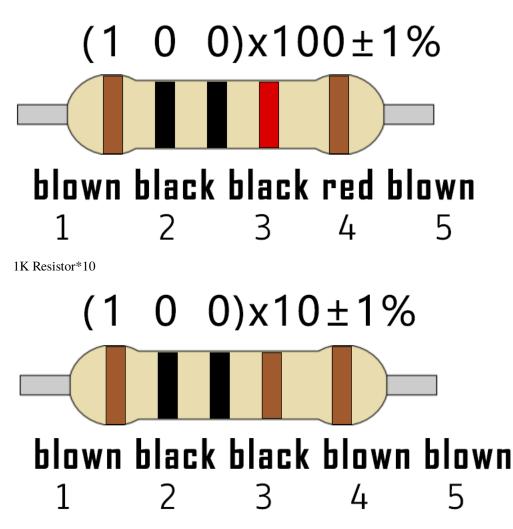
	1st Digit	2nd Digit	3rd Digit	Multiplier	Tolerance
Black		0	0	x1	
Brown	1	1	1	x10	± 1%
Red	2	2	2	x100	± 2%
Orange	3	3	3	x1K	± 3%
Yellow	4	4	4	x10K	± 4%
Green	5	5	5	x100K	±0.5%
Blue	6	6	6	x1M	±0.25%
Violet	7	7	7	x10M	±0.10%
Grey	8	8	8	x100M	±0.05%
White	9	9	9	x1G	
Gold				÷ 10	± 5%
Silver				÷ 100	± 10%

In this kit, we provide three five-band resistors with different resistance values. We three five-band resistors as an example.

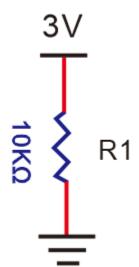
<sup>220</sup> Resistor\*10



10K Resistor\*10



In the same voltage, there will be less current and more resistance. The connection between current(I), voltage(V), and resistance ( $\mathbb{R}$ ) can be expressed by the formula: I=U/R. In the figure below, if the voltage is 3V, the current through R1 is: I = U / R = 3 V / 10 K = 0.0003A = 0.3 mA.

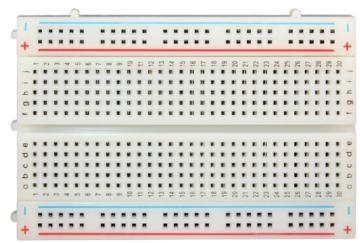


Don't connect a low resistance directly to the two poles of the power supply, which will cause excessive current to

damage the electronic components. Resistors do not have positive and negative poles.

#### **3Bread board**

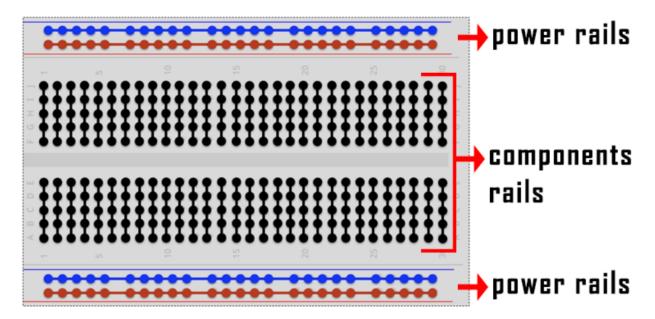
Breadboards are used to build and test circuits quickly before completing any circuit design. There are many holes in the breadboard that can be inserted into circuit components such as integrated circuits and resistors. A typical breadboard is shown below



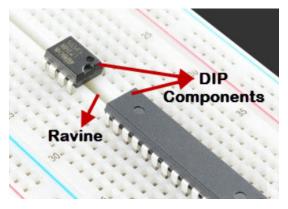
The breadboard has strips of metal, which run underneath the board and connect the holes on the top of the board. The metal strips are laid out as shown below. Note that the top and bottom rows of holes are connected horizontallywhile the remaining holes are connected vertically.



The first two rows (top) and the last two rows (bottom) of the breadboard are used for the positive pole (+) and negative pole (-) of the power supply respectively. The conductive layout of the breadboard is shown in the figure below:



When we connect DIP (Dual In-line Packages) components, such as integrated circuits, microcontrollers, chips and so on, we can see that a groove in the middle isolates the middle part, so the top and bottom of the groove is not connected. DIP components can be connected as shown in the following diagram:



# HORIZONTAL ROWS POWER BUS Each series of 5 sockets marked Each side of the breadboard has a pair of a-e and f-j are connected. vertical connections marked - and + Components connected to a row + **POWER**: Each + sign runs power will be connected to any other anywhere in the vertical column. part inserted in the same row. - GROUND: Each – sign runs to ground anywhere in the vertical column. CENTERLINE This line divides the breadboard in half, restricting electricity to one half or the other.

#### **4Power Supply**

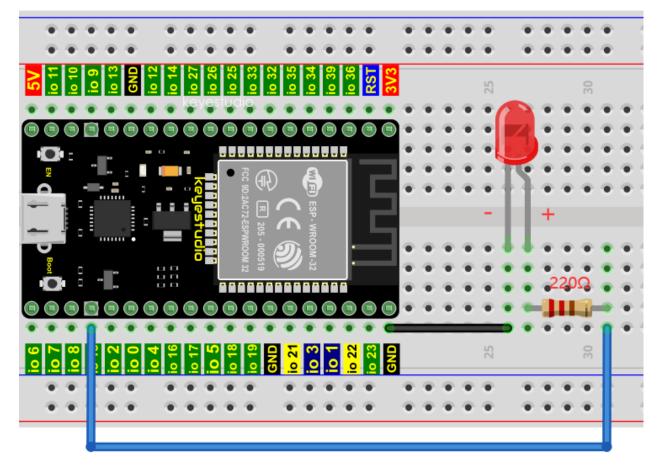
The ESP32 needs 3.3V-5V power supply. In this project, we will connect the ESP32 to the computer via an USB cable.

#### 5.3.4 4.Wiring Diagram

First, disconnect all power from the ESP32. Then build the circuit according to the wiring diagram. After the circuit is built and verified correctly, connect the ESP32 to your computer via a USB cable.

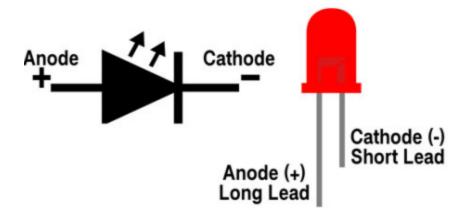
Note: Avoid any possible short circuits (especially connecting 3.3V and GND)!

**WARNING:** A short circuit can cause high current in your circuit, create excessive component heat and cause permanent damage to your hardware!

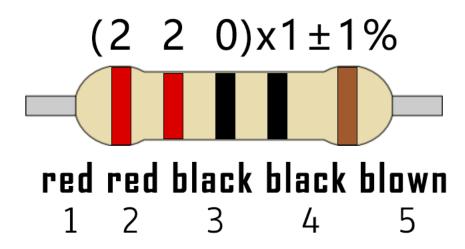


Note:

How to connect a LED



How to identify the 220 Five-band resistor



#### 5.3.5 5.Test Code

```
/*
* Filename : Turn On LED
* Description : Make an led on.
* Auther : http//www.keyestudio.com
*/
#define LED BUILTIN 15
// the setup function runs once when you press reset or power the board
void setup() {
 // initialize digital pin LED_BUILTIN as an output.
 pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
 digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
}
```

Before uploading the code to ESP32click"Tools"→"Board" and select"ESP32 Wrover Module".

Auto Format		Ctrl+T			Ø	
Archive Sketch Fix Encoding &	Reload				•	
*** Manage Librarie	5	Ctrl+Shift+I	****		1	ESP32 Wrover Module
e Serial Monitor		Ctrl+Shift+M				ESP32 Pico Kit
tio Serial Plotter		Ctrl+Shift+L				TTGO LoRa32-OLED V1
WiFi101 / WiFiN	NA Firmware Update	er				XinaBox CW02
B					_	SparkFun ESP32 Thing
Board: "ESP32 W		2	Boards Manager			u-blox NINA-W10 series (ESP32)
D Upload Speed: "		[	Arduino AVR Boards		,	Widora AIR
i Flash Frequency			Arduino Mbed OS R	2040 Boa	rds	Electronic SweetPeas - ESP320
Flash Mode: "Q		;	ESP32 Arduino		;	Nano32
Partition Schem		;	ESP8266 Boards (2.5.	0)	>	LOLIN D32
Core Debug Lev	el: "None"	>	H is the voltage lev	vel)		LOLIN D32 PRO
Port: "COM3"		>	* ***********	******	e l	WEMOS LOLIN32
Get Board Info						Dongsen Tech Pocket 32
Programmer		>			-	"WeMos" WiFi&Bluetooth Batte
Burn Bootloade						ESPea32
						Noduino Quantum
						Node32s
						Hornbill ESP32 Dev
	ESP3	2 Wrover Module D	efault, QIO, 80MHz, 921600, N	lone on CC	M3	Hornbill ESP32 Minima
	2010		21011, 210, 001112, 021000, I			EiroPostlo ESD22

Select the serial port.

Project_02_Turn_0	Dn_LED   Arduino 1.8.16			_		$\times$
File Edit Sketch Too	ols Help					
	Auto Format	Ctrl+T				Ø
	Archive Sketch					
Project_02_Tu	Fix Encoding & Reload					
//*******	Manage Libraries	Ctrl+Shift+I	*****			^
/* * Filename	Serial Monitor	Ctrl+Shift+M				
* Descriptio	Serial Plotter	Ctrl+Shift+L				
* Auther */ #define LED B	WiFi101 / WiFiNINA Firmware Updater					
fuerine ppp_p	Board: "ESP32 Wrover Module"	>				
// the setup	Upload Speed: "921600"	>	r the board			
<pre>void setup()     // initiali</pre>	Flash Frequency: "80MHz"	>				
pinMode (LED	Flash Mode: "QIO"	>				
<pre>} void loop() {</pre>	Partition Scheme: "Default"	>				
digitalWrit	Core Debug Level: "None"	>	H is the volt	age leve	el)	
}	Port: "COM3"	3	Serial ports			
//********	Get Board Info		🗸 сомз	*****	*****	*
	Programmer	>				~
	Burn Bootloader					
Invalid library f	ound in C:\Users\Administrator	Desktop\Ardu	ino\hardware\e	spressi	f\esp3	2\1:
Invalid library f	Cound in C:\Users\Administrator	Desktop\Ardu	ino\hardware\e	spressi	f\esp3	
<						~
1	ESP32	Wrover Module, De	fault, QIO, 80MHz, 9	21600, <u>No</u>	ne on C	-
1	ESP32	Wrover Module, De	fault, QIO, 80MHz, 9	21600, No	ne on C	омз

**Note:** For macOS users, if the uploading fails, please set the baud rate to 115200 before clicking

Project_02_Turn_0	On_LED   Arduino 1.8.16			_		×
File Edit Sketch Too	ols Help					
	Auto Format	Ctrl+T				Ø
	Archive Sketch					
Project_02_Tu	Fix Encoding & Reload					
//****	Manage Libraries	Ctrl+Shift+I	******	*		^
/* * Filename	Serial Monitor	Ctrl+Shift+M				
* Descriptio	Serial Plotter	Ctrl+Shift+L				
* Auther */	WiFi101 / WiFiNINA Firmware Updater					
<pre>#define LED_B</pre>	Board: "ESP32 Wrover Module"	>				
// the setup	Upload Speed: "921600"	;	921600			
<pre>void setup() // initiali</pre>	Flash Frequency: "80MHz"	;	115200			
pinMode (LED	Flash Mode: "QIO"	;	256000			
<pre>} void loop() {</pre>	Partition Scheme: "Default"	;	230400			
digitalWrit	Core Debug Level: "None"	;	512000	ltage leve	el)	
}	Port: "COM3"	>		1		
//******	Get Board Info		******	********	*****	¢.
	Programmer	>				~
	Burn Bootloader					
Invalid library f	Cound in C:\Users\Administrator Cound in C:\Users\Administrator					2\1: ~
< 1	ESP32	Wrover Module, De	efault, QIO, 8 <u>0M</u> H	z, 921600 <u>, No</u>	ne on <u>CC</u>	> СМЗ

Click to download the code to ESP32.



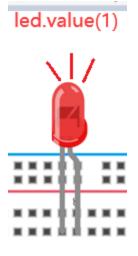
 Uploading...
 Image: Compression of the second sec

#### The code is uploaded successfully

Project_02_Turn_On_LED   Arduino 1.8.16	_		$\times$
<u>File Edit Sketch Tools H</u> elp			
			Ø
Project_02_Turn_On_LED			
//*************************************			^
* Filename : Turn On LED * Description : Make an led on.			
* Auther : http//www.keyestudio.com */			
#define LED_BUILTIN 15			
<pre>// the setup function runs once when you press reset or power the board void setup() { // initialize digital pin LED_BUILTIN as an output. pinMode(LED_BUILTIN, OUTPUT); } void loop() { digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the volta);</pre>	age le	vel)	
} //***********************************	*****	******	* •
Done uploading.			
Invalid library found in C:\Users\Administrator\Desktop\Arduino\hardware\e Invalid library found in C:\Users\Administrator\Desktop\Arduino\hardware\e			
invalla library found in C. (osers Administrator (besktop Aradino (nardware (e	spress	rr (espa	× 1.
<			>
6 ESP32 Wrover Module, Default, QIO, 80MHz, 9	21600, N	lone on C	омз

## 5.3.6 6.Test Result

After uploading the code successfully, power up with a USB cable and the LED will lit up.

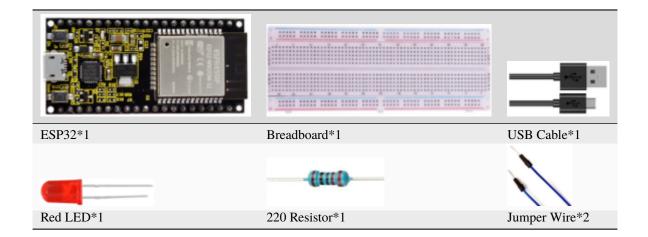


# 5.4 Project 03LED Flashing

### 5.4.1 1.Introduction

In this project, we will show you the LED flashing effect. We will work to use the ESP32's digital pin to turn on the LED and make it flash.

### 5.4.2 2.Components

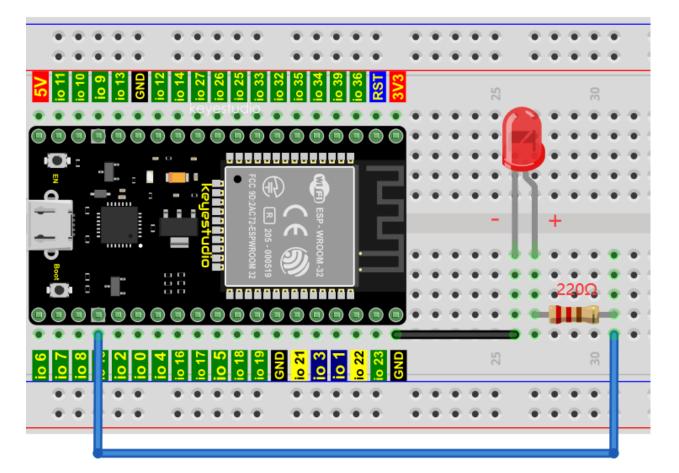


### 5.4.3 3.Wiring Diagram

First, disconnect all power from the ESP32. Then build the circuit according to the wiring diagram. After the circuit is built and verified correctly, connect the ESP32 to your computer via a USB cable.

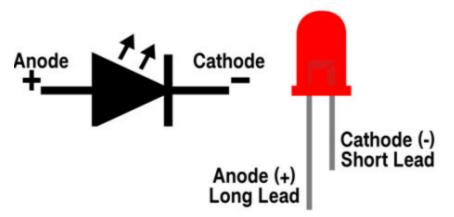
Note: Avoid any possible short circuits (especially connecting 3.3V and GND)!

**WARNING:** A short circuit can cause high current in your circuit, create excessive component heat and cause permanent damage to your hardware!

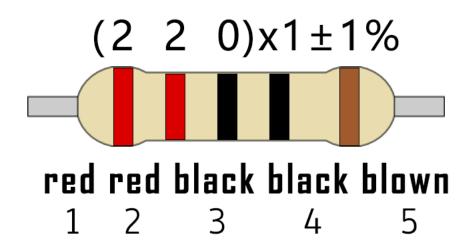


#### Note:

How to connect a LED



How to identify the 220 Five-band resistor



## 5.4.4 4.Test Code

```
/*
* Filename : External LED flashing
* Description : Make an led blinking.
* Auther : http//www.keyestudio.com
*/
#define PIN_LED 15 //define the led pin
// the setup function runs once when you press reset or power the board
void setup() {
 // initialize digital pin LED as an output.
 pinMode(PIN_LED, OUTPUT);
}
// the loop function runs over and over again forever
void loop() {
 digitalWrite(PIN_LED, HIGH); // turn the LED on (HIGH is the voltage level)
 delay(500);
                           // wait for 0.5s
 digitalWrite(PIN_LED, LOW);
                         // turn the LED off by making the voltage LOW
                           // wait for 0.5s
 delay(500);
}
```

Before uploading the code to ESP32, please check the configuration of Arduino IDE.

Click "Tools" to confirm the board type and port, as shown below:

Project_03_LED_F	Flashing   Arduino 1.8.16		>	×
File Edit Sketch To	ols Help			
Project_03_LE	Auto Format Archive Sketch	Ctrl+T	Q.	2
*/ #define PIN_L	Fix Encoding & Reload Manage Libraries Serial Monitor	Ctrl+Shift+I Ctrl+Shift+M		^
<pre>// the setup void setup() // initiali pinMode(PIN</pre>	Serial Plotter WiFi101 / WiFiNINA Firmware Update	Ctrl+Shift+L r	r the board	
<pre>} // the loop f void loop() {   digitalWrit   delay(500);   digitalWrit   delay(500); } //**********************************</pre>	Board: "ESP32 Wrover Module" Upload Speed: "921600" Flash Frequency: "80MHz" Flash Mode: "QIO" Partition Scheme: "Default" Core Debug Level: "None" Port: "COM3"	> > > > > > > > > > > > > > > > > > > >	the voltage level) ng the voltage LOW **********	<
Invalid librar Invalid librar < 7	Get Board Info Programmer Burn Bootloader	>	.no\hardware\espressif\esp32\. .no\hardware\espressif\esp32\. fault, QIO, 80MHz, 921600, None on COM:	1 ~
		- wiover wodure, De	1444, 410, 8010112, 821000, None on CONA	-

Click to download the code to ESP32.

Project_03_LED_Flashing   Arduino 1.8.16	_		$\times$
<u>Eile Edit Sketch Tools H</u> elp			
			Ø
Project_03_LED_Flashing			
#define PIN_LED 15 //define the led pin			^
<pre>// the setup function runs once when you press reset or power the board void setup() {</pre>			
<pre>// initialize digital pin LED as an output.     pinMode(PIN_LED, OUTPUT); }</pre>			
<pre>// the loop function runs over and over again forever void loop() { digitalWrite(PIN_LED, HIGH); // turn the LED on (HIGH is the voltage let delay(500); // wait for 0.5s digitalWrite(PIN_LED, LOW); // turn the LED off by making the voltage is delay(500); // wait for 0.5s</pre>			
} //***********************************	****	*****	* ¥
Uploading			
Hash of data verified. Compressed 229264 bytes to 107533 Writing at 0x00024000 (85 %)			^ ~
7 ESP32 Wrover Module, Default, QIO, 80MHz, 921	600, No	one on C	омз

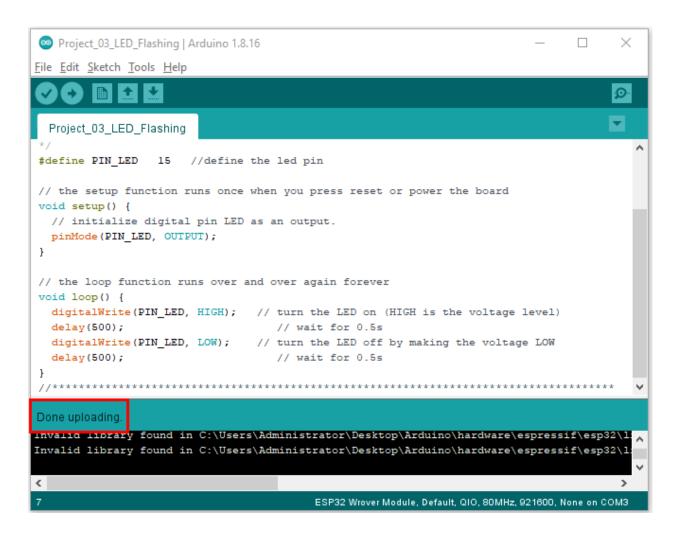
Note: If uploading the code fails, you can press the Boot button on ESP32 after click and release the Boot



after the percentage of

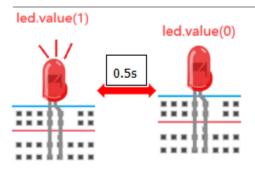
uploading progress appears, as shown below:

The code is uploaded successfully



# 5.4.5 5.Test Result

After uploading the code successfully, power up with a USB cable and the LED will start flashing.



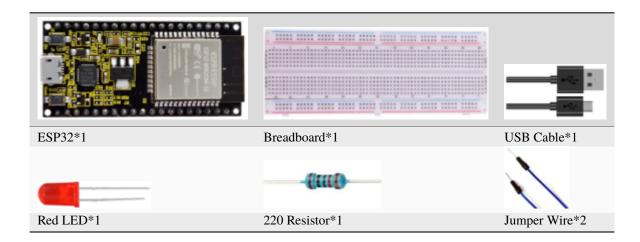
# 5.5 Project 04: Breathing Led

### 5.5.1 1.Introduction

In previous studies, we know that LEDs have on/off state, so how to enter the intermediate state? How to output an intermediate state to make the LED half bright? That's what we're going to learn.

Breathing light, that is, LED is turned from off to on gradually, and gradually from on to off, just like "breathing". However, how to control the brightness of a LED? We will use ESP32's PWM to achieve this target.

### 5.5.2 2.Components

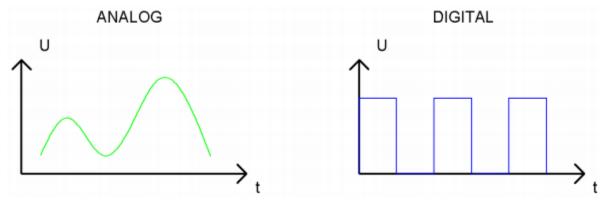


### 5.5.3 3.Component Knowledge



#### **Analog & Digital**

An Analog Signal is a continuous signal in both time and value. On the contrary, a digital signal or discrete time signal is a time series consisting of a sequence of quantities. Most signals in life are analog signals. A familiar example of an analog signal would be how the temperature throughout the day continuously changes and could not change instantaneously from  $0^{\circ}$ C to  $10^{\circ}$ C. However, digital signals can instantaneously change in value. This change is expressed in numbers as 1 and 0 (the basis of binary code). Their differences can be seen more easily when compared, as shown below:



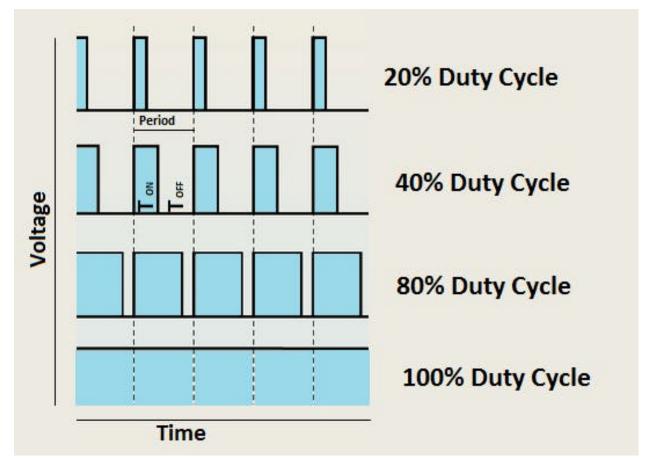
In practical application, we often use binary as the digital signal, that is a series of 0's and 1's. Since a binary signal only has two values (0 or 1), it has great stability and reliability. Lastly, both analog and digital signals can be converted into each other.

#### PWM

PWM, Pulse-Width Modulation, is a very effective method for using digital signals to control analog circuits. Common processors cannot directly output analog signals. PWM technology makes it very convenient to achieve this conversion (translation of digital to analog signals).

PWM technology uses digital pins to send certain frequencies of square waves, that is, the output of high levels and low levels, which alternately last for a while. The total time for each set of high levels and low levels is generally fixed, which is called the period (Note: the reciprocal of the period is frequency). The time of high level outputs is generally called "pulse width", and the duty cycle is the percentage of the ratio of pulse duration, or pulse width (PW) to the total period (T) of the waveform.

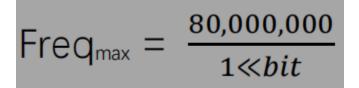
The longer the output of high levels last, the longer the duty cycle and the higher the corresponding voltage in the analog signal will be. The following figures show how the analog signal voltages vary between 0V-3.3V(high level is 3.3V) corresponding to the pulse width 0%-100%:



The longer the PWM duty cycle is, the higher the output power will be. Therefore, we can use PWM to control the brightness of an LED or the speed of DC motor and so on. It is evident from the above that PWM is not real analog, and the effective value of the voltage is equivalent to the corresponding analog. Then we can control the output power of the LED and other output modules to achieve different effects.

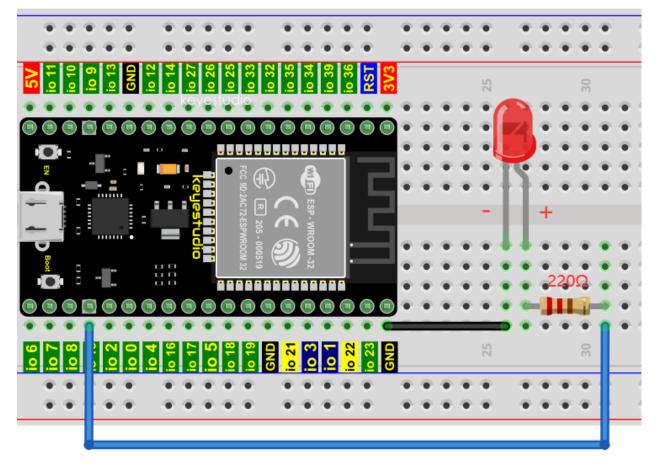
#### ESP32 and PWM:

On ESP32, the LEDC(PWM) controller has 16 separate channels, each of which can independently control frequency, duty cycle, and even accuracy. Unlike traditional PWM pins, the PWM output pins of ESP32 are configurable, with one or more PWM output pins per channel. The relationship between the maximum frequency and bit precision is shown in the following formula, where the maximum value of bit is 31.



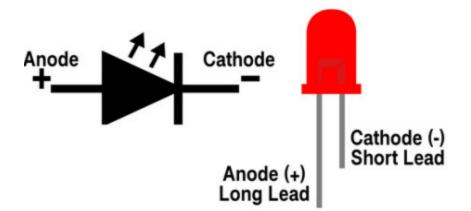
For example, generate a PWM with an 8-bit precision (28=256. Values range from 0 to 255) with a maximum frequency of 80,000,000/255 = 312,500Hz.

### 5.5.4 4.Wiring Diagram

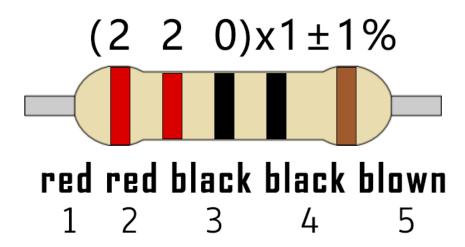


#### Note:

How to connect a LED



How to identify the 220 Five-band resistor



### 5.5.5 5.Test Code

The design of this project makes the GPIO15 output PWM, and the pulse width gradually increases from 0% to 100%, and then gradually decreases from 100% to 0%.

```
/*
* Filename : Breathing Led
* Description : Make led light fade in and out, just like breathing.
* Auther : http//www.keyestudio.com
*/
#define PIN_LED 15 //define the led pin
#define CHN
              0 //define the pwm channel
#define FRQ
              1000 //define the pwm frequency
#define PWM_BIT 8
                   //define the pwm precision
void setup() {
 ledcSetup(CHN, FRQ, PWM_BIT); //setup pwm channel
 ledcAttachPin(PIN_LED, CHN); //attach the led pin to pwm channel
}
void loop() {
```

```
for (int i = 0; i < 255; i++) { //make light fade in
    ledcWrite(CHN, i);
    delay(10);
    for (int i = 255; i > -1; i--) { //make light fade out
    ledcWrite(CHN, i);
    delay(10);
    }
}
```

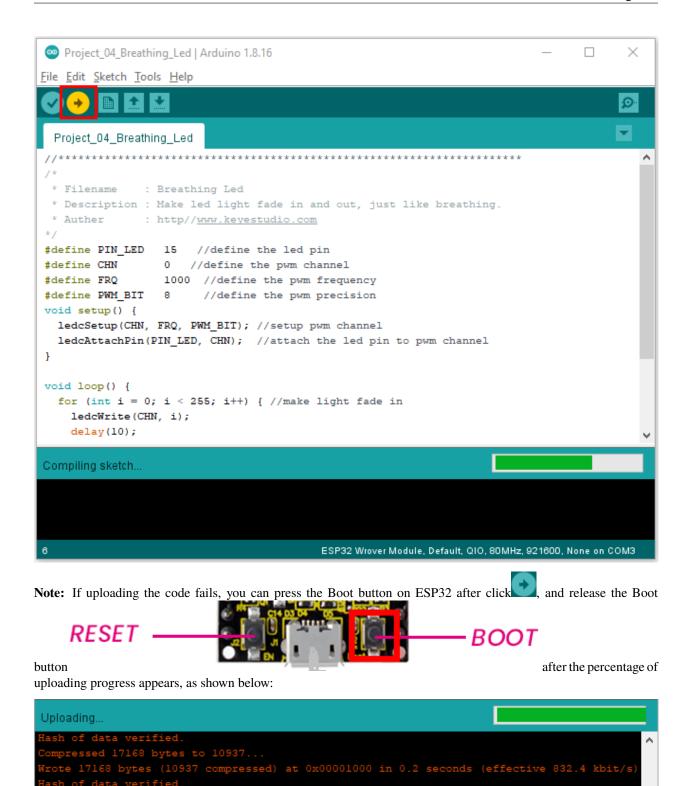
Before uploading Project Code to ESP32, please check the configuration of Arduino IDE.

Click "**Tools**" to confirm the board type and port as shown below:

🤍 Project_04_Br	eathing_Led   Arduino 1.8.16			—		×
File Edit Sketch	Tools Help					
	Auto Format	Ctrl+T				Ø
	Archive Sketch					_
Project_04_Bre	Fix Encoding & Reload					
//********	Manage Libraries	Ctrl+Shift+I	****			^
/* * Filename	Serial Monitor	Ctrl+Shift+M				
* Descriptio	Serial Plotter	Ctrl+Shift+L	reathing.			
* Auther */	WiFi101 / WiFiNINA Firmware Updater					
#define PIN_L						
#define CHN	Board: "ESP32 Wrover Module"	>				
#define FRQ	Upload Speed: "921600"	>				
<pre>#define PWM_B void setup()</pre>	Flash Frequency: "80MHz"	>				
ledcSetup(C	Flash Mode: "QIO"	>				
ledcAttachP	Partition Scheme: "Default"	>	channel			
}	Core Debug Level: "None"	>				
<pre>void loop() {</pre>	Port: "COM3"	>				
for (int i ledcWrite	Get Board Info					
delay(10)	Programmer	>				~
	Burn Bootloader					
6	ESP32	Wrover Module, De	fault, QIO, 80MHz, 9	21600, N	lone on (	сомз

Click to download the code to ESP32.

(continued from previous page)



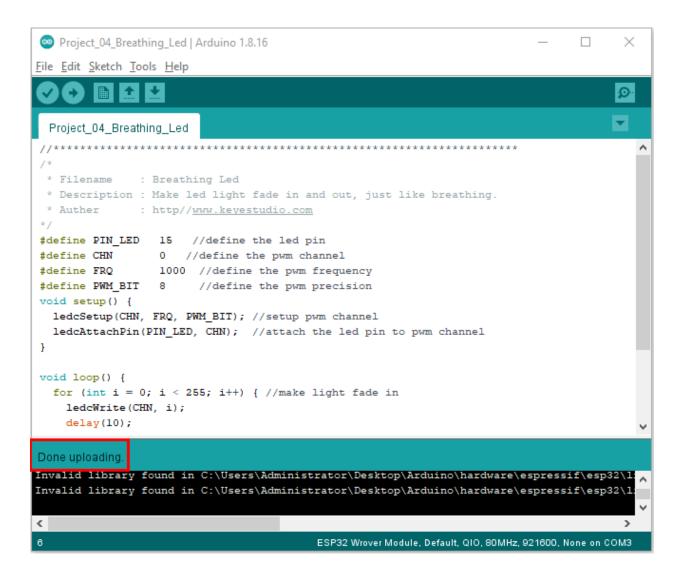
The code is uploaded successfully!

₹

32464 byt

х

ESP32 Wrover Module, Default, QIO, 80MHz, 921600, None on COM3



### 5.5.6 6.Test Result

After uploading the code successfully, power up with a USB cable and the LED is turned from ON to OFF and then back from OFF to ON gradually like breathing.

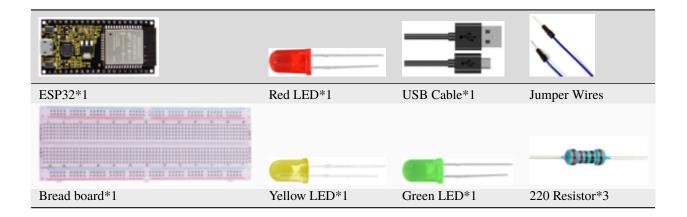


# 5.6 Project 05Traffic Lights

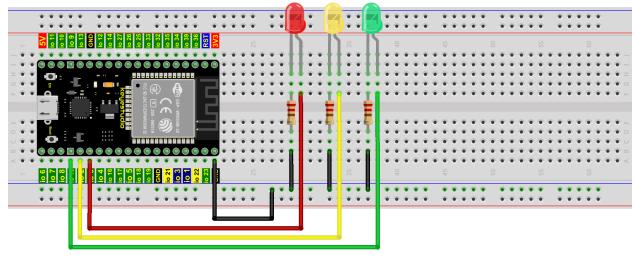
### 5.6.1 1.Introduction

Traffic lights are closely related to people's daily lives, which generally show red, yellow, and green. Everyone should obey the traffic rules, which can avoid many traffic accidents. In this project, we will use ESP32 and some LEDs (red, green and yellow) to simulate the traffic lights.

### 5.6.2 2.Components



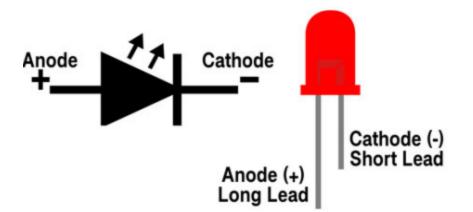
# 5.6.3 3.Wiring Diagram



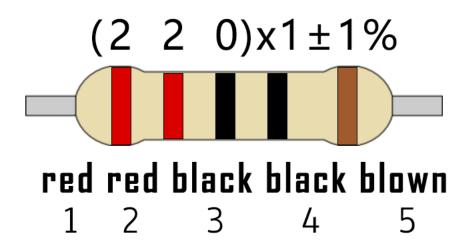
fritzing

Note:

How to connect a LED



How to identify the 220 Five-band resistor



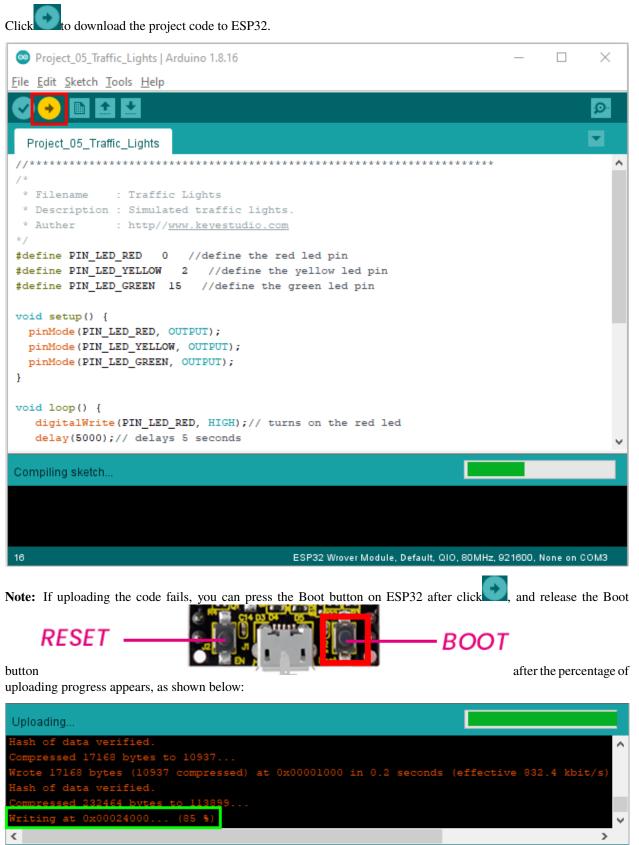
#### 5.6.4 4.Test Code

```
/*
* Filename : Traffic Lights
* Description : Simulated traffic lights.
* Auther : http//www.keyestudio.com
*/
#define PIN_LED_RED 0 //define the red led pin
#define PIN_LED_YELLOW 2 //define the yellow led pin
#define PIN_LED_GREEN 15 //define the green led pin
void setup() {
 pinMode(PIN_LED_RED, OUTPUT);
 pinMode(PIN_LED_YELLOW, OUTPUT);
 pinMode(PIN_LED_GREEN, OUTPUT);
}
void loop() {
  digitalWrite(PIN_LED_GREEN, HIGH);// turns on the green led
```

Before uploading the code to ESP32, please check the configuration of Arduino IDE.

Click "Tools" to confirm the board type and port as shown below:

Project_05_Traffic_	Lights   Arduino 1.8.16			_		×
File Edit Sketch Tool	ls Help		-			
00 1	Auto Format	Ctrl+T				Ø
	Archive Sketch					_
Project_05_Tra	Fix Encoding & Reload					
//*******	Manage Libraries	Ctrl+Shift+I	*****			^
/* * Filename	Serial Monitor	Ctrl+Shift+M				
* Descriptio	Serial Plotter	Ctrl+Shift+L				
* Auther	WEI101 (MEENINIA Einsteinen Undeten					
*/ #define PIN L	WiFi101 / WiFiNINA Firmware Updater					
#define PIN_L	Board: "ESP32 Wrover Module"	>				
<pre>#define PIN_L</pre>	Upload Speed: "921600"	>				
void setup()	Flash Frequency: "80MHz"	>				
pinMode(PIN	Flash Mode: "QIO"	>				
pinMode(PIN pinMode(PIN	Partition Scheme: "Default"	>				
}	Core Debug Level: "None"	>				
	Port: "COM3"	>				
<pre>void loop() {     digitalWri</pre>	Get Board Info		1 <b>4</b>			
delay(5000	Drogrammer	>				~
	Programmer Burn Bootloader					
Invalid library found in C:\Users\Administrator\Desktop\Arduino\hardware\espressif\esp32\l Invalid library found in C:\Users\Administrator\Desktop\Arduino\hardware\espressif\esp32\l						
						~
<						>
16	ESP32	Wrover Module, De	fault, QIO, 80MHz, 9	21600, N	lone on (	сомз



ESP32 Wrover Module, Default, QIO, 80MHz, 921600, None on COM3

#### The code is uploaded successfully

Project_05_Traffic_Lights   Arduino 1.8.16	_		$\times$
<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp			
			ø
Project_05_Traffic_Lights			
//*************************************			^
/*			
* Filename : Traffic Lights * Description : Simulated traffic lights.			
* Auther : http://www.keyestudio.com			
*/			
<pre>#define PIN_LED_RED 0 //define the red led pin</pre>			
#define PIN_LED_YELLOW 2 //define the yellow led pin			
<pre>#define PIN_LED_GREEN 15 //define the green led pin</pre>			
<pre>void setup() {     pinMode(PIN_LED_RED, OUTPUT);     pinMode(PIN_LED_YELLOW, OUTPUT);     pinMode(PIN_LED_GREEN, OUTPUT); }</pre>			
<pre>void loop() {     digitalWrite(PIN_LED_RED, HIGH);// turns on the red led     delay(5000);// delays 5 seconds</pre>			v
Done uploading.			
<pre>invalu library found in C:\Users\Administrator\Desktop\Arduino\hardware\e Invalid library found in C:\Users\Administrator\Desktop\Arduino\hardware\e</pre>			1.1
			$\sim$
<			>
16 ESP32 Wrover Module, Default, QIO, 80MHz, 9	921600, N	lone on (	сомз

### 5.6.5 5.Test Result

After uploading the code successfully, power up with a USB cable and what you'll see are below:

First, the green light will be on for five seconds then off;

Next, the yellow light blinks three times then goes off;

Then, the red light goes on for five seconds then goes off;

Repeat step 1 to 3 above.

# 5.7 Project 06: RGB LED

### 5.7.1 1.Introduction



RGB is composed of three colors (red, green and blue), which can emit different colors by mixing these three colors.

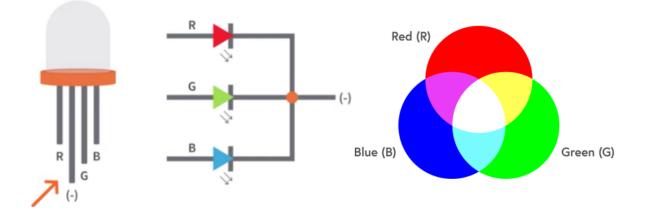
In this project, we will introduce the RGB and show you how to use ESP32 to control the RGB to emit different color lights .RGB is pretty basic, but it's also a great way to learn the fundamentals of electronics and coding.

### 5.7.2 2.Components

ESP32*1	RGB LED	Jumper Wires
	-(1111)-	-
Breadboard*1	220 Resistor*3	USB Cable*1

## 5.7.3 3.Component Knowledge

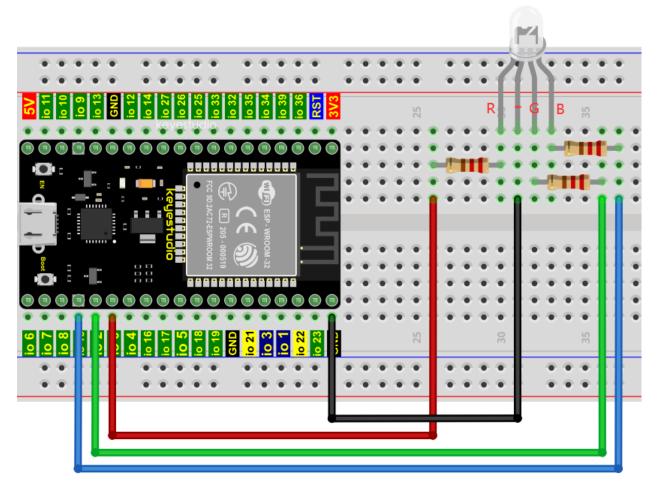
Most monitors adopt the RGB color standard, and all colors on a computer screen are a mixture of red, green and blue in varying proportions.



This RGB LED has 4 pins, with each color (red, green, blue) and a common cathode. To change its brightness, we can use the PWM of the ESP32 pins, which can give different duty cycle signals to the RGB to produce different colors of light.

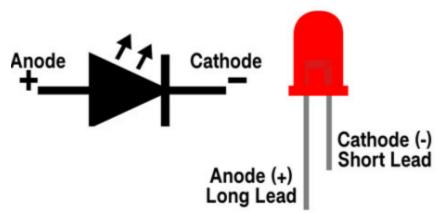
If we use three 10-bit PWM to control the RGB, in theory, we can create 2.10\*210\*210=1,073,741,824(1billion) colors through different combinations.

# 5.7.4 4.Wiring Diagram

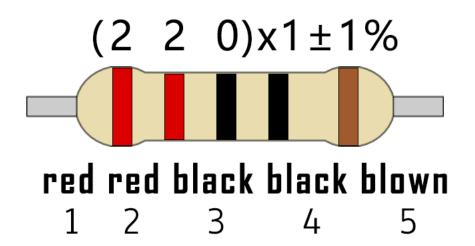


#### Note:

How to connect a LED



How to identify the 220 Five-band resistor



## 5.7.5 5.Test Code

```
/*
* Filename : RGB LED
* Description : Use RGBLED to show random color.
* Auther : http//www.keyestudio.com
*/
int ledPins[] = {0, 2, 15}; //define red, green, blue led pins
const byte chns[] = {0, 1, 2}; //define the pwm channels
int red, green, blue;
void setup() {
 for (int i = 0; i < 3; i++) { //setup the pwm channels,1KHz,8bit</pre>
   ledcSetup(chns[i], 1000, 8);
   ledcAttachPin(ledPins[i], chns[i]);
 }
}
void loop() {
 red = random(0, 256);
 green = random((0, 256);
 blue = random(0, 256);
 setColor(red, green, blue);
 delay(200);
}
void setColor(byte r, byte g, byte b) {
 ledcWrite(chns[0], 255 - r); //Common anode LED, low level to turn on the led.
 ledcWrite(chns[1], 255 - g);
 ledcWrite(chns[2], 255 - b);
}
```

## 5.7.6 6.Test Result

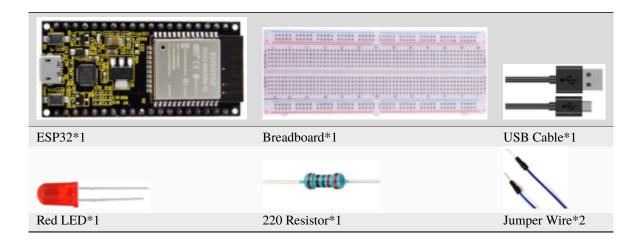
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the RGB LED starts to display random colors.

# 5.8 Project 07: Flowing Water Light

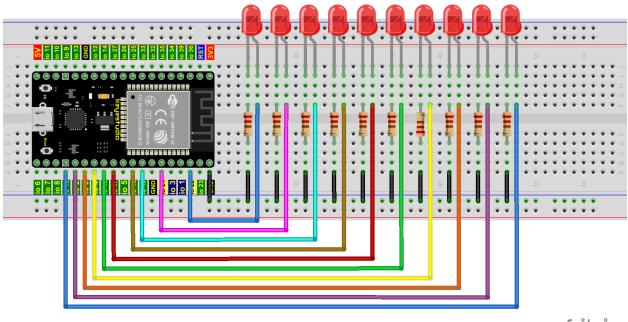
### 5.8.1 1.Introduction

In our daily life, we can see many billboards composed of different colors of LED. They constantly change the light (like water) to attract customers' attention. In this project, we will use ESP32 to control 10 LEDs to achieve the effect of flowing water.

### 5.8.2 2.Components



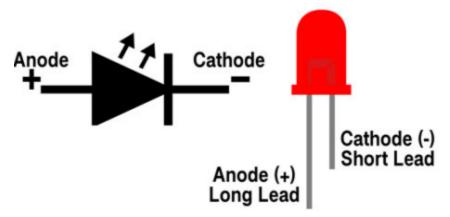
# 5.8.3 3.Wiring Diagram



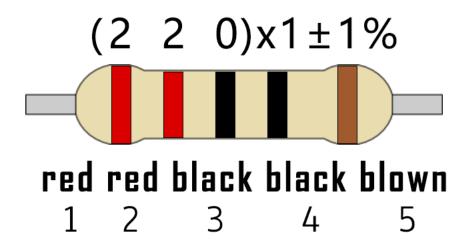
```
fritzing
```

Note:

How to connect a LED



How to identify the 220 Five-band resistor



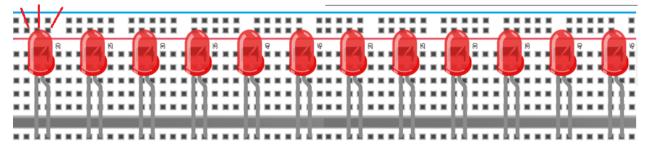
### 5.8.4 4.Test Code

This project is designed to make a flowing water lamp. Actions: First turn LED #1 ON, then turn it OFF. Then turn LED #2 ON, and then turn it OFF... and repeat the same to all 10 LEDs until the last LED is turns OFF. This process is repeated to achieve the "movements" of flowing water.

```
/*
* Filename
            : Flowing Water Light
* Description : Using ten leds to demonstrate flowing lamp.
* Auther
           : http//www.keyestudio.com
*/
byte ledPins[] = {22, 21, 19, 18, 17, 16, 4, 0, 2, 15};
int ledCounts;
void setup() {
 ledCounts = sizeof(ledPins);
 for (int i = 0; i < ledCounts; i++) {
   pinMode(ledPins[i], OUTPUT);
 }
}
void loop() {
 for (int i = 0; i < ledCounts; i++) {
   digitalWrite(ledPins[i], HIGH);
   delay(100);
   digitalWrite(ledPins[i], LOW);
 }
 for (int i = ledCounts - 1; i > -1; i - -) {
   digitalWrite(ledPins[i], HIGH);
   delay(100);
   digitalWrite(ledPins[i], LOW);
 }
}
//*******
              *******
```

## 5.8.5 5.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that 10 LEDs will light up from left to right and then back from right to left.



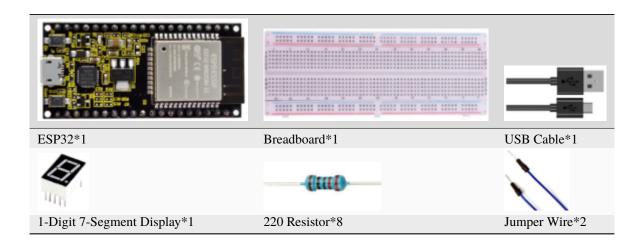
# 5.9 Project 081-Digit Digital Tube

### 5.9.1 1.Introduction

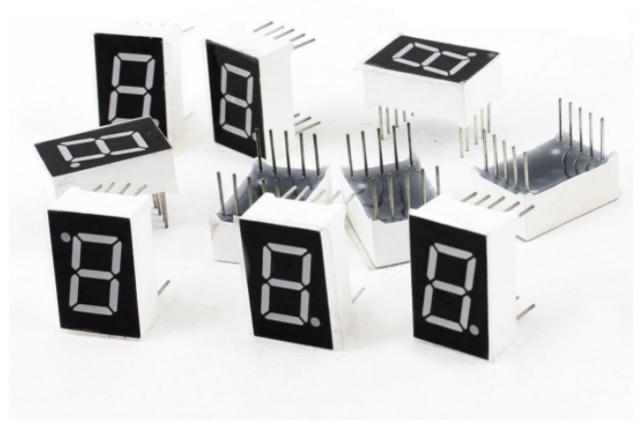
A 1-Digit 7-Segment Display is an electronic display device that displays decimal numbers. It is widely used in digital clocks, electronic meters, basic calculators and other electronic devices that display digital information.

Though they may not look modern enough, they are an alternative to more complex dot matrix displays and are easy to use in limited light conditions and strong sunlight. In this project, we will use ESP32 to control 1-Digit 7-segment display displays numbers.

### 5.9.2 2.Components



#### 5.9.3 3.Component Knowledge

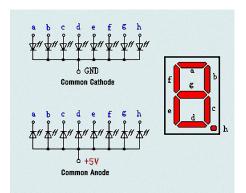


#### 1-Digit 7-Segment Display principle:

Digital tube display is a semiconductor light emitting device, its basic unit is a light-emitting diode (LED). The digital tube display can be divided into 7-segment display and 8-segment display according to the number of segments. The 8-segment display has one more LED unit than the 7-segment display (used for decimal point display). Each segment of the 7-segment display is a separate LED. According to the connection mode of the LED unit, the digital tube can be divided into a common anode digital tube and a common cathode digital tube.

In the common cathode 7-segment display, all the cathodes (or negative electrodes) of the segmented LEDs are connected together, so you should connect the common cathode to GND. To light up a segmented LED, you can set its associated pin to "HIGH".

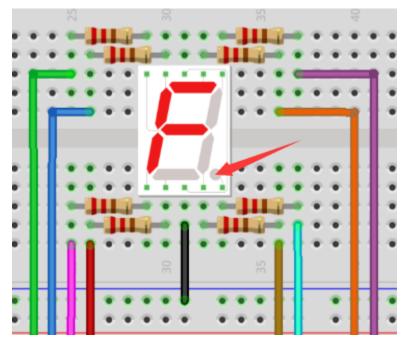
In the common anode 7-segment display, the LED anodes (positive electrodes) of all segments are connected together, so you should connect the common anode to"+5V". To light up a segmented LED, you can set its associated pin to"LOW".

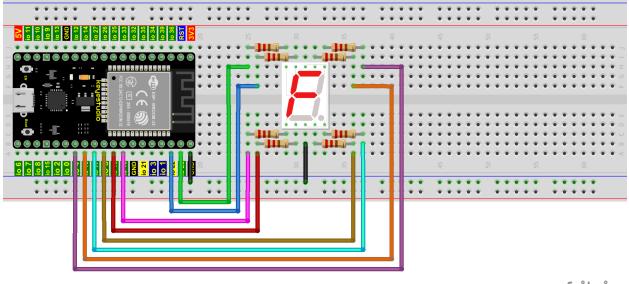


Each part of the digital tube is composed of an LED. So when you use it, you also need to use a current limiting resistor. Otherwise, the LED will be damaged. In this experiment, we use an ordinary common cathode one-digit digital tube. As we mentioned above, you should connect the common cathode to GND. To light up a segmented LED, you can set its associated pin to "HIGH".

# 5.9.4 4.Wiring Diagram

Note: The direction of the 7-segment display inserted into the breadboard is consistent with the wiring diagram, with one more point in the lower right corner.





fritzing

### 5.9.5 5.Test Code

The digital display is divided into 7 segments, and the decimal point display is divided into 1 segment. When certain numbers are displayed, the corresponding segment will be lit. For example, when the number 1 is displayed, segments b and c will be turned on.

```
/*
* Filename
            : 1-Digit Digital Tube
* Description : One Digit Tube displays numbers from 9 to 0.
* Auther
           : http//www.keyestudio.com
*/
// sets the IO PIN for every segment
int a=16; // digital PIN 16 for segment a
int b=4; // digital PIN 4 for segment b
int c=5; // digital PIN 5 for segment c
int d=18; // digital PIN 18 for segment d
int e=19; // digital PIN 19 for segment e
int f=22; // digital PIN 22 for segment f
int g=23; // digital PIN 23 for segment g
int dp=17; // digital PIN 17 for segment dp
void digital_0(void) // displays number 0
{
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,HIGH);
digitalWrite(f,HIGH);
digitalWrite(g,LOW);
digitalWrite(dp,LOW);
}
void digital_1(void) // displays number 1
```

```
{
digitalWrite(a,LOW);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,LOW);
digitalWrite(e,LOW);
digitalWrite(f,LOW);
digitalWrite(g,LOW);
digitalWrite(dp,LOW);
}
void digital_2(void) // displays number 2
{
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,LOW);
digitalWrite(d,HIGH);
digitalWrite(e,HIGH);
digitalWrite(f,LOW);
digitalWrite(g,HIGH);
digitalWrite(dp,LOW);
}
void digital_3(void) // displays number 3
{
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(f,LOW);
digitalWrite(e,LOW);
digitalWrite(dp,LOW);
digitalWrite(g,HIGH);
}
void digital_4(void) // displays number 4
{
digitalWrite(a,LOW);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,LOW);
digitalWrite(e,LOW);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);
digitalWrite(dp,LOW);
}
void digital_5(void) // displays number 5
{
digitalWrite(a,HIGH);
digitalWrite(b,LOW);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,LOW);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);
```

```
digitalWrite(dp,LOW);
}
void digital_6(void) // displays number 6
{
digitalWrite(a,HIGH);
digitalWrite(b,LOW);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,HIGH);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);
digitalWrite(dp,LOW);
}
void digital_7(void) // displays number 7
{
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,LOW);
digitalWrite(e,LOW);
digitalWrite(f,LOW);
digitalWrite(g,LOW);
digitalWrite(dp,LOW);
}
void digital_8(void) // displays number 8
{
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,HIGH);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);
digitalWrite(dp,LOW);
}
void digital_9(void) // displays number 9
{
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,LOW);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);
digitalWrite(dp,LOW);
}
void setup()
{
  // initialize digital pin LED as an output.
  pinMode(a, OUTPUT);
  pinMode(b, OUTPUT);
  pinMode(c, OUTPUT);
```

```
pinMode(d, OUTPUT);
 pinMode(e, OUTPUT);
 pinMode(f, OUTPUT);
 pinMode(g, OUTPUT);
 pinMode(dp, OUTPUT);
}
void loop()
{
while(1)
{
digital_9();// displays number 9
delay(1000); // waits a sencond
digital_8();// displays number 8
delay(1000); // waits a sencond
digital_7();// displays number 7
delay(1000); // waits a sencond
digital_6();// displays number 6
delay(1000); // waits a sencond
digital_5();// displays number 5
delay(1000); // waits a sencond
digital_4();// displays number 4
delay(1000); // waits a sencond
digital_3();// displays number 3
delay(1000); // waits a sencond
digital_2();// displays number 2
delay(1000); // waits a sencond
digital_1();// displays number 1
delay(1000);// waits a sencond
digital_0();// displays number 0
delay(1000);// waits a sencond
}}
```

### 5.9.6 6.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the 1-Digit 7-Segment Display will display numbers from 9 to 0.

# 5.10 Project 094-Digit Digital Tube

### 5.10.1 1.Introduction

A 4-digit 7-segment display is a very practical display device and it is used for devices such as electronic clocks, score counters and the number of people in the park. Because of the low price, easy to use, more and more projects will use 4 Digit 7-segment display. In this project, we use ESP32 to control 4-digit 7-segment display to display digits.

## 5.10.2 2.Components

ESP32*1	Breadboard*1	USB Cable*1
EEEE	-((((())-	$\mathbf{X}$
4-digit 7-segment display Module*1	220 Resistor*8	Jumper Wire*2

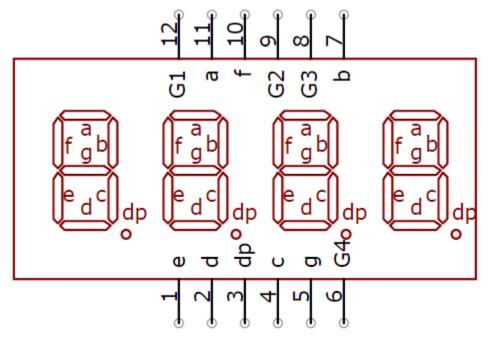
## 5.10.3 3.Component Knowledge



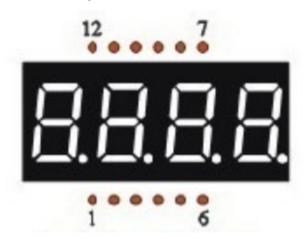
#### 4-digit 7-segment display

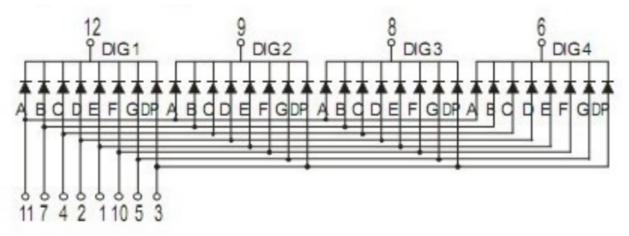
It is a device with common cathode and anode, its display principle is similar to the 1-Digit digital tube display. They all have eight GPIO ports to control the digital tube display, that is 8 leds. However, here is 4-digit, so it needs four GPIO ports to control the bit selection end. Our 4 - digit digital tube is common cathode.

The following figure shows the pin diagram of the 4-digit digital tube. G1, G2, G3 and G4 are the control pins.

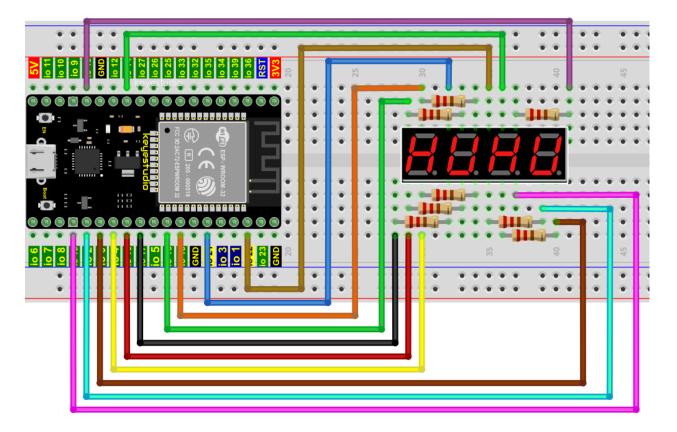


Schematic Diagram





# 5.10.4 4.Wiring Diagram



# 5.10.5 5.Test Code

//************************************	
<pre>/*  * Filename : 4-Digit Digital Tube  * Description : Four Digit Tube displays numbers from 0 to 9999.  * Auther : http//www.keyestudio.com</pre>	
*/	
<pre>#define d_a 18 //Define nixie tube a to pin 18 #define d_b 13 #define d_c 2 #define d_d 16 #define d_e 17 #define d_f 19 #define d_g 0 #define d_dp 4</pre>	
<pre>#define G1 21 //Define the first set of nixtube G1 to pin 21 #define G2 22 #define G3 14 #define G4 15</pre>	
//Nixie tube 0-F code value	
(continues on	next page

Chapter 5. Arduino Tutorial

(continued from previous page)

```
unsigned char num[17][8] =
{
//abcdefgdp
                                  //0
  \{1, 1, 1, 1, 1, 1, 1, 0, 0\},\
  \{0, 1, 1, 0, 0, 0, 0, 0\},\
                                  //1
                                   //2
  \{1, 1, 0, 1, 1, 0, 1, 0\},\
  \{1, 1, 1, 1, 1, 0, 0, 1, 0\},\
                                   //3
  \{0, 1, 1, 0, 0, 1, 1, 0\},\
                                  //4
  \{1, 0, 1, 1, 0, 1, 1, 0\},\
                                   //5
                                   //6
  \{1, 0, 1, 1, 1, 1, 1, 0\},\
                                   //7
  \{1, 1, 1, 0, 0, 0, 0, 0\},\
  \{1, 1, 1, 1, 1, 1, 1, 1, 0\},\
                                  //8
  \{1, 1, 1, 1, 1, 0, 1, 1, 0\},\
                                   //9
  \{1, 1, 1, 0, 1, 1, 1, 1\},\
                                   //A
  \{1, 1, 1, 1, 1, 1, 1, 1, 1\},\
                                  //B
  \{1, 0, 0, 1, 1, 1, 0, 1\},\
                                  //C
  \{1, 1, 1, 1, 1, 1, 1, 0, 1\},\
                                   //D
  \{1, 0, 0, 1, 1, 1, 1, 1\},\
                                   //E
  \{1, 0, 0, 0, 1, 1, 1, 1\},\
                                  //F
  \{0, 0, 0, 0, 0, 0, 0, 0, 1\},\
                                  //.
};
void setup()
{
  pinMode(d_a,OUTPUT);
                          //Set to output pin
  pinMode(d_b,OUTPUT);
  pinMode(d_c,OUTPUT);
  pinMode(d_d,OUTPUT);
  pinMode(d_e,OUTPUT);
  pinMode(d_f,OUTPUT);
  pinMode(d_g,OUTPUT);
  pinMode(d_dp,OUTPUT);
  pinMode(G1,OUTPUT);
  pinMode(G2,OUTPUT);
  pinMode(G3,OUTPUT);
  pinMode(G4,OUTPUT);
}
void loop()
{
  //Start counting from 0 and gradually increase by 1 to 9999, repeating.
  for(int l = 0;l < 10;l++ )</pre>
  {
    for(int k = 0; k < 10; k++)
    {
      for(int j = 0; j < 10; j++)
      {
        for(int i = 0;i < 10;i++)
        {
           //125 flashes a second equals one second.
```

```
(continued from previous page)
```

```
//1000/8=125
          for(int q = 0;q<125;q++)
          {
            Display(1,1);//The first nixie tube shows the value of L.
            delay(2);
            Display(2,k);
            delay(2);
            Display(3,j);
            delay(2);
            Display(4,i);
            delay(2);
          }
       }
      }
   }
 }
}
//Display functions: g ranges from 1 to 4,num ranges from 0 to 9
void Display(unsigned char g, unsigned char n)
{
  digitalWrite(d_a,LOW);
                              //Remove the light
  digitalWrite(d_b,LOW);
  digitalWrite(d_c,LOW);
  digitalWrite(d_d,LOW);
  digitalWrite(d_e,LOW);
  digitalWrite(d_f,LOW);
  digitalWrite(d_g,LOW);
  digitalWrite(d_dp,LOW);
  switch(g)
                     //Gate a choice
  {
   case 1:
      digitalWrite(G1,LOW);
                              //Choose the first digit
      digitalWrite(G2,HIGH);
      digitalWrite(G3,HIGH);
      digitalWrite(G4,HIGH);
      break:
   case 2:
      digitalWrite(G1,HIGH);
      digitalWrite(G2,LOW);
                              //Choose the second bit
      digitalWrite(G3,HIGH);
      digitalWrite(G4,HIGH);
      break:
   case 3:
      digitalWrite(G1,HIGH);
      digitalWrite(G2,HIGH);
                              //Choose the third bit
      digitalWrite(G3,LOW);
      digitalWrite(G4,HIGH);
      break;
    case 4:
```

(continued from previous page)

```
digitalWrite(G1,HIGH);
     digitalWrite(G2,HIGH);
     digitalWrite(G3,HIGH);
     digitalWrite(G4,LOW);
                         //Choose the fourth bit
     break:
   default:break;
 }
 digitalWrite(d_a,num[n][0]);
                               //a Queries the code value table
 digitalWrite(d_b,num[n][1]);
 digitalWrite(d_c,num[n][2]);
 digitalWrite(d_d,num[n][3]);
 digitalWrite(d_e,num[n][4]);
 digitalWrite(d_f,num[n][5]);
 digitalWrite(d_g,num[n][6]);
 digitalWrite(d_dp,num[n][7]);
}
```

#### 5.10.6 6.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the 4-digit 7-segment display displays 0-9999and repeat these actions in an infinite loop.

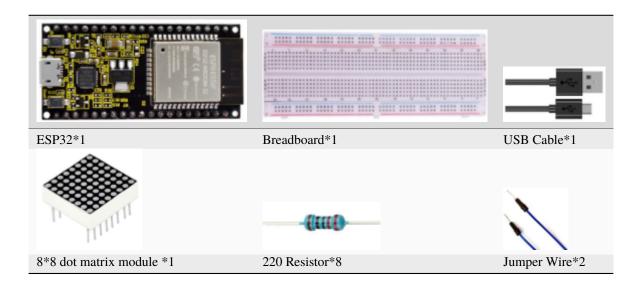
# 5.11 Project 108×8 Dot-matrix Display

### 5.11.1 1.Introduction

Dot matrix display is an electronic digital display device that can display information on machine, clocks, public transport departure indicators and many other devices.

In this project, we will use ESP32 to control 8x8 LED dot matrix to display digits.

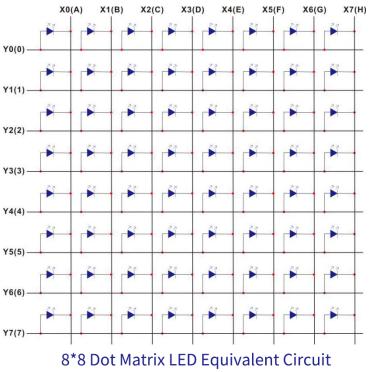
# 5.11.2 2.Components

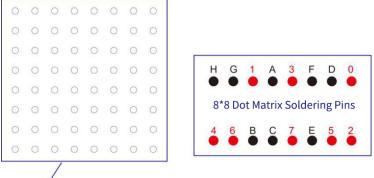


## 5.11.3 3.Component Knowledge

#### 8\*8 dot matrix module

The 8\*8 dot matrix is composed of 64 LEDs, including row common anode and row common cathode. Our module is row common anode, that is, each row has a line connecting the positive pole of the LED, and the column is connecting the negative pole of the LED lamp, as shown in the following figure :

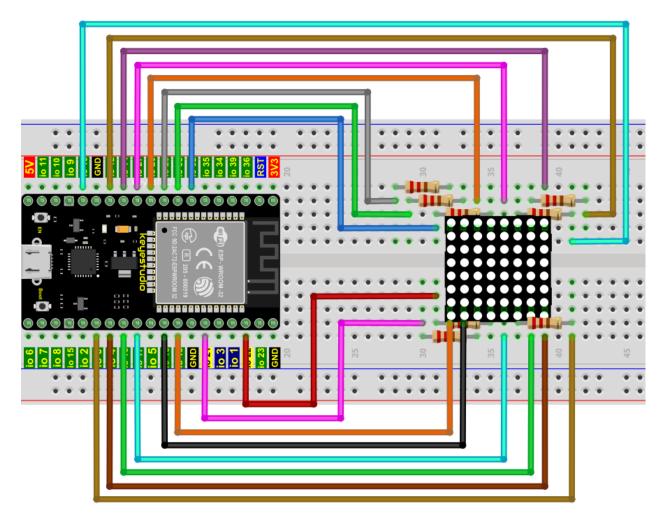




788BS Silk Print

8\*8 Dot Matrix Outlook and Pinouts

# 5.11.4 4.Wiring Diagram



#### 5.11.5 5.Test Code

```
/*
* Filename : 8×8 Dot-matrix Display
* Description : 8×8 Dot-matrix displays numbers from 0 to 9.
* Auther
               : http//www.keyestudio.com
*/
int R[] = {14,26,4,27,19,16,18,17};
int C[] = {32,21,22,12,0,13,33,25};
unsigned char data_0[8][8] =
{
\{0, 0, 1, 1, 1, 0, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 1, 1, 1, 0, 0, 0\}
};
unsigned char data_1[8][8] =
{
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 1, 1, 0, 0, 0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 1, 1, 1, 0, 0\}
};
unsigned char data_2[8][8] =
{
\{0, 0, 1, 1, 1, 0, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 1, 0, 0, 0, 0\},\
\{0, 0, 1, 0, 0, 0, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 0, 0, 0\}
};
unsigned char data_3[8][8] =
{
\{0, 0, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 1, 1, 1, 1, 0, 0\},\
```

(continued from previous page)

```
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0,0,0,0,0,0,1,0,0\},\
\{0, 0, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 0, 0, 0\}
};
unsigned char data_4[8][8] =
{
\{0, 1, 0, 0, 0, 0, 0, 0\},\
\{0, 1, 0, 0, 1, 0, 0, 0\},\
\{0, 1, 0, 0, 1, 0, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 1, 1, 0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0,0,0,0,1,0,0,0\},\
\{0, 0, 0, 0, 0, 0, 0, 0, 0\}
};
unsigned char data_5[8][8] =
{
\{0, 1, 0, 0, 0, 0, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 0, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0,0,0,0,0,0,0,0,0\}
};
unsigned char data_6[8][8] =
{
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 0, 0, 0\},\
\{0, 1, 0, 0, 0, 0, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 0, 0, 0\}
};
unsigned char data_7[8][8] =
{
\{0,0,0,0,0,0,0,0,0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 1, 0, 0, 0, 0\},\
\{0, 0, 1, 0, 0, 0, 0, 0\},\
\{0, 1, 0, 0, 0, 0, 0, 0\},\
\{0,0,0,0,0,0,0,0,0\}
```

};

(continued from previous page)

```
unsigned char data_8[8][8] =
{
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 0, 0, 0\}
};
unsigned char data_9[8][8] =
{
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0,0,0,0,0,0,0,0,0\}
};
void Display(unsigned char dat[8][8])
{
for(int c = 0; c<8;c++)
{
digitalWrite(C[c],LOW);
for(int r = 0;r<8;r++)
{
digitalWrite(R[r],dat[r][c]);
}
delay(1);
Clear();
}
}
void Clear()
{
for(int i = 0;i<8;i++)</pre>
{
digitalWrite(R[i],LOW);
digitalWrite(C[i],HIGH);
}
}
void setup(){
  for(int i = 0;i<8;i++)</pre>
  {
```

(continued from previous page)

```
pinMode(R[i],OUTPUT);
   pinMode(C[i],OUTPUT);
 }
}
void loop(){
 for (int i = 1; i <= 100; i = i + (1)) {</pre>
   Display(data_0);
 }
 for (int i = 1; i <= 100; i = i + (1)) {</pre>
   Display(data_1);
 3
 for (int i = 1; i <= 100; i = i + (1)) {</pre>
   Display(data_2);
 }
 for (int i = 1; i <= 100; i = i + (1)) {</pre>
   Display(data_3);
 }
 for (int i = 1; i <= 100; i = i + (1)) {</pre>
   Display(data_4);
 }
 for (int i = 1; i <= 100; i = i + (1)) {</pre>
   Display(data_5);
 }
 for (int i = 1; i <= 100; i = i + (1)) {</pre>
   Display(data_6);
 }
 for (int i = 1; i <= 100; i = i + (1)) {
   Display(data_7);
 }
 for (int i = 1; i <= 100; i = i + (1)) {</pre>
   Display(data_8);
 }
 for (int i = 1; i <= 100; i = i + (1)) {
   Display(data_9);
 }
}
```

### 5.11.6 6.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the 8\*8 dot matrix displays the numbers 0~9 respectively.

# 5.12 Project 1174HC595N Control 8 LEDs

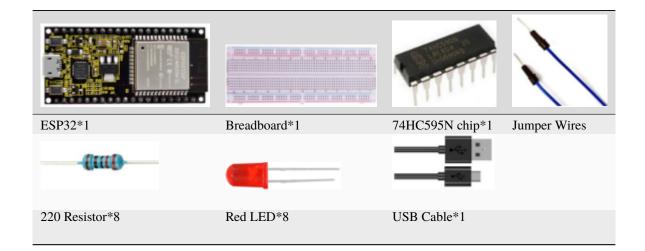
# 5.12.1 1.Introduction

In previous projects, we learned how to light up an LED.

With only 32 IO ports on ESP32, how do we light up a lot of leds? Sometimes it is possible to run out of pins on the ESP32, and you need to extend it with the shift register. You can use the 74HC595N chip to control 8 outputs at a time, taking up only a few pins on your microcontroller. In addition, you can also connect multiple registers together to further expand the output.

In this project, we will use an ESP32a 74HC595 chip and LEDs to make a flowing water light to understand the function of the 74HC595 chip.

# 5.12.2 2.Components

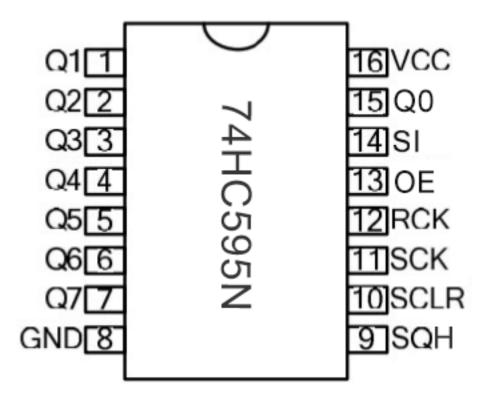


# 5.12.3 3.Component Knowledge



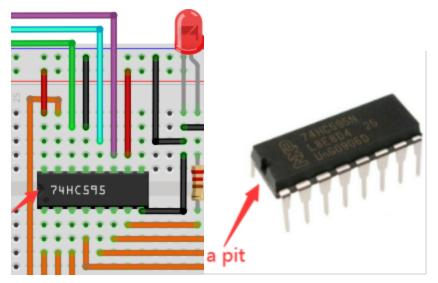
#### 74HC595N Chip:

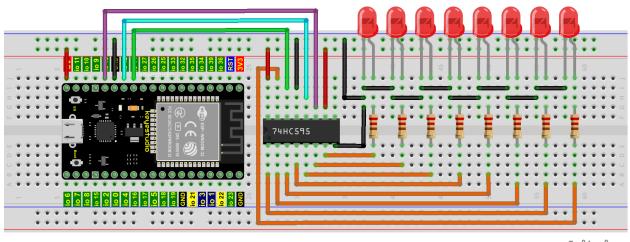
The 74HC595 chip is used to convert serial data into parallel data. A 74HC595 chip can convert the serial data of one byte into 8 bits, and send its corresponding level to each of the 8 ports correspondingly. With this characteristic, the 74HC595 chip can be used to expand the IO ports of an ESP32. At least 3 ports are required to control the 8 ports of the 74HC595 chip.



The ports of the 74HC595 chip are described as follows

Note: Note the orientation the 74HC595N chip is inserted.





fritzing

## 5.12.4 5.Test Code

```
/*
* Filename : 74HC595N Control 8 LEDs
* Description : Use 74HC575N to drive ten leds to display the flowing light.
           : http//www.keyestudio.com
* Auther
*/
int dataPin = 14; // Pin connected to DS of 74HC595(Pin14)
int latchPin = 12; // Pin connected to ST_CP of 74HC595(Pin12)
int clockPin = 13; // Pin connected to SH_CP of 74HC595(Pin11)
void setup() {
 // set pins to output
 pinMode(latchPin, OUTPUT);
 pinMode(clockPin, OUTPUT);
 pinMode(dataPin, OUTPUT);
}
void loop() {
 // Define a one-byte variable to use the 8 bits to represent the state of 8 LEDs of
\rightarrow LED bar graph.
// This variable is assigned to 0x01, that is binary 00000001, which indicates only
→one LED light on.
byte x = 0x01; // 0b 0000 0001
 for (int j = 0; j < 8; j++) { // Let led light up from right to left</pre>
   writeTo595(LSBFIRST, x);
   \mathbf{x} \ll 1; // make the variable move one bit to left once, then the bright LED move.
\rightarrow one step to the left once.
   delay(50);
 }
 delay(100);
                //0b 1000 0000
 x = 0x80;
 for (int j = 0; j < 8; j++) { // Let led light up from left to right
   writeTo595(LSBFIRST, x);
   x >>= 1;
```

(continued from previous page)

```
delay(50);
}
delay(100);

void writeTo595(int order, byte _data ) {
    // Output low level to latchPin
    digitalWrite(latchPin, LOW);
    // Send serial data to 74HC595
    shiftOut(dataPin, clockPin, order, _data);
    // Output high level to latchPin, and 74HC595 will update the data to the parallel_
    output port.
    digitalWrite(latchPin, HIGH);
}
```

### 5.12.5 6.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the 8 LEDs start flashing in flowing water mode.

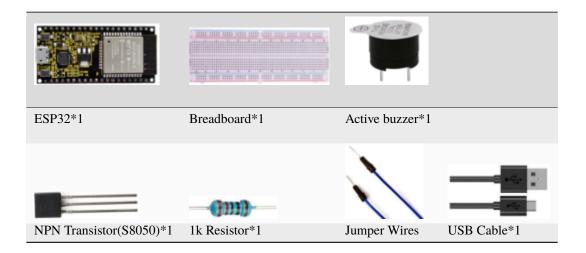
# 5.13 Project 12Active Buzzer

### 5.13.1 1.Introduction

Active buzzer is a sound component that is widely used as a sound component for computersprintersalarmselectronic toys and phonestimers etc. It has an internal vibration source, just by connecting to a 5V power supply, it can continuously buzz.

In this project, we will use ESP32 to control the active buzzer to beep.

# 5.13.2 2.Components



## 5.13.3 3.Component Knowledge



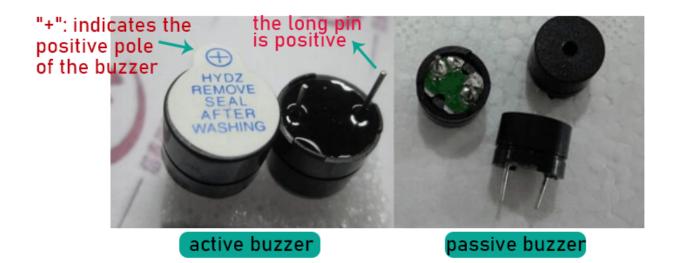
#### Active buzzer:

Active buzzer inside has a simple oscillator circuit, which can convert constant direct current into a certain frequency pulse signal. Once active buzzer receives a high level, it will produce sound.

Passive buzzer is an internal without vibration source integrated electronic buzzer, it must be driven by 2k to 5k square wave, rather than a DC signal.

The two buzzers are very similar in appearance, but one buzzer with a green circuit board is a passive buzzer, while the other buzzer with black tape is an active buzzer.

Passive buzzers don't have positive polarity, but active buzzers have. As shown below:

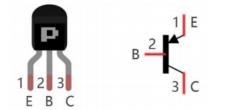


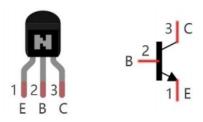
#### **Transistor:**



Because the buzzer requires such large current that GPIO of ESP32 output capability cannot meet the requirement, a transistor of NPN type is needed here to amplify the current.

Transistor, the full name: semiconductor transistor, is a semiconductor device that controls current. Transistorcan be used to amplify weak signal, or works as a switch. It has three electrodes(PINs): base (b), collector © and emitter (e). When there is current passing between "be", "ce" will allow several-fold current (transistor magnification) pass, at this point, transistor works in the amplifying area. When current between "be" exceeds a certain value, "ce" will not allow current to increase any longer, at this point, transistor works in the saturation area. Transistor has two types as shown below: PNP and NPN,





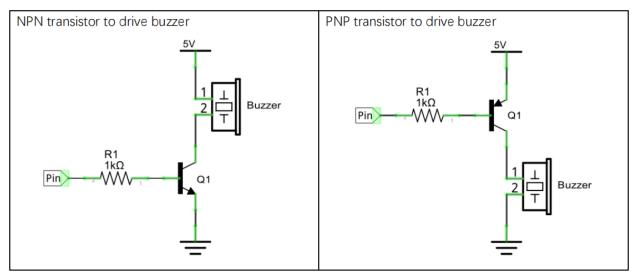
#### **PNP transistor NPN transistor**

In our kit, the PNP transistor is marked with 8550, and the NPN transistor is marked with 8050.

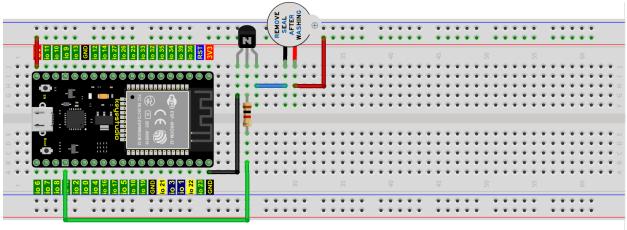
Based on the transistor's characteristics, it is often used as a switch in digital circuits. As micro-controller's capacity to output current is very weak, we will use transistor to amplify current and drive large-current components.

When using NPN transistor to drive buzzer, we often adopt the following method. If GPIO outputs high level, current will flow through R1, the transistor will get conducted, and the buzzer will sound. If GPIO outputs low level, no current flows through R1, the transistor will not be conducted, and buzzer will not sound.

When using PNP transistor to drive buzzer, we often adopt the following method. If GPIO outputs low level, current will flow through R1, the transistor will get conducted, and the buzzer will sound. If GPIO outputs high level, no current flows through R1, the transistor will not be conducted, and buzzer will not sound.



# 5.13.4 4.Wiring Diagram



### fritzing

Note: The buzzer power supply in this circuit is 5V. On a 3.3V power supply, the buzzer can work, but it will reduce the loudness.

### 5.13.5 5.Test Code

```
/*
* Filename : Active Buzzer
* Description : Active buzzer beeps.
* Auther : http//www.keyestudio.com
*/
#define buzzerPin 15 //define buzzer pins
void setup ()
{
 pinMode (buzzerPin, OUTPUT);
}
void loop ()
{
 digitalWrite (buzzerPin, HIGH);
 delay (500);
 digitalWrite (buzzerPin, LOW);
 delay (500);
}
```

# 5.13.6 6.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the active buzzer beeps.

# 5.14 Project 13Passive Buzzer

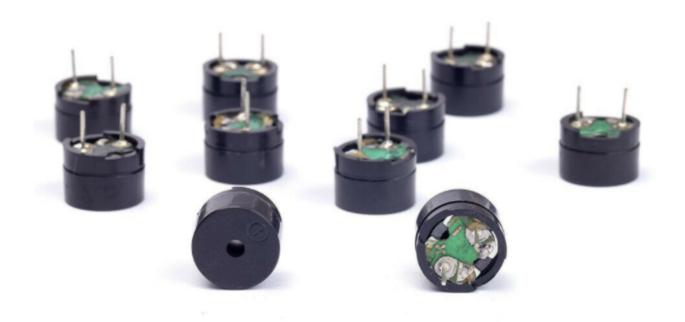
# 5.14.1 1.Introduction

In a previous project, we studied an active buzzer, which can only make a sound and may make you feel very monotonous. In this project, we will learn a passive buzzer and use the ESP32 control it to work. Unlike the active buzzer, the passive buzzer can emit sounds of different frequencies.

# 5.14.2 2.Components

ESP32*1	Breadboard*1	Passive Buzzer *1	
	-()11)	$\mathbf{X}$	
NPN Transistor(S8050)*1	1kResistor*1	Jumper Wires	USB Cable*1

# 5.14.3 3.Component Knowledge



#### **Passive buzzer:**

A passive buzzer is an integrated electronic buzzer with no internal vibration source and it has to be driven by 2K-5K square waves, not DC signals.

The two buzzers are very similar in appearance, but one buzzer with a green circuit board is a passive buzzer and the other buzzer with black tape is an active buzzer.

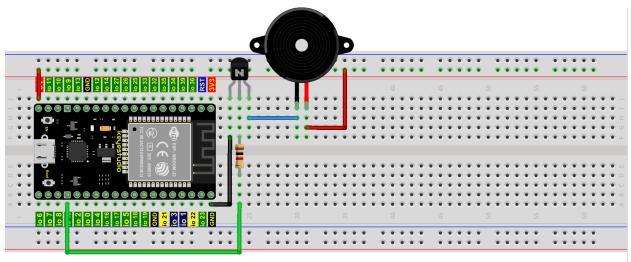
Passive buzzers cannot distinguish between positive polarity while active buzzers can.



#### **Transistor:**

Please refer to Project 12.

# 5.14.4 4.Wiring Diagram



# fritzing

### 5.14.5 5.Test Code

```
(continued from previous page)
// LEDC timer uses 13 bit accuracy
#define LEDC_TIMER_13_BIT 13
// Define tool I/O ports
#define BUZZER PIN 15
//Create a musical melody list, Super Mario
int melody[] = {330, 330, 330, 262, 330, 392, 196, 262, 196, 165, 220, 247, 233, 220,
→196, 330, 392, 440, 349, 392, 330, 262, 294, 247, 262, 196, 165, 220, 247, 233, 220, 1
→196, 330, 392,440, 349, 392, 330, 262, 294, 247, 392, 370, 330, 311, 330, 208, 220, .
\rightarrow 262, 220, 262,
294, 392, 370, 330, 311, 330, 523, 523, 523, 392, 370, 330, 311, 330, 208, 220, 262,220,
→262, 294, 311, 294, 262, 262, 262, 262, 262, 294, 330, 262, 220, 196, 262, 262,262, .
→262, 294, 330, 262, 262, 262, 262, 294, 330, 262, 220, 196};
//Create a list of tone durations
\leftrightarrow4,8,4,8,4,8,2,8,4,4,8,4,1,8,4,4,8,4,8,4,8,2};
void setup() {
pinMode(BUZZER_PIN, OUTPUT); // Set the buzzer to output mode
}
void loop() {
 int noteDuration; //Create a variable of noteDuration
 for (int i = 0; i < sizeof(noteDurations); ++i)</pre>
 {
     noteDuration = 800/noteDurations[i];
     ledcSetup(LEDC_CHANNEL_0, melody[i]*2, LEDC_TIMER_13_BIT);
     ledcAttachPin(BUZZER_PIN, LEDC_CHANNEL_0);
     ledcWrite(LEDC_CHANNEL_0, 50);
     delay(noteDuration * 1.30); //delay
 }
}
```

# 5.14.6 6.Test Result

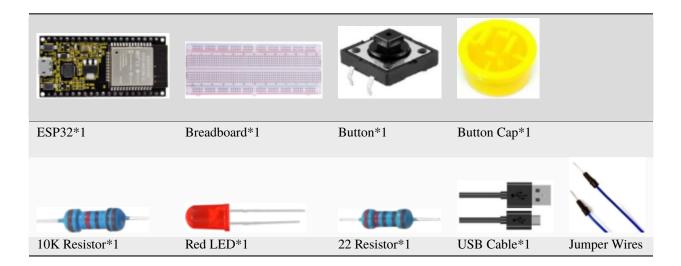
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the passive buzzer plays music.

# 5.15 Project 14: Mini Table Lamp

#### 5.15.1 1.Introduction

Do you know that the ESP32 can light up an LED when you press a button? In this project, we will use a ESP32a button switch and an LED to make a mini table lamp.

## 5.15.2 2.Components



### 5.15.3 3.Component Knowledge



#### **Button:**

A button can control the circuit on and off, the button is plugged into a circuit, the circuit is disconnected when the button is not pressed. The circuit works when you press the button, but breaks again when you release it.

Why does it only work when you press it? It starts from the internal structure of the button, which don't allow current to travel from one end of the button to the other before it is pressed; When pressed, a metal strip inside the button connects the two sides to allow electricity to pass through.

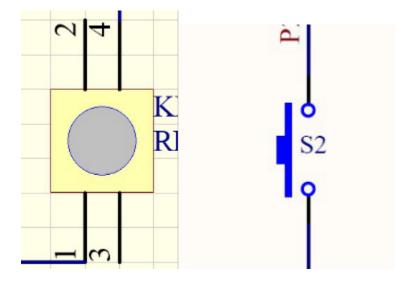
The internal structure of the button is shown in the figure



Before the button is pressed, 1 and 2 are on, 3 and 4 are also on, but 1, 3 or 1, 4 or 2, 3 or 2, 4 are off (not working). Only when the button is pressed, 1, 3 or 1, 4 or 2, 3 or 2, 4 are on.

The button switch is one of the most commonly used components in circuit design.

#### Schematic diagram of the button:

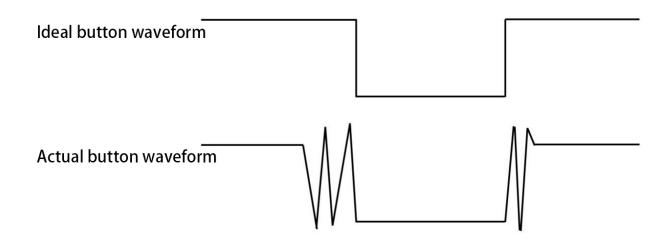


#### What is button [shake](javascript:;)?

We think of the switch circuit as "press the button and turn it on immediately", "press it again and turn it off immediately". In fact, this is not the case.

The button usually uses a mechanical elastic switch, and the mechanical elastic switch will produce a series of [shake](javascript:;) due to the elastic action at the moment when the mechanical contact is opened and closed (usually about 10ms).

As a result, the button switch will not immediately and stably turn on the circuit when it is closed, and it will not be completely and instantaneously disconnected when it is turned off.



#### How to eliminate the [shake](javascript:;)?

There are two common methods, namely fix [shake](javascript:;) in the software and hardware. We only discuss the [shake](javascript:;) removal in the software.

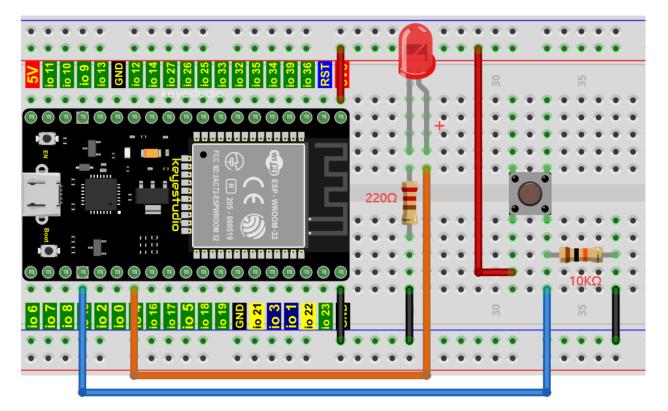
We already know that the [shake](javascript:;) time generated by elasticity is about 10ms, and the delay command can be used to delay the execution time of the command to achieve the effect of [shake](javascript:;) removal.

Therefore, we delay 0.02s in the code to achieve the key anti-shake function.

# Effect excluding jitter

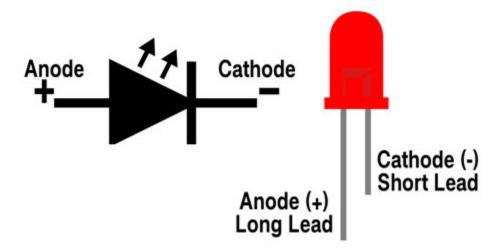


# 5.15.4 4.Wiring Diagram

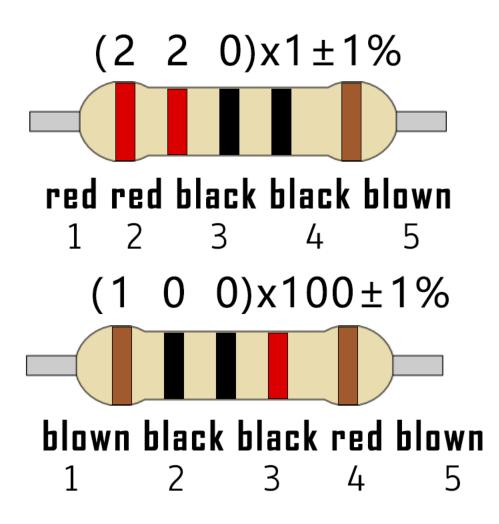


#### Note:

How to connect the LED



How to identify the 220 5-band resistor and 10K 5-band resistor



5.15.5 5.Test code

```
/*
* Filename : Mini Table Lamp
* Description : Make a table lamp.
* Auther : http//www.keyestudio.com
*/
#define PIN_LED
              4
#define PIN_BUTTON 15
bool ledState = false;
void setup() {
 // initialize digital pin PIN_LED as an output.
 pinMode(PIN_LED, OUTPUT);
 pinMode(PIN_BUTTON, INPUT);
}
// the loop function runs over and over again forever
void loop() {
 if (digitalRead(PIN_BUTTON) == LOW) {
```

(continued from previous page)

### 5.15.6 6.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that press the push button switch, the LED turns on; When it is released, the LED is still on. Press it again, and the LED turns off. When it is released, the LED stays off. Doesn't it look like a mini table lamp?

# 5.16 Project 15Tilt and LED

#### 5.16.1 1.Introduction

The ancients without electronic clock, so the hourglass are invented to measure time. The hourglass has a large capacity on both sides, and which is filled with fine sand on one side.

What's more, there is a small channel in the middle, which can make the hourglass stand upright, the side with fine sand is on the top. due to the effect of gravity, the fine sand will flow down through the channel to the other side of the hourglass.

When the sand reaches the bottom, turn it upside down and record the number of times it has gone through the hourglass, therefore, the next day we can know the approximate time of the day by it.

In this project, we will use ESP32 to control the tilt switch and LED lights to simulate an hourglass to make an electronic hourglass.

## 5.16.2 2.Components

	15200 304.5300		-((11())-
ESP32*1	Tilt Switch*1	Red LED*4	10K Resistor*1
Breadboard*1	220 Resistor*4	USB Cable*1	Jumper Wires

# 5.16.3 3.Component Knowledge



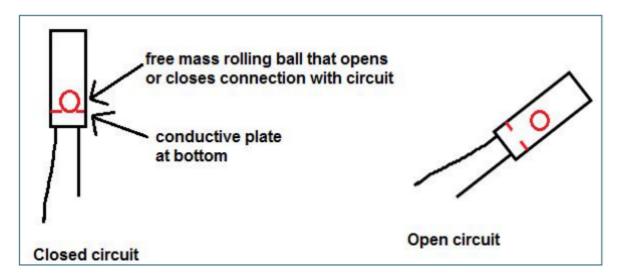
Tilt switch is also called digital switch. Inside is a metal ball that can roll.

The principle of rolling the metal ball to contact with the conductive plate at the bottom, which is used to control the on and off of the circuit. When it is a rolling ball tilt sensing switch with single directional trigger, the tilt sensor is tilted toward the trigger end (two gold-plated pin ends), the tilt switch is in a closed circuit and the voltage at the analog port is about 5V(binary number is 1023).

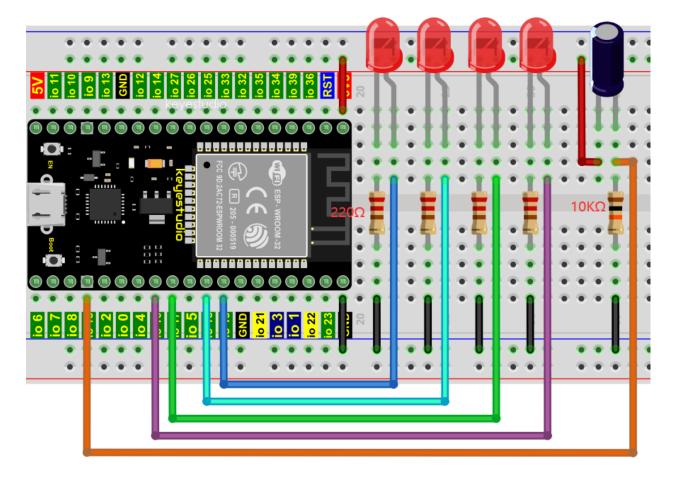
In this way, the LED will light up. When the tilting switch is in horizontal position or tilting to the other end, the tilting switch is in open state the voltage of the analog port is about 0V (binary number is 0), the LED will turn off. In the

program, we judge the state of the switch based on whether the voltage value of the analog port is greater than 2.5V (binary number is 512).

The internal structure of the tilt switch is used here to illustrate how it works, as shown below:

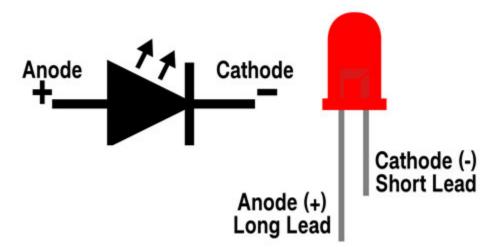


### 5.16.4 4.Wiring Diagram

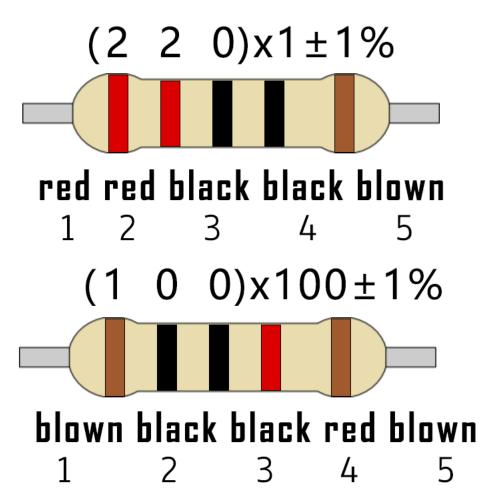


#### Note:

How to connect the LED



How to identify the 220 5-band resistor and 10K 5-band resistor



### 5.16.5 5.Test Code

```
/*
* Filename : Tilt And LED
* Description : Tilt switches and four leds to simulate an hourglass.
* Auther : http//www.keyestudio.com
*/
#define SWITCH_PIN 15 // the tilt switch is connected to Pin15
byte switch_state = 0;
void setup()
{
    for(int i=16;i<20;i++)</pre>
 {
      pinMode(i, OUTPUT);
 }
   pinMode(SWITCH_PIN, INPUT);
for(int i=16;i<20;i++)</pre>
 {
   digitalWrite(i, ♥);
 }
 Serial.begin(9600);
}
void loop()
{
switch_state = digitalRead(SWITCH_PIN);
Serial.println(switch_state);
if (switch_state == 0)
{
for(int i=16;i<20;i++)</pre>
 {
   digitalWrite(i,1);
   delay(500);
 }
 }
  if (switch_state == 1)
{
  for(int i=19;i>15;i--)
  {
   digitalWrite(i, ◊);
   delay(500);
  }
 }
}
```

# 5.16.6 6.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that when you tilt the breadboard to an angle, the LEDs will light up one by one. When you turn the breadboard to the original angle, the LEDs will turn off one by one. Like the hourglass, the sand will leak out over time.

# 5.17 Project 16: I2C 128×32 LCD

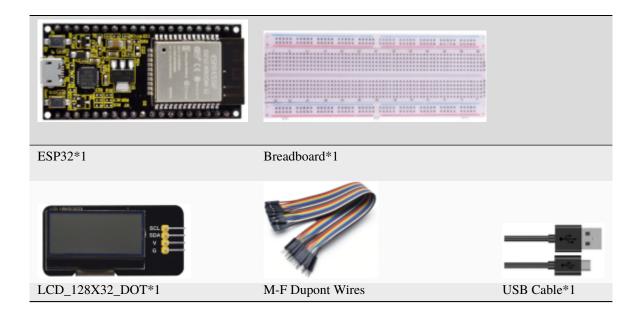
## 5.17.1 1.Introduction

In everyday life, we can do all kinds of experiments with the display module and also DIY a variety of small objects.

For example, you can make a temperature meter with a temperature sensor and display, or make a distance meter with an ultrasonic module and display.

In this project, we will use the LCD\_128X32\_DOT module as the display and connect it to the ESP32, which will be used to control the LCD\_128X32\_DOT display to display various English words, common symbols and numbers.

# 5.17.2 2.Components



## 5.17.3 3.Component Knowledge



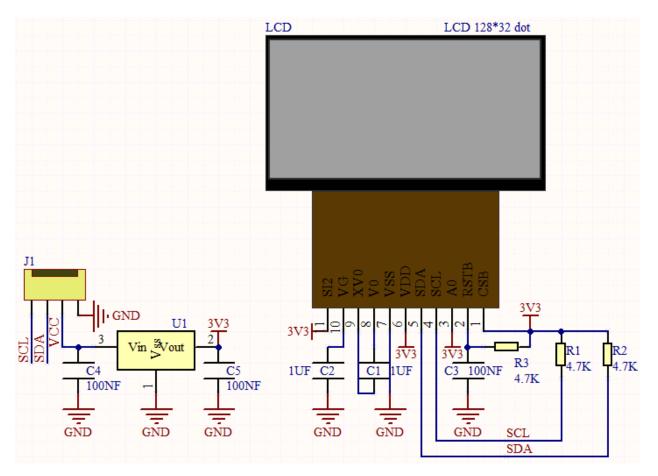
#### LCD\_128X32\_DOT:

It is an LCD module with 128\*32 pixels and its driver chip is ST7567A.

The module uses the IIC communication mode, while the code contains a library of all alphabets and common symbols that can be called directly. When using, we can also set it in the code so that the English letters and symbols show different text sizes.

To make it easy to set up the pattern display, we also provide a mold capture software that converts a specific pattern into control code and then copies it directly into the test code for use.

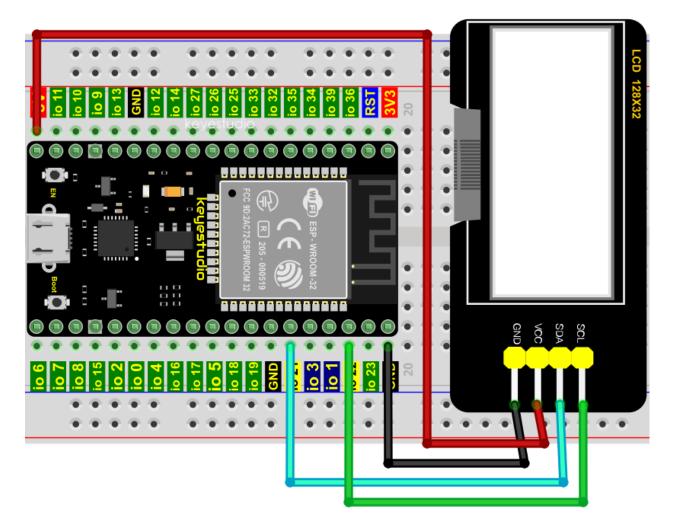
# Schematic diagram of LCD\_128X32\_DOT



#### Features:

- Pixel: 128\*32 character
- Operating voltage(chip)4.5V to 5.5V
- Operating current100mA (5.0V)
- Optimal operating voltage(module):5.0V

# 5.17.4 4.Wiring Diagram



# 5.17.5 5.Adding the lcd128\_32\_io library

This code uses a library named "**lcd128\_32\_io**", if you haven't installed it yet, please do so before learning. The steps to add third-party libraries are as follows:

Open the Arduino IDEclick "Sketch"  $\rightarrow$  "Include Library"  $\rightarrow$  "Add .ZIP Library...".

Click on the link to download the library fileArduino C "lcd128\_32\_io.h" Librarie

Select the LCD\_128X32.ZIP file and then click"Open".

	Project_17_I2C_128_32_LCD   Arduino 1.8.16 File Edit Sketch Tools Help					_		×
Correction Sket	Verify/Compile Upload		Ctrl+R Ctrl+U					ø
Project	Upload Using Pr	ogrammer	Ctrl+Shift+U					
//***** /*	Export compiled	-	Ctrl+Alt+S		∆ Manage Libraries	Ctrl+Shift+I	***	^
* File * Desc	Show Sketch Fol	der	Ctrl+K		Add ZID Liberry			
* Auth	Include Library		2		Add .ZIP Library			
*/	Add File				Arduino libraries			
<pre>#include "1</pre>	lcd128_32_io.h	"			Bridge			
//Create 1CD128 *32 pin, sda>21, scl>2				Esplora				
lcd lcd(21,	, 22);				Ethernet			
void setup	0 4				Firmata			
<pre>lcd.Init(); //initialize</pre>				GSM				
	r(); //clear				LiquidCrystal			
}					Mouse			
<pre>void loop()</pre>	•				Robot IR Remote			
<pre>lcd.Cursor(0, 4); //Set display position lcd.Display("KEYESTUDIO"); //Setting the d</pre> Robot Motor					~			
-							1	
🥯 Select a zip f	ïle or a folder contair	ing the libra	ry you'd like to add	d				×
Look in: 📙 3. Libraries						0		
<b>e</b>	LCD_128X32	-	0					
Recent Items	File name:	learning Ki	t Basic Edition\	2. W	indows System\2. C_Tutor	rial\3. Libraries	0	pen
	Files of type: ZIP files or folders 🗸 🗸						Ca	ncel

### 5.17.6 6.Test Code

}

(continued from previous page)

# 5.17.7 7.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the 128X32LCD module display will show "KEYESTUDIO" at the first line "ABCDEFGHIJKLMNOPQR" will be displayed at the second line "123456789 $\pm$ \*/<>=\$@" will be shown at the third line and "%^&(){}:;'|?,.~\[]" will be displayed at the fourth line.

# 5.18 Project 17Small Fan

# 5.18.1 1.Introduction

In hot summer, we need electric fans to cool us down, so in this project, we will use the ESP32 to control a DC motor and small fan blades to make a small electric fan.

# 5.18.2 2.Components

		1 • • • • • • • • • • • • • • • • • • •	
ESP32*1	Breadboard*1	6 AA Battery Holder*1	Breadboard Power Module*1
AA Battery(Self-prepared)*6	Fan*1	DC Motor*1	NPN Transistor (S8050)*1
	<b>(</b> ))))		
PNP Transistor (S8550)*1	1K Resistor*1	Jumper Wire	Diode*1
USB Ca- ble*1			

# 5.18.3 3.Component Knowledge

### Keyestudio Breadboard Power Supply Module



### Introduction

This breadboard power supply module is compatible with 5V and 3.3V, which can be applied to MB102 breadboard. The module contains two channels of independent control, powered by the USB all the way.

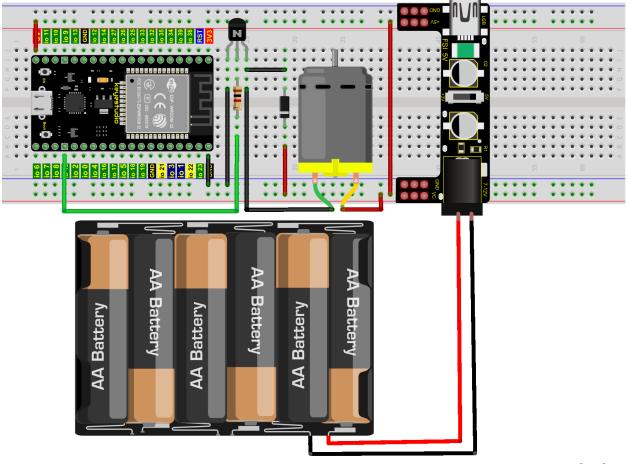
The output voltage is constant for the DC5V, and another way is powered by DC 7-12V, output controlled by the slide switch, respectively for DC5V and DC3.3V.

If the other power supply is DC 7-12v, when the slide switch is switched to +5V, the output voltages of the left and right lines of the module are DC 5V. When the slide switch is switched to +3V, the output voltage of the USB power supply terminal of the module is DC5V, and the output voltage of the DC 7-12V power supply terminal of the other power supply is DC3.3V.

### Specification

- Applied to MB102 breadboard
- Input voltageDC 7-12V or powered by USB
- Output voltage3.3V or 5V
- Max output current<700mA
- Up and down two channels of independent control, one of which can be switched to 3.3V or 5V
- Comes with two sets of DC output pins, easy for external use

## 5.18.4 4.Wiring Diagram 1



fritzing

(Note: Connect the wires and then install a small fan blade on the DC motor. )

## 5.18.5 5. Test Code

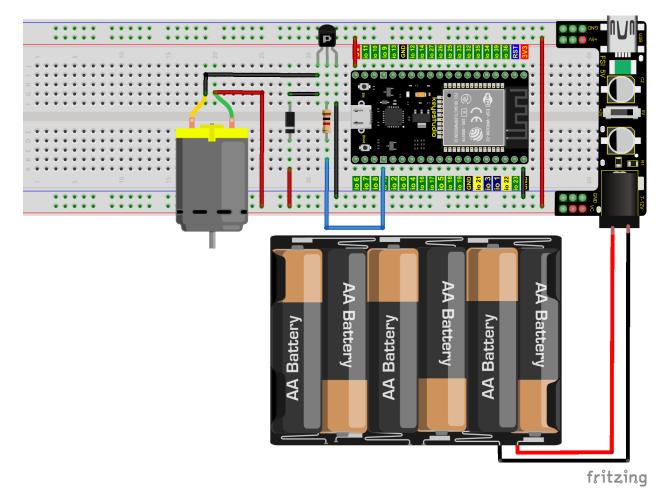
```
/*
* Filename : Small_Fan
* Description : S8050 triode drives the motor working
* Auther : http//www.keyestudio.com
*/
void setup() {
 pinMode(15, OUTPUT); // Initialize pin 15 as output.
}
void loop() {
 digitalWrite(15, HIGH); // Turn on the motor (HIGH means HIGH level)
 delay(4000); // Delay 4 seconds
digitalWrite(15, LOW); // Reduce the voltage and turn off the motor
 delay(2000);
                    // Delay 2 seconds
}
```

# 5.18.6 6.Test Result 1

Upload the code to the ESP32 and power up. The motor rotates for 4s, stops for 2s, in loop way.

# 5.18.7 7. Wiring Diagram 2

We use the S8550 PNP transistor to control the motor.



Note: wire up and connect a fan on the motor.

## 5.18.8 8.Test Code 2

```
/*
* Filename : Small_Fan
* Description : S8550 triode drives the motor working
* Auther
          : http//www.keyestudio.com
*/
void setup() {
 pinMode(15, OUTPUT); // Initialize pin 15 as output.
}
void loop() {
 digitalWrite(15, LOW);
                      // Turn on the motor (LOW means LOW level)
 delay(4000);
                      // Delay 4 seconds
 digitalWrite(15, HIGH);
                      // Raise the voltage and turn off the motor
                       // Delay 2 seconds
 delay(2000);
}
```

## 5.18.9 9.Test Result 2

Upload the code to the ESP32 and power up. The motor rotates for 4s, stops for 2s, in loop way.

# 5.19 Project 18Dimming Light

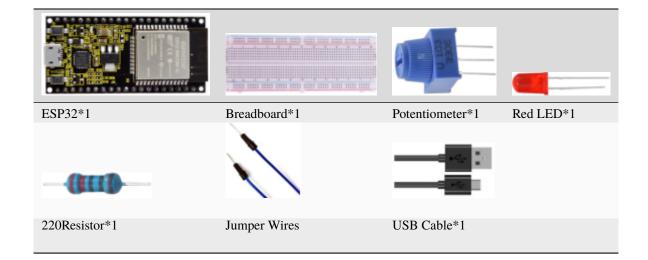
### 5.19.1 1.Introduction

A potentiometer is a three-terminal resistor with sliding or rotating contacts that forms an adjustable voltage divider. It works by changing the position of the sliding contacts across a uniform resistance.

In the potentiometer, the entire input voltage is applied across the whole length of the resistor, and the output voltage is the voltage drop between the fixed and sliding contact.

In this project, we will learn how to use ESP32 to read the values of the potentiometer, and make a dimming lamp with LED.

## 5.19.2 2.Components



## 5.19.3 3.Component Knowledge



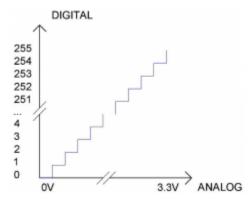
### Adjustable potentiometer:

It is a kind of resistor and an analog electronic component, which has two states of 0 and 1(high level and low level). The analog quantity is different, its data state presents a linear state such as  $1 \sim 1024$ .

### ADC:

An ADC is an electronic integrated circuit used to convert analog signals such as voltages to digital or binary form consisting of 1s and 0s.

The range of our ADC on ESP32 is 12 bits, that means the resolution is  $2^{12}=4096$ , and it represents a range (at 3.3V) will be divided equally to 4096 parts. The rage of analog values corresponds to ADC values. So the more bits the ADC has, the denser the partition of analog will be and the greater the precision of the resulting conversion.



Subsection 1: the analog in rang of 0V—3.3/4095 V corresponds to digital 0; Subsection 2: the analog in rang of 3.3/4095 V—2\*3.3 /4095 V corresponds to digital 1;

•••

The following analog will be divided accordingly.

The conversion formula is as follows:

$$ADCValue = \frac{Ana \log Voltage}{3.3} * 4095$$

### DAC

The reversing of this process requires a DAC, Digital-to-Analog Converter. The digital I/O port can output high level and low level (0 or 1), but cannot output an intermediate voltage value. This is where a DAC is useful. ESP32 has two DAC output pins with 8-bit accuracy, GPIO25 and GPIO26, which can divide VCC (here is 3.3V) into 2^8=256 parts.

For example, when the digital quantity is 1, the output voltage value is 3.3/256 \* 1 V, and when the digital quantity is 128, the output voltage value is 3.3/256 \* 128 = 1.65 V, the higher the accuracy of DAC, the higher the accuracy of output voltage value will be.

The conversion formula is as follows:

Ana 
$$\log Voltage = \frac{DACValue}{255} * 3.3(V)$$

### ADC on ESP32

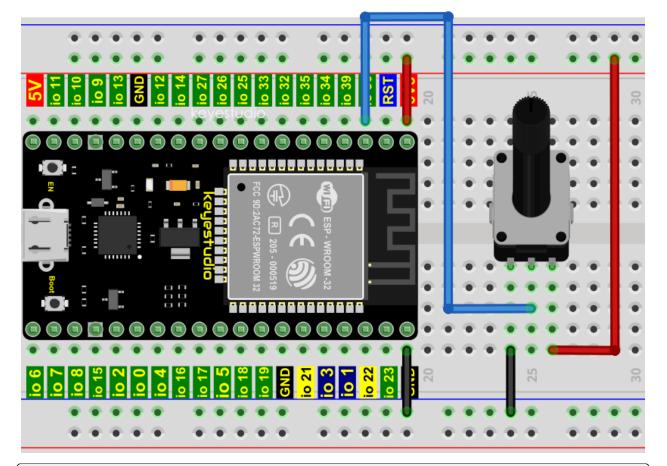
ESP32 has 16 pins can be used to measure analog signals. GPIO pin sequence number and analog pin definition are shown in the following table

ESP32 has two 8-bit digital analog converters to be connected to GPIO25 and GPIO26 pins, respectively, and it is immutable. As shown in the following table

The DAC pin number is already defined in ESP32's code base; for example, you can replace GPIO25 with DAC1 in the code.

Read the ADC value, DAC value and voltage value of the potentiometer.

We connect the potentiometer to the analog IO port of ESP32 to read the ADC value, DAC value and voltage value of the potentiometer, please refer to the wiring diagram below



```
//In loop()the analogRead() function is used to obtain the ADC value,
//and then the map() function is used to convert the value into an 8-bit precision DAC_
-value.
//The input and output voltage are calculated according to the previous formula,
//and the information is finally printed out.
void loop() {
    int adcVal = analogRead(PIN_ANALOG_IN);
    int dacVal = map(adcVal, 0, 4095, 0, 255);
    double voltage = adcVal / 4095.0 * 3.3;
    Serial.printf("ADC Val: %d, \t DAC Val: %d, \t Voltage: %.2fV\n", adcVal, dacVal,_
    +voltage);
    delay(200);
}
```

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable, open the serial monitor and set the baud rate to 115200. You will see that the serial monitor window will print out the ADC value, DAC value and voltage value of the potentiometer.

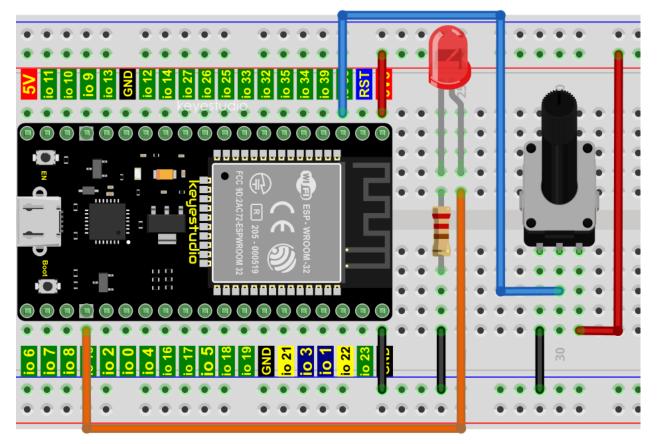
When turning the potentiometer handle, the ADC value, DAC value and voltage value will change. As shown below:

💿 сомз									_		$\times$
											Send
ADC Val:	76, DAC	Val:	4,	Voltage:	0.06V						^
ADC Val:	402, DAC	Val:	25,	Voltage:	0.32V						
ADC Val:	908, DAC	Val:	56,	Voltage:	0.73V						
ADC Val:	909, DAC	Val:	56,	Voltage:	0.73V						
ADC Val:	1577, DAC	Val:	98,	Voltage:	1.27V						
ADC Val:	1711, DAC	Val:	106,	Voltage:	1.38V						
ADC Val:	1904, DAC	Val:	118,	Voltage:	1.53V						
ADC Val:	2265, DAC	Val:	141,	Voltage:	1.83V						
ADC Val:	2271, DAC	Val:	141,	Voltage:	1.83V						
ADC Val:	2555, DAC	Val:	159,	Voltage:	2.06V						
ADC Val:	2547, DAC	Val:	158,	Voltage:	2.05V						
ADC Val:	3003, DAC	Val:	187,	Voltage:	2.42V						
ADC Val:	3327, DAC	Val:	207,	Voltage:	2.68V						
ADC Val:	4095, DAC	Val:	255,	Voltage:	3.30V						
ADC Val:	4095, DAC	Val:	255,	Voltage:	3.30V						
											~
Autoscr	oll 🗌 Show t	imesta	mp			Newline	~	115200 b	aud 🗸	Clea	r output

# 5.19.4 4.Wiring diagram of the dimming lamp

In the previous step, we read the ADC value, DAC value and voltage value of the potentiometer.

Now we need to convert the ADC value of the potentiometer into the brightness of the LED to make a lamp that can adjust the brightness. The wiring diagram is as follow:



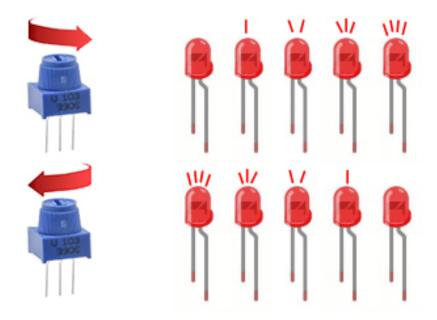
## 5.19.5 5.Test Code

```
/*
* Filename
            : Dimming Light
* Description : Controlling the brightness of LED by potentiometer.
            : http//www.keyestudio.com
* Auther
*/
#define PIN_ANALOG_IN
                    36 //the pin of the potentiometer
#define PIN LED
              15 // the pin of the LED
#define CHAN
                   0
void setup() {
 ledcSetup(CHAN, 1000, 12);
 ledcAttachPin(PIN_LED, CHAN);
}
void loop() {
 int adcVal = analogRead(PIN_ANALOG_IN); //read adc
```

	(continued from previous page)
<pre>int pwmVal = adcVal;</pre>	// adcVal re-map to pwmVal
<pre>ledcWrite(CHAN, pwmVal);</pre>	// set the pulse width.
<pre>delay(10);</pre>	
}	
//********	*********************

## 5.19.6 6.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that turn the potentiometer handle and the brightness of the LED will change accordingly.



# 5.20 Project 19Flame Alarm

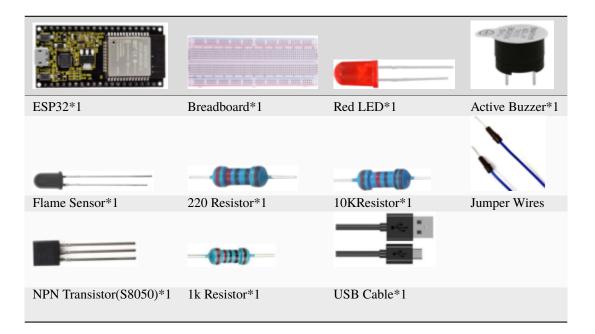
## 5.20.1 1.Introduction

Fire is a terrible disaster and fire alarm systems are very useful in houses, commercial buildings and factories.

In this project, we will use ESP32 to control a flame sensor, a buzzer and a LED to simulate fire alarm devices. This is a meaningful maker activity.

. .

## 5.20.2 2.Components

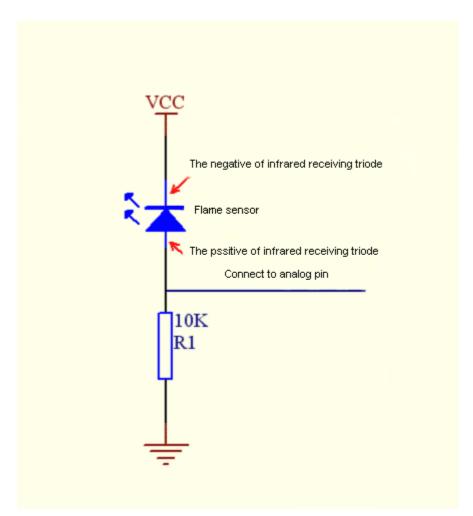


# 5.20.3 3.Component Knowledge



The flame emits a certain amount IR light that is invisible to the human eye, but our flame sensor can detect it and alert a microcontroller (such as ESP32) that a fire has been detected.

It has a specially designed infrared receiver tube to detect the flame and then convert the flame brightness into a fluctuating level signal. The short pin of the receiving triode is negative pole and the other long pin is positive pole. We should connect the short pin (negative) to 5V and the long pin (positive) to the analog pin, a resistor and GND. As shown in the figure below

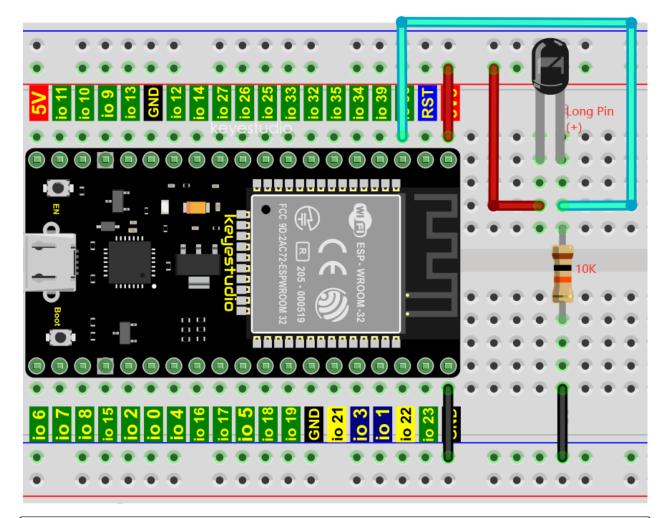


#### Note:

Since vulnerable to radio frequency radiation and temperature changes, the flame sensor should be kept away from heat sources like radiators, heaters and air conditioners, as well as direct irradiation of sunlight, headlights and incandescent light.

## 5.20.4 4. Read the ADC value, DAC value and voltage value of the flame sensor

We first use a simple code to read the ADC value, DAC value and voltage value of the flame sensor and print them out. Please refer to the wiring diagram below



```
/*
* Filename : Read Analog Value Of Flame Sensor
* Description : Basic usage of ADCDAC and Voltage
* Auther : http//www.keyestudio.com
*/
#define PIN_ANALOG_IN 36 //the pin of the Flame sensor
void setup() {
 Serial.begin(115200);
}
//In loop()the analogRead() function is used to obtain the ADC value,
//and then the map() function is used to convert the value into an 8-bit precision DAC
\rightarrow value.
//The input and output voltage are calculated according to the previous formula,
//and the information is finally printed out.
void loop() {
 int adcVal = analogRead(PIN_ANALOG_IN);
 int dacVal = map(adcVal, 0, 4095, 0, 255);
 double voltage = adcVal / 4095.0 * 3.3;
```

```
Serial.printf("ADC Val: %d, \t DAC Val: %d, \t Voltage: %.2fV\n", adcVal, dacVal,_

voltage);

delay(200);

}
```

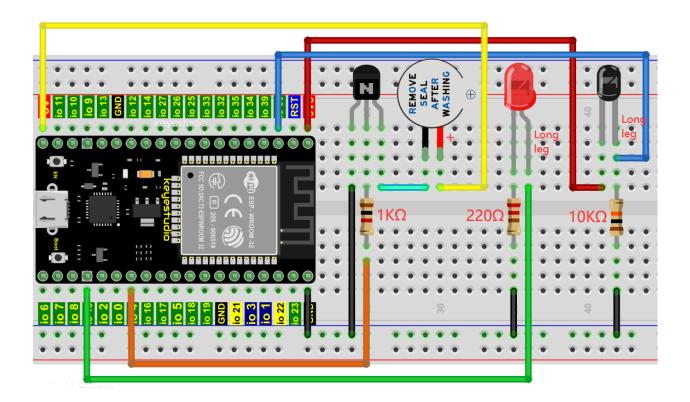
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable, open the serial monitor and set the baud rate to 115200. You will see that the serial monitor window will print out the ADC value, DAC value and voltage value of the flame sensor.

When the sensor is closed to fire, the ADC value, DAC value and voltage value will get greater. Conversely, the ADC value, DAC value and voltage value decrease.

0	COM3								_		$\times$
											Send
ADC	Val:	76,	DAC Val: 4,	, Voltage:	0.06V						^
ADC	Val:	402,	DAC Val: 29	5, Voltage:	0.32V						
ADC	Val:	908,	DAC Val: 50	6, Voltage:	0.73V						
ADC	Val:	909,	DAC Val: 50	6, Voltage:	0.73V						
ADC	Val:	1577,	DAC Val: 98	8, Voltage:	1.27V						
ADC	Val:	1711,	DAC Val: 10	06, Voltage:	1.38V						
ADC	Val:	1904,	DAC Val: 11	18, Voltage:	1.53V						
ADC	Val:	2265,	DAC Val: 14	41, Voltage:	1.83V						
ADC	Val:	2271,	DAC Val: 14	41, Voltage:	1.83V						
ADC	Val:	2555,	DAC Val: 19	59, Voltage:	2.06V						
ADC	Val:	2547,	DAC Val: 19	58, Voltage:	2.05V						
ADC	Val:	3003,	DAC Val: 18	87, Voltage:	2.42V						
ADC	Val:	3327,	DAC Val: 20	07, Voltage:	2.68V						
ADC	Val:	4095,	DAC Val: 29	55, Voltage:	3.30V						
ADC	Val:	4095,	DAC Val: 29	55, Voltage:	3.30V						
				_							~
<b>∠</b> !	Autosci	roll 🗌	Show timestamp			Newline	~	115200 baud	~	Clear	output

# 5.20.5 5.Wiring diagram of the flame alarm

Next, we will use a flame sensor, a buzzer, and a LED to make an interesting project, that is flame alarm. When flame is detected, the LED flashes and the buzzer alarms.



### 5.20.6 6.Test Code

```
value = get value()
if value >500:
    buzzer.value(1) the threshold of 500 in the code can be reset as required)
```

```
/*
* Filename : Flame Alarm
* Description : Controlling the buzzer and LED by flame sensor.
* Auther : http//www.keyestudio.com
*/
#define PIN_ADC0
                36 //the pin of the flame sensor
#define PIN_LED 15 // the pin of the LED
#define PIN_BUZZER 4 // the pin of the buzzer
void setup() {
 pinMode(PIN_LED, OUTPUT);
 pinMode(PIN_BUZZER, OUTPUT);
 pinMode(PIN_ADC0, INPUT);
}
void loop() {
 int adcVal = analogRead(PIN_ADC0); //read the ADC value of flame sensor
 if (adcVal >= 500) {
   digitalWrite (PIN_BUZZER, HIGH); //turn on buzzer
   digitalWrite(PIN_LED, HIGH); // turn on LED
```

```
delay(500); // wait a second.
digitalWrite (PIN_BUZZER, LOW);
digitalWrite(PIN_LED, LOW); // turn off LED
delay(500); // wait a second
}
else
{
digitalWrite(PIN_LED, LOW); //turn off LED
digitalWrite (PIN_BUZZER, LOW); //turn off buzzer
}
}
```

### 5.20.7 7.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that when the flame sensor detects the flame, the LED will flash and the buzzer will alarm; otherwise, the LED does not light up and the buzzer does not sound.

# 5.21 Project 20Night Lamp

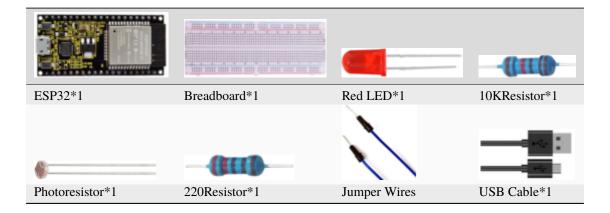
### 5.21.1 1.Introduction

Sensors or components are ubiquitous in our daily life. For example, some public street lamps will automatically turn on at night and turn off during the day.

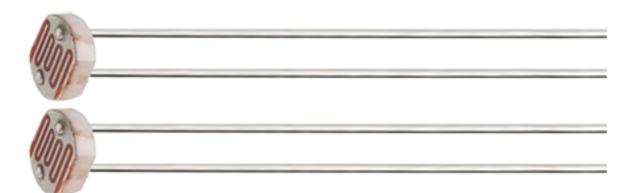
Why? In fact, this make use of a photosensitive element that senses the intensity of external ambient light. When the outdoor brightness decreases at night, the street lights will turn on automatically; In the daytime, the street lights will automatically turn off.

The principle of which is very simple, In this Project, we use a ESP32 to control a LED to achieve the effect of the street light.

## 5.21.2 2.Components



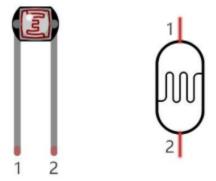
# 5.21.3 3.Component Knowledge



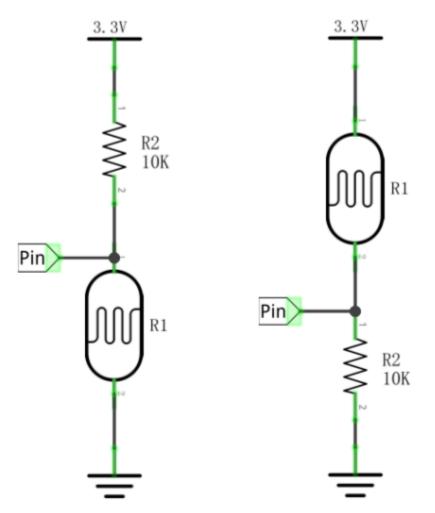
### **Photoresistor :**

It is a kind of photosensitive resistance, its principle is that the photoresistor surface receives brightness (light) to reduce the resistance, the resistance value will change with the detected intensity of the ambient light. With this characteristic, we can use the photosensitive resistance to detect the light intensity.

Photosensitive resistance and its electronic symbol are as follows



The following circuit is used to detect changes in resistance values of photoresistors

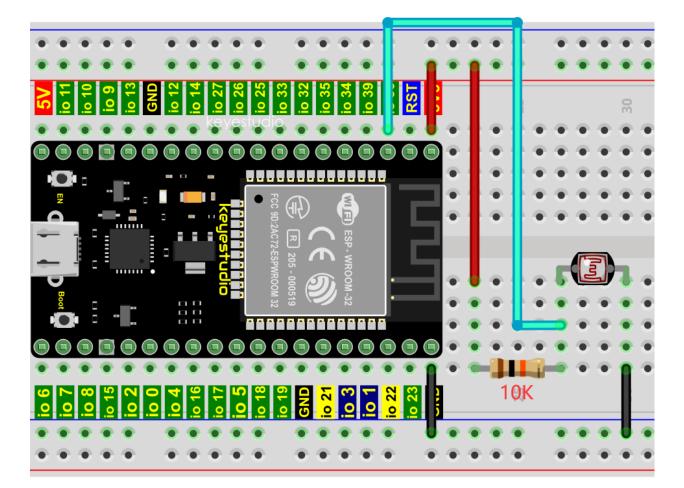


In the circuit above, when the resistance of the photoresistor changes due to the change of light intensity, the voltage between the photoresistor and resistance R2 will also change.

Thus, the intensity of light can be obtained by measuring this voltage.

# 5.21.4 4.Read the ADC value, DAC value and voltage value of the photoresistor

We first use a simple code to read the ADC value, DAC value and voltage value of the photoresistor and print them out. Please refer to the following wiring diagram



```
/*
* Filename : Read Photosensitive Analog Value
* Description : Basic usage of ADC
* Auther : http//www.keyestudio.com
*/
#define PIN_ANALOG_IN 36 //the pin of the photosensitive sensor
void setup() {
 Serial.begin(115200);
}
//In loop()the analogRead() function is used to obtain the ADC value,
//and then the map() function is used to convert the value into an 8-bit precision DAC.
\rightarrow value.
//The input and output voltage are calculated according to the previous formula,
//and the information is finally printed out.
void loop() {
 int adcVal = analogRead(PIN_ANALOG_IN);
 int dacVal = map(adcVal, 0, 4095, 0, 255);
 double voltage = adcVal / 4095.0 * 3.3;
 Serial.printf("ADC Val: %d, \t DAC Val: %d, \t Voltage: %.2fV\n", adcVal, dacVal, __
→voltage);
                                                                    (continues on next page)
```

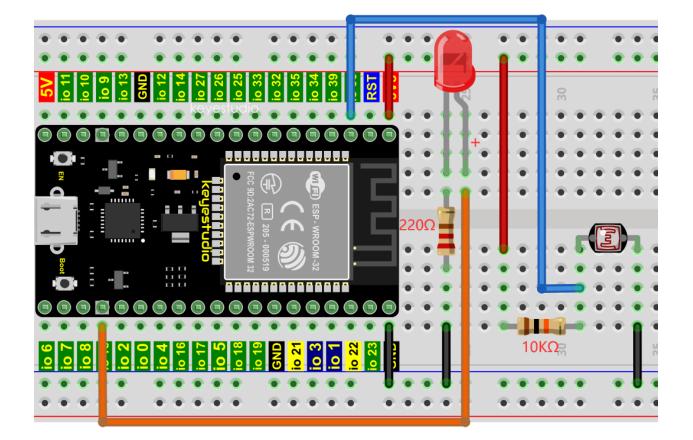
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable, open the serial monitor and set the baud rate to 115200.

You will see that the serial monitor window will print out the ADC valueDAC value and voltage value of the photoresistor. When the light intensity around the photoresistor is gradually reduced, the ADC value, DAC value and voltage value will gradually increase. On the contrary, the ADC value, DAC value and voltage value decrease gradually.

0	COM3					_	$\Box$ $\times$
							Send
ADC	Val:	76,	DAC Val: 4,	Voltage: 0.06V			^
ADC	Val:	402,	DAC Val: 25,	Voltage: 0.32V			
ADC	Val:	908,	DAC Val: 56,	Voltage: 0.73V			
ADC	Val:	909,	DAC Val: 56,	Voltage: 0.73V			
ADC	Val:	1577,	DAC Val: 98,	Voltage: 1.27V			
ADC	Val:	1711,	DAC Val: 106,	Voltage: 1.38V			
ADC	Val:	1904,	DAC Val: 118,	Voltage: 1.53V			
ADC	Val:	2265,	DAC Val: 141,	Voltage: 1.83V			
ADC	Val:	2271,	DAC Val: 141,	Voltage: 1.83V			
ADC	Val:	2555,	DAC Val: 159,	Voltage: 2.06V			
ADC	Val:	2547,	DAC Val: 158,	Voltage: 2.05V			
ADC	Val:	3003,	DAC Val: 187,	Voltage: 2.42V			
ADC	Val:	3327,	DAC Val: 207,	Voltage: 2.68V			
ADC	Val:	4095,	DAC Val: 255,	Voltage: 3.30V			
ADC	Val:	4095,	DAC Val: 255,	Voltage: 3.30V			
							~
	Autosc	roll 🗌	Show timestamp		Newline $\sim$	115200 baud $\lor$	Clear output

# 5.21.5 5.Wiring diagram of the light-controlled lamp

We made a small dimming lamp in the front, now we will make a light controlled lamp. The principle is the same, that is, the ESP32 takes the ADC value of the sensor, and then adjusts the brightness of the LED.



# 5.21.6 6.Test Code

```
/*
* Filename : Night Lamp
* Description : Controlling the brightness of LED by photosensitive sensor.
* Auther
           : http//www.keyestudio.com
*/
#define PIN_ANALOG_IN 36 // the pin of the photosensitive sensor
#define PIN_LED 15 // the pin of the LED
#define CHAN
                     0
#define LIGHT_MIN
                     372
#define LIGHT_MAX
                     2048
void setup() {
 ledcSetup(CHAN, 1000, 12);
 ledcAttachPin(PIN_LED, CHAN);
}
void loop() {
 int adcVal = analogRead(PIN_ANALOG_IN); //read adc
 int pwmVal = map(constrain(adcVal, LIGHT_MIN, LIGHT_MAX), LIGHT_MIN, LIGHT_MAX, 0,
\rightarrow 4095); // adcVal re-map to pwmVal
ledcWrite(CHAN, pwmVal); // set the pulse width.
 delay(10);
}
```

## 5.21.7 7.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that when the intensity of light around the photoresistor is reduced, the LED will be bright, on the contrary, the LED will be dim.

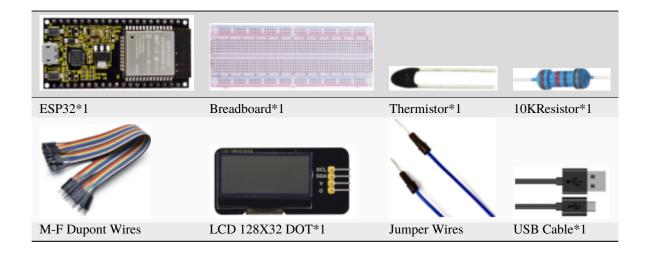
# 5.22 Project 21: Temperature Instrument

# 5.22.1 1.Introduction

Thermistor is a kind of resistor whose resistance depends on temperature changes, which is widely used in gardening, home alarm systems and other devices.

Therefore, we can use the features to make a temperature instrument.

# 5.22.2 2.Components



# 5.22.3 3.Component Knowledge

Thermistor: It is a temperature sensitive resistor.

When it senses a change in temperature, the resistance of the thermistor will change. We can take advantage of this characteristic to detect temperature intensity. The thermistor and its electronic symbol are shown below:



The relationship between resistance and temperature of the thermistor is

$$Rt = R * EXP[B * \left(\frac{1}{T2} - \frac{1}{T1}\right)]$$

**Rt** is the thermistor resistance under T2 temperature;

**R** is the nominal resistance of thermistor under T1 temperature;

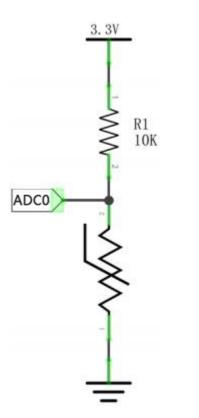
**EXP[n]** is nth power of e;

**B** is temperature index;

T1, T2 is Kelvin temperature (absolute temperature). Kelvin temperature=273.15 + Celsius temperature.

Parameters : B=3950, R=10k, T1=25.

The circuit connection method of the thermistor is similar to the photoresistor, as shown below



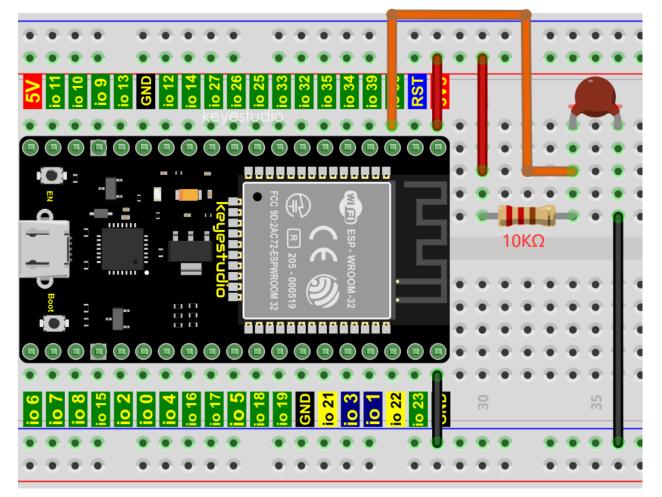
We can use the value measured by the ADC converter to obtain the resistance of thermistor, and then we can use the formula to obtain the temperature value.

Therefore, the temperature formula can be derived as:

$$T2 = 1/\left(\frac{1}{T1} + \ln\left(\frac{Rt}{R}\right)/B\right)$$

## 5.22.4 4.Read the value of the Thermistor

First we will learn the thermistor to read the current ADC value, voltage value and temperature value and print them out. Please connect the wirings according to the wiring diagram below

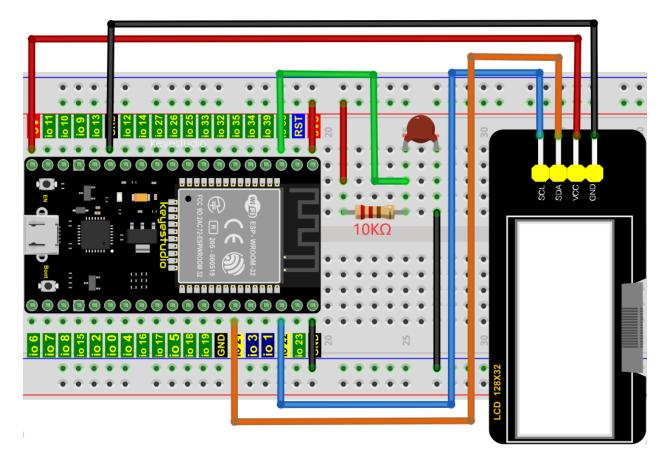


```
//*******
                       /*
* Filename
           : Thermomter
* Description : Making a thermometer by thermistor.
* Auther
           : http//www.keyestudio.com
*/
#define PIN_ANALOG_IN 36
void setup() {
 Serial.begin(115200);
}
void loop() {
 int adcValue = analogRead(PIN_ANALOG_IN);
                                                         //read ADC pin
 double voltage = (float)adcValue / 4095.0 * 3.3;
                                                         // calculate voltage
 double Rt = 10 * voltage / (3.3 - voltage);
                                                         //calculate resistance
→value of thermistor
 double tempK = 1 / (1 / (273.15 + 25) + log(Rt / 10) / 3950.0); //calculate_
```

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable, open the serial monitor and set the baud rate to 115200.

You will see that the monitor prints out the thermistor's current ADC value, voltage value and temperature value. Try pinching the thermistor with your index finger and thumb (don't touch wires) for a while, and you will see the temperature increasing.

💿 C	OM3			_	
					Send
ADC v	alue : 1861,	Voltage : 1.50V,	Temperature : 29.17C		^
ADC v	alue : 1870,	Voltage : 1.51V,	Temperature : 28.96C		
ADC v	alue : 1872,	Voltage : 1.51V,	Temperature : 28.92C		
ADC v	alue : 1871,	Voltage : 1.51V,	Temperature : 28.94C		
ADC v	alue : 1876,	Voltage : 1.51V,	Temperature : 28.83C		
ADC v	alue : 1874,	Voltage : 1.51V,	Temperature : 28.87C		
ADC v	alue : 1875,	Voltage : 1.51V,	Temperature : 28.85C		
ADC v	alue : 1875,	Voltage : 1.51V,	Temperature : 28.85C		
ADC v	alue : 1887,	Voltage : 1.52V,	Temperature : 28.58C		
ADC v	alue : 1909,	Voltage : 1.54V,	Temperature : 28.08C		
ADC v	alue : 1851,	Voltage : 1.49V,	Temperature : 29.40C		
ADC v	alue : 1879,	Voltage : 1.51V,	Temperature : 28.76C		
ADC v	alue : 1883,	Voltage : 1.52V,	Temperature : 28.67C		
ADC v	alue : 1862,	Voltage : 1.50V,	Temperature : 29.15C		
ADC v	alue : 1810,	Voltage : 1.46V,	Temperature : 30.34C		
					~
Aut	toscroll 🗌 Show time:	stamp	Newline $\checkmark$ 115200 baud	~	Clear output



# 5.22.5 5.Wiring diagram of the temperature instrument

# 5.22.6 6.Adding the lcd128\_32\_io library

 $Open \ the \ Arduino \ IDE click \ ``Sketch" \rightarrow ``Include \ Library" \rightarrow ``Add \ .ZIP \ Library...''.$ 

In the pop-up window, find the file named **"2. Windows System\2. C\_Tutorial\3. Libraries\LCD\_128X32.ZIP"**, which locates in this directory. Select the LCD\_128X32.ZIP file and then click "Open".

	1.2_Temperature_Instr tch Tools Help	ument   Arduino 1.8.16	Manage Librar	∆ ries	Ctrl+Shift+I		×
Project	Export compiled Binary		Add .ZIP Libra Arduino librari Bridge Esplora				2- 2- ^
void lo	Show Sketch Folde	r Ctrl+K	Ethernet Firmata				
double t double t lcd.Curs lcd.Disp lcd.Curs lcd.Disp lcd.Curs lcd.Disp lcd.Curs lcd.Disp	Add File tt = 10 * Voltage sempK = 1 / (1 / sempC = tempK - 2 sor(0,0); //Set d		Mouse Robot IR Remo			ltage istance peratur peratur	e
<ul> <li>Select a zip</li> </ul>	file or a folder containing	the library you'd like to add					<b>&gt;</b>
Look in Cook in Recent Items	: 3. Libraries	- 1			- v 🌶 📂 🛄		2

## 5.22.7 7.Test Code

Desktop

<u>\_\_\_</u>

File name:

Files of type:

ZIP files or folders

//*************************************
/*
* Filename : Temperature Instrument
* Description : LCD displays the temperature of thermistor.
* Auther : http//www.keyestudio.com
*/
<pre>#include "lcd128_32_io.h"</pre>
#define PIN_ANALOG_IN 36
<pre>lcd lcd(21, 22); //Create lCD128 *32 pinsda-&gt;21 scl-&gt;22</pre>
(continues on next page)

? Learning Kit Basic Edition\2. Windows System\2. C\_Tutorial\3. Libraries

Chapter 5. Arduino Tutorial

0pen

Cancel

 $\sim$ 

```
void setup() {
 lcd.Init(); //initialize
 lcd.Clear(); //clear
}
char string[10];
void loop() {
 int adcValue = analogRead(PIN_ANALOG_IN);
                                                            //read ADC pin
 double voltage = (float)adcValue / 4095.0 * 3.3;
                                                            // calculate voltage
 double Rt = 10 * voltage / (3.3 - voltage);
                                                             //calculate resistance
\rightarrow value of thermistor
 double tempK = 1 / (1 / (273.15 + 25) + log(Rt / 10) / 3950.0); //calculate_
→temperature (Kelvin)
 double tempC = tempK - 273.15;
                                                             //calculate
→temperature (Celsius)
 lcd.Cursor(0,0); //Set display position
 lcd.Display("Voltage:"); //Setting the display
 lcd.Cursor(0,8);
 lcd.DisplayNum(voltage);
 lcd.Cursor(0,11);
 lcd.Display("V");
 lcd.Cursor(2, 0);
 lcd.Display("tempC:");
 lcd.Cursor(2,8);
 lcd.DisplayNum(tempC);
 lcd.Cursor(2,11);
 lcd.Display("C");
 delay(200);
}
```

## 5.22.8 8.Test Result

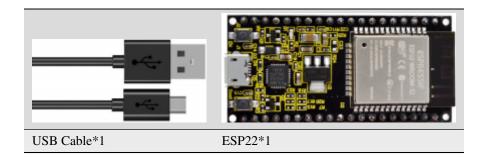
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the LCD 128X32 DOT displays the voltage value of the thermistor and the temperature value in the current environment.

# 5.23 Project 22Bluetooth

This chapter mainly introduces how to make simple data transmission through Bluetooth of ESP22 and mobile phones. Project 22.1 is classic Bluetooth while project 22.2 is Bluetooth control LED.

# 5.23.1 Project 22.1Classic Bluetooth

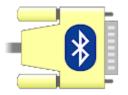
### 1.Components



In this tutorial we need to use a Bluetooth APP called serial Bluetooth terminal to assist in the experiment.

Download link: https://www.appsapk.com/serial-Bluetooth-terminal/.

Here is its sign



### 2.Component Knowledge

Bluetooth is a short-distance communication system that can be divided into two types, namely low power Bluetooth (BLE) and classic Bluetooth. There are two modes for simple data transfer: master mode and slave mode.

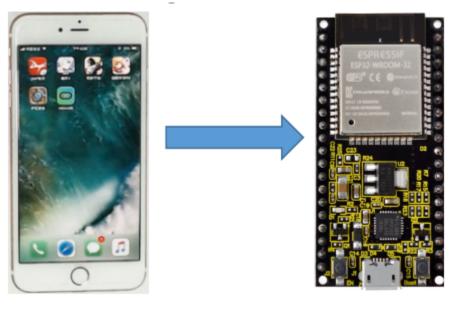
### Master Mode:

In this mode, work is done on the master device and can be connected to the slave device. When the device initiates a connection request in the main mode, information such as the address and pairing password of other Bluetooth devices are required. Once paired, you can connect directly to them.

#### Slave Mode:

A Bluetooth module in the slave mode can only accept connection requests from the host, but cannot initiate connection requests. After being connected to a host device, it can send and receive data through the host device.

Bluetooth devices can interact with each other, when they interact, the Bluetooth device in the main mode searches for nearby devices. While a connection is established, they can exchange data. For example, when a mobile phone exchanges data with ESP22, the mobile phone is usually in master mode and the ESP22 is in slave mode.

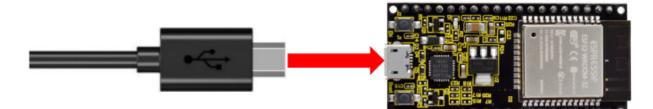


Master

Slave

# 3.Wiring Diagram

We can use a USB cable to connect ESP22 mainboard to the USB port on a computer.



### 4.Test Code

```
Project_22.1_Classic_Bluetooth | Arduino 1.8.16
                                                                             \times
File Edit Sketch Tools Help
       +
             +
   Ð
 Project_22.1_Classic_Bluetooth
Α
/*
 * Filename : Classic Bluetooth--SerialToSerialBT
 * Description : ESP32 communicates with the phone by bluetooth and print phone's data vi
 * Auther : http//www.keyestudio.com
*/
#include "BluetoothSerial.h"
BluetoothSerial SerialBT;
String buffer;
void setup() {
  Serial.begin(115200);
  SerialBT.begin("ESP32test"); //Bluetooth device name
  Serial.println("\nThe device started, now you can pair it with bluetooth!");
ł
void loop() {
  if (Serial.available()) {
<
                                                                             >
Invalid library found in C:\Users\Administrator\Desktop\Arduino\libraries\DHT: no headers
Invalid library found in C:\Users\Administrator\Desktop\Arduino\libraries\examples: no he
Invalid library found in C:\Users\Administrator\Desktop\Arduino\libraries\TM1650: no head 🗹
<
                                        ESP32 Wrover Module, Default, QIO, 80MHz, 921600, None on COM3
```

```
//*
    * Filename : Classic Bluetooth--SerialToSerialBT
    * Description : ESP32 communicates with the phone by bluetooth and print phone's data_
    .via a serial port
    * Auther : http//www.keyestudio.com
    */
    #include "BluetoothSerial.h"
BluetoothSerial SerialBT;
String buffer;
void setup() {
    Serial.begin(115200);
    SerialBT.begin("ESP32test"); //Bluetooth device name
    Serial.println("\nThe device started, now you can pair it with bluetooth!");
}
void loop() {
```

### 6.Test Result

Compile and upload the code to the ESP22. After uploading successfully, we will use a USB cable to power on. Open the serial monitor and set the baud rate to **115200**.

When you see the serial monitor prints out the character string as below, it indicates that the Bluetooth of ESP22 is ready and waiting for connection with a phone. (If open the serial monitor and set the baud rate to 115200, the information is not displayed, please press the RESET button of the ESP22)

RESET		000	<b>-</b> BO	ОТ		
🐵 сомз				—		$\times$
					5	Send
ets Jun 8 2016 00:22:57						^
<pre>rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH configsip: 0, SPIWP:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,H mode:DIO, clock div:1 load:0x3fff0018,len:4 load:0x3fff001c,len:1496 load:0x40078000,len:8596 load:0x40080400,len:6980 entry 0x400806f4</pre>	_ ld_drv:0x	00,wp_drv:	0x00			
The device started, now you can pair it with blue	tooth!					
						*
Autoscroll 🗌 Show timestamp	Newline	~	115200 baud	~ (	lear out	tput

Make sure that the Bluetooth of your phone has been turned on and "Serial Bluetooth Terminal" has been installed.



Click"Search"search for the nearby Bluetooth and select to connect the"ESP22 test".

4611 CMCC	1	11:24	* 🛈 🌲 🖽	<b>4</b> G 93%
<	Blu	etooth	ı	
Bluetoo	oth			-•
Renam MyBlueto	e this device			>
Paired de	vices			
Ō	ESP32test			ູ່ໃໃ
Available	devices			
*	4E:17:CF:AA	:84:38		
With the E devices.	Bluetooth enable, this	device wi	ll be visible to	nearby

Turn on software APP, click the left of the terminal. Select "Devices" .

toncc	11:25	* 🛈 🛸 🖪	D 4G 9	3% 🔲	the CMCC	11:25	\$ °O \$ E	<b>4</b> G 9	93% 🔲
≡ Termi	nal	••	ī	÷				Î	
					Serial Bluetoot	h Terminal			
					Terminal				
					Devices				
					Settings				
					Info				

Select ESP22test in classic Bluetooth mode, and a successful connecting prompt will appear as shown below.

← Devices	٠						
BLUETOOTH CLASSIC	BLUETOOTH LE	11:25:41.094 Connecting to ESP32test 11:25:41.195 Connected					
ESP32test C4:4F:33:22:B6:3B		M1	M2	М3	M4	M5	M6
							>

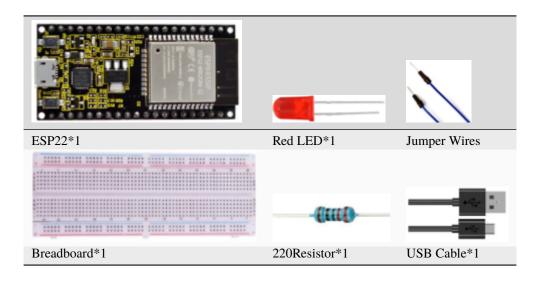
Data can be transferred between your phone and a computer via ESP22 now.

Send "Hello", When the computer receives it, which will reply with "Hi!".

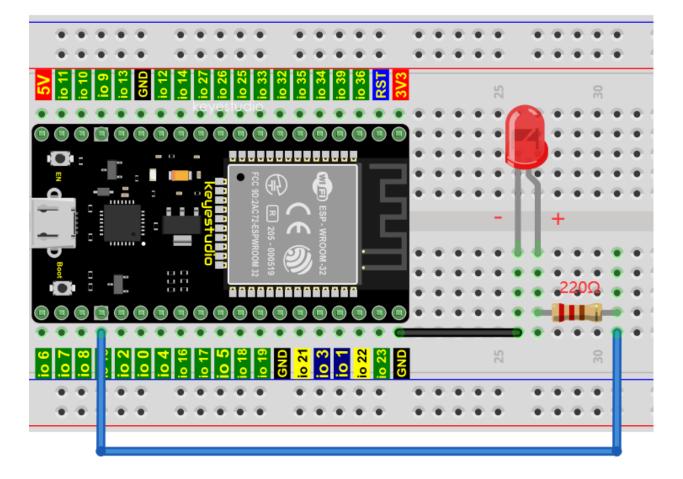
COM3	$ \Box$ $\times$
Hi!	Send
<pre>ets Jun 8 2016 00:22:57 rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT) configsip: 0, SPIWP:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00 mode:DIO, clock div:1 load:0x3fff0018,len:4 load:0x3fff001c,len:1496 load:0x40078000,len:8596 load:0x40078000,len:6980 entry 0x400806f4 The device started, now you can pair it with bluetooth!</pre>	~
Hello!	
Autoscroll Show timestamp Newline V 115200 baud	✓ Clear output
Image: Second secon	
11:25:41.094 Connecting to ESP32test 11:25:41.195 Connected 11:26:11.913 Hello! 11:26:24.759 Hi!	
M1 M2 M3 M4 M5 M6	

# 5.23.2 Project 22.2Bluetooth Control LED

### 1.Components



## 2.Wiring Diagram



#### 3.Test Code

```
Project_22.2_Bluetooth_Control_LED | Arduino 1.8.16
                                                                            \times
File Edit Sketch Tools Help
    Ð
       ÷
             +
                                                                                 Ø
  Project_22.2_Bluetooth_Control_LED
*********
 /*
 * Filename : Bluetooth Control LED
 * Description : The phone controls esp32's led via bluetooth.
               When the phone sends "LED on," ESP32's LED lights turn on.
               When the phone sends "LED off," ESP32's LED lights turn off.
 * Auther
             : http//<u>www.keyestudio.com</u>
 */
 #include "BluetoothSerial.h"
 #include "string.h"
 #define LED 15
 BluetoothSerial SerialBT;
 char buffer[20];
 static int count = 0;
 void setup() {
  pinMode(LED, OUTPUT);
  SerialBT.begin("ESP32test"); //Bluetooth device name
  Serial.begin(115200);
  Serial.println("\nThe device started, now you can pair it with bluetooth!");
                                            ESP32 Wrover Module, Default, QIO, 80MHz, 921600, None on COM3
/*
* Filename : Bluetooth Control LED
 * Description : The phone controls esp32's led via bluetooth.
               When the phone sends "LED_on," ESP32's LED lights turn on.
               When the phone sends "LED_off," ESP32's LED lights turn off.
* Auther
              : http//www.keyestudio.com
*/
#include "BluetoothSerial.h"
#include "string.h"
#define LED 15
BluetoothSerial SerialBT;
char buffer[20];
static int count = 0;
void setup() {
 pinMode(LED, OUTPUT);
 SerialBT.begin("ESP32test"); //Bluetooth device name
 Serial.begin(115200);
```

(continues on next page)

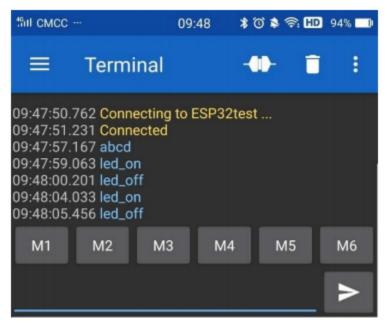
(continued from previous page)

```
Serial.println("\nThe device started, now you can pair it with bluetooth!");
}
void loop() {
 while(SerialBT.available())
 {
   buffer[count] = SerialBT.read();
   count++;
 }
 if(count>0){
   Serial.print(buffer);
   if(strncmp(buffer,"led_on",6)==0){
     digitalWrite(LED,HIGH);
   }
   if(strncmp(buffer,"led_off",7)==0){
     digitalWrite(LED,LOW);
   }
   count=0;
   memset(buffer,0,20);
 }
}
```

#### 4.Test Result

Compile and upload the code to the ESP22. After uploading successfully, we will use a USB cable to power on. The APP operation is the same as the project 22.1. To make the external LED on and off, simply change the sending content to "led\_on" and "led\_off".

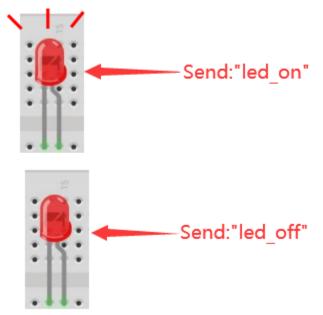
Moving the APP to send data:



The serial monitor will display as follows:

© COM4			_		×
					Send
abcd					^
led_on					
led_off					
led_on					
led_off					
					~
Autoscroll Show timestamp	Newline $\checkmark$	115200 baud	~	Clear	output

#### **LED** Circumstance



**Note:** If the sent content is not "led-on 'or" led-off ", the status of the LED will not change. If the LED is on, it remains on when irrelevant content is received; Conversely, if the LED is off, it continues to be off when irrelevant content is received.

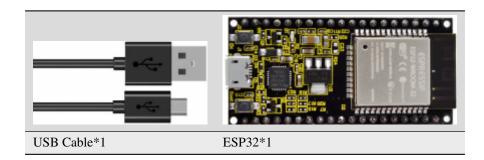
# 5.24 Project 23WiFi Station Mode

## 5.24.1 1.Introduction

ESP32 has three different WiFi operating modes: Station mode, AP mode and AP+Station mode. All WiFi programming projects must be configured with WiFi operating mode before using, otherwise WiFi cannot be used.

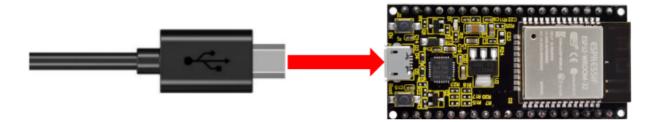
In this project, we are going to learn the WiFi Station mode of the ESP32.

## 5.24.2 2.Components



### 5.24.3 3.Wiring Diagram

Plug the ESP32 to the USB port of your PC



## 5.24.4 4.Component Knowledge

#### Station mode

When setting Station mode, the ESP32 is taken as a WiFi client. It can connect to the router network and communicate with other devices on the router via a WiFi connection.

As shown in the figure below, the PC and the router have been connected. If the ESP32 wants to communicate with the PC, the PC and the router need to be connected.



### 5.24.5 5.Test Code

```
Project_23_WiFi_Station_Mode | Arduino 1.8.16
                                                                             \times
File Edit Sketch Tools Help
  Project_23_WiFi_Station_Mode
/*
 * Filename : WiFi Station
 * Description : Connect to your router using ESP32
 * Auther : http//www.keyestudio.com
*/
#include <WiFi.h> //Include the WiFi Library header file of ESP32.
//Enter correct router name and password.
const char *ssid Router = "ChinaNet-2.4G-ODF0"; //Enter the router name
const char *password_Router = "ChinaNet@233"; //Enter the router password
void setup(){
 Serial.begin(115200);
  delay(2000);
  Serial.println("Setup start");
 WiFi.begin (ssid Router, password Router);//Set ESP32 in Station mode and connect it to yc
  Serial.println(String("Connecting to ")+ssid_Router);
//Check whether ESP32 has connected to router successfully every 0.5s.
<
                                                                                    >
                                        ESP32 Wrover Module, Default, QIO, 80MHz, 921600, None on COM3
```

Since WiFi names and passwords vary from place to place, thereby users need to enter the correct WiFi names and passwords in the box shown below before the program code runs.

```
Project_23_WiFi_Station_Mode | Arduino 1.8.16
                                                                           \times
File Edit Sketch Tools Help
             +
           +
    +
        Project_23_WiFi_Station_Mode
 //****
        /*
 * Filename : WiFi Station
 * Description : Connect to your router using ESP32 Enter the correct Router
 * Auther : http//<u>www.keyestudio.com</u>
                                                name and password.
 */
 #include <WiFi.h> //Include the WiFi Library header file of ESP32.
 //Enter correct router name and password.
                         = "ChinaNet-2.4G-ODFO"; //Enter the router name
 const char *ssid Router
 const char *password Router = "ChinaNet@233"; //Enter the router password
 void setup(){
  Serial.begin(115200);
  delay(2000);
  Serial.println("Setup start");
 <
                                                                                >
Done Saving.
                                           ESP32 Wrover Module, Default, QIO, 80MHz, 921600, None on COM3
/*
* Filename : WiFi Station
* Description : Connect to your router using ESP32
* Auther : http//www.keyestudio.com
*/
#include <WiFi.h> //Include the WiFi Library header file of ESP32.
//Enter correct router name and password.
const char *ssid_Router = "ChinaNet-2.4G-0DF0"; //Enter the router name
const char *password_Router = "ChinaNet@233"; //Enter the router password
void setup(){
 Serial.begin(115200);
 delay(2000);
 Serial.println("Setup start");
 WiFi.begin(ssid_Router, password_Router);//Set ESP32 in Station mode and connect it to_
\rightarrow your router.
 Serial.println(String("Connecting to ")+ssid_Router);
//Check whether ESP32 has connected to router successfully every 0.5s.
 while (WiFi.status() != WL_CONNECTED){
   delay(500);
   Serial.print(".");
```

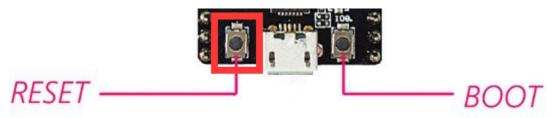
(continues on next page)

(continued from previous page)

### 5.24.6 6.Test Result

After making sure the router name and password are entered correctly, compile and upload the code to ESP32, open serial monitor and set baud rate to 115200.

When ESP32 successfully connects to "ssid\_Router", serial monitor will print out the IP address, then monitor will display as follows: (If open the serial monitor and set the baud rate to 115200, the information is not displayed, please press the RESET button of the ESP32)



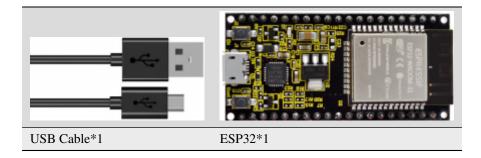
© COM20 -	- [	) ×
		Send
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00		<u>^</u>
mode:DIO, clock div:1		
load:0x3fff0018,len:4		
load:0x3fff001c,len:1496		
load:0x40078000,len:8596		
load:0x40080400,len:6980		
entry 0x400806f4		
Setup start		
Connecting to ChinaNet-2.4G-0DF0		
•••••		
Connected, IP address:		
192.168.1.108		
Setup End		
		~
🗹 Autoscroll 🗌 Show timestamp 🛛 Newline 🗸 115200 baud	~ C1	lear output

# 5.25 Project 24WiFi AP Mode

### 5.25.1 1.Introduction

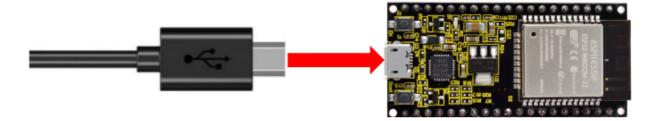
In this project, we are going to learn the WiFi AP mode of the ESP32.

### 5.25.2 2.Components



### 5.25.3 3.Wiring Diagram

Plug the ESP32 mainboard to the USB port of your PC



## 5.25.4 4.Component Knowledge

#### **AP Mode:**

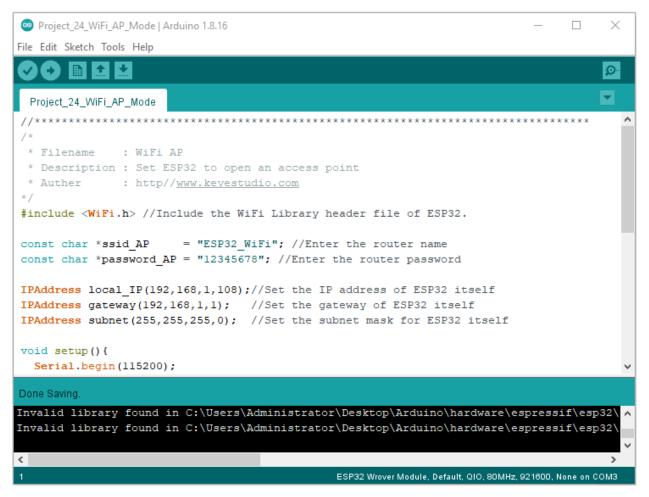
When setting AP mode, a hotspot network will be created, waiting for other WiFi devices to connect. As shown below;

Take the ESP32 as the hotspot, if a phone or PC needs to communicate with the ESP32, it must be connected to the ESP32's hotspot.

Communication is only possible after a connection is established via the ESP32.



### 5.25.5 5.Test Code



Before running the code , you can make any changes to the ESP32 AP name and password in the box as shown below, but in a default circumstance, it doesn't need to modify.



(continued from previous page)

```
Serial.println("Setting soft-AP ... ");
boolean result = WiFi.softAP(ssid_AP, password_AP);
if(result){
   Serial.println("Ready");
   Serial.println(String("Soft-AP IP address = ") + WiFi.softAPIP().toString());
   Serial.println(String("MAC address = ") + WiFi.softAPmacAddress().c_str());
}else{
   Serial.println("Failed!");
}
Serial.println("Setup End");
}
void loop() {
}
```

### 5.25.6 6.Test Result

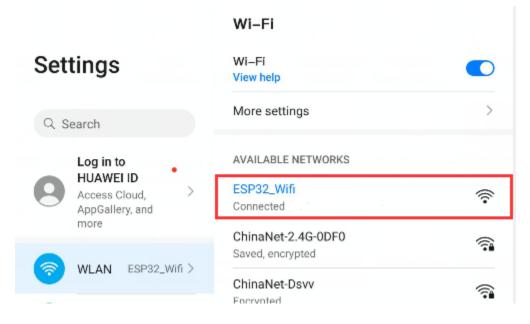
Enter the ESP32 AP name and password correctly, compile and upload the code to ESP32, open the serial monitor and set the baud rate to **115200**, then monitor will display as follows:

(If open the serial monitor and set the baud rate to 115200, the information is not displayed, please press the RESET button of the ESP32)



© COM20			$\times$
			Send
configsip: 0, SPIWP:0xee			^
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00			
mode:DIO, clock div:1			
load:0x3fff0018,len:4			
load:0x3fff001c,len:1496			
load:0x40078000,len:8596			
load:0x40080400,len:6980			
entry 0x400806f4			
Setting soft-AP configuration			
Ready			
Setting soft-AP			
Ready			
Soft-AP IP address = 192.168.1.108			
MAC address = 58:BF:25:8A:19:D1			
Setup End			
			~
Autoscroll Show timestamp Newline ~ 115200 baud	~	Clear o	utput

When observing the printed information of the serial monitor, turn on the WiFi scanning function of the mobile phone, you can see the ssid\_AP on ESP32, which is dubbed "ESP32\_Wifi" in this code. You can connect to it either by typing the password "12245678" or by modifying the code to change its AP name and password.

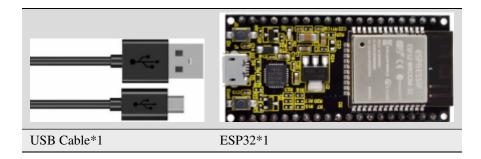


# 5.26 Project 25WiFi Station+AP Mode

## 5.26.1 1.Introduction

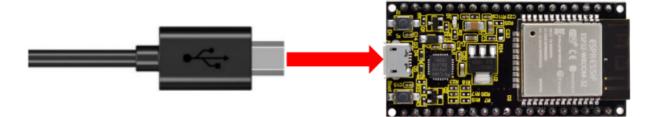
In this project, we are going to learn the AP+Station mode of the ESP32.

## 5.26.2 2.Components



### 5.26.3 3.Wiring Diagram

Plug the ESP32 mainboard to the USB port of your PC



### 5.26.4 4.Component Knowledge

#### **AP+Station mode:**

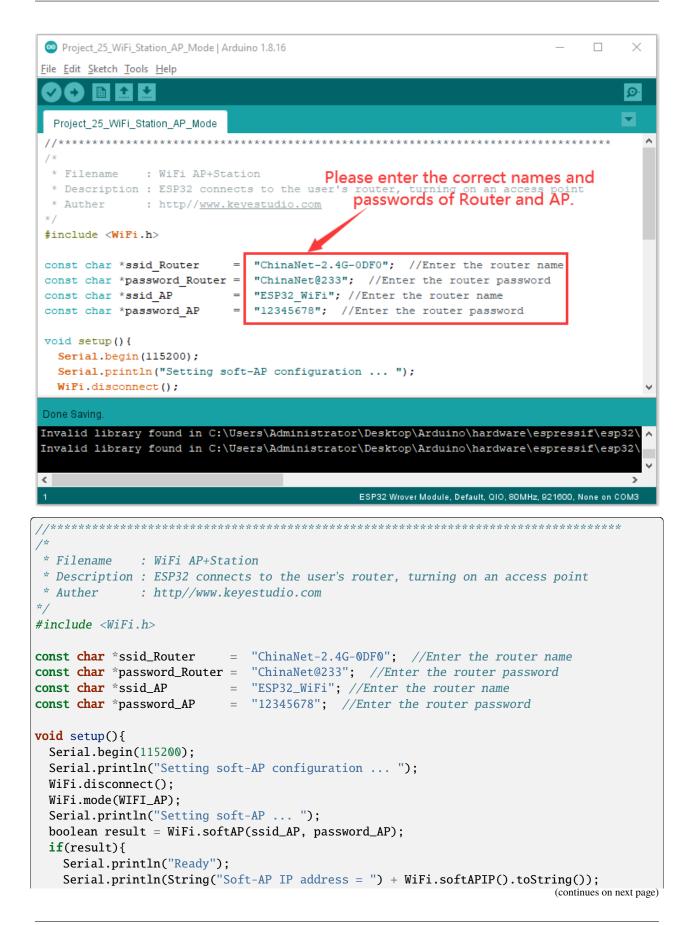
In addition to the AP mode and the Station mode, **AP+Station mode** can be used at the same time. Turn on the Station mode of the ESP32, connect it to the router network, and it can communicate with the Internet through the router. Then turn on the AP mode to create a hotspot network.

Other WiFi devices can be connected to the router network or the hotspot network to communicate with the ESP32.

## 5.26.5 5.Test Code

🐵 Project_25_WiFi_Station_AP_Mode   Arduino 1.8.16 -	$\times$
<u>File Edit Sketch Tools H</u> elp	
	ø
Project_25_WiFi_Station_AP_Mode	
//*************************************	^
/*	
* Filename : WiFi AP+Station	
* Description : ESP32 connects to the user's router, turning on an access point	
* Auther : http// <u>www.keyestudio.com</u> */	
<pre>#include <wifi.h></wifi.h></pre>	
<pre>const char *ssid_Router = "ChinaNet-2.4G-ODFO"; //Enter the router name const char *password_Router = "ChinaNet@233"; //Enter the router password const char *ssid_AP = "ESP32_WiFi"; //Enter the router name const char *password_AP = "12345678"; //Enter the router password void setup(){ Serial.begin(115200); Serial.println("Setting soft-AP configuration ");</pre>	
WiFi.disconnect();	~
Done Saving.	
1 ESP32 Wrover Module, Default, QIO, 80MHz, 921600, None on	сомз

Before running the code, you need to modify the ssid\_Router, password\_Router, ssid\_AP and password\_AP, as shown in the box below:



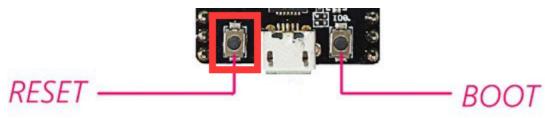
(continued from previous page)

```
Serial.println(String("MAC address = ") + WiFi.softAPmacAddress().c_str());
 }else{
   Serial.println("Failed!");
 }
 Serial.println("\nSetting Station configuration ... ");
 WiFi.begin(ssid_Router, password_Router);
 Serial.println(String("Connecting to ")+ ssid_Router);
 while (WiFi.status() != WL_CONNECTED){
   delay(500);
   Serial.print(".");
 }
 Serial.println("\nConnected, IP address: ");
 Serial.println(WiFi.localIP());
 Serial.println("Setup End");
}
void loop() {
}
```

6.Test Result

Ensure that the code in the program has been modified correctly, compile and upload the code to the ESP32. After uploading successfullywe will use a USB cable to power on.

Open the serial monitor and set the baud rate to 115200, then monitor will display as follows: (If open the serial monitor and set the baud rate to 115200, the information is not displayed, please press the RESET button of the ESP32)



💿 COM3				_		$\times$
						Send
load:0x40078000,len:8596						^
load:0x40080400,len:6980						
entry 0x400806f4						
Setting soft-AP configuration						
Setting soft-AP						
Ready						
Soft-AP IP address = 192.168.4.1						
MAC address = 58:BF:25:8A:19:D1						
Setting Station configuration						
Connecting to ChinaNet-2.4G-0DF0						
Connected, IP address:						
192.168.1.157						
Setup End						
						Y
Autoscroll Show timestamp	[	Newline $\checkmark$	115200 baud	$\sim$	Clear o	utput

Open the WiFi scanning function of the mobile phone, you can see the ssid\_AP.

	Wi–Fi	
Settings	Wi–Fi View help	
Q Search	More settings	>
Log in to	AVAILABLE NETWORKS	
Access Cloud, > AppGallery, and	ESP32_Wifi Connected	Ŵ
more	ChinaNet-2.4G-0DF0 Saved, encrypted	() <b>:</b>
WLAN ESP32_Wifi >	ChinaNet-Dsvv	6
	Encrynted	•

#### CHAPTER

# **GETTING STARTED WITH PYTHON**

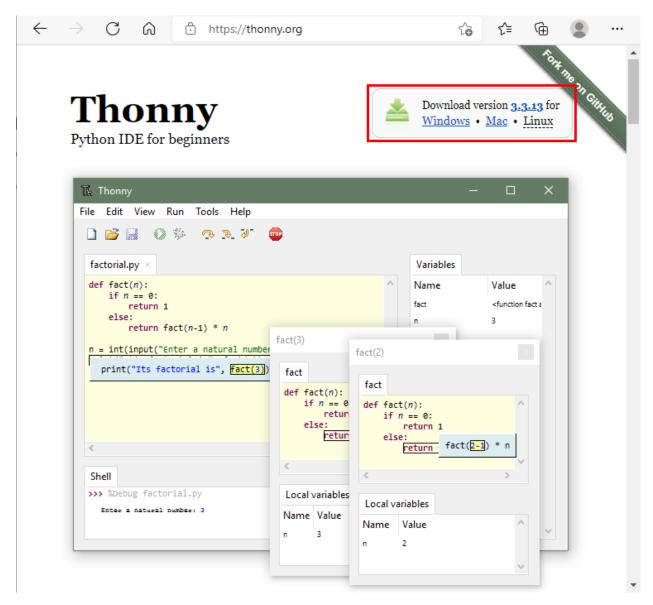
1.Install Thonny

Thonny is a free and open source software platform with small size, simple interface, simple operation and rich functions. It is a Python IDE suitable for beginners. In this tutorial, we use this IDE to develop a ESP32. Thonny supports multiple operating systems including Windows, Mac OS, Linux.

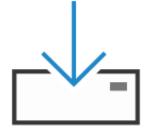
2. Download Thonny

1) Enter the websitehttps://thonny.org to download the latest version of Thonny.

(2)Thonny open-source code libraryhttps://github.com/thonny/thonny.

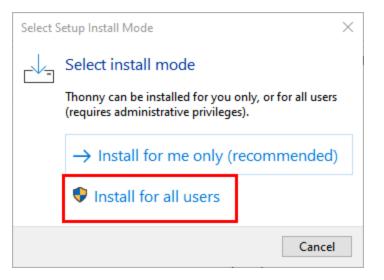


1. The downloaded Thonny icon is as follow:



thonny-3.3.13.exe

2.Double-click"thonny-3.3.13.exe" and	select	install	mode.	You	can	choose
→ Install for me only (recomm	ended)					



3. You can also keep selecting Next to finish the installation.



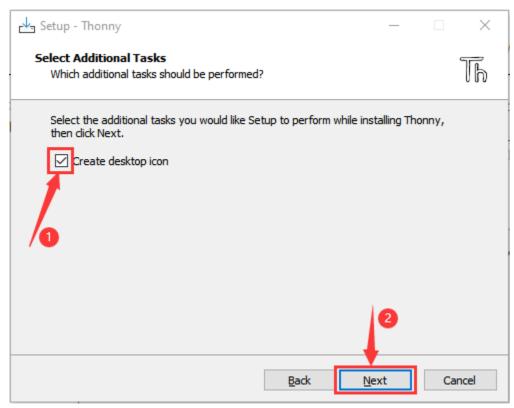
License Agreement		<u> </u>
Please read the following important information before continuing.		Uh
Please read the following License Agreement. You must accept the terms agreement before continuing with the installation.	s of this	
The MIT License (MIT)		^
Copyright (c) 2020 Aivar Annamaa		
Permission is hereby granted, free of charge, to any person obtaining a this software and associated documentation files (the "Software"), to de Software without restriction, including without limitation the rights to use	eal in the	
modify, merge, publish, distribute, sublicense, and/or sell copies of the s		~
● I accept the agreement		
○ I <u>d</u> o not accept the agreement		

4.If you want to change the route of installing Thonnyjust click"Browse..."to select a new route and click OK.

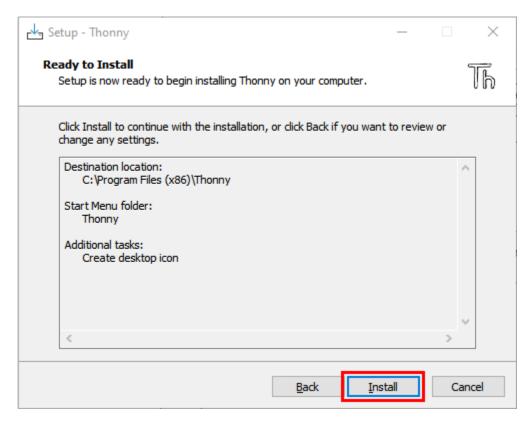
Setup - Thonny	_		×
Select Destination Location			
Where should Thonny be installed?			lh
Setup will install Thonny into the following folder.			
To continue, click Next. If you would like to select a different folder,	click Bro	wse.	
C:\Program Files (x86)\Thonny	Br	owse	
2 At least 99.5 MB of free disk space is required.		0	
At least 53.5 Mb of thee disk space is required.			
<u>B</u> ack <u>N</u> e	ext	Can	cel

🛃 Setup - Thonny	_		×
Select Start Menu Folder Where should Setup place the program's shortcuts?			Th
To continue, click Next. If you would like to select a different folder,			
Thonny		owse	
<u>B</u> ack Nex	t	Car	ncel

5. Click Create desktop icon, you will view Thonny on your desktop.



6.Click"Install"



#### 7. Wait for a while but don't click \*\*Cancel

Setup - Thonny	_		$\times$
<b>Installing</b> Please wait while Setup installs Thonny on your computer.			Th
Extracting files C:\Program Files (x86)\Thonny\Lib\idlelib\rpc.py			
		Ca	ancel

#### G. Click\*\*"Finish"\*\*

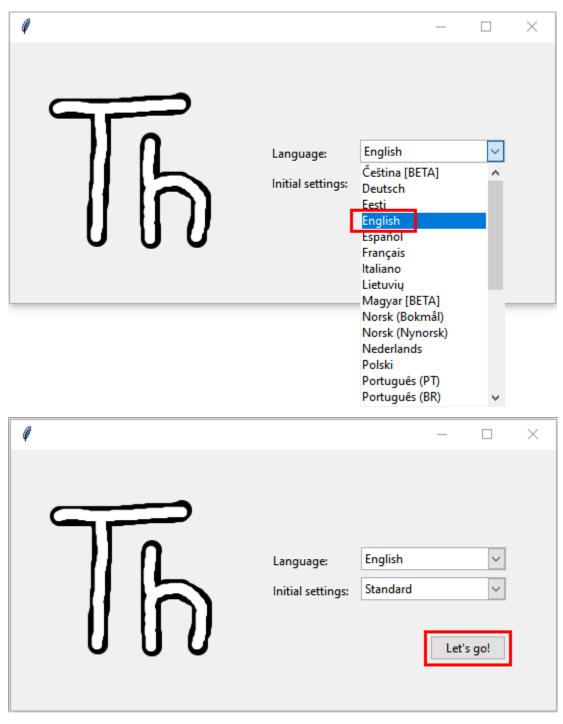




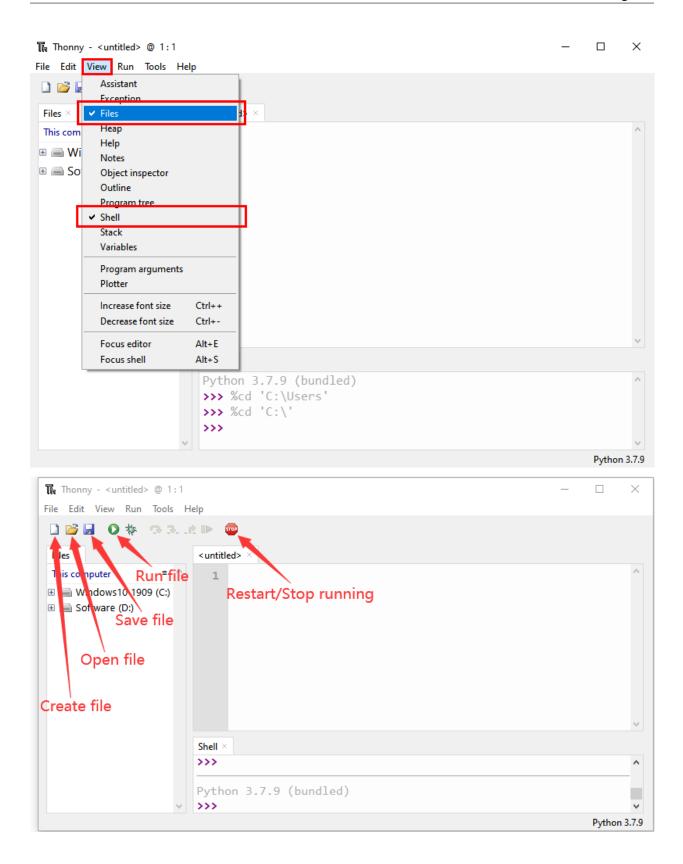
#### 2.Basic Setting

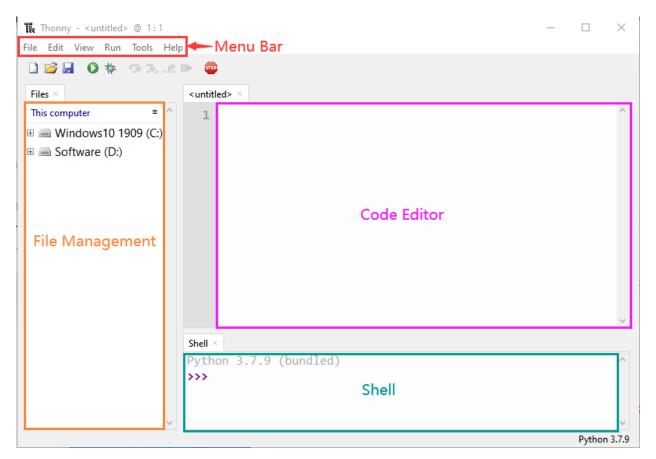
Double-click Thonny, choose language and initial settings and click Let's go

0			—		$\times$
Th	Language: Initial settings:	English Standard	Let's	<ul><li>✓</li><li>✓</li><li>; go!</li></ul>	



Click"View"→"File"and"Shell"





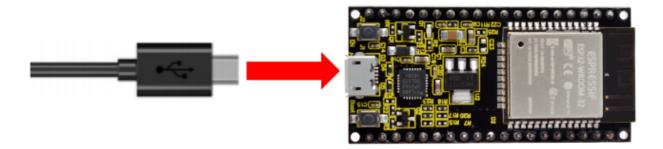
#### Install the CP2102 driver

Before using the Thonny, we need to install the CP2102 driver in the computer.

#### Windows system

Check if the CP2102 driver has been installed

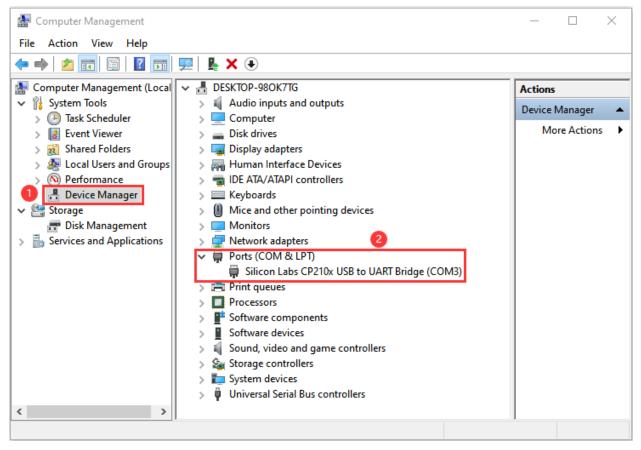
1. Interface the ESP32 with your PC with a USB cable



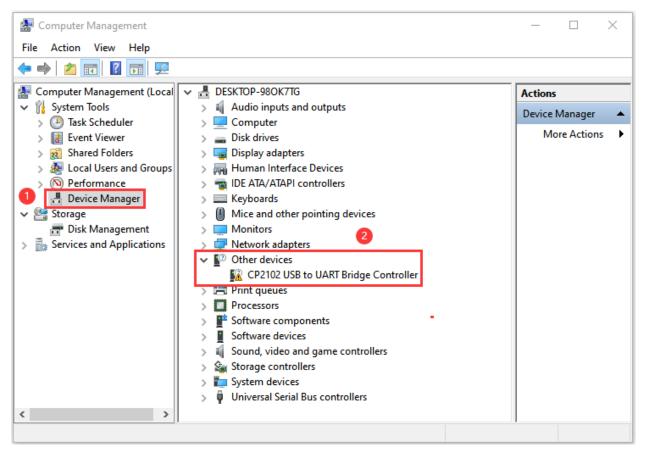
2. Click"This PC"and right-click Manage"

Carlo Bin		
	<b>Open</b> Pin to Quick access	
0.025	Manage Pin to Start	
Contre	Map network drive Disconnect network drive	
Pane	Create shortcut Delete	
a <sup>l h</sup> Thonny	Properties	

3.Click"Device Manager", if the CP2102 driver has been installed Silicon Labs CP210x USB to UART Bridge(COMx) will be shown.



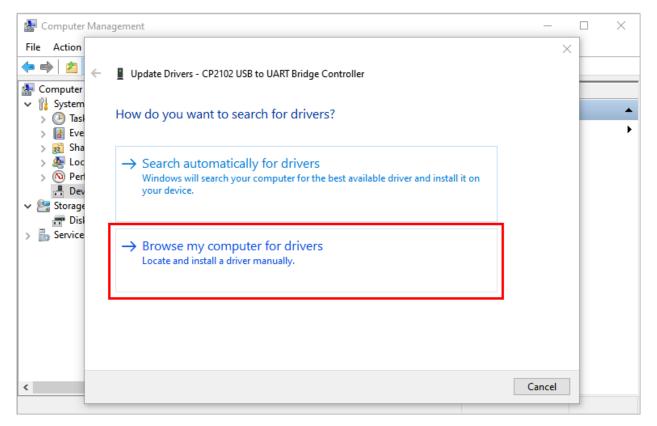
If the CP2102 has not been installed



Click"CP2102USB to UART Bridge Controller" and Update driver".

🛃 Computer Management			_		×
File Action View Help					
🗢 🔿 🙍 🗊 🗐 🗐 🖉 🗾 😕	٢				
🚪 Computer Management (Local 🔽 🛃 DESk	CTOP-98OK7TG		Actions	;	
✓				Manager	
> 🕑 Task Scheduler > 💻 Computer				-	
> 🛃 Event Viewer 🛛 🔰 ک 🚃 🛛	Disk drives		Mo	ore Action	s
> 👸 Shared Folders 🛛 🔰 🙀 🛛	Display adapters				
> 🜆 Local Users and Groups 🔰 🛛 کې 🐺 H	> 👰 Local Users and Groups 🛛 > 👼 Human Interface Devices				
> 🔞 Performance 🛛 🔰 📷 🛛	> 🔊 Performance > 📷 IDE ATA/ATAPI controllers				
📇 Device Manager 🔰 🛌 K	🕂 Device Manager > 🔤 Keyboards				
🗸 🔄 Storage 🛛 🔰 🗎 N	lice and other pointing devices				
	> 🛄 Monitors				
	ns 🔰 > 💭 Network adapters				
	Other devices				
-	CP2102 USB to UART Bridge Controller	Update driver			
> 🖻 P	rint queues	Disable device			
	rocessors	Uninstall device			
	oftware components	Uninstall device	2		
	oftware devices	Scan for hardwa	are chang	es	
	ound, video and game controllers		-	·	
	torage controllers	Properties			
	ystem devices				_
	Iniversal Serial Bus controllers				
< >>					
aunches the Update Driver Wizard for the selecte	d device.				

Click"Browse my computer for drivers ".



Click Browse... to choose CP210x\_6.7.4 ("4. Python Tutorial\1.Development Environment Configuration\CP2102 Driver File-Windows") and click Next

🛃 Computer Management	- 🗆 X
File Action	×
←	2 USB to UART Bridge Controller
✓      ✓      ✓      ✓      ✓      System     ✓      ✓      ✓      ✓      ✓      System     ✓      ✓	n your computer
> 👸 Sha Search for drivers in this lo	
>   Per  Get started with Python(  A  De  V   Storage	CP2102 Driver File-Windows\CP210x_6.7.4 V Browse
> Disl	
	m a list of available drivers on my computer ailable drivers compatible with the device, and all drivers in the e device.
٤	Next Cancel

### The CP2102 driver will be installed

Compute	× □ ×
File Action	
Compute V Syster Windows has successfully updated your drivers	
Control Ta:     Contro Ta:     Control Ta:     Control Ta:     Control Ta:     Control Ta	•
> 😥 Sh > 🕭 Lo > 🔞 Pe 🤍 Silicon Labs CP210x USB to UART Bridge	
E De	
> Di: > B Servic	
<	Close
Somputer Management	x
E Computer Management File Action View Help	– – ×
	X
File       Action       View       Help         (= =)       (2)       (2)       (2)       (2)       (2)         (a)       (2)       (2)       (2)       (2)       (2)       (2)         (a)       (a)       (a)       (a)       (a)       (a)       (a)       (a)         (a)       (b)       (b)       (b)       (b)       (b)       (b)       (c)         (a)       (b)       (b)       (b)       (b)       (b)       (c)       (c)         (b)       (c)	×
File       Action       View       Help         Image: Second sec	
File       Action       View       Help         Image: Second state of the second	Actions Device Manager
File       Action       View       Help         Image: Second state of the second	Actions
File       Action       View       Help         Image: System Tools       Image: System Tools       Image: System Tools       Image: System Tools         Image: System Tools       Image: System Tools       Image: System Tools       Image: System Tools       Image: System Tools         Image: System Tools<	Actions Device Manager
File       Action       View       Help         Image: Solution of the stress of the	Actions Device Manager
File       Action       View       Help         Image: Solution of the system of the	Actions Device Manager
File       Action       View       Help         Image: Solution of the stress of the	Actions Device Manager
File       Action       View       Help         Image: Solution of the state	Actions Device Manager
File       Action       View       Help         Image: Storage       Image: Storage       Image: Storage       Image: Storage       Image: Storage         Image: Storage       Image: Storage       Image: Storage       Image: Storage       Image: Storage       Image: Storage         Image: Storage<	Actions Device Manager
File       Action       View       Help         Image: Storage       Image: Storage       Image: Storage       Image: Storage       Image: Storage         Image: Storage       Imag	Actions Device Manager
File       Action       View       Help            ←          ←          ←          ←          ←            ←          ←          ←          ←          ←            ←          ←          ←          ←          ←            ←          ←          ←          ←          ←            ←          ←          ←          ←          ←          ←              ←	Actions Device Manager
File       Action       View       Help            ←          ←          ←          ←          ←            ←          ←          ←          ←          ←            ←          ←          ←          ←          ←            ←          ←          ←          ←          ←            ←          ←          ←          ←          ←          ←              ←                    ←          ←	Actions Device Manager
File       Action       View       Help	Actions Device Manager
File       Action       View       Help	Actions Device Manager
File       Action       View       Help <ul> <li></li></ul>	Actions Device Manager
File       Action       View       Help <ul> <li></li></ul>	Actions Device Manager
File       Action       View       Help <ul> <li></li></ul>	Actions Device Manager
File       Action       View       Help <ul> <li></li></ul>	Actions Device Manager

MAC System

#### Download link for CP2102CP2102-Driver-File-MAC.zip

Download MacOS version

#### Download for WinCE

Platform	Software	Release Notes
WinCE 6.0 (2.1)	Download VCP (276 KB)	Download WinCE 6.0 Revision History
🙀 WinCE 5.0 (2.1)	Download VCP (271 KB)	Download WinCE 5.0 Revision History

# Download for Macintosh OSX (v5.3.5)

Platform	Software	Release Notes
🗳 Mac OSX	Download VCP (832 KB)	Download Mac VCP Revision History

### **Download for Linux**

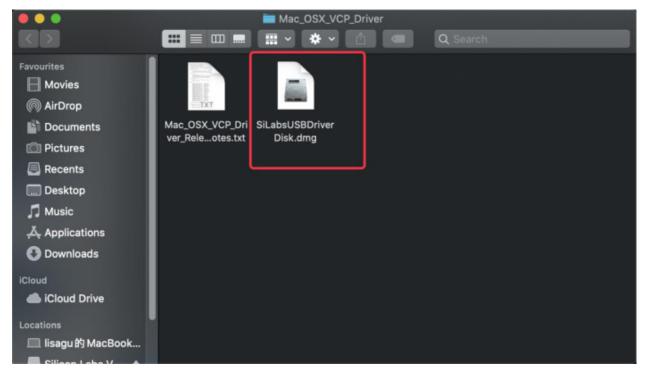
Platform	Software	Release Notes
∆ Linux 3.x.x and 4.x.x	Download VCP (10.0 KB)	Download Linux 3.x.x and 4.x.x VCP Revision History
∆ Linux 2.6.x	Download VCP (10.2 KB)	Download Linux 2.6.x VCP Revision History

\*Note: The Linux 3.x.x and 4.x.x version of the driver is maintained in the current Linux 3.x.x and 4.x.x tree at www.kernel.org.

#### Unzip the downloaded package



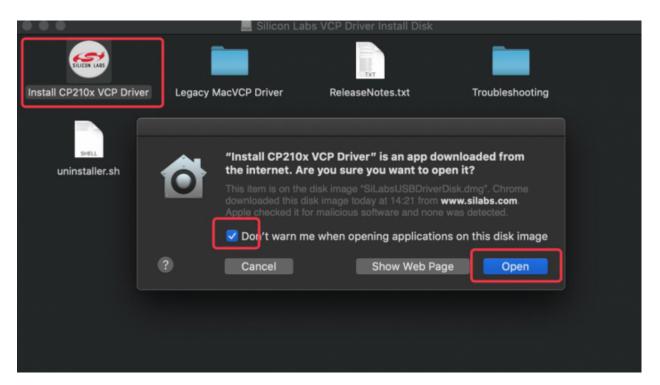
Open the folder and double-click "SiLabsUSBDriverDisk.dmg" file



### Then you can see the following file

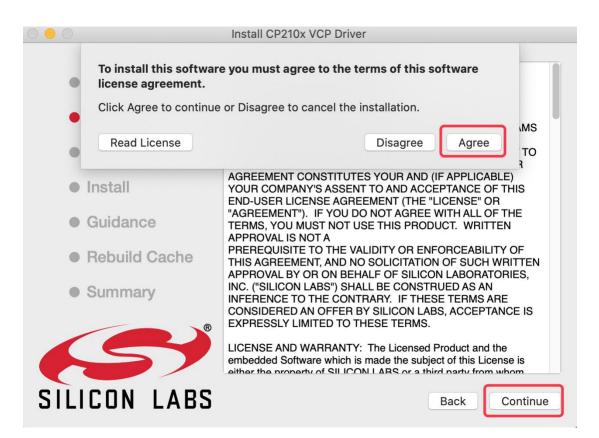
	📃 Silicon Labs	VCP Driver Install Disk		
STUEDR LARS		TXT		
Install CP210x VCP Driver	Legacy MacVCP Driver	ReleaseNotes.txt	Troubleshooting	
SHELL				
uninstaller.sh				

Double-click\*\*"Install CP210x VCP Driver"tap\*\*"**Don't warn me when opening application on this disk image**"and click"**Open**"



#### Click"Continue"

	Install CP210x VCP Driver
Introduction	You will be guided through the steps necessary to install
License	the virtual COM port driver for the Silicon Labs CP210x USB to UART Bridge.
Info	
Install	
Guidance	
Rebuild Cache	
Summary	
(S)	
SILICON LABS	Back Continue
Click"Agree"then tap"Continue"	



Click"Continue"then input your user password

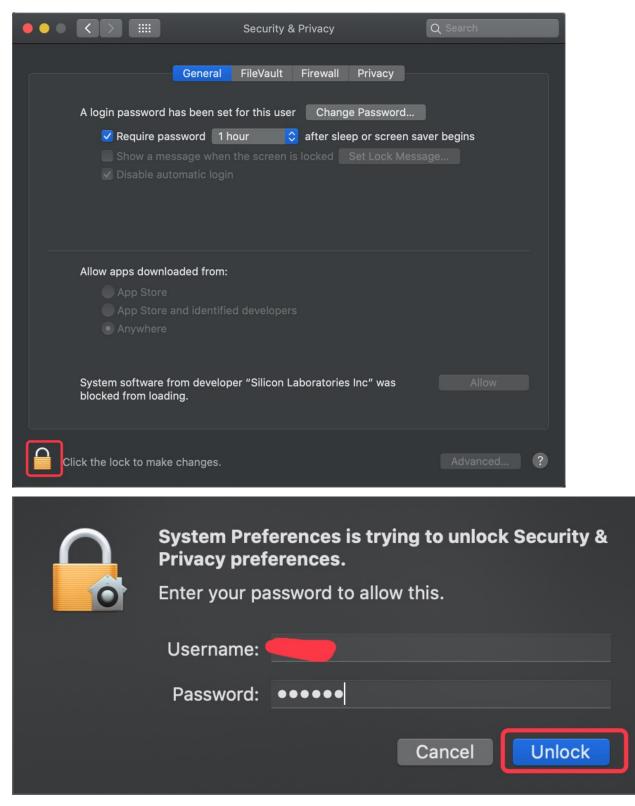
	Install CP210x VCP Driver
Introduction	To Be Installed: Version 5.3.5
License	Currently Installed: None
<ul> <li>Info</li> </ul>	Version 5.3.5 will be installed in /Library/Extensions/.
Install	
Guidance	
Rebuild Cache	
Summary	
S	You will be prompted to enter your password.
SILICON LABS	Back Continue

Install CP210x VCP Driver is trying to install a new helper tool. Enter your password to allow this.
Username:
Password:
Cancel Install Helper

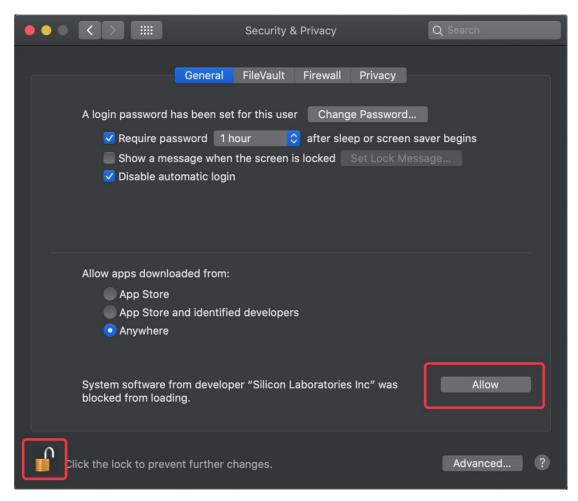
Select"Select Open Security Preferences"

	System Extension Blocked
0	A program tried to load new system extension(s) signed by "Silicon Laboratories Inc", which will be incompatible with a future version of macOS. If you want to enable these extensions, open Security & Privacy System Preferences.
	Open Security Preferences OK
	incompatible with a future version of macOS. If you want to enable these extensions, open Security & Privacy System Preferences.

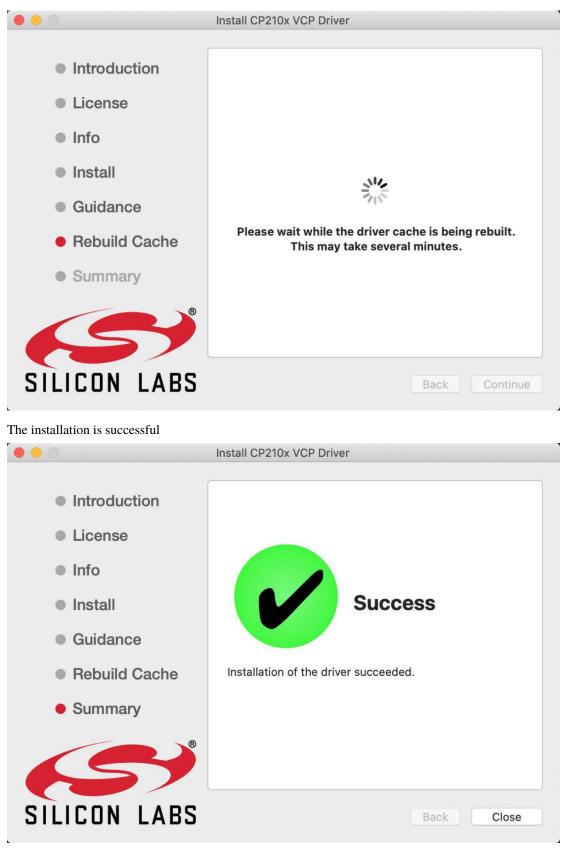
Click on security lock and enter your user password to authorize.



When you see that the lock is opened, click "Allow".



Return to the installation interface and wait for the installation as prompted.



**3.Burn Micropython firmware** 

To run a Python program on the ESP32 board, we need to burn the firmware to the ESP32 board first.

Download Micropython firmware

microPython websitehttp://micropython.org/

ESP32 firmwarehttps://micropython.org/download/esp32/

# Firmware

Releases

v1.18 (2022-01-17) .bin [.elf] [.map] [Release notes] (latest) v1.17 (2021-09-02) .bin [.elf] [.map] [Release notes] v1.16 (2021-06-23) .bin [.elf] [.map] [Release notes] v1.15 (2021-04-18) .bin [.elf] [.map] [Release notes] v1.14 (2021-02-02) .bin [.elf] [.map] [Release notes] v1.13 (2020-09-02) .bin [.elf] [.map] [Release notes] v1.12 (2019-12-20) .bin [.elf] [.map] [Release notes]

Nightly builds

```
v1.18-121-gd8a7bf83c (2022-02-10) .bin [.elf] [.map]
v1.18-107-gaca40127b (2022-02-09) .bin [.elf] [.map]
v1.18-105-gada836b83 (2022-02-08) .bin [.elf] [.map]
v1.18-103-g6f7d6c567 (2022-02-08) .bin [.elf] [.map]
```

# Firmware (Compiled with IDF 3.x)

Releases

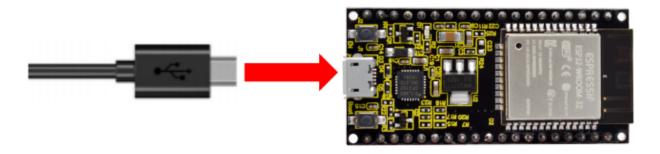
v1.14 (2021-02-02) .bin [.elf] [.map] [Release notes] (latest) v1.13 (2020-09-02) .bin [.elf] [.map] [Release notes] v1.12 (2019-12-20) .bin [.elf] [.map] [Release notes] v1.11 (2019-05-29) .bin [.elf] [.map] [Release notes] v1.10 (2019-01-25) .bin [.elf] [.map] [Release notes] v1.9.4 (2018-05-11) .bin [.elf] [.map] [Release notes]

The firmware we use\*\*esp32-20210902-v1.17.bin

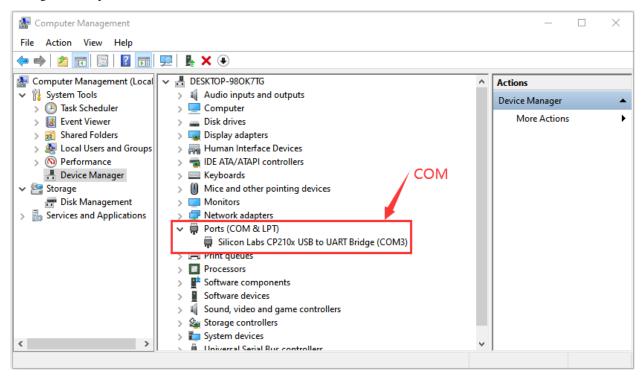
FirmwareDownload Python Firmware

Burn the Micropython firmware

Connect the ESP32 to your PC with a USB cable



Make sure the driver has been installed successfully and the COM port can be identified correctly. Open Device Manager and expand "Ports".



Open Thonnyclick"run"and"Select interpreter..."

🏗 Thonny - <ur< th=""><th>ntitled&gt; @ 1:1</th><th></th><th>- 0</th><th>×</th></ur<>	ntitled> @ 1:1		- 0	×
File Edit View	Run Tools Help			
	Select interpreter			
Files ×	Run current script Debug current script (nicer)	F5 Ctrl+F5		
This computer	Debug current script (faster)	Shift+F5		^
🗉 📾 Window:	🐚 Debug current script (birdseye)	Ctrl+Shift+B		
🗉 📾 Software	Step over	F6		
	🎨 Step into	F7		
	📌 Step out			
	🐌 Resume	F8		
	📫 Run to cursor	Ctrl+F8		
	Step back	Ctrl+B		
	Run current script in terminal	Ctrl+T		
	Dock user windows Pygame Zero mode			~
	5 Stop/Restart backend	Ctrl+F2		^
	Interrupt execution	Ctrl+C	to COM22: port not found	
	Send EOF / Soft reboot Disconnect	Ctrl+D	d or disconnected. Use 'Stop/Restart' to restart.	
			D:\pyth	on.exe

Select Micropython (ESP32) and Silicon Labs CP210x USB to UART Bridge(COM3) and click "Install or update firmware".

🏹 Thonn	y options								×
Which	interpreter o	r device	Theme & Font should Thonny u			Shell	Assistant	0	
Deta Con (loo If yo Con If yo (imp	necting via U nect your de k for your de ou can't find i necting via V our device sup	ISB cable vice to th vice nam t, you m vebREPL oports W setup), c	: e computer and e, "USB Serial" o ay need to instal (EXPERIMENTAL ebREPL, first con onnect your con	r "UART"). I proper USB driv .): inect via serial, r	ver first. nake sure V	VebREPL			
	or WebREPI on Labs CP2	10x USB t	o UART Bridge (	COM3)					~
							Instal	3 Il or update f	irmware
								ОК	Cancel

Select"Silicon Labs CP210x USB to UART Bridge(COM3)"click "Browse…"and choose the firmware **esp32-20210902-v1.17.bin.** Check"Erase flash before installing"and"Flash mode"then click"Install".

If you haven't downloaded the firmware, please click on the link to downloadDownload Python Firmware

(NoteIf you fail to install the firmwarepress the Boot button on the ESP32 board and click"Install"

	~	
ESP32 firmware installer	×	
This dialog allows installing or updating firmware on ESP32 using the most common set If you need to set other options, then please use 'esptool' on the command line. Note that there are many variants of MicroPython for ESP devices. If the firmware provid at micropython.org/download doesn't work for your device, then there may exist better alternatives look around in your device's documentation or at MicroPython forum.	led	
Port Silicon Labs CP210x USB to UART Bridge (COM3) V Reload		
Firmware rted with Python/Python_Firmware/esp32-20220117-v1.18.bin Browse		
Flash mode From image file (keep) () Quad I/O (qio)		
O Dual I/O (dio) O Dual Output (dout)		
Erase flash before installing		
	ncel	
Install	ncel	×
Install Car	ncel	×
Install Car		×
Install Car $\overline{\mathbf{k}}$ Open $\leftarrow \rightarrow \checkmark \uparrow \square \ll 1. \text{ Development } Python_Firmware \lor O \checkmark Search$	ch Python_Firmware	
Install Car $\overline{\mathbf{k}}$ Open $\leftarrow \rightarrow \checkmark \uparrow \square \ll 1. Development > Python_Firmware \checkmark \overline{\mathbf{O}} SearceOrganize \checkmark New folder$	ch Python_Firmware	?
Install Car The Open ← → ~ ↑	ch Python_Firmware	?
Install Car	ch Python_Firmware	•

🕞 ESP32 firr	nware installer	$\times$
If you need Note that t at micropy	allows installing or updating firmware on ESP32 using the most common settings to set other options, then please use 'esptool' on the command line. there are many variants of MicroPython for ESP devices. If the firmware provided thon.org/download doesn't work for your device, then there may exist better s look around in your device's documentation or at MicroPython forum.	<b>.</b>
Port Firmware	Silicon Labs CP210x USB to UART Bridge (COM3) class for Python/Python_Firmware/esp32-20210902-v1.17.bin Browse	
<ul> <li>From i</li> <li>Dual I/</li> </ul>	mage file (keep) (Quad I/O (qio) O (dio) (Dual Output (dout) ash before installing	
	Install Cancel	

Then click Close and OK

The ESP32 fire	mware installer	$\times$
If you nee Note that at micropy	g allows installing or updating firmware on ESP32 using the most common settings d to set other options, then please use 'esptool' on the command line. there are many variants of MicroPython for ESP devices. If the firmware provided ython.org/download doesn't work for your device, then there may exist better es look around in your device's documentation or at MicroPython forum.	j.
Port Firmware	Silicon Labs CP210x USB to UART Bridge (COM3) V Reload on (Important) /Python_Firmware/esp32-20210902-v1.17.bin Browse	
~	nde mage file (keep) () Quad I/O (qio) (O (dio) () Dual Output (dout)	
	Nriting at 0x0006521c (19 %)	

😱 ESP32 firm	mware installer	$\times$
If you need Note that t at micropy	g allows installing or updating firmware on ESP32 using the most common settings d to set other options, then please use 'esptool' on the command line. there are many variants of MicroPython for ESP devices. If the firmware provided ython.org/download doesn't work for your device, then there may exist better es look around in your device's documentation or at MicroPython forum.	
Port	Silicon Labs CP210x USB to UART Bridge (COM3)  Reload	
Firmware Flash mo	class for Python/Python_Firmware/esp32-20210902-v1.17.bin Browse	
From in	mage file (keep) 🔿 Quad I/O (qio)	
🔿 Dual I/	O (dio) O Dual Output (dout)	
✓ Erase fl	lash before installing	
<u> </u>	Done! Install Close	]

File       The Thonny options       ×         Image: Constraint of the state of the st	
General Interpreter Editor Theme & Font Run & Debug Terminal Shell Assistant	
Files       Which interpreter or device should Thonny use for running your code?         MicroPython (ESP32)       •         Details       Connecting via USB cable: Connect your device to the computer and select corresponding port below (look for your device name, "USB Serial" or "UART"). If you can't find it, you may need to install proper USB driver first.         Connecting via WebREPL (EXPERIMENTAL): If your device supports WebREPL, first connect via serial, make sure WebREPL is enabled (import webrepl_setup), connect your computer and device to same network and select < WebREPL > below         Port or WebREPL       Port or WebREPL	
Silicon Labs CP210x USB to UART Bridge (COM3)	n.exe

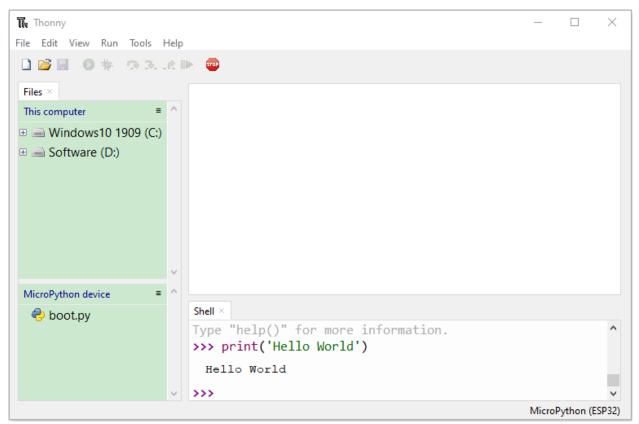
Turn off all windows and turn to the main page and click <sup>eee</sup> "STOP.

The Thonny	_		$\times$
File Edit View Run Tools Help			
🗋 🗃 📕 💿 🚸 👁 🗈 🗈 🖿 🗰 🛑 STOP Button			
Files ×			
This computer = ^			
⊞ → Windows10 1909 (C:)			
🗉 📾 Software (D:)			
/:Root directory			
Inter space of ESP32			
used to save files.			
MicroPython device Shell ×			_
<pre>   boot.py   MicroPython v1.17 on 2021-09-02; ESP32 module </pre>	with	ESP3	2
Type "help()" for more information.			_
✓ >>>			¥
	MicroPy	thon (ES	SP32)

### Test Code

Test the Shell commander

Input print('hello world') in the"Shell"and press Enter



### Run the test code(online)

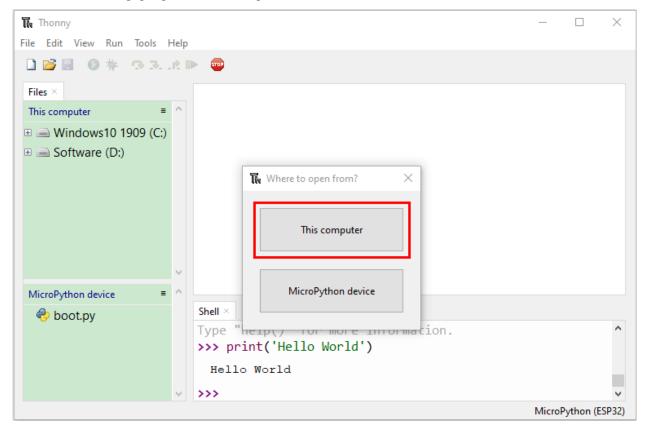
Connect the ESP32 to your PC. Users can program and debug programs with Thonny.

Open Thonny and click Open.

If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

The Thonny				$\times$
File Edit View Run Tools Help				
D 📸 📄 🔷 🌞 🛪 🖈 🛯 Files ×				
This computer				
🗉 📾 Windows10 1909 (C:)				
🗉 📾 Software (D:)				
×				
MicroPython device				
😔 boot.py	Shell ×			
	Type "help()" for more information.			^
	<pre>&gt;&gt;&gt; print('Hello World')</pre>			
	Hello World			
×	>>>			~
		MicroP	ython (ES	P32)

When a new window pops up, click"This computer"



🏹 Open			×
← → * ↑	« 2. Python Projects » Project 01: Hello World	ע גע אין	ch Project 01: Hello W
Organize 👻 🛛 N	lew folder		III 🔹 🚺 😯
1 Ouish second	^ Name ^	Date modified	Type Size
<ul> <li>Quick access</li> <li>Desktop</li> <li>Downloads</li> <li>Documents</li> </ul>	Th Project_01_HelloWorld	10/15/2021 8:30 AM	Thonny
i socurrens	File name: Project_01_HelloWorld	✓ Pythou fi Ope	

Select the file"Project\_01\_HelloWorld.py"

Click O, "Hello World" will be printed in the "Shell" monitor.

Thonny - C:\Users\Administrator\De	sktop\KS5003(KS5004)Keyestudio ESP32 42 in 1 Sensor Kit\Windows — 🛛 🛛 🔀	<
File Edit View Run Tools Help		
🗋 😂 🖩 💽 🔶 🤋 R. 18 🕨	500 · · · · · · · · · · · · · · · · · ·	
Files ×	lesson_01_HelloWorld.py $ imes$	
This computer =	1 print("Hello World!")	
<ul> <li></li></ul>	<pre>2 print("Welcome Keyestudio") 3</pre>	
	Shell × MicroPuthon v1 17 on 2021 00 02: 55022 modulo with 55022	
MicroPython device	MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32 . Type "help()" for more information.	
micror years acree	>>> %Run -c \$EDITOR CONTENT	
😔 boot.py	Hello World! Welcome Keyestudio	
	>>>	1
	MicroPython (ESP3	2)

Note: Press the reset button to reboot

#### Run the test code(offline)

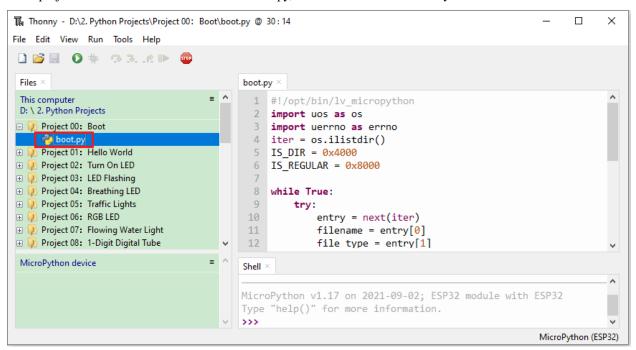
After rebooting the ESP32, run the boot.py file under the root directory first then run your code file.

So, we need to add a guide program to run the code of users.

Move the file"4. Python Tutorial\2. Python Projects" to the disk(D)the route is "D:/2. Python Projects", then open the "Thonny".

🏗 Thonny - <untitled> @ 1:1</untitled>			$\times$
File Edit View Run Tools Help			
🗋 😂 🛃 🛛 🕸 🕾 🐼 🕪 🚥			
Files $\times$		<untitled> ×</untitled>	
This computer	= ^	1	^
🚍 Windows10 1909 (C:)			
🗄 🚞 Software (D:)			
			$\vee$
	~	Shell ×	
MicroPython device	= ^		^
😔 boot.py		MicroPython v1.17 on 2021-09-02; ESP32 module with ESP3	2
		Type "help()" for more information.	
	× .	>>>	¥

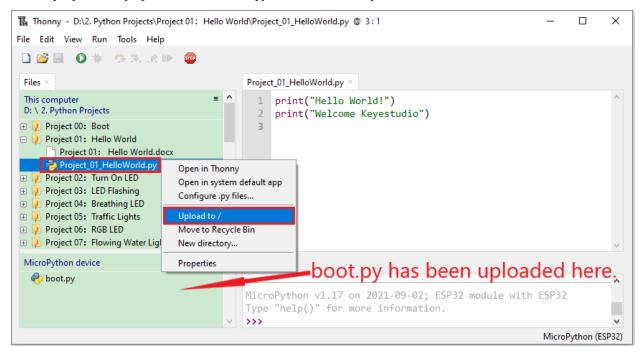
Click project 00. Boot.Boot and double-click boot.py, then the code under MicroPython device can run offline.



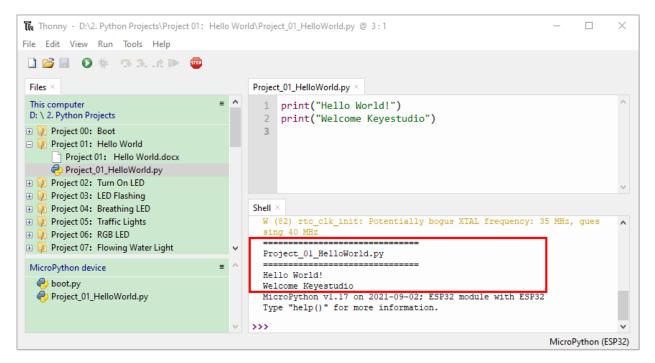
If you want to run the code offline, you nee to upload boot.py and program code to MicroPython device, then press the ESP32's reset button. We will take the project 00 and project 01 as an example. Select boot.py and right-click Upload to /.

ि Thonny - D:\2. Python Projects\Project 00: Boot\boo	ot.py @ 30:14 —	×
File Edit View Run Tools Help		
🗋 🚰 📕 🔹 🐐 👁 3e 🕨 👳		
Files ×	boot.py ×	
This computer = D: \ 2. Python Projects	1 #!/opt/bin/lv_micropython 2 import uos as os	^
	<pre>3 import uerrno as errno 4 iter = os.ilistdir() 5 IS_DIR = 0x4000 6 IS_REGULAR = 0x8000 7</pre>	
⊕ Project 03: Ett     Upload to /     ⊕ Project 05: Tra     Move to Recycle Bin	8 while True: 9 try:	
Image: Project 06: RG New directory	10 entry = next(iter)	
Project 07: Flo     Project 08: 1-0     Properties     Project 08: 1-0     Properties     Properies     Properies	11filename = entry[0]12file type = entry[1]	v
MicroPython device	shell No code has been uploaded here.	
	MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32 Type "help()" for more information.	~ ~
	MicroPytho	n (ESP32)

Similarly, upload the project\_01\_Helloworld. py file to the "MicroPython Device".

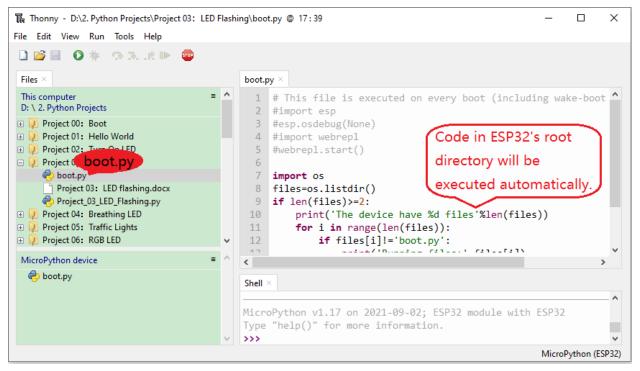


Press the Reset button, you will view code running in the Shell monitor

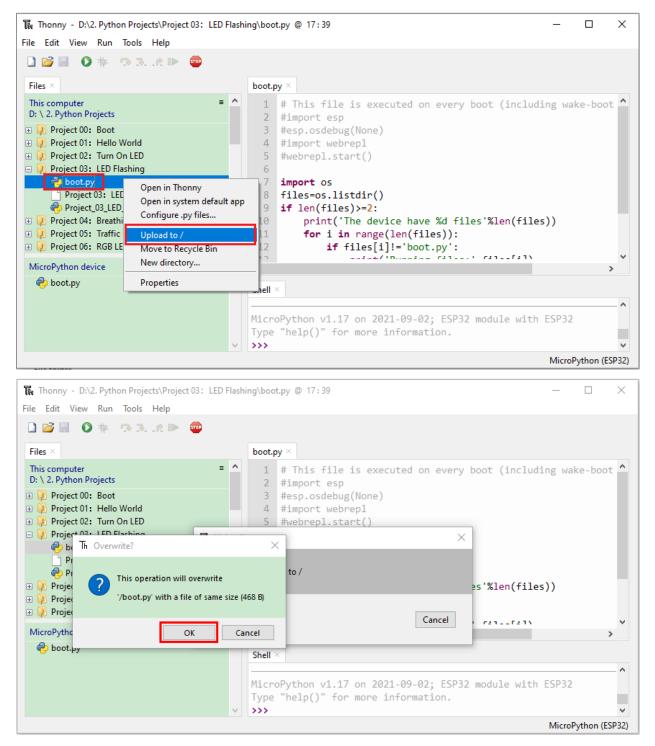


Upload the code to the ESP32

We take the boot.py as an example. If we add a boot.py in each code directory, reboot the ESP32, the boot.py will run first.

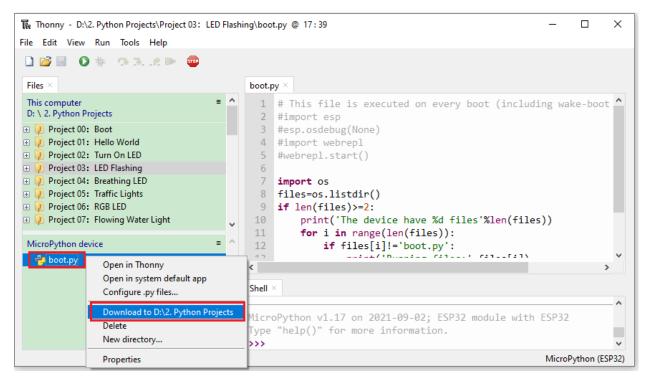


Select "boot.py" in the file Project 03LED Flashing, right-click to select "Upload to /". Then the code will be uploaded to the root directory of the ESP32 and click OK.



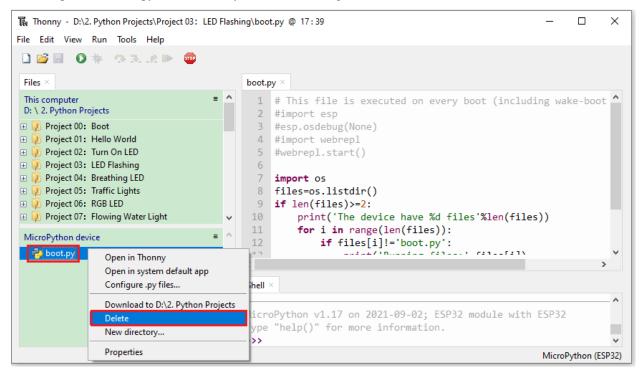
Download the code to your PC:

MicroPython device<boot.py, then right-click Download to...



Delete files of the ESP32

For example, click"boot.py"in the MicroPython device and right-click Delete.



Select boot.py in the Project 03LED Flashing folder, right-click Move to Recycle Bin to delete it.

Thonny - D:\2. Python Projects\Project 03: LED Flashi File Edit View Run Tools Help          D I III IIIIIIIIIIIIIIIIIIIIIIIIIIIIII	boot.py ×
This computer       Image: Computer of the second sec	<pre>1 # This file is executed on every boot (including wake-boot 2 #import esp 3 #esp.osdebug(None) 4 #import webrep1 5 #webrep1.start() 6 7 import os 8 files=os.listdir() 9 if len(files)&gt;=2: 10 print('The device have %d files'%len(files)) 11 for i in range(len(files)): 12 if files[i]!='boot.py': 14 contact('Duration files'' files[i]) </pre>
MicroPython device Properties	Shell × MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32 Type "help()" for more information. >>> MicroPython (ESP32

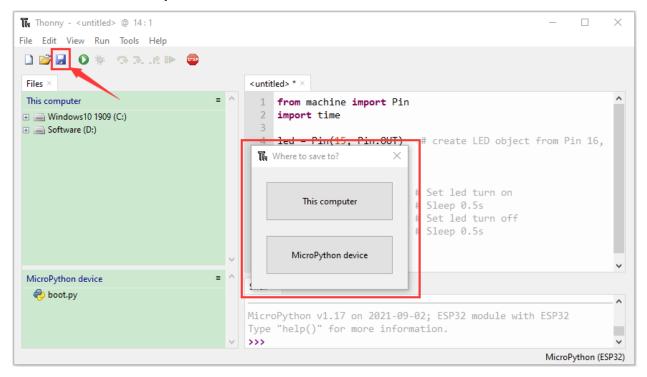
Create and save code

 $\label{eq:click-file} Click ``File" \rightarrow ``New" to create and edit code.$ 

🔚 Thonny				- 🗆	×
File Edit View Run	Tools Help				
🗋 New	Ctrl+N	<b></b>			
💕 Open	Ctrl+O	1 -			
Recent files	+				
Close	Ctrl+W	= ^			
Close all	Ctrl+Shift+W				
Save	Ctrl+S				
Save All files	Ctrl+Alt+S				
Save as	Ctrl+Shift+S				
Save copy					
Move / rename					
Print	Ctrl+P				
Exit	Alt+F4				
		~			
MicroPython device		= ^			
			Shell × MicroPython v1.17 on 2021- Type "help()" for more inf	e with ESP3	2
		~	>>>		Υ
				MicroPython (E	SP32)

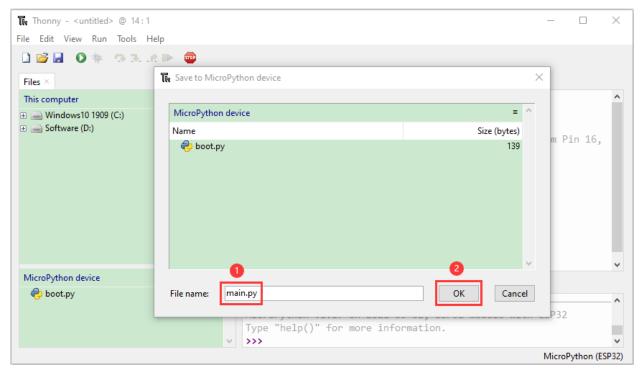
Enter the code in the new file. We take the Project\_03\_LED\_Flashing.py as an example.



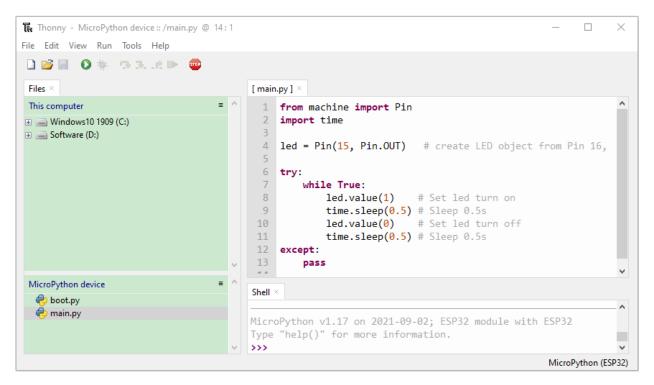


Click  $\blacksquare$  to save the code to your PC or the ESP32.

Select MicroPython device and enter main.py in the new page and click OK.

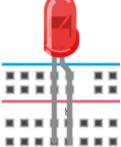


Then the code will be uploaded to the ESP32.



Disconnect the USB cable and connect it, you can see the effect of the LED flashing continuously in the circuit on a cycle.





### CHAPTER

### SEVEN

### **PYTHON TUTORIAL**

### 7.1 Download Python code files

Click on the link to download the Python code fileDownload Python Codes

# 7.2 Development Environment Configuration

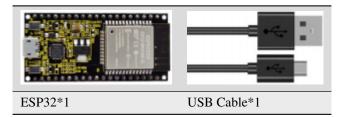
Click on the link to enter the development environment setup tutorial: Pythin Development Environment Configuration

# 7.3 Project 01: Hello World

1. Overview

For ESP32 beginners, we will start with some simple things. In this project, you only need a ESP32 mainboard, a USB cable and Raspberry Pi to complete the "Hello World!" project, which is a test of communication between the ESP32 mainboard and the Raspberry Pi as well as a primary project.

2. Components



3. Wiring Diagram

In this project, we will use a USB cable to connect the ESP32 to Raspberry Pi.



#### **Running code online**

To run the ESP32 online, you need to connect the ESP32 to the computer, which allows you to compile or debug programs using Thonny software.

Advantages:

- 1. You can use the Thonny software to compile or debug programs.
  - 2. Through the "Shell" window, you can view error messages and output results generated during the running of the program as well as query related function information online to help improve the program.

Disadvantages:

- 1. To run the ESP32 online, you must connect the ESP32 to a computer and run it with the Thonny software.
- 2. If the ESP32 is disconnected from the computer , when they reconnect, the program won't run again.

Basic Operation:

1. Open Thonny and click"Open...".

🏗 Thonny - <untitled> @ 1:1</untitled>		- 0	×
File Edit View Run Tools Help			
□ 📴 🖬 • O 🚸 🗇 3	<pre>vuntitled&gt; ×</pre>		
This computer =	1		^
<ul> <li>Windows10 1909 (C:)</li> <li>Software (D:)</li> <li>MicroPython device =</li> </ul>			
😔 boot.py			$\sim$
	Shell ×		
			^
	MicroPython v1.17 on 2021-09-02;		
	Type "help()" for more informati	ion.	
· · · · · · · · · · · · · · · · · · ·	>>>	MicroPython (E	✓ SP32)

2. Click"This computer"in the new pop-up window.

T	Where to open from?	$\times$
	This computer	
	MicroPython device	

In the new dialog boxselect"Project\_01\_HelloWorld.py",click"Open". (If you haven't downloaded the code file, please click on the link to download it:Download Python Codes)

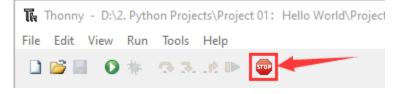
The Open				×
← → · ↑ 📙 « Soft	tware (D:) > 2. Python Projects > Project 0	1: Hello World 🛛 🗸	ට 🔎 Search F	Project 01: Hello W
Organize 🔻 New folder	r			== • • • •
🕹 Downloads \land	Name	Date modified	Туре	Size
h Music	Th Project_01_HelloWorld	10/15/2021 8:30 AM	Python file	1 KB
Pictures		-		
Videos		0		
🏪 Windows10 1909 🗸		N	2	
File na	me: Project_01_HelloWorld		<ul> <li>Pythor files</li> </ul>	~
			Open	Cancel

3. Click C"Run current script" to execute the program "Hello World!", "Welcome Keyestudio", which will be printed in the "Shell" window.

iile Edit View Run Tools Help	TOP			
Files ×		Project_01_HelloWorld.py ×		
This computer D: \ 2. Python Projects Project 00: Boot Project 01: Hello World Project 01: Hello World.docx Project 01: Hello World.py Project 02: Turn On LED Project 03: LED Flashing Project 04: Breathing LED	= ^	<pre>1 print("Hello World!") 2 print("Welcome Keyestudio") 3</pre>		^
	v	Shell × MicroPython v1.17 on 2021-09-02; ESP32 module with ESP	932	^
MicroPython device	= ^	Type "help()" for more information. >>> %Run -c \$EDITOR_CONTENT Hello World! Welcome Keyestudio		ļ

#### Exit running online

When running online, click "" "Stop /Restart Backend" or press "Ctrl+C" on the Thonny to exit the program.



### 5. Test Code

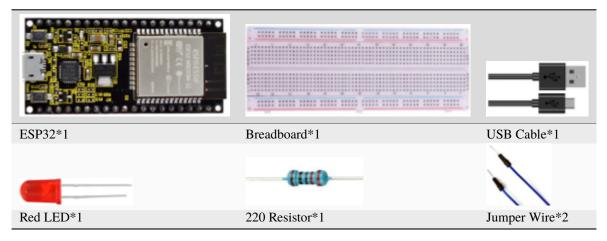
```
print("Hello World!")
print("Welcome Keyestudio")
```

### 7.4 Project 02: Turn On LED

1. Introduction

In this project, we will show you how to light up the LED. We use the ESP32's digital pin to turn on the LED so that the LED is lit up.

2. Components

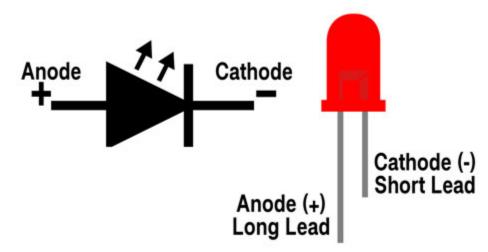


3. Component Knowledge

#### 1LED:

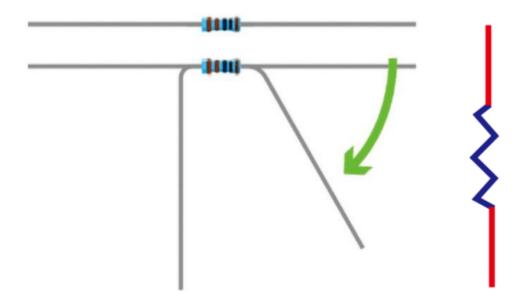


The LED is a semiconductor known as "light-emitting diode", which is an electronic device made from semiconducting materials(silicon, selenium, germanium, etc.). It has an anode and a cathode, the short lead is cathode, which connects to GND, the long lead is anode, which connects to 3.3V or 5V.



### 2Five-band resistor

A resistor is an electronic component in a circuit that restricts or regulates the flow current to flow. On the left is the appearance of the resistor and on the right is the symbol for the resistance in the circuit. Its unit is(). 1 m = 1000 k1k = 10007)



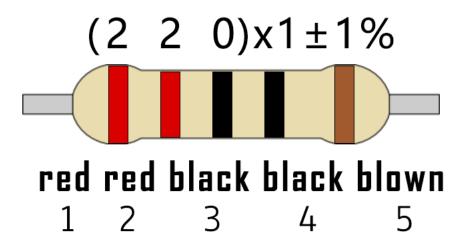
We can use resistors to protect sensitive components, such as LED. The strength of the resistance is marked on the body of the resistor with an electronic color code. Each color code represents a number, and you can refer to it in a resistance card.

- -Color 1 1st Digit.
- -Color 2 2nd Digit.
- -Color 3 3rd Digit.
- -Color 4 Multiplier.
- -Color 5 Tolerance.

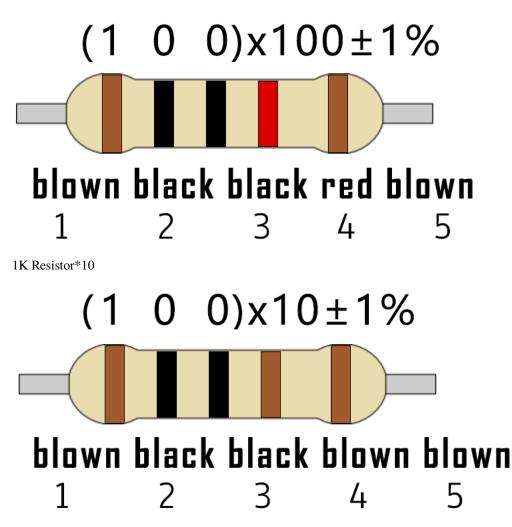
	1st Digit	2nd Digit	3rd Digit	Multiplier	Tolerance
Black		0	0	x1	
Brown	1	1	1	x10	± 1%
Red	2	2	2	x100	± 2%
Orange	3	3	3	x1K	± 3%
Yellow	4	4	4	x10K	± 4%
Green	5	5	5	x100K	±0.5%
Blue	6	6	6	x1M	±0.25%
Violet	7	7	7	x10M	±0.10%
Grey	8	8	8	x100M	±0.05%
White	9	9	9	x1G	
Gold				÷ 10	± 5%
Silver				÷ 100	± 10%

In this kit, we provide three five-band resistors with different resistance values. We three five-band resistors as an example.

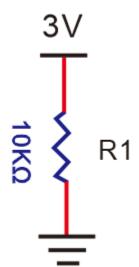
<sup>220</sup> Resistor\*10



10K Resistor\*10



In the same voltage, there will be less current and more resistance. The connection between current(I), voltage(V), and resistance<sup>®</sup> can be expressed by the formula: I=U/R. In the figure below, if the voltage is 3V, the current through R1 is: I = U / R = 3 V / 10 K = 0.0003A = 0.3mA.

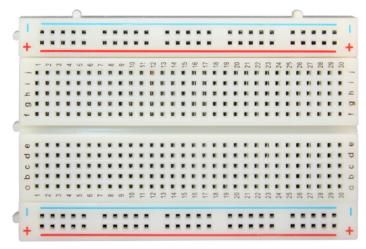


Don't connect a low resistance directly to the two poles of the power supply, which will cause excessive current to

damage the electronic components. Resistors do not have positive and negative poles.

#### **3Bread board**

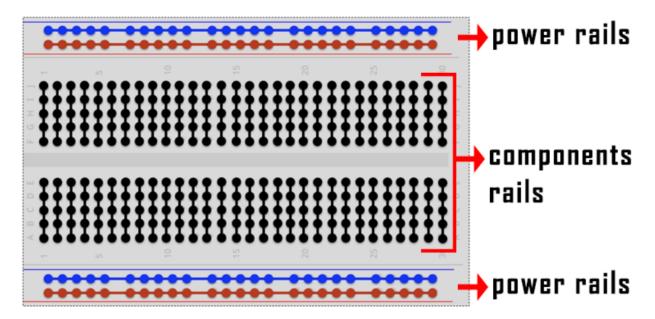
Breadboards are used to build and test circuits quickly before completing any circuit design. There are many holes in the breadboard that can be inserted into circuit components such as integrated circuits and resistors. A typical breadboard is shown below



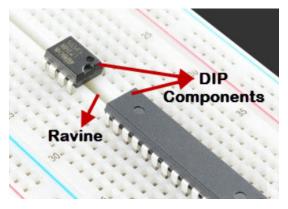
The breadboard has strips of metal, which run underneath the board and connect the holes on the top of the board. The metal strips are laid out as shown below. Note that the top and bottom rows of holes are connected horizontallywhile the remaining holes are connected vertically.

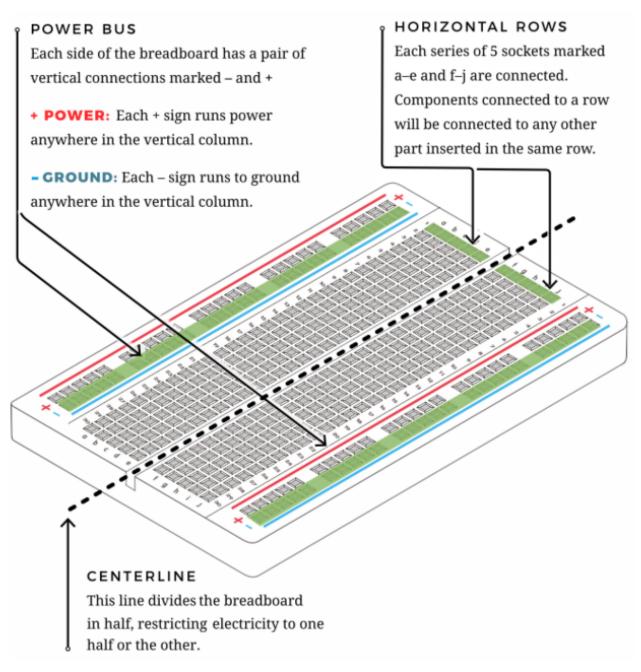


The first two rows (top) and the last two rows (bottom) of the breadboard are used for the positive pole (+) and negative pole (-) of the power supply respectively. The conductive layout of the breadboard is shown in the figure below:



When we connect DIP (Dual In-line Packages) components, such as integrated circuits, microcontrollers, chips and so on, we can see that a groove in the middle isolates the middle part, so the top and bottom of the groove is not connected. DIP components can be connected as shown in the following diagram:





# 4) Power Supply

The ESP32 needs 3.3V-5V power supply. In this project, we will connect the ESP32 to the computer via an USB cable.

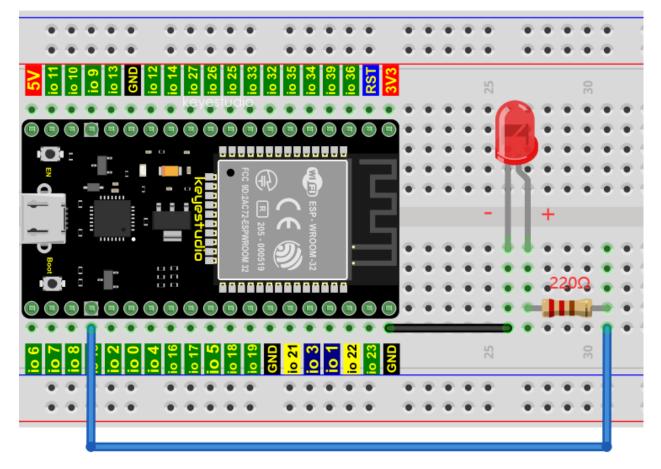


4.Wiring Diagram

First, disconnect all power from the ESP32. Then build the circuit according to the wiring diagram. After the circuit is built and verified correctly, connect the ESP32 to your computer via a USB cable.

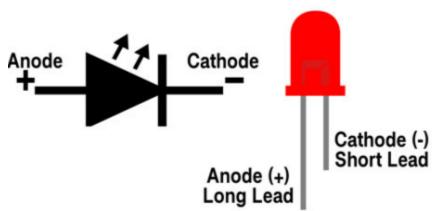
Note: Avoid any possible short circuits (especially connecting 3.3V and GND)!

**WARNING:** A short circuit can cause high current in your circuit, create excessive component heat and cause permanent damage to your hardware!

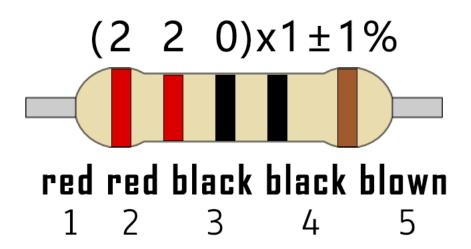


Note:

How to connect a LED



How to identify the 220 Five-band resistor



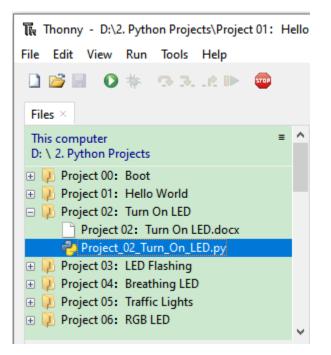
# 5. Test Code

If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

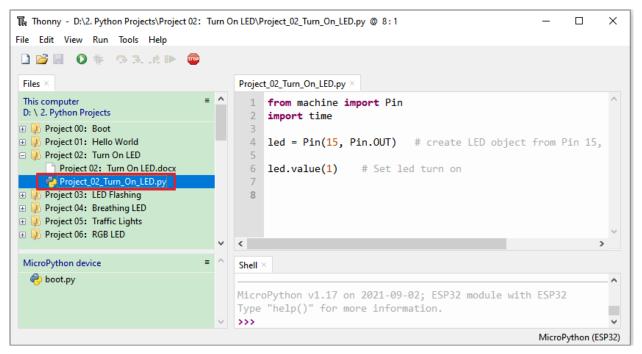
2. Python Projects		_	$\Box$ $\times$
Share View			~ ?
→ This PC → Software (D:) → 2. Pyth	on Projects v さ	,○ Search 2	. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	
Frit mouth a culture			

#### Exit running online

Open"Thonny", click"This computer"→"D:"→"2. Python Projects"→"Project 02Turn On LED"

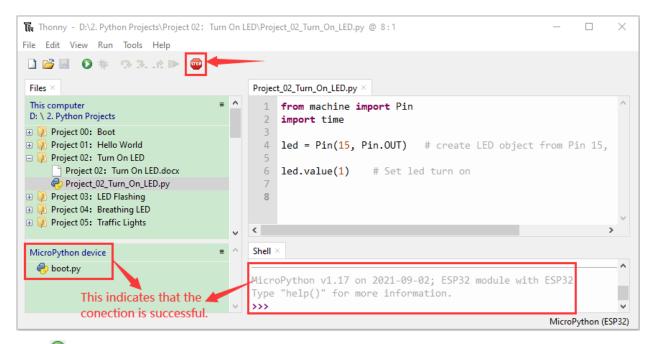


Click"Project 02Turn On LED", double-click"Project\_02\_Turn\_On\_LED.py"to open it, as shown below;

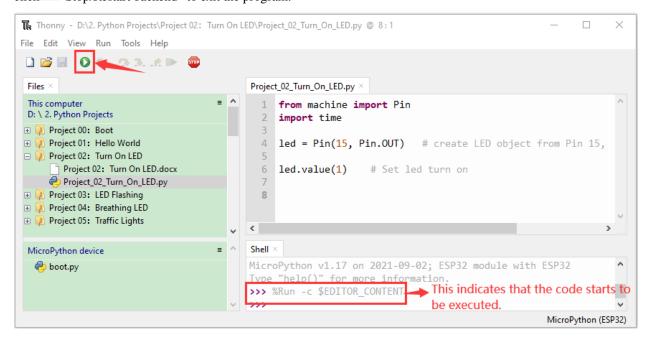


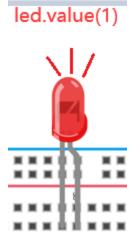


Connect the ESP32 to your PC. Click <sup>22</sup> "Stop/Restart backend" then go to the Shell window to check.

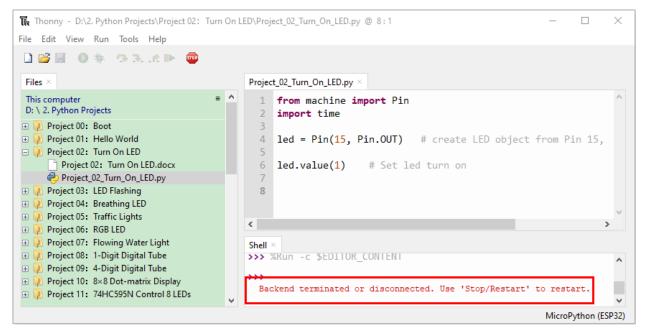


Click O"Run current script" the code starts to be executed and the LED in the circuit lit up. Press "Ctrl+C" or click "Stop/Restart backend" to exit the program.



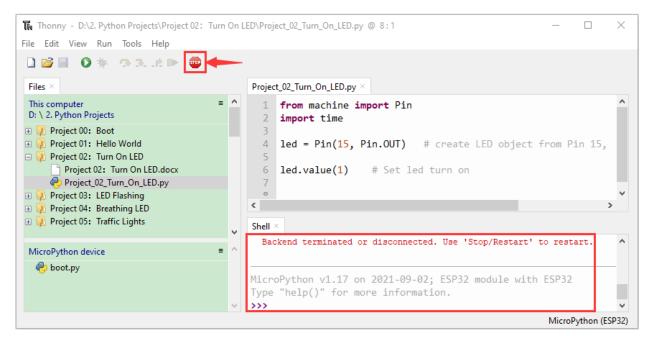


**Note**: This is the code running online. If you disconnect USB cable and power up the ESP32 or press its reset button, LED is not bright and the following messages will be displayed in the "**Shell**" window of Thonny:



Code running offlineUpload the code to ESP32

Make sure the ESP32 has been connected to the computer, click <sup>222</sup> "Stop/Restart backend".



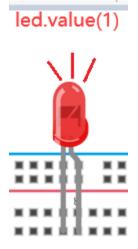
As shown below, right-click the file"Project\_02\_Turn\_On\_LED.py"select "Upload to /"to upload the code to ESP32.

🕞 Thonny - D:\2. Python Projects\Proje	ect 02: Turn On L	ED\Proj	ect_02_Turn_On_LED.py @ 8:1	×
File Edit View Run Tools Help				
0 🐸 🖩 0 🐐 🤉 3e 🕨	STOP			
Files $\times$		Projec	t_02_Turn_On_LED.py $ imes$	
This computer D: \ 2. Python Projects	= ^	1	<pre>from machine import Pin import time</pre>	^
<ul> <li></li></ul>		3 4 5 6	<pre>led = Pin(15, Pin.OUT) # create LED object from Pin 15, led.value(1) # Set led turn on</pre>	
<ul> <li>Project_02_Turn_On_LED.py</li> <li>Project 03: LED Flashing</li> <li>Project 04: Breathing LED</li> <li>Project 05: Traffic Lights</li> </ul>	Open in Thon Open in syster Configure .py	m defau	lt app	>
MicroPython device	Upload to / Move to Recy New directory		rminated or disconnected. Use 'Stop/Restart' to restart.	^
	Properties	Type	v1.17 on 2021-09-02; ESP32 module with ESP32 neip()" for more information.	
			MicroPython (E	SP32)

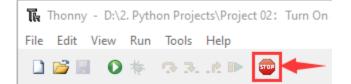
Upload"boot.py"in the same way.

🙀 Thonny - D:\2. Python Projects\Project 02: Turn On L	.ED\Proj	ect_02_Turn_On_LED.py @ 8:1 -	×
File Edit View Run Tools Help			
🗋 😂 🖩 🛛 🌞 🦘 3e 🕪 🥶			
Files $\times$	Projec	t_02_Turn_On_LED.py ×	
This computer       =         D: \ 2. Python Projects       =         Project 00: Boot       •         Description       •         Project 01: He       •         Project 02: Tui       •         Project 03: LEI       •         Project 04: Bre       •         Project 05: Tra       Move to Recycle Bin         Project 06: RG       New directory	1 2 3 4 5 6 7 <b>8</b>	<pre>from machine import Pin import time led = Pin(15, Pin.OUT)  # create LED object from Pin 15, led.value(1)  # Set led turn on</pre>	~
MicroPython device Properties			~
<ul> <li>boot.py</li> <li>Project_02_Turn_On_LED.py</li> </ul>	< Shell		>
Make sure you have uploaded Project_ 02_Turn_On_LED.py and boot.py here.		oPython v1.17 on 2021-09-02; ESP32 module with ESP32 "help()" for more information.	- ^
		MicroPython (E	SP32)

Press the reset button of ESP32 and you can see LED is  $\ensuremath{\mathsf{ON}}$  .



NoteCodes here is run offline. If you want to stop running offline and enter"Shell", just click "Stop/Restart backend"in Thonny.

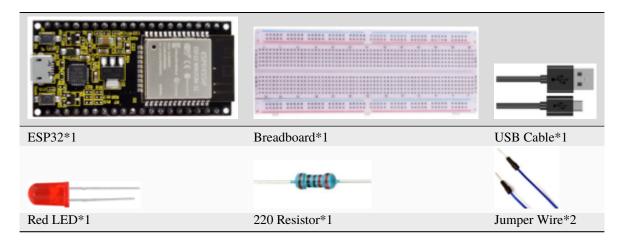


# 7.5 Project 03LED Flashing

### 1.Introduction

In this project, we will show you the LED flashing effect. We use the ESP32's digital pin to turn on the LED and make it flashing.

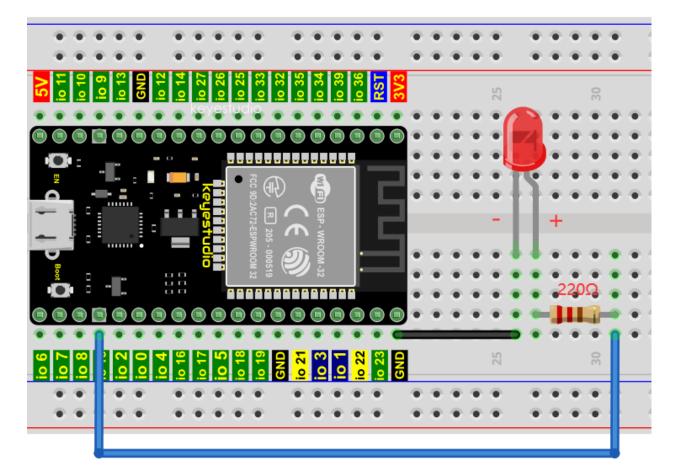
2.Components



### 3.Wiring diagram

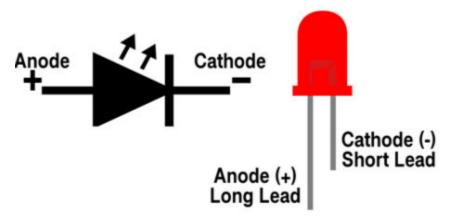
First, disconnect all power from the ESP32. Then build the circuit according to the wiring diagram. After the circuit is built and verified correct, connect the ESP32 to your computer using a USB cable.

Note: Avoid any possible short circuits (especially connecting 3.3V and GND)!

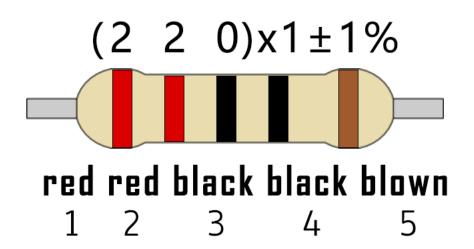


Note:

How to connect a LED



How to identify the 220 Five-color ring resistor



4.Project code

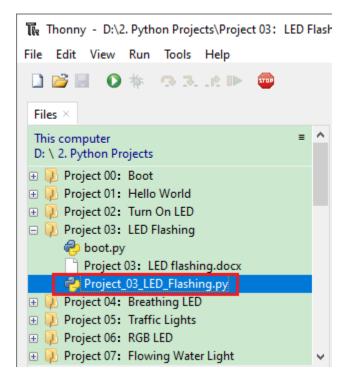
Codes used in this tutorial are saved in "2. Python Projects".

If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

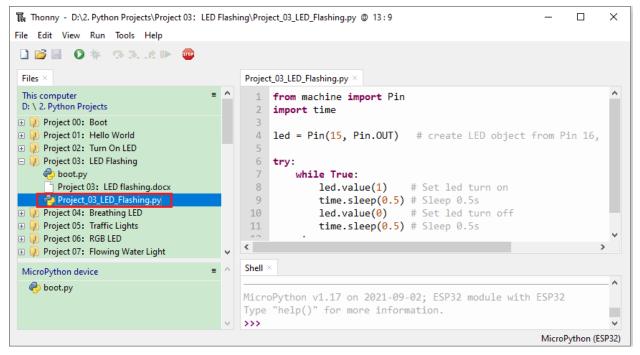
Python Projects		—	
Share View			~ (
> This PC > Software (D:) > 2. Pytł	non Projects 🗸 🗸	ר א פער אין	2. Python
Name	Date modified	Туре	Size
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
Project 02: Turn On LED	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

Code running online:

Open"Thonny"click"This computer"→"D:"→"2. Python Projects"→"Project 03LED Flashing".



Expand folder"Project 03: LED Flashing" and double left-click "Project\_03\_LED\_Flashing.py" to open it. As shown in the illustration below



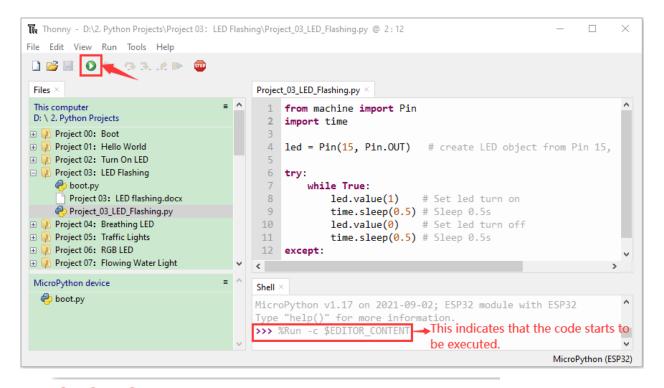
(continued from previous page)

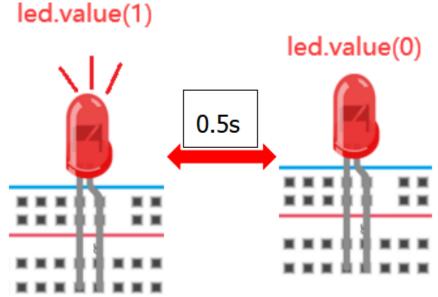
```
while True:
    led.value(1)  # Set led turn on
    time.sleep(0.5) # Sleep 0.5s
    led.value(0)  # Set led turn off
    time.sleep(0.5) # Sleep 0.5s
except:
    pass
```

Make sure the ESP32 has been connected to the computer. Click "" "Stop/Restart backend" and see what will display in the "Shell" window.

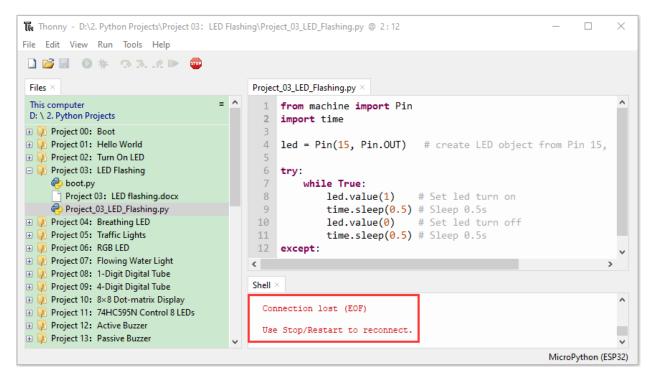


Click O "Run current script" the code starts to be executed and you can see the LED flash. Press"Ctrl+C" or click "Stop/Restart backend" to exit the program.

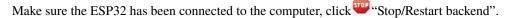


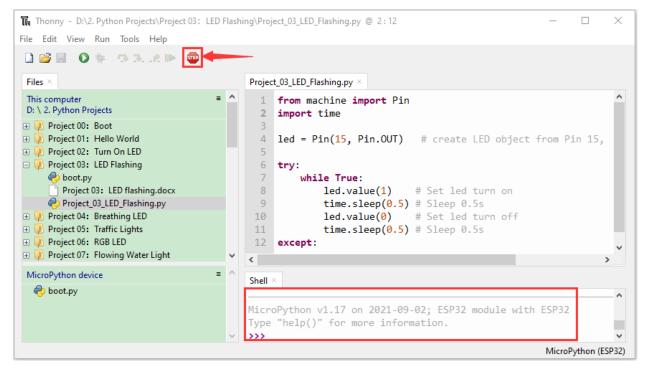


**Note**: This is the code running online. If you disconnect USB cable and power up the ESP32 or press its reset button, the LED in the circuit will stop flashing and the following messages will be displayed in the "Shell"



Code running offlineUpload the code to ESP32





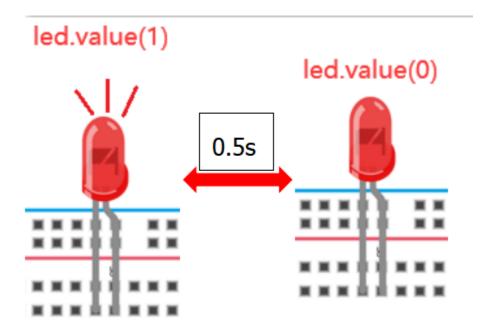
As shown below, right-click the file"Project\_03\_LED\_Flashing.py"select "Upload to /"to upload the code to ESP32.

ि Thonny - D:∖2. Python Projects\Projects File Edit View Run Tools Help	ct 03: LED F	lashing\Pi	roject_03_L	ED_Flashing.py @ 2:12 —	×
	<b>500</b>				
Files ×		Proj	ect_03_LED	)_Flashing.py $ imes$	
This computer D: \ 2. Python Projects 	=	^ 1 2	impor	machine <b>import</b> Pin • <b>t</b> time	^
Project 00: Boot     Project 01: Hello World     Project 02: Turn On LED     Project 03: LED Flashing			led =	• Pin(15, Pin.OUT) # create LED object from Pin 15,	
boot.py		7	/ v	<pre>/hile True: led.value(1)  # Set led turn on</pre>	
<ul> <li>Project_03_LED_Flashing.py</li> <li>Project 04: Breathing LED</li> <li>Project 05: Traffic Lights</li> <li>Project 06: RGB LED</li> </ul>		Thonny system def e .py files		<pre>time.sleep(0.5) # Sleep 0.5s led.value(0) # Set led turn off time.sleep(0.5) # Sleep 0.5s</pre>	ļ
⊕ Project 07: Flowing Water Light     ■	Upload to Move to	o / Recycle Bi	n.	· · · · · · · · · · · · · · · · · · ·	· ~
MicroPython device	New dire	1 A A A A A A A A A A A A A A A A A A A			
- bookpy	Propertie	S	<del>. or y cric</del>	v1.17 on 2021-09-02; ESP32 module with ESP32	^
		v Typ		()" for more information.	~
				MicroPython (ES	P32)

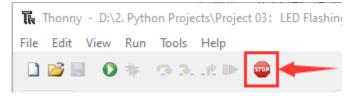
# Upload"boot.py"in the same way.

🙀 Thonny - D:\2. Python Projects\Project 03: LED Flas	hing\Proj	ject_03_LED_Flashing.py @ 2:12 —	×
File Edit View Run Tools Help			
🗋 🐸 📕 💿 🐐 🗇 3e 🕨 👼			
Files ×	Projec	rt_03_LED_Flashing.py $ imes$	
This computer D: \ 2. Python Projects □ ↓ Project 00: Boot	1 2 3	<pre>from machine import Pin import time</pre>	^
boot.py     Open in Thonny     Open in system default app     Project 01: Hel     Project 02: Tur     Project 03: LED	4 5 6 7	<pre>led = Pin(15, Pin.OUT) # create LED object from Pin 15, try:    while True:</pre>	
표 🕖 Project 04: Bre Upload to /	8	<pre>led.value(1) # Set led turn on</pre>	
🗉 💫 Project 05: Trat 🛛 Move to Recycle Bin	9	<pre>time.sleep(0.5) # Sleep 0.5s</pre>	
🗉 권 Project 06: RGE 🛛 New directory	10	<pre>led.value(0) # Set led turn off</pre>	
Project 07: Flov     Properties	11	<pre>time.sleep(0.5) # Sleep 0.5s</pre>	
MicroPython device	12	except:	~
e boot.py	<		>
Project_03_LED_Flashing.py	Shell	×	
Make sure you have uploaded			^
	Micr	oPython v1.17 on 2021-09-02; ESP32 module with ESP32	
Project_03_LED_Flashing.py and	Type	"help()" for more information.	
boot.py here.	>>>		Υ.
		MicroPython (E	SP32)

Press the reset button of ESP32 and you can see the LED flash



NoteCodes here is run offline. If you want to stop running offline and enter"Shell", just click ""Stop/Restart backend"in Thonny.



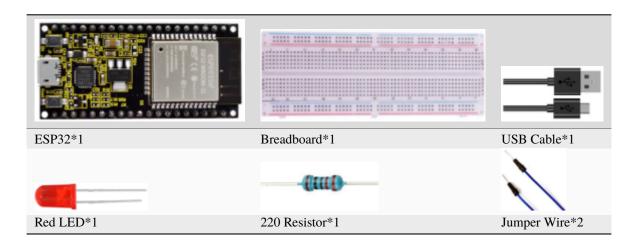
# 7.6 Project 04: Breathing Led

1.Introduction

In previous studies, we know that LEDs have on/off state, so how to enter the intermediate state? How to output an intermediate state to make the LED half bright? That's what we're going to learn.

Breathing light, that is, LED is turned from off to on gradually, and gradually from on to off, just like "breathing". So, how to control the brightness of a LED? We will use ESP32's PWM to achieve this target.

2.Components

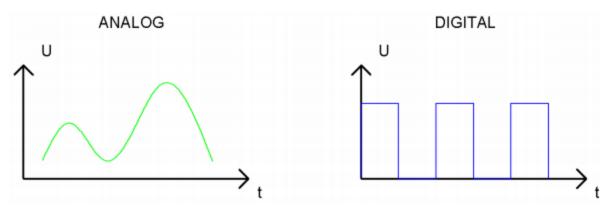


#### 3.Component knowledge



#### Analog & Digital:

An Analog Signal is a continuous signal in both time and value. On the contrary, a Digital Signal or discrete time signal is a time series consisting of a sequence of quantities. Most signals in life are analog signals. A familiar example of an Analog Signal would be how the temperature throughout the day is continuously changing and could not suddenly change instantaneously from 0°C to 10°C. However, Digital Signals can instantaneously change in value. This change is expressed in numbers as 1 and 0 (the basis of binary code). Their differences can more easily be seen when compared when graphed as below.



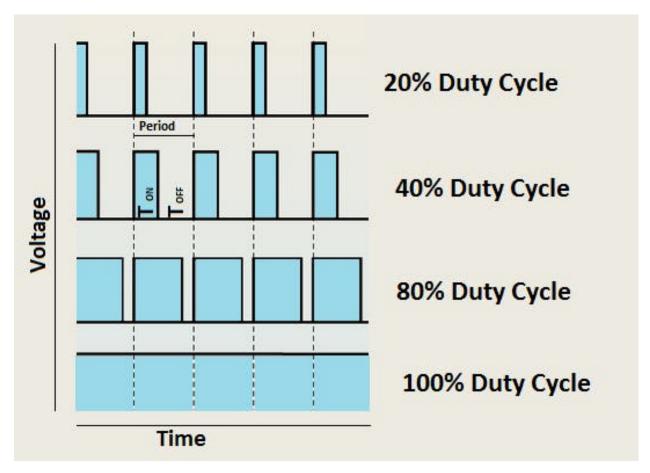
In practical application, we often use binary as the digital signal, that is a series of 0's and 1's. Since a binary signal only has two values (0 or 1), it has great stability and reliability. Lastly, both analog and digital signals can be converted into the other.

# PWM

PWM, Pulse-Width Modulation, is a very effective method for using digital signals to control analog circuits. Common processors cannot directly output analog signals. PWM technology makes it very convenient to achieve this conversion (translation of digital to analog signals).

PWM technology uses digital pins to send certain frequencies of square waves, that is, the output of high levels and low levels, which alternately last for a while. The total time for each set of high levels and low levels is generally fixed, which is called the period (Note: the reciprocal of the period is frequency). The time of high level outputs are generally called "pulse width", and the duty cycle is the percentage of the ratio of pulse duration, or pulse width (PW) to the total period(T) of the waveform.

The longer the output of high levels last, the longer the duty cycle and the higher the corresponding voltage in the analog signal will be. The following figures show how the analog signal voltages vary between 0V-3V3 (high level is 3V3) corresponding to the pulse width 0%-100%:

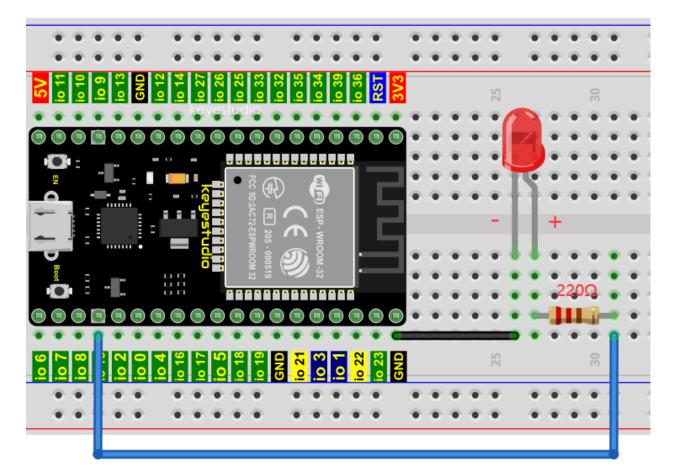


The longer the PWM duty cycle is, the higher the output power will be. Now that we understand this relationship, we can use PWM to control the brightness of an LED or the speed of DC motor and so on. It is evident from the above that PWM is not real analog, and the effective value of the voltage is equivalent to the corresponding analog. So, we can control the output power of the LED and other output modules to achieve different effects.

# ESP32 and PWM:

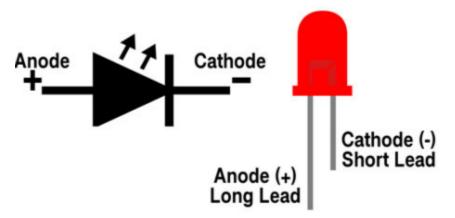
The ESP32 PWM controller has 8 independent channels, each of which can independently control frequency, duty cycle, and even accuracy. Unlike traditional PWM pins, the PWM output pins of ESP32 are configurable and they can be configured to PWM.

4.Wiring diagram

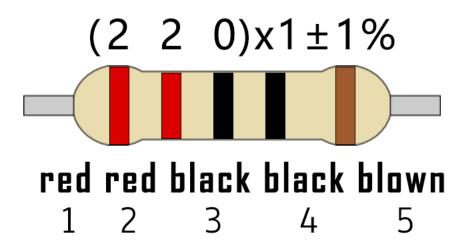


# Note:

How to connect a LED



How to identify the 220 Five-color ring resistor



5.Project code

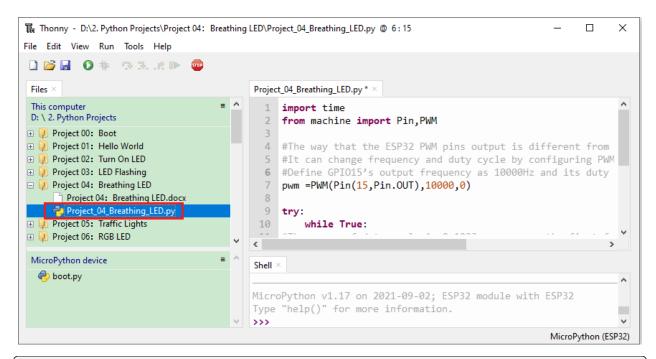
The design of this project makes the GP15 output PWM, and the pulse width gradually increases from 0% to 100%, and then gradually decreases from 100% to 0%.

Codes used in this tutorial are saved in"2. Python Projects".

If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		—	) ×
Share View			~ ?
→ This PC → Software (D:) → 2. Pyth	on Projects v さ		Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01:Hello World	2/17/2022 10:21 AM	File folder	
	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 04Breathing Led", and double left-click "Project\_04\_Breathing\_LED.py".



# import time from machine import Pin,PWM

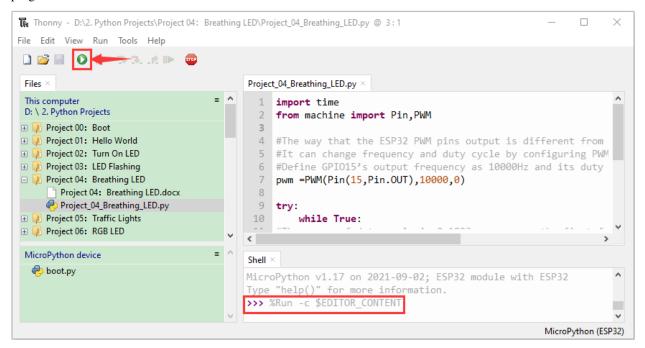
```
#The way that the ESP32 PWM pins output is different from traditionally controllers.
#It can change frequency and duty cycle by configuring PWM's parameters at the
→initialization stage.
#Define GPI015's output frequency as 10000Hz and its duty cycle as 0, and assign them to.
\rightarrow PWM.
pwm = PWM(Pin(15, Pin.OUT), 10000, 0)
trv:
    while True:
#The range of duty cycle is 0-1023, so we use the first for loop to control PWM to.
\hookrightarrow change the duty
#cycle value,making PWM output 0% -100%; Use the second for loop to make PWM output 100%-
\rightarrow 0\%.
        for i in range(0,1023):
             pwm.duty(i)
             time.sleep_ms(1)
        for i in range(0,1023):
             pwm.duty(1023-i)
             time.sleep_ms(1)
except:
#Each time PWM is used, the hardware Timer will be turned ON to cooperate it. Therefore,...
\rightarrow after each use of PWM,
#deinit() needs to be called to turned OFF the timer. Otherwise, the PWM may fail to
\rightarrow work next time.
    pwm.deinit()
```

6.Project result

😱 Thonny - D:\2. Python Projects\Project 04: Breathing LED\Project\_04\_Breathing\_LED.py @ 3:1  $\times$ File Edit View Run Tools Help 🗋 💕 🛃 🛛 🐐 🔿 🥆 🖻 🕨 Files Project\_04\_Breathing\_LED.py \* This computer import time D: \ 2. Python Projects from machine import Pin, PWM 🗉 🕖 Project 00: Boot 3 🗄 📜 Project 01: Hello World #The way that the ESP32 PWM pins output is different from 4 표 🔑 Project 02: Turn On LED #It can change frequency and duty cycle by configuring PWM 표 📜 Project 03: LED Flashing 6 #Define GPI015's output frequency as 10000Hz and its duty 🖃 💫 Project 04: Breathing LED pwm =PWM(Pin(15,Pin.OUT),10000,0) Project 04: Breathing LED.docx Project\_04\_Breathing\_LED.py try: 🗄 🕖 Project 05: Traffic Lights while True: 🕀 🕖 Project 06: RGB LED < | MicroPython device Shell doot.py MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32 Type "help()" for more information. >>> MicroPython (ESP32)

Make sure the ESP32 has been connected to the computer, click "" "Stop/Restart backend".

Click O"Run current script", the code starts to be executed and you'll see that the LED is turned from ON to OFF and then back from OFF to ON gradually like breathing. Press"Ctrl+C"or click ""Stop/Restart backend" to exit the program.



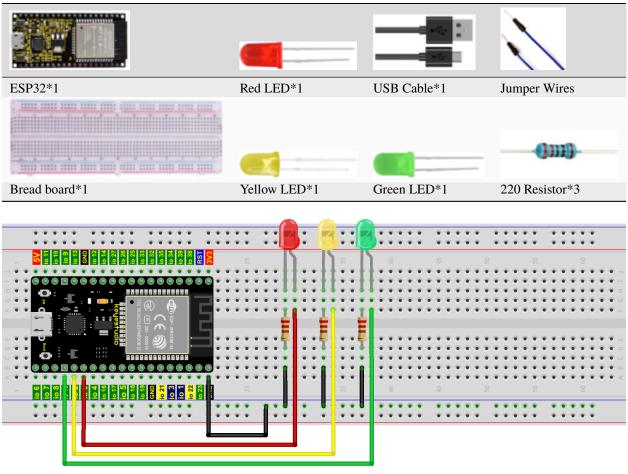


# 7.7 Project 05Traffic Lights

#### 1.Introduction

Traffic lights are closely related to people's daily life, which generally show red, yellow, and green. Everyone should obey the traffic rules, which can avoid many traffic accidents. In this project, we will use ESP32 and some LEDs (red, green and yellow) to simulate the traffic lights.

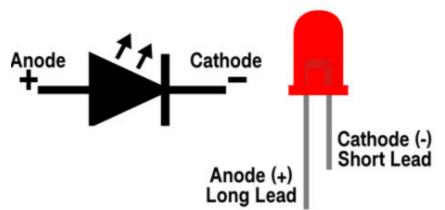
2.Components



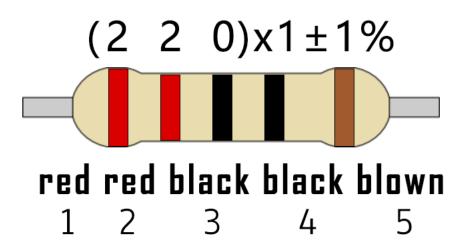
fritzing

# Note:

How to connect a LED



How to identify the 220 Five-color ring resistor



4.Project code

Codes used in this tutorial are saved in"2. Python Projects".

If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

Python Projects		_	$\Box$ $\times$
Share View			~ 🧃
→ This PC → Software (D:) → 2. Pytł	non Projects 🗸 🗸	,	. Python
Name	Date modified	Туре	Size /
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
Project 02: Turn On LED	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 05Traffic Lights". and double left-click "Project\_05\_Traffic\_Lights.py".



# from machine import Pin import time

led\_red = Pin(0, Pin.OUT) # create red led object from Pin 0, Set Pin 0 to output led\_yellow = Pin(2, Pin.OUT) # create yellow led object from Pin 2, Set Pin 2 to output led\_green = Pin(15, Pin.OUT) # create green led object from Pin 15, Set Pin 15 to output

#### while True:

```
led_red.value(1) # Set red led turn on
time.sleep(5) # Sleep 5s
led_red.value(0) # Set red led turn off
led_yellow.value(1)
time.sleep(0.5)
led_yellow.value(0)
time.sleep(0.5)
led_yellow.value(1)
time.sleep(0.5)
led_yellow.value(0)
time.sleep(0.5)
led_yellow.value(1)
time.sleep(0.5)
led_yellow.value(0)
time.sleep(0.5)
led_green.value(1)
time.sleep(5)
led_green.value(0)
```

5.Project result

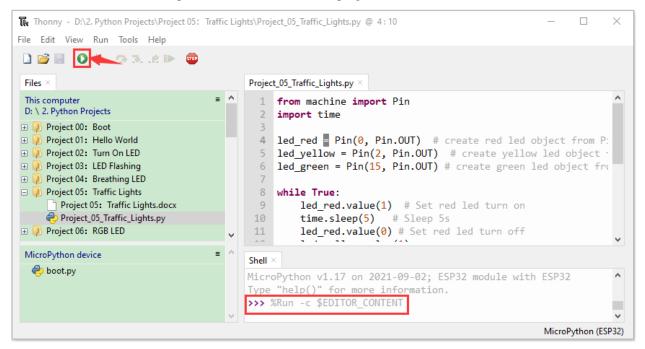
Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".

🗋 🗃 📓 🗶 🎋 😳 3e 🕨 📴 🗲 Files ×		Projec	t_05_Traffic_Lights.py $ imes$	
This computer D: \ 2. Python Projects D: \ 2. Python Projects D: \ 2. Python Project 00: Boot D: Project 01: Hello World D: Project 02: Turn On LED D: Project 03: LED Flashing D: Project 04: Breathing LED D: Project 05: Traffic Lights D: Project 05: Traffic Lights.docx D: Project 06: RGB LED D: \ 2. Project 06: RGB LED	~	1 2 3 4 5 6 7 8 9 10 11	<pre>from machine import Pin import time led_red Pin(0, Pin.OUT) # create red led object from Pi led_yellow = Pin(2, Pin.OUT) # create yellow led object from led_green = Pin(15, Pin.OUT) # create green led object from while True:     led_red.value(1) # Set red led turn on     time.sleep(5) # Sleep 5s     led_red.value(0) # Set red led turn off</pre>	1
MicroPython device ≡	~	Shell >	x Python v1.17 on 2021-09-02; ESP32 module with ESP32 "help()" for more information.	- 1

Click O"Run current script", the code starts to be executed and you'll see are below:

First, the green light will be on for five seconds and then off; Next, the yellow light blinks three times and then goes off; Then, the red light goes on for five seconds and then goes off; Repeat steps 1 to 3 above.

Press "Ctrl+C" or click "Stop/Restart backend" to exit the program.



# 7.8 Project 06: RGB LED

1.Introduction



RGB is composed of three colors (red, green and blue), which can emit different colors of light by mixing these three basic colors.

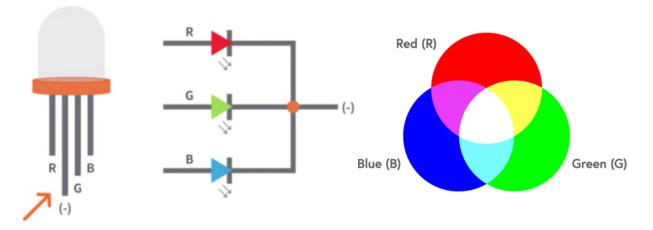
In this project, we will introduce the RGB and show you how to use ESP32 to control the RGB to emit different color light. RGB is pretty basic, but it's also a great way to learn the fundamentals of electronics and coding.

2.Components

ESP32*1	RGB LED	Jumper Wires
	-((660))	
· · · · · · · · · · · · · · · · · · ·	220 D *2	
Breadboard*1	220 Resistor*3	USB Cable*1

#### 3.Component knowledge

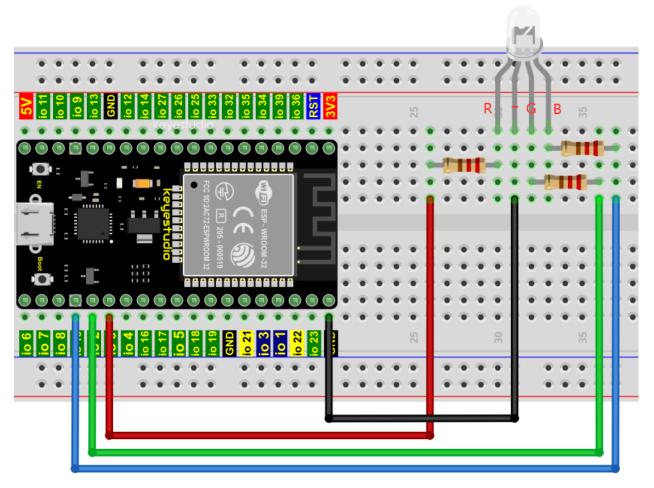
Most monitors adopt the RGB color standard, and all colors on a computer screen are a mixture of red, green and blue in varying proportions.



This RGB LED has 4 pins, each color (red, green, blue) and a common cathode, to change its brightness, we can use the PWM of the ESP32 pins, which can give different duty cycle signals to the RGB to produce different colors of light.

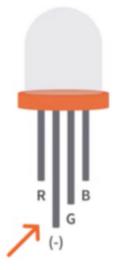
If we use three 10-bit PWM to control the RGB, in theory, we can create  $2 \ 10*210*210 = 1,073,741,824(1 \text{ billion})$  colors through different combinations.

4.Wiring diagram

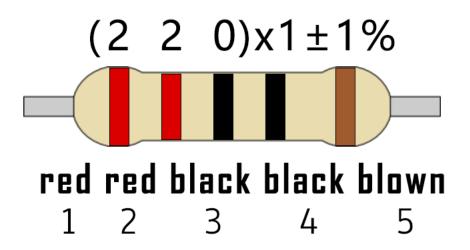


### Note

The longest pin (common cathode) of the RGB LED is connected to GND.



How to identify the 220 Five-color ring resistor



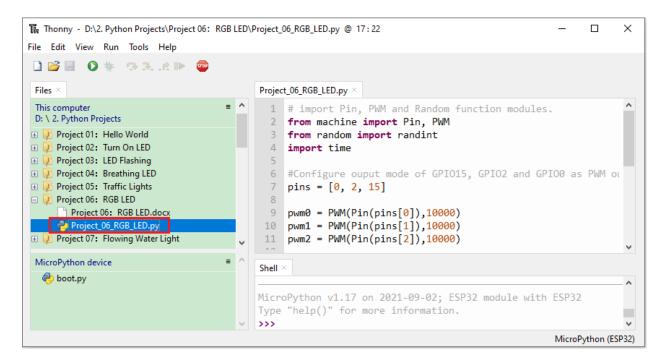
5.Project code

Codes used in this tutorial are saved in"2. Python Projects".

If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects			$\Box$ $\times$
Share View			~ ?
→ This PC → Software (D:) → 2. Pytho	on Projects 🗸 Č	⊖ Search 2	. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
Project 02: Turn On LED	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 06RGB LED", and double left-click "Project\_06\_RGB\_LED.py".



```
# import Pin, PWM and Random function modules.
from machine import Pin, PWM
from random import randint
import time
#Configure ouput mode of GPI015, GPI02 and GPI00 as PWM output and PWM frequency as_
\rightarrow 10000Hz.
pins = [0, 2, 15]
pwm0 = PWM(Pin(pins[0]), 10000)
pwm1 = PWM(Pin(pins[1]), 10000)
pwm2 = PWM(Pin(pins[2]), 10000)
#define a function to set the color of RGBLED.
def setColor(r, g, b):
    pwm0.duty(1023-r)
    pwm1.duty(1023-g)
    pwm2.duty(1023-b)
try:
    while True:
        red = randint(0, 1023)
        green = randint((0, 1023))
        blue = randint(0, 1023)
        setColor(red, green, blue)
        time.sleep_ms(200)
except:
    pwm0.deinit()
    pwm1.deinit()
    pwm2.deinit()
```

6.Project result

Thonny - D:\2. Python Projects\Project 06: RGB LED\Project\_06\_RGB\_LED.py @ 6:15  $\times$ File Edit View Run Tools Help 🗋 💕 📓 🔹 🐐 👁 🥆 🖻 Files > Project\_06\_RGB\_LED.py  $\wedge$ This computer 1 # import Pin, PWM and Random function modules. D: \ 2. Python Projects from machine import Pin, PWM 🕀 🕖 Project 01: Hello World from random import randint 🗉 📜 Project 02: Turn On LED 4 import time 🕀 🕖 Project 03: LED Flashing 🕀 🔑 Project 04: Breathing LED 6 #Configure ouput mode of GPI015, GPI02 and GPI00 as PWM or 🕀 🕖 Project 05: Traffic Lights pins = [0, 2, 15]🖃 📜 Project 06: RGB LED Project 06: RGB LED.docx 9 pwm0 = PWM(Pin(pins[0]),10000) Project\_06\_RGB\_LED.py 10 pwm1 = PWM(Pin(pins[1]),10000) 표 🕕 Project 07: Flowing Water Light 11 pwm2 = PWM(Pin(pins[2]),10000) MicroPython device = Shell × eboot.py MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32 Type "help()" for more information. >>> MicroPython (ESP32)

Click Circle Click



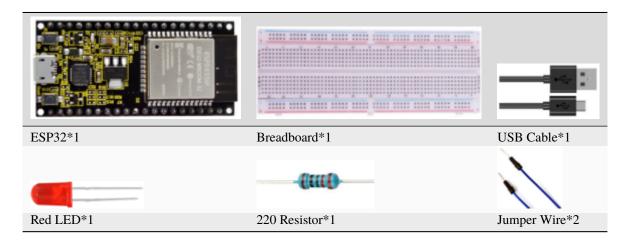
Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".

# 7.9 Project 07: Flowing Water Light

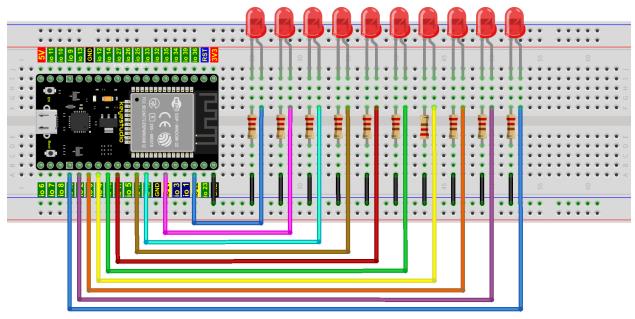
#### 1.Introduction

In our daily life, we can see many billboards composed of different colors of LED. They constantly change the light (like water) to attract customers' attention. In this project, we will use ESP32 to control 10 leds to achieve the effect of flowing water.

2.Components



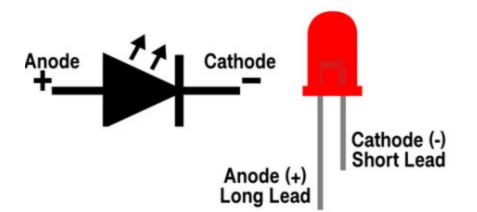
3.Wiring diagram :



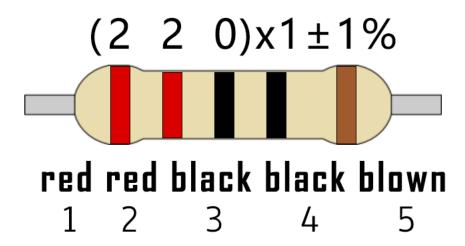
fritzing

#### Note:

How to connect a LED



How to identify the 220 Five-color ring resistor



4.Project code

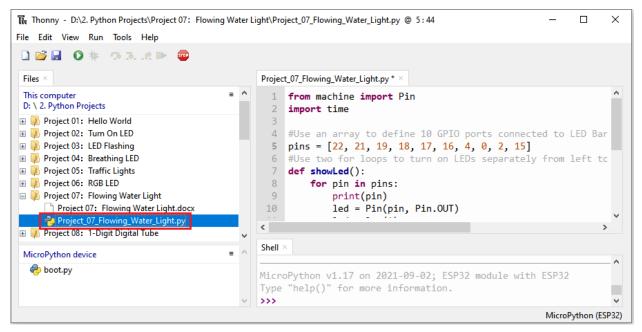
This project is designed to make a flowing water lamp. Which are these actions: First turn LED #1 ON, then turn it OFF. Then turn LED #2 ON, and then turn it OFF... and repeat the same to all 10 LEDs until the last LED is turns OFF. This process is repeated to achieve the "movements" of flowing water.

Codes used in this tutorial are saved in"2. Python Projects".

If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

Python Projects		—	
Share View			~ 🥐
> This PC > Software (D:) > 2. Pytl	hon Projects 🛛 🗸 🗸	ල 🔎 Searc	h 2. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AI	M File folder	
Project 01: Hello World	2/17/2022 10:21 AI	M File folder	
	2/17/2022 11:10 AI	M File folder	
Project 03: LED Flashing	2/17/2022 11:12 AI	M File folder	
Project 04: Breathing LED	2/17/2022 10:21 AI	M File folder	

Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 07Flowing Water Light", and double left-click "Project\_07\_Flowing\_Water\_Light.py".



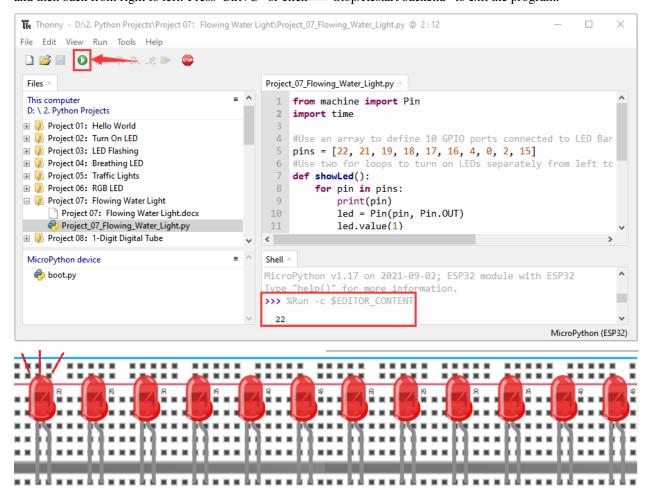
### from machine import Pin import time #Use an array to define 10 GPIO ports connected to LED Bar Graph for easier operation. pins = [22, 21, 19, 18, 17, 16, 4, 0, 2, 15] #Use two for loops to turn on LEDs separately from left to right and then back from. *→right* to left def showLed(): for pin in pins: print(pin) led = Pin(pin, Pin.OUT) led.value(1) time.sleep\_ms(100) led.value(♥) time.sleep\_ms(100) for pin in reversed(pins): print(pin) led = Pin(pin, Pin.OUT) led.value(1) time.sleep\_ms(100) led.value(0) time.sleep\_ms(100) while True: showLed()

#### 5.Project result

Make sure the ESP32 has been connected to the computer,  $click \overset{\textcircled{}}{=} "Stop/Restart backend"$ .

Thonny - D:\2. Python Projects\Project 07: Flowing	Water l	r Light\Project_07_Flowing_Water_Light.py @ 2:12 -	×
File Edit View Run Tools Help			
- 🗋 🚰 🛃 - 🔘 🚸 - 🗇 3 e 🕨 🛑 🛑		-	
Files $\times$		Project_07_Flowing_Water_Light.py * ×	
This computer D: \ 2. Python Projects D: \ 2. Python Projects D: \ 2. Python Project 01: Hello World D: \ 2. Project 01: Hello World D: \ 2. Project 02: Turn On LED D: \ 2. Project 03: LED Flashing D: \ 2. Project 03: LED Flashing D: \ 2. Project 04: Breathing LED D: \ 2. Project 05: Traffic Lights D: \ 2. Project 05: RGB LED D: \ 2. Project 07: Flowing Water Light.docx	= ^	<pre>1 from machine import Pin 2 import time 3 4 #Use an array to define 10 GPIO ports connected to LED F 5 pins = [22, 21, 19, 18, 17, 16, 4, 0, 2, 15] 6 #Use two for loops to turn on LEDs separately from left 7 def showLed(): 8 for pin in pins: 9 print(pin) 10 led = Pin(pin, Pin.OUT)</pre>	
<ul> <li>Project_07_Flowing_Water_Light.py</li> <li>Project 08: 1-Digit Digital Tube</li> </ul>	- 1	<	>
MicroPython device	= ^	Shell × MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32 Type "help()" for more information.	^
		MicroPython	(ESP32)

Click Circle Click

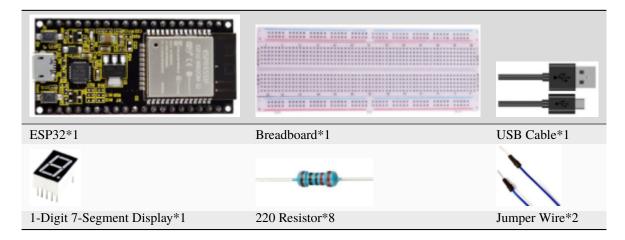


# 7.10 Project 081-Digit Digital Tube

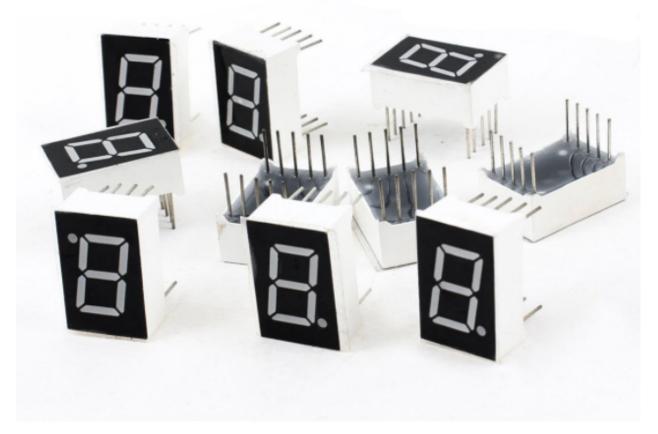
#### 1.Introduction

A 1-Digit 7-Segment Display is an electronic display device that displays decimal numbers. It is widely used in digital clocks, electronic meters, basic calculators and other electronic devices that display digital information. Even though they may not look modern enough, they are an alternative to more complex dot matrix displays and are easy to use in limited light conditions and strong sunlight. In this project, we will use ESP32 to control 1-Digit 7-segment display to display numbers.

2.Components



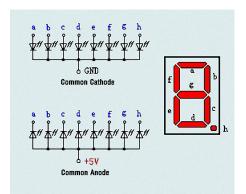
3.Component knowledge



**1-Digit 7-Segment Display principle:** Digital tube display is a semiconductor light emitting device, its basic unit is a light-emitting diode (LED). The digital tube display can be divided into 7-segment display and 8-segment display according to the number of segments. The 8-segment display has one more LED unit than the 7-segment display (used for decimal point display). Each segment of the 7-segment display is a separate LED. According to the connection mode of the LED unit, the digital tube can be divided into a common anode digital tube and a common cathode digital tube.

In the common cathode 7-segment display, all the cathodes (or negative electrodes) of the segmented LEDs are connected together, so you should connect the common cathode to GND. To light up a segmented LED, you can set its associated pin to "HIGH".

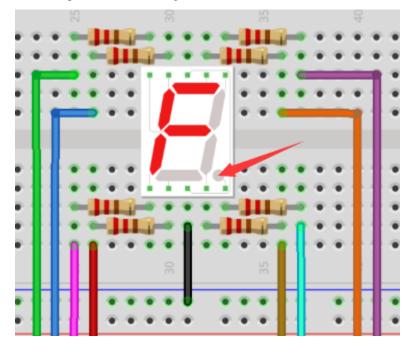
In the common anode 7-segment display, the LED anodes (positive electrodes) of all segments are connected together, so you should connect the common anode to"+5V". To light up a segmented LED, you can set its associated pin to"LOW".

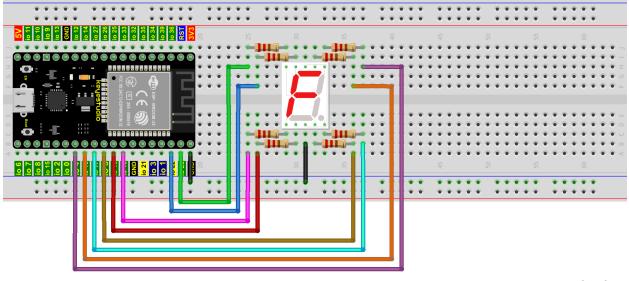


Each part of the digital tube is composed of an LED. So when you use it, you also need to use a current limiting resistor. Otherwise, the LED will be damaged. In this experiment, we use an ordinary common cathode one-digit digital tube. As we mentioned above, you should connect the common cathode to GND. To light up a segmented LED, you can set its associated pin to "HIGH".

#### 4.Wiring diagram

Note: The direction of the 7-segment display inserted into the breadboard is consistent with the wiring diagram, with one more point in the lower right corner.





### fritzing

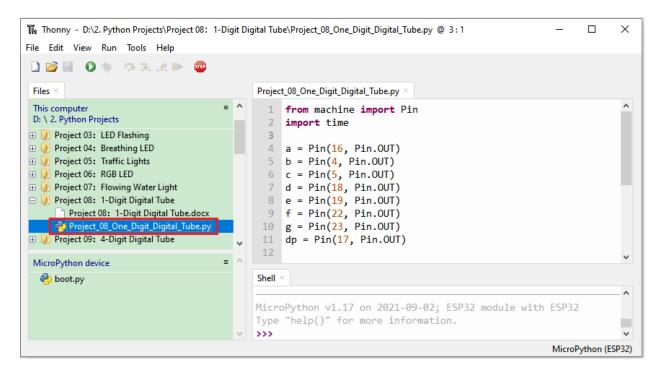
#### 5.Project code

The digital display is divided into 7 segments, and the decimal point display is divided into 1 segment. When certain numbers are displayed, the corresponding segment will be lit. For example, when the number 1 is displayed, segments b and c will be turned on.

Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		—	$\times$
Share View			~ ?
☐ > This PC > Software (D:) > 2. Python Proj	ects 〜 ご	🔎 Search 2. Pytho	on
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
Project 02:Turn On LED	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

 $Open "Thonny" click "This computer" \rightarrow "D:" \rightarrow "2. Python Projects" \rightarrow "Project 08: 1-Digit Digital Tube", and double left-click "Project_08_One_Digit_Digital_Tube.py".$ 



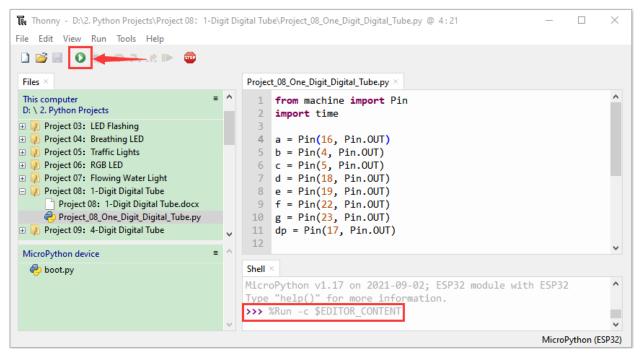
```
from machine import Pin
import time
a = Pin(16, Pin.OUT)
b = Pin(4, Pin.OUT)
c = Pin(5, Pin.OUT)
d = Pin(18, Pin.OUT)
e = Pin(19, Pin.OUT)
f = Pin(22, Pin.OUT)
g = Pin(23, Pin.OUT)
dp = Pin(17, Pin.OUT)
pins = [Pin(id,Pin.OUT) for id in [16, 4, 5, 18, 19, 22, 23, 17]]
def show(code):
    for i in range(0, 8):
        pins[i].value(~code & 1)
        code = code >> 1
#Select code from 0 to 9
mask_digits = [0xc0, 0xf9, 0xa4, 0xb0, 0x99, 0x92, 0x82, 0xf8,0x80, 0x90]
for code in reversed(mask_digits):
    show(code)
    time.sleep(1)
```

6.Project result

Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".

nis computer			t_08_One_Digit_Digital_Tube.py 🖄	
<ul> <li>1 2. Python Projects</li> <li>Project 03: LED Flashing</li> <li>Project 04: Breathing LED</li> <li>Project 05: Traffic Lights</li> <li>Project 06: RGB LED</li> <li>Project 07: Flowing Water Light</li> <li>Project 08: 1-Digit Digital Tube</li> <li>Project 08: 1-Digit Digital Tube.docx</li> <li>Project 09: 4-Digit Digital Tube</li> <li>Project 09: 4-Digit Digital Tube</li> </ul>		1 2 3 4 5 6 7 8 9 10 11	<pre>from machine import Pin import time a = Pin(16, Pin.OUT) b = Pin(4, Pin.OUT) c = Pin(5, Pin.OUT) d = Pin(18, Pin.OUT) e = Pin(19, Pin.OUT) g = Pin(22, Pin.OUT) dp = Pin(17, Pin.OUT)</pre>	^
AicroPython device ≡ ♦ boot.py	^	12 Shell	<	~

Click Circle Click

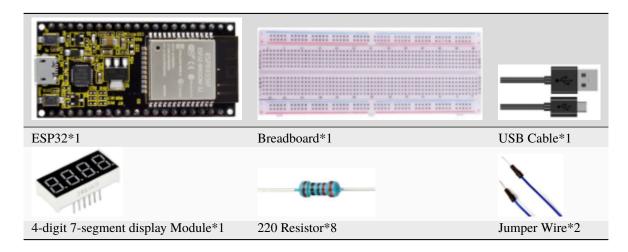


# 7.11 Project 094-digit Digital Tube

#### 1.Introduction

The 4-digit 7-segment display is a very practical display device and it is used for devices such as electronic clocks, score counters and the number of people in the park. Because of the low price, easy to use, more and more projects will use the 4 Digit 7-segment display. In this project, we use ESP32 to control the 4-digit 7-segment display to display digits.

2.Components

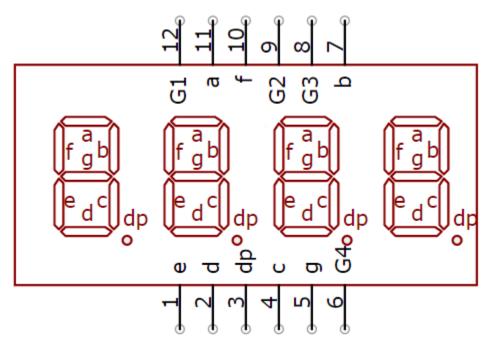


#### 3.Component Knowledge

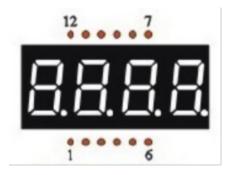


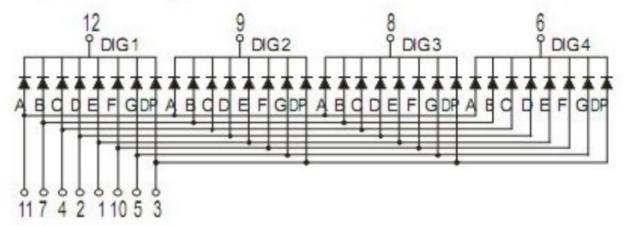
\*\*4-digit 7-segment display\*\*It is a device with common cathode and anode, its display principle is similar to the 1-Digit digital tube display. Both of them have eight GPIO ports to control the digital tube display, that is 8 leds. However, here is 4-digit, so it needs four GPIO ports to control the bit selection end. Our 4 - digit digital tube is common cathode.

The following figure shows the pin diagram of the 4-digit digital tube. G1, G2, G3 and G4 are the control pins.

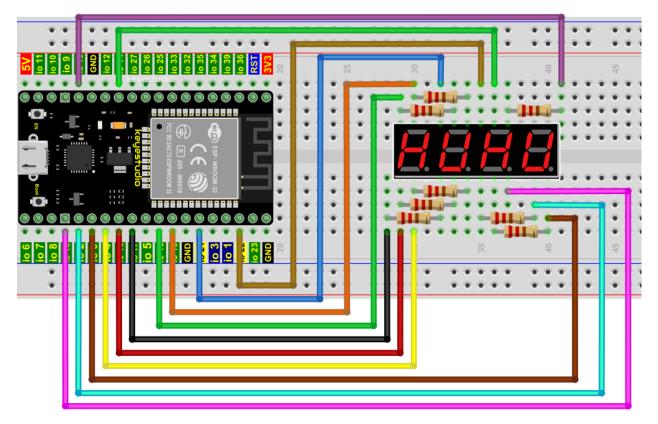


Schematic Diagram





4.Wiring Diagram

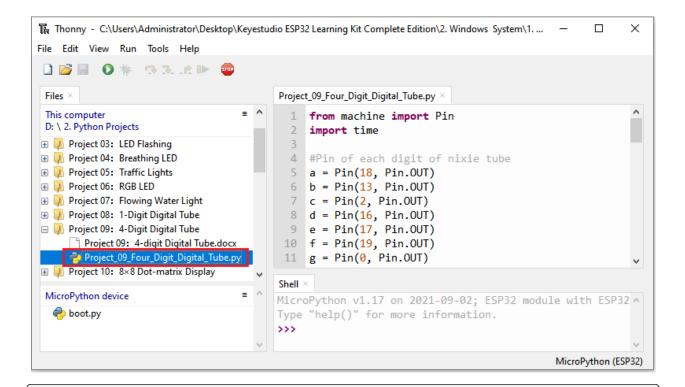


#### 5.Test Code

Codes used in this tutorial are saved in "2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

Python Projects Share View			_	· ×
> This PC > Software (D:) > 2. Pyth	on Projects 🛛 🗸 🗸	Ū	,	2. Python
Name	Date modified	1	Туре	Size 🖍
Project 00: Boot	2/17/2022 10:21 AI	M I	File folder	
Project 01: Hello World	2/17/2022 10:21 AI	M F	File folder	
Project 02: Turn On LED	2/17/2022 11:10 AI	M F	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AI	M F	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AI	M I	File folder	

 $Open "Thonny" click "This computer" \rightarrow "D:" \rightarrow "2. Python Projects" \rightarrow "Project 094-Digit Digital Tube", then double left-click "Project_09_Four_Digit_Digital_Tube.py".$ 



```
from machine import Pin
import time
#Pin of each digit of nixie tube
a = Pin(18, Pin.OUT)
b = Pin(13, Pin.OUT)
c = Pin(2, Pin.OUT)
d = Pin(16, Pin.OUT)
e = Pin(17, Pin.OUT)
f = Pin(19, Pin.OUT)
g = Pin(0, Pin.OUT)
dp = Pin(4, Pin.OUT)
G1 = Pin(21, Pin.OUT)
G2 = Pin(22, Pin.OUT)
G3 = Pin(14, Pin.OUT)
G4 = Pin(15, Pin.OUT)
#digital tube a to dp corresponding development board pins
d_Pins=[Pin(i,Pin.OUT) for i in [18,13,2,16,17,19,0,4]]
#Pin corresponding to digital tube segment G1, G2, G3, and G4
w_Pins=[Pin(i,Pin.OUT) for i in [21,22,14,15]]
number={
    '0':
    [1,1,1,1,1,1,0,0],#0
    '1':
    [0,1,1,0,0,0,0,0],#1
    '2':
    [1,1,0,1,1,0,1,0],#2
```

(continues on next page)

(continued from previous page)

```
'3':
    [1,1,1,1,0,0,1,0],#3
    '4':
    [0,1,1,0,0,1,1,0],#4
    '5':
    [1,0,1,1,0,1,1,0],#5
    '6':
    [1,0,1,1,1,1,1,0],#6
    '7':
    [1,1,1,0,0,0,0,0],#7
    '8':
    [1,1,1,1,1,1,1,0],#8
    '9':
    [1,1,1,1,0,1,1,0],#9
}
def display(num,dp):
    global number
    count=0
    for pin in d_Pins:#displays the value of num
        pin.value(number[num][count])
        count += 1
    if dp==1:
        d_Pins[7].value(0)
def clear():
    for i in w_Pins:
        i.value(≬)
    for i in d_Pins:
        i.value(1)
def showData(num):
   #the hundreds, thousands, ones, and fractional values of a numeric value
    d\_num=num
    location=d_num.find('.')
    if location>0:
        d_num=d_num.replace('.','')
        while len(d_num)<4:</pre>
            d_num = 0 + d_num
        for i in range(0,4):
            time.sleep(2)
            clear()
            w_Pins[3-i].value(1)
            if i==location-1:
                display(d_num[i],1)
            else:
                display(d_num[i], 0)
    if location<0:
        for i in range(0,4):
            time.sleep(2)
            clear()
            w_Pins[3-i].value(1)
            display(d_num[i],0)
while True:
```

(continues on next page)

(continued from previous page)

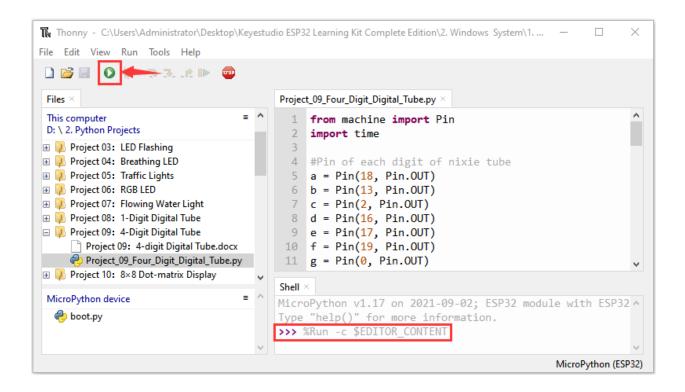
num='9016'
showData(num)

#### 6.Test Result

Make sure the ESP32 has been connected to the computer, then click <sup>999</sup> "Stop/Restart backend".

ile Edit View Run Tools Help	
Files × This computer = ^ D: \ 2. Python Projects Project 03: LED Flashing Project 04: Breathing LED Project 05: Traffic Lights Project 05: Traffic Lights Project 07: Flowing Water Light Project 08: 1-Digit Digital Tube Project 09: 4-Digit Digital Tube.docx Project 09: 4-digit Digital Tube.docx Project 09: 4-digit Digital Tube.docx Project 09: 4-digit Digital Tube.docx	Project_09_Four_Digit_Digital_Tube.py × 1 from machine import Pin 2 import time 3 4 #Pin of each digit of nixie tube 5 a = Pin(18, Pin.OUT) 6 b = Pin(13, Pin.OUT) 7 c = Pin(2, Pin.OUT) 8 d = Pin(16, Pin.OUT) 9 e = Pin(17, Pin.OUT) 10 f = Pin(19, Pin.OUT) 11 g = Pin(0, Pin.OUT)
MicroPython device =	Shell × MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32 ^ Type "help()" for more information. >>>

Click Circle Click

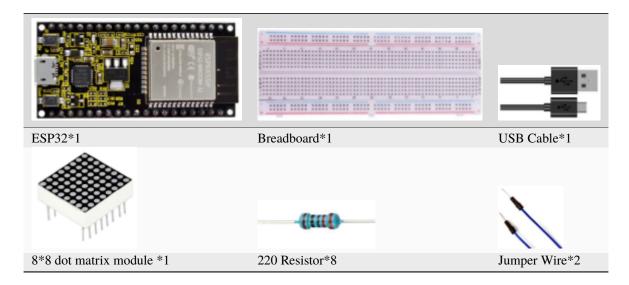


### 7.12 Project 108×8 Dot-matrix Display

#### 1.Introduction

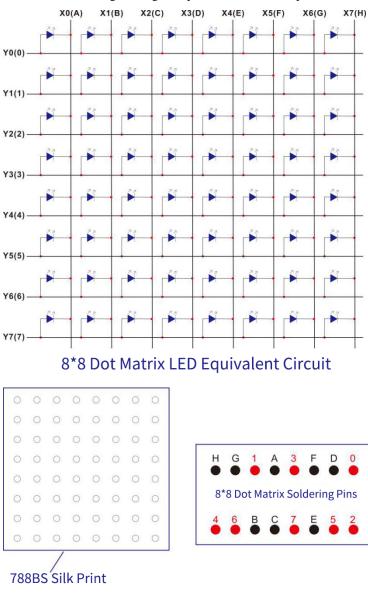
Dot matrix display is an electronic digital display device that can display information on machine, clocks, public transport departure indicators and many other devices. In this project, we will use ESP32 to control 8x8 LED dot matrix in a way that lights it up.

#### 2.Components



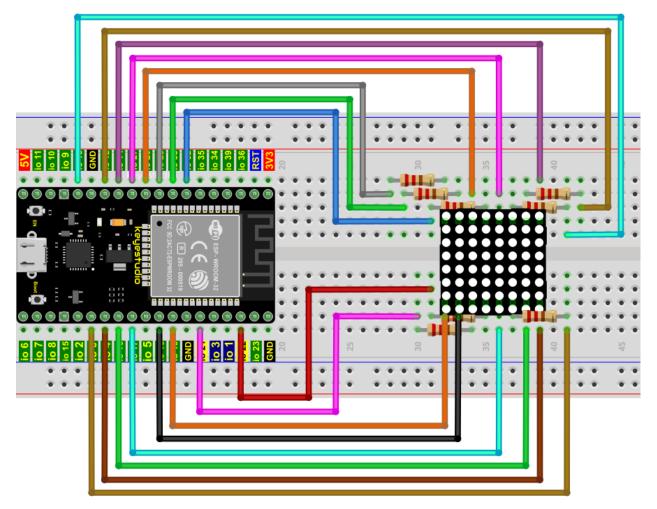
#### 3.Component Knowledge

**8\*8** dot matrix module The 8\*8 dot matrix is composed of 64 LEDs, including row common anode and row common cathode. Our module is row common anode, each row has a line connecting the positive pole of the LED, and the column is connecting the negative pole of the LED lamp, as shown in the following figure :



### 8\*8 Dot Matrix Outlook and Pinouts

4.Wiring Diagram



#### 5.Test Code

The code used in this tutorial is saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		—	$\Box$ $\times$
Share View			~ 🕐
→ This PC → Software (D:) → 2. Pyth	non Projects 🗸 🗸 🗸	ට 🔎 Search	2. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
Project 02: Turn On LED	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

 $Open "Thonny" click "This computer" \rightarrow "D:" \rightarrow "2. Python Projects" \rightarrow "Project 108 \times 8 Dot-matrix Display", then double left-click "Project_10_8 \times 8_Dot_Matrix_Display.py".$ 

l 🖻 🖩 🛛 🐐 🔿 3e 🕨 🥮		
Files ×	Project_10_8×8_Dot_Matrix_Display.py ×	
This computer = Chi \ 2. Python Projects = Project 05: Traffic Lights = Project 06: RGB LED = Project 07: Flowing Water Light = Project 08: 1-Digit Digital Tube = Project 09: 4-Digit Digital Tube = Project 10: 8×8 Dot-matrix Display.docx = Project 10: 8×8 Dot-matrix Display.docx = Project 11: 74HC595N Control 8 LEDs = Project 11: 74HC595N Control 8 LEDs = Project 11: 40 + 1 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 +	<pre>1 from machine import Pin 2 import time 3 4 #Define the pin of the row and Set to output. 5 row1 = Pin(14, Pin.OUT) 6 row2 = Pin(26, Pin.OUT) 7 row3 = Pin(4, Pin.OUT) 8 row4 = Pin(27, Pin.OUT) 9 row5 = Pin(19, Pin.OUT) 10 row6 = Pin(16, Pin.OUT) 11 row7 = Pin(18, Pin.OUT)</pre>	
Project 12: Active Buzzer      MicroPython device	Shell × MicroPython v1.17 on 2021-09-02; ESP32 module with	SP32
e boot.py	Type "help()" for more information.	- 25 22 7

from machine import Pin
import time
#Define the pin of the row and Set to output.
row1 = Pin(14, Pin.OUT)
row2 = Pin(26, Pin.OUT)
row3 = Pin(4, Pin.OUT)
<pre>row4 = Pin(27, Pin.OUT)</pre>
row5 = Pin(19, Pin.OUT)
row6 = Pin(16, Pin.OUT)
row7 = Pin(18, Pin.OUT)
row8 = Pin(17, Pin.OUT)
#Define the pins of the column and Set to output
col1 = Pin(32, Pin.0UT)
col2 = Pin(21, Pin.0UT)
col3 = Pin(22, Pin.OUT)
col4 = Pin(12, Pin.OUT)
<pre>col5 = Pin(0, Pin.OUT) col6 = Pin(13, Pin.OUT)</pre>
col7 = Pin(33, Pin.OUT)
col8 = Pin(25, Pin.001)
cord = rin(23, rin, oor)
#Sets the pin of the column to low level
col1.value(0)
col2.value(0)
col3.value(0)
col4.value(0)
col5.value(0)
col6.value(0)
<pre>col7.value(0)</pre>
col8.value(♥)

(continues on next page)

(continued from previous page)

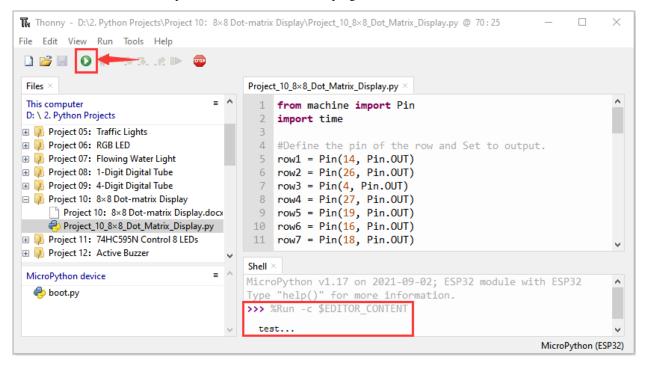
```
#Since the column of the lattice has been set to low level,
#the corresponding row of the lattice will light up when the pin of the row is at high.
\rightarrow level
def Row(d):
    if(d ==1):
        row1.value(1) #Light the first line
    if(d ==2):
        row2.value(1) #Light the second line
    if(d ==3):
        row3.value(1)
    if(d ==4):
        row4.value(1)
    if(d ==5):
        row5.value(1)
    if(d ==6):
        row6.value(1)
    if(d ==7):
        row7.value(1)
    if(d ==8):
        row8.value(1)
#Close the lattice
def off():
    row1.value(◊)
    row2.value(♥)
   row3.value())
   row4.value())
    row5.value())
   row6.value(≬)
   row7.value())
    row8.value(≬)
try:
    print("test...")
    while True:
         for num in range(1,10): #Light the lattice line by line
             Row(num)
             if(num == 9): #Because the lattice has only 8 rows, and I'm limiting it.
\rightarrow here, is equal to 9
                off()
                            #Close the lattice
             time.sleep(0.2)
except:
    pass
```

6.Test Result

Make sure the ESP32 has been connected to the computer, then click = 1000 (Stop/Restart backend").

iles ×		Project_10_8×8_Dot_Matrix_Display.py ×	
Inscriptient Inscriptient Inscriptient Project 05: Traffic Lights Project 06: RGB LED Project 07: Flowing Water Light Project 08: 1-Digit Digital Tube Project 09: 4-Digit Digital Tube Project 09: 4-Digit Digital Tube Project 09: 4-Digit Digital Tube Project 10: 8×8 Dot-matrix Display Project 10: 8×8 Dot-matrix Display.do Project 10: 8×8 Dot-Matrix_Display.do Project 11: 74HC595N Control 8 LEDs	1	<pre>1 from machine import Pin 2 import time 3 4 #Define the pin of the row and Set to output. 5 row1 = Pin(14, Pin.OUT) 6 row2 = Pin(26, Pin.OUT) 7 row3 = Pin(4, Pin.OUT) 8 row4 = Pin(27, Pin.OUT) 9 row5 = Pin(19, Pin.OUT) 10 row6 = Pin(16, Pin.OUT) 11 row7 = Pin(18, Pin.OUT)</pre>	,
Project 12: Active Buzzer           /icroPython device	•	Shell × MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32	

Click Circle Click



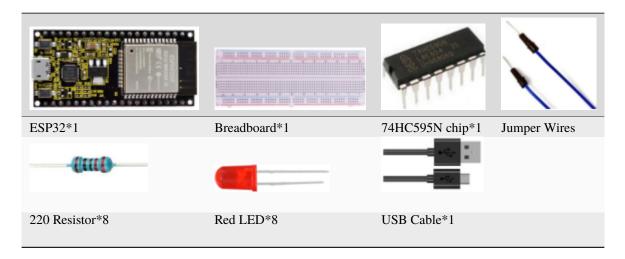
# 7.13 Project 1174HC595N Control 8 LEDs

#### 1.Introduction

In previous projects, we learned how to light up an LED.

With only 32 IO ports on ESP32, how do we light up a lot of leds? Sometimes it is possible to run out of pins on the ESP32, and you need to extend it with the shift register. You can use the 74HC595N chip to control 8 outputs at a time, taking up only a few pins on your microcontroller. In addition, you can also connect multiple registers together to further expand the output. In this project, we will use ESP32, 74HC595 chip and LED to make a flowing water light to understand the function of the 74HC595 chip.

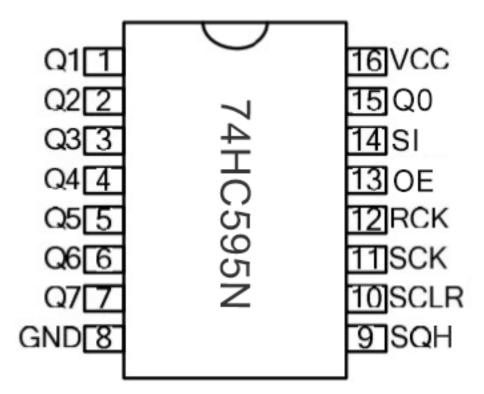
2.Components



3.Component knowledge



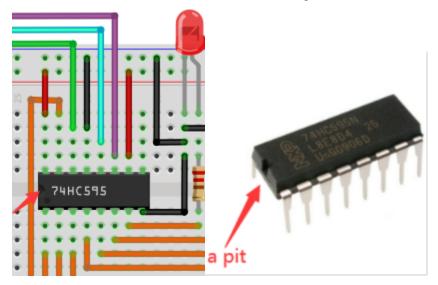
**74HC595N Chip:** The 74HC595 chip is used to convert serial data into parallel data. A 74HC595 chip can convert the serial data of one byte into 8 bits, and send its corresponding level to each of the 8 ports correspondingly. With this characteristic, the 74HC595 chip can be used to expand the IO ports of an ESP32. At least 3 ports are required to control the 8 ports of the 74HC595 chip.

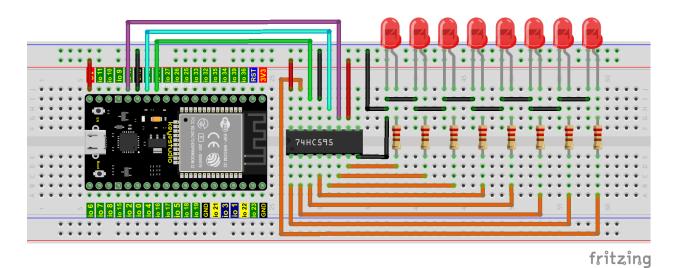


The ports of the 74HC595 chip are described as follows

#### 4.Wiring diagram

Note: Note the orientation in which the 74HC595N chip is inserted.





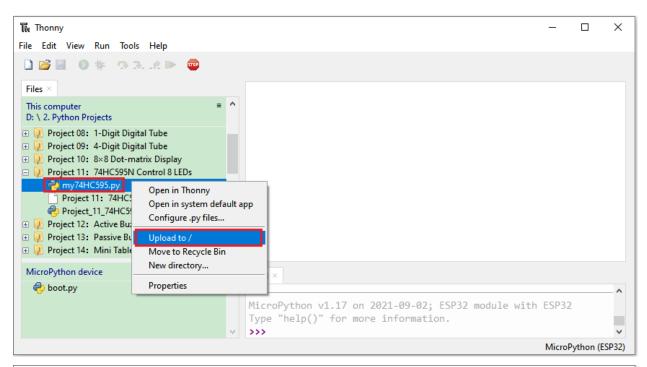
#### 5.Project code

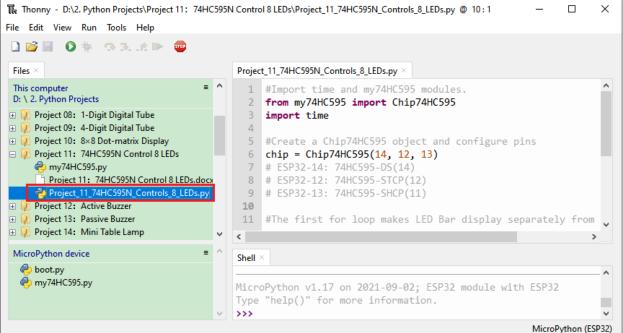
Codes used in this tutorial are saved in "2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		_	$\Box$ $\times$
Share View			~ 🕐
→ This PC → Software (D:) → 2. Pyth	on Projects 🛛 🗸 🗸	ට 🔎 Sea	rch 2. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AN	A File folder	
Project 01: Hello World	2/17/2022 10:21 AN	A File folder	
Project 02: Turn On LED	2/17/2022 11:10 AN	A File folder	
Project 03: LED Flashing	2/17/2022 11:12 AN	A File folder	
Project 04: Breathing LED	2/17/2022 10:21 AN	A File folder	

Open"Thonny"click"This computer"→"D:"→"2. Python Projects"→"Project 1174HC595N Control 8 LEDs".

Select"my74HC595.py", right click your mouse to select"Upload to /"wait for "my74HC595.py" to be uploaded to ESP32, and then double left-click "Project\_11\_74HC595N\_Controls\_8\_LEDs.py".





#Import time and my74HC595 modules. from my74HC595 import Chip74HC595 import time #Create a Chip74HC595 object and configure pins chip = Chip74HC595(14, 12, 13) # ESP32-14: 74HC595-DS(14) # ESP32-12: 74HC595-STCP(12) # ESP32-13: 74HC595-STCP(11) (continues on next page)

(continued from previous page)

```
#The first for loop makes LED Bar display separately from left to right
#while the second for loop make it display separately from right to left.
while True:
    x = 0x01
    for count in range(8):
        chip.shiftOut(1, x)
        x = x<<1;
        time.sleep_ms(300)
    x = 0x01
    for count in range(8):
        chip.shiftOut(0, x)
        x = x<<1
        time.sleep_ms(300)</pre>
```

#### 6.Project result

Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".

Project 12: Active Buzzer  Project 13: Passive Buzzer  Received and the second	J	10	#The first for loop makes LED Bar display separately from
<ul> <li>Project 09: 4-Digit Digital Tube</li> <li>Project 10: 8×8 Dot-matrix Display</li> <li>Project 11: 74HC595N Control 8 LEDs</li> <li>my74HC595.py</li> <li>Project 11: 74HC595N Control 8 LEDs.docx</li> <li>Project 11: 74HC595N_Control 8 LEDs.docx</li> </ul>		6 7	<pre>#Create a Chip74HC595 object and configure pins chip = Chip74HC595(14, 12, 13) # ESP32-14: 74HC595-DS(14) # ESP32-12: 74HC595-STCP(12) # ESP32-13: 74HC595-SHCP(11)</pre>
Files × This computer = D: \ 2. Python Projects D: \ 2. Python Project 08: 1-Digit Digital Tube D: Project 09: 4-Digit Digital Tube		1 2 3	t_11_74HC595N_Controls_8_LEDs.py × #Import time and my74HC595 modules. from my74HC595 import Chip74HC595 import time

Click O"Run current script", the code starts to be executed and you'll see that the 8 LEDs start flashing in flowing water mode. Press"Ctrl+C"or click ""Stop/Restart backend" to exit the program.

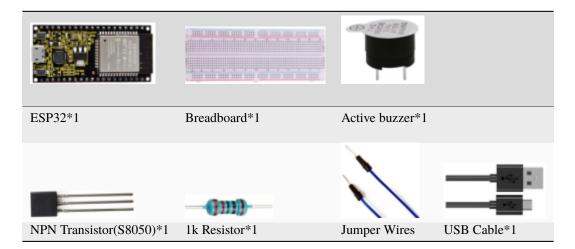
Files ×	Pro	ect_11_74HC595N_Controls_8_LEDs.py $\times$
This computer D: \ 2. Python Projects D: \ 2. Python Projects D: \ 2. Python Project 09: 4-Digit Digital Tube D: Project 09: 4-Digit Digital Tube D: Project 10: 8×8 Dot-matrix Display D: Project 11: 74HC595N Control 8 LEDs Project 11: 74HC595N Control 8 LEDs.docx Project 11: 74HC595N Control 8 LEDs.docx Project 11: 74HC595N_Controls_8_LEDs.py D: Project 12: Active Buzzer D: \ 2. Project 13: Passive Buzzer	_	<pre>#Import time and my74HC595 modules. from my74HC595 import Chip74HC595 import time #Create a Chip74HC595 object and configure pins chip = Chip74HC595(14, 12, 13) # ESP32-14: 74HC595-DS(14) # ESP32-12: 74HC595-STCP(12) # ESP32-13: 74HC595-SHCP(11)</pre>
Project 14: Mini Table Lamp  MicroPython device	× <	>
e boot.py boot.py my74HC595.py	Тур	<pre>roPython v1.17 on 2021-09-02; ESP32 module with ESP32 e "help()" for more information. %Run -c \$EDITOR_CONTENT</pre>

### 7.14 Project 12Active Buzzer

1.Introduction

Active buzzer is a sound component that is widely used as a sound component for computersprintersalarmselectronic toys and phonestimers etc. It has an internal vibration source, just by connecting to a 5V power supply, it can continuously buzz. In this project, we will use ESP32 to control the active buzzer to beep.

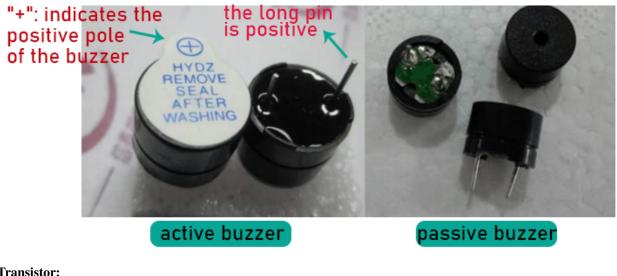
2.Components



3.Component knowledge



Active buzzer: Active buzzer inside has a simple oscillator circuit, which can convert constant direct current into a certain frequency pulse signal. Once active buzzer receives a high level, it will produce sound. Passive buzzer is an internal without vibration source integrated electronic buzzer, it must be driven by 2k to 5k square wave, rather than a DC signal. The two buzzers are very similar in appearance, but one buzzer with a green circuit board is a passive buzzer, while the other buzzer with black tape is an active buzzer. Passive buzzers don't have positive polarity, but active buzzers have. As shown below:

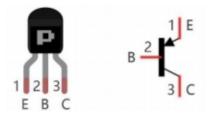


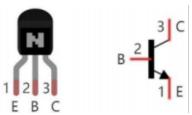
#### **Transistor:**



Because the buzzer requires such large current that GPIO of ESP32 output capability cannot meet the requirement, a transistor of NPN type is needed here to amplify the current.

Transistor, the full name: semiconductor transistor, is a semiconductor device that controls current. Transistorcan be used to amplify weak signal, or works as a switch. It has three electrodes(PINs): base (b), collector © and emitter (e). When there is current passing between "be", "ce" will allow several-fold current (transistor magnification) pass, at this point, transistor works in the amplifying area. When current between "be" exceeds a certain value, "ce" will not allow current to increase any longer, at this point, transistor works in the saturation area. Transistor has two types as shown below: PNP and NPN.



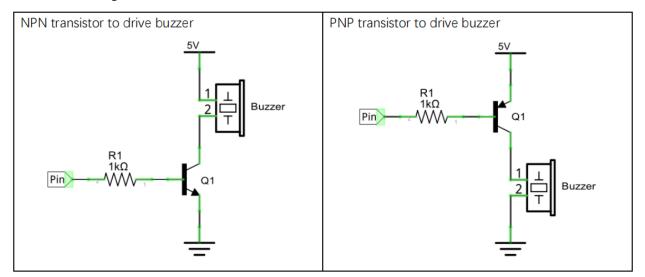


In our kit, the PNP transistor is marked with 8550, and the NPN transistor is marked with 8050.

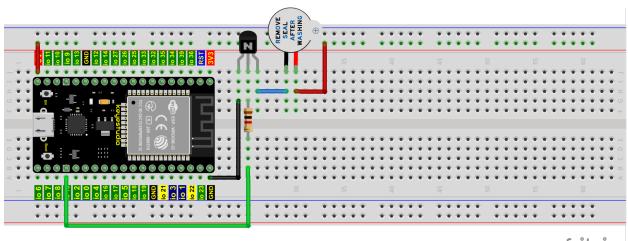
Based on the transistor's characteristics, it is often used as a switch in digital circuits. As micro-controller's capacity to output current is very weak, we will use transistor to amplify current and drive large-current components.

When using NPN transistor to drive buzzer, we often adopt the following method. If GPIO outputs high level, current will flow through R1, the transistor will get conducted, and the buzzer will sound. If GPIO outputs low level, no current flows through R1, the transistor will not be conducted, and buzzer will not sound.

When using PNP transistor to drive buzzer, we often adopt the following method. If GPIO outputs low level, current will flow through R1, the transistor will get conducted, and the buzzer will sound. If GPIO outputs high level, no current flows through R1, the transistor will not be conducted, and buzzer will not sound.



4.Wiring diagram



fritzing

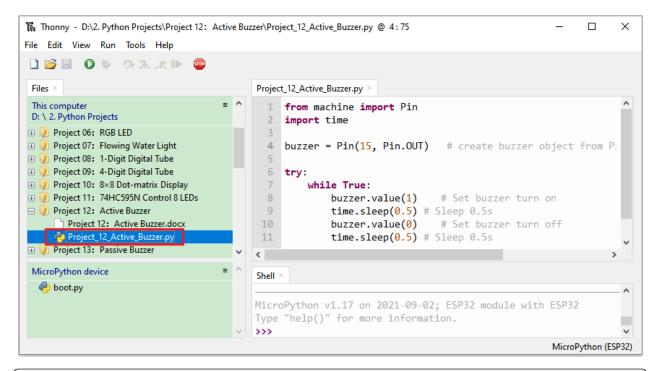
Note: The buzzer power supply in this circuit is 5V. On a 3.3V power supply, the buzzer can work, but will reduce the loudness.

5.Project code

Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		—	$\times$
Share View			~ ?
→ This PC → Software (D:) → 2. Python Proje	ects ~ ె	,○ Search 2. Pyth	on
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
- Project 02: Turn On LED	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

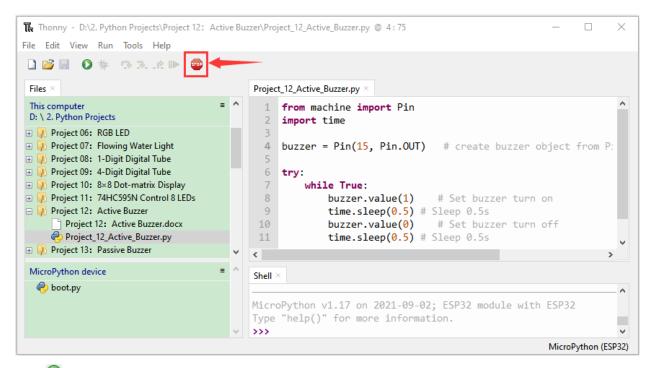
Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 12: Active Buzzer", and then double left-click "Project\_12\_Active\_Buzzer.py".



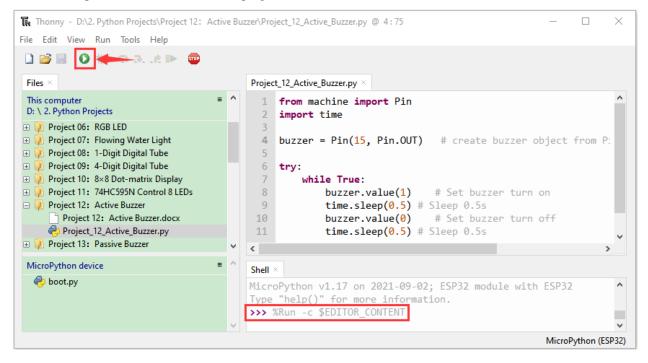
```
from machine import Pin
import time
buzzer = Pin(15, Pin.OUT)  # create buzzer object from Pin 15, Set Pin 15 to output
try:
    while True:
        buzzer.value(1)  # Set buzzer turn on
        time.sleep(0.5) # Sleep 0.5s
        buzzer.value(0)  # Set buzzer turn off
        time.sleep(0.5) # Sleep 0.5s
except:
    pass
```

6.Project result

Make sure the ESP32 has been connected to the computer,  $click \stackrel{\text{result}}{=} "Stop/Restart backend"$ .



Click C'Run current script", the code starts to be executed and you'll see that the active buzzer beeps. Press"Ctrl+C"or click Ctrl+C start backend" to exit the program.

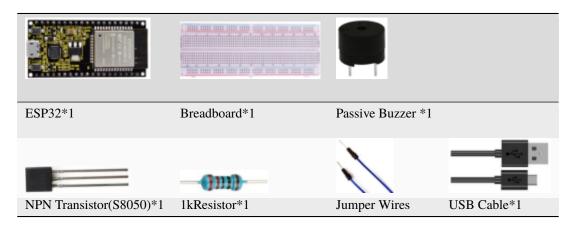


### 7.15 Project 13Passive Buzzer

1.Introduction:

In a previous project, we studied an active buzzer, which can only make a sound and may make you feel very monotonous. In this project, we will learn a passive buzzer and use the ESP32 control it to work. Unlike the active buzzer, the passive buzzer can emit sounds of different frequencies.

2.Components



3.Component knowledge



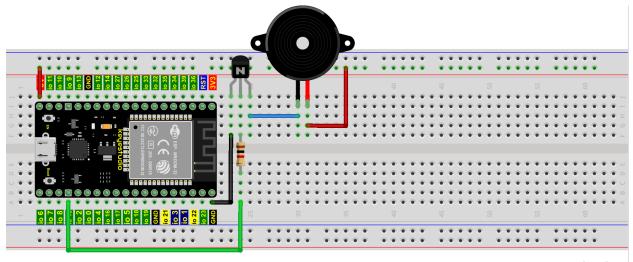
Passive buzzer: A passive buzzer is an integrated electronic buzzer with no internal vibration source and it has to be

driven by 2K-5K square waves, not DC signals. The two buzzers are very similar in appearance, but one buzzer with a green circuit board is a passive buzzer and the other buzzer with black tape is an active buzzer. Passive buzzers cannot distinguish between positive polarity while active buzzers can.



Transistor: Please refer to Project 12.

4. Wiring diagram:



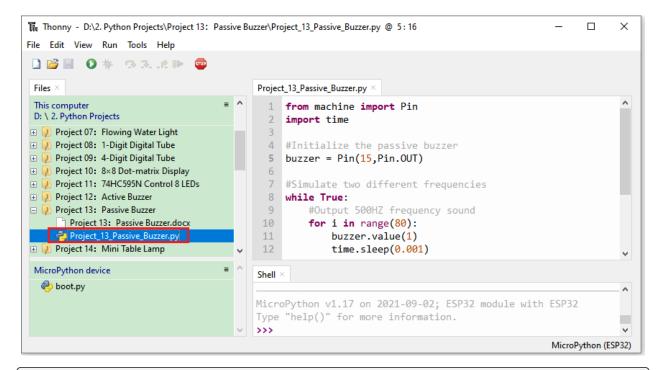
fritzing

#### 5.Project code

Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		_	$\Box$ $\times$
Share View			~ 🕐
→ This PC → Software (D:) → 2. Pyth	hon Projects 🗸 🗸	,	. Python
Name	Date modified	Туре	Size ^
	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
	2/17/2022 11:10 AM	File folder	
	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

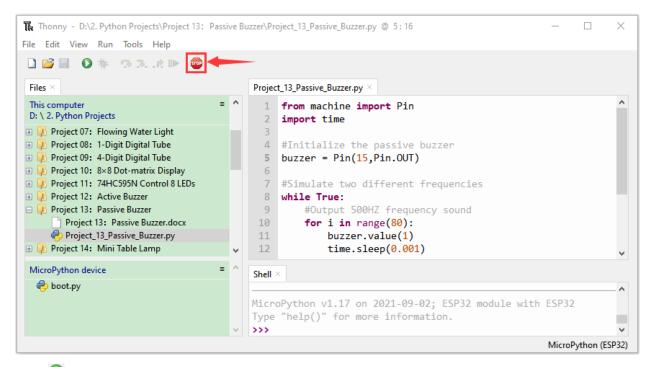
Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 13: Passive Buzzer", and then double left-click "Project\_13\_Passive\_Buzzer.py".

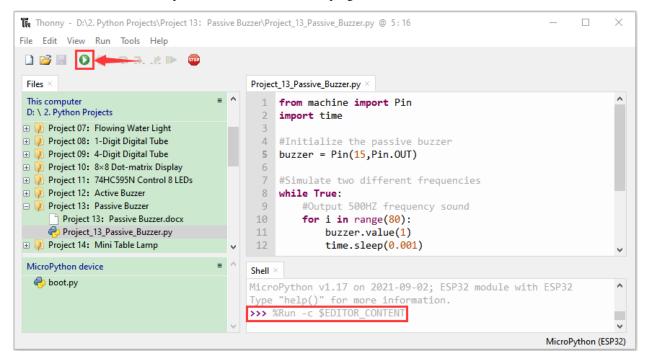


```
from machine import Pin
import time
#Initialize the passive buzzer
buzzer = Pin(15, Pin.OUT)
#Simulate two different frequencies
while True:
    #Output 500HZ frequency sound
    for i in range(80):
        buzzer.value(1)
        time.sleep(0.001)
        buzzer.value(0)
        time.sleep(0.001)
    #Output 250HZ frequency sound
    for i in range(100):
       buzzer.value(1)
        time.sleep(0.002)
        buzzer.value(0)
        time.sleep(0.002)
```

6.Project result

Make sure the ESP32 has been connected to the computer, click <sup>10</sup> "Stop/Restart backend".



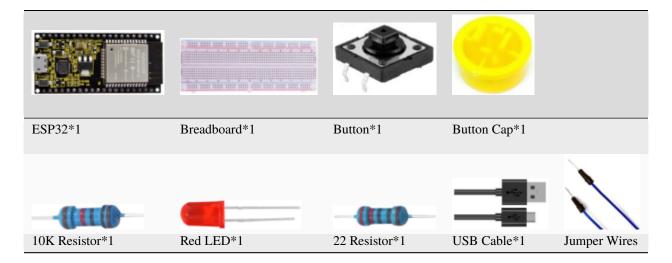


# 7.16 Project 14: Mini Table Lamp

1.Introduction

Do you know that the ESP32 can light up an LED when you press a button? In this project, we will use ESP32a button switch and an LED to make a mini table lamp.

2.Components



3.Component knowledge



Button: A button can control the circuit on and off, the button is plugged into a circuit, the circuit is disconnected when the button is not pressed. The circuit works when you press the button, but breaks again when you release it. Why does it only work when you press it? It starts from the internal structure of the button, which don't allow current to travel from one end of the button to the other before it is pressed; When pressed, a metal strip inside the button connects the two sides to allow electricity to pass through.



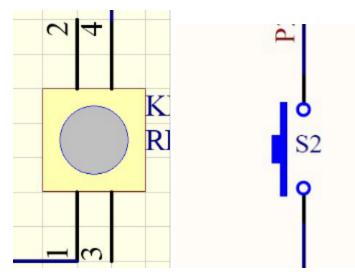
. Before the button is pressed, 1 and 2 are

The internal structure of the button is shown in the figure on, 3 and 4 are also on, but 1, 3 or 1, 4 or 2, 3 or 2, 4 are off(not working). Only when the button is pressed, 1, 3 or 1,

4 or 2, 3 or 2, 4 are on.

The button switch is one of the most commonly used components in circuit design.

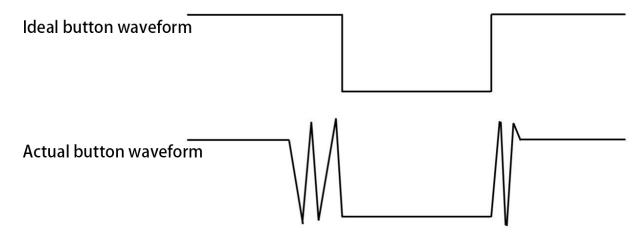
### Schematic diagram of the button:



### What is button shake javascript?

We think of the switch circuit as "press the button and turn it on immediately", "press it again and turn it off immediately". In fact, this is not the case.

The button usually uses a mechanical elastic switch, and the mechanical elastic switch will produce a series of shake javascript due to the elastic action at the moment when the mechanical contact is opened and closed (usually about 10ms). As a result, the button switch will not immediately and stably turn on the circuit when it is closed, and it will not be completely and instantaneously disconnected when it is turned off.



### How to eliminate the [shake](javascript:;)?

There are two common methods, namely fix [shake](javascript:;) in the software and hardware. We only discuss the [shake](javascript:;) removal in the software.

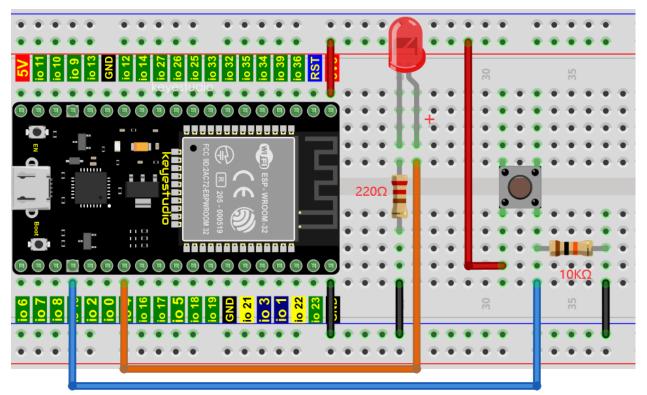
We already know that the [shake](javascript:;) time generated by elasticity is about 10ms, and the delay command can be used to delay the execution time of the command to achieve the effect of [shake](javascript:;) removal.

Therefore, we delay 0.02s in the code to achieve the key anti-shake function.



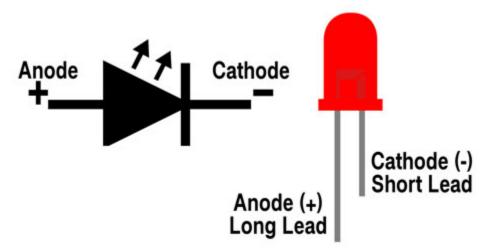


4.Wiring Diagram

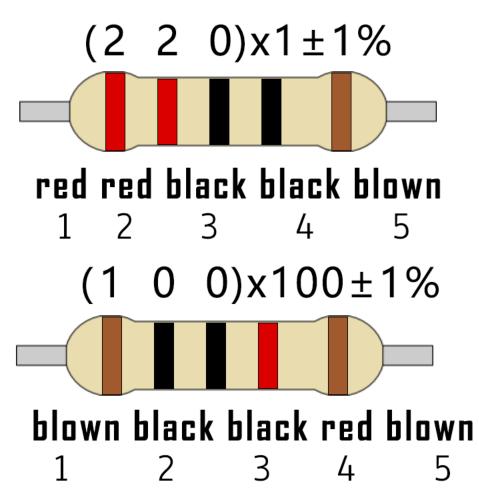


### Note:

How to connect the LED

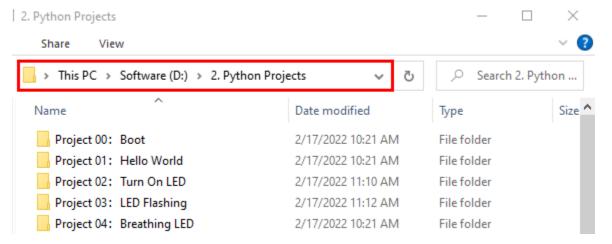


How to identify the 220 5-band resistor and 10K 5-band resistor

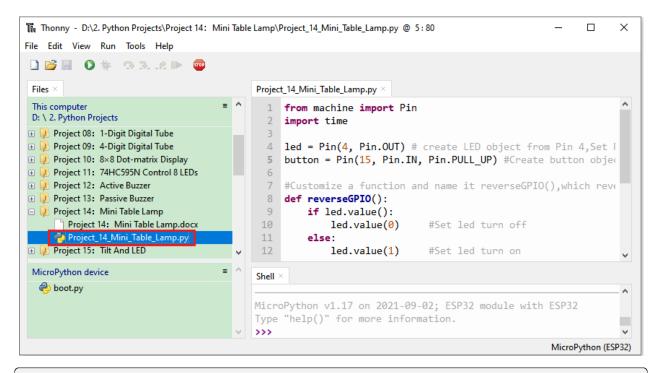


5.Project code

Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes



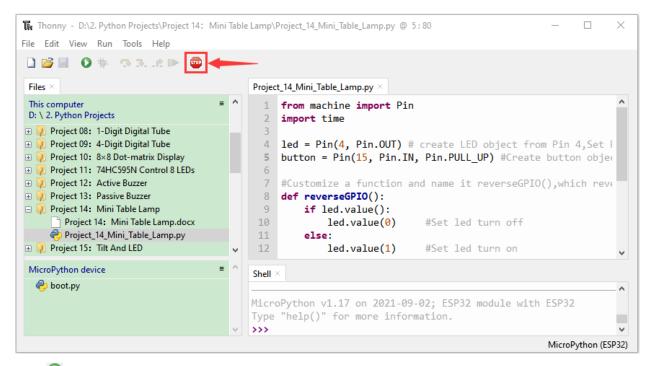
Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 14: Mini Table Lamp", and then double left-click "Project\_14\_Mini\_Table\_Lamp.py".



```
from machine import Pin
import time
led = Pin(4, Pin.OUT) # create LED object from Pin 4,Set Pin 4 to output
button = Pin(15, Pin.IN, Pin.PULL_UP) #Create button object from Pin15, Set GP15 to input
#Customize a function and name it reverseGPIO(), which reverses the output level of the
\hookrightarrow LED
def reverseGPIO():
    if led.value():
        led.value(♥)
                          #Set led turn off
    else:
        led.value(1)
                          #Set led turn on
try:
    while True:
        if not button.value():
            time.sleep_ms(20)
            if not button.value():
                reverseGPI0()
                while not button.value():
                    time.sleep_ms(20)
except:
    pass
```

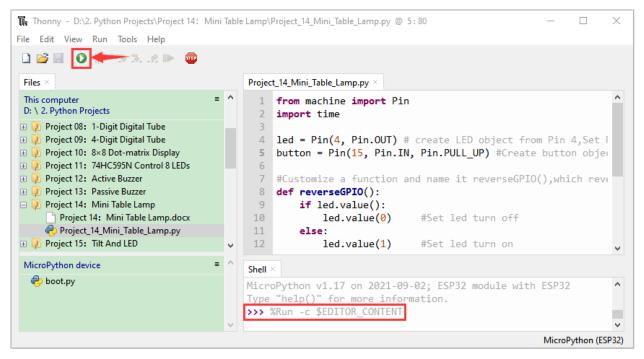
6.Project result

Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".



Click Circle Click Circle Content script", the code starts to be executed and you'll see that press the push button switch, the LED turns on. When it is released, the LED is still on. Press it again, and the LED turns off. When it is released, the LED

stays off. Doesn't it look like a mini table lamp? Press"Ctrl+C"or click ""Stop/Restart backend"to exit the program.



# 7.17 Project 15Tilt And LED

#### 1.Introduction

The ancients without electronic clock, so the hourglass are invented to measure time. The hourglass has a large capacity on both sides, and which is filled with fine sand on one side. What's more, there is a small channel in the middle, which can make the hourglass stand upright, the side with fine sand is on the top. due to the effect of gravity, the fine sand will flow down through the channel to the other side of the hourglass. When the sand reaches the bottom, turn it upside down and record the number of times it has gone through the hourglass, therefore, the next day we can know the approximate time of the day by it.

In this project, we will use ESP32 to control the tilt switch and LED lights to simulate an hourglass and make an electronic hourglass.

2.Components

	5500 5145344		-((11())-
ESP32*1	Tilt Switch*1	Red LED*4	10K Resistor*1
Breadboard*1	220 Resistor*4	USB Cable*1	Jumper Wires

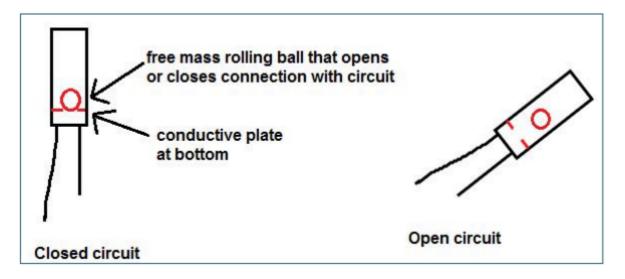
3.Component knowledge



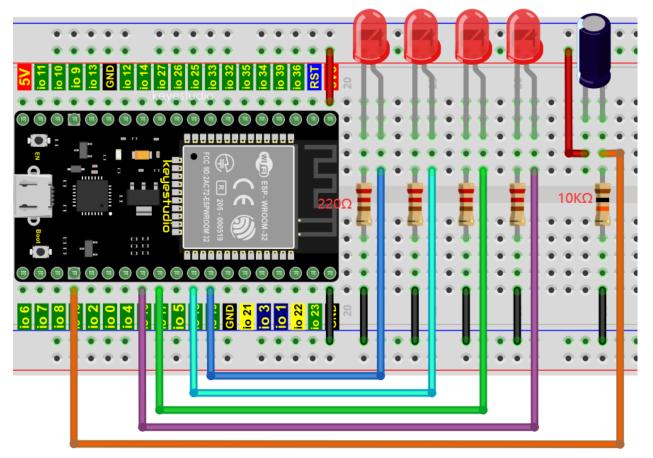
Tilt switch is also called digital switch. Inside is a metal ball that can roll. The principle of rolling the metal ball to contact with the conductive plate at the bottom, which is used to control the on and off of the circuit. When it is a rolling ball tilt sensing switch with single directional trigger, the tilt sensor is tilted toward the trigger end (two gold-plated pin ends), the tilt switch is in a closed circuit and the voltage at the analog port is about 5V(binary number is 1023).

In this way, the LED will light up. When the tilting switch is in horizontal position or tilting to the other end, the tilting switch is in open state the voltage of the analog port is about 0V (binary number is 0), the LED will turn off. In the program, we judge the state of the switch based on whether the voltage value of the analog port is greater than 2.5V (binary number is 512).

The internal structure of the tilt switch is used here to illustrate how it works, as shown below:

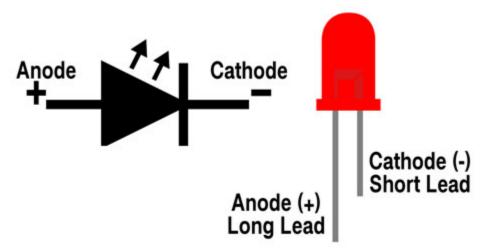


# 4.Wiring Diagram

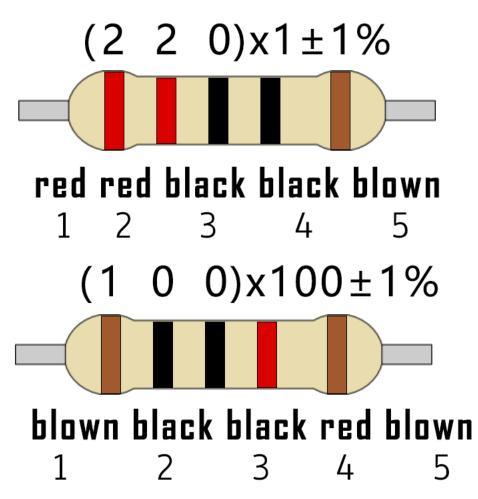


# Note:

How to connect the LED

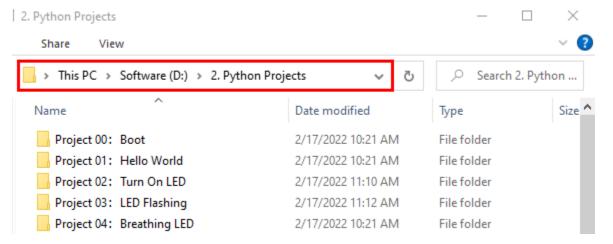


How to identify the 220 5-band resistor and 10K 5-band resistor

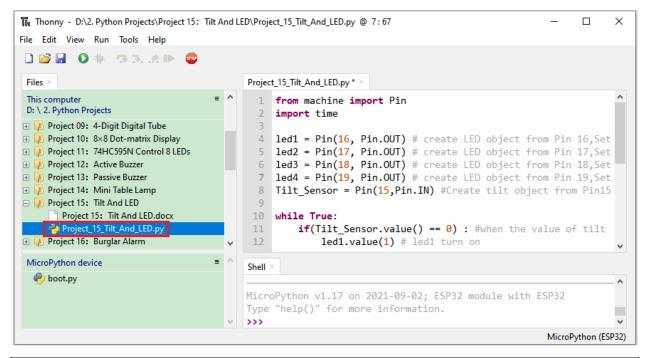


5.Project code

Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes



Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 15: Tilt And LED", and then double left-click "Project\_15\_Tilt\_And\_LED.py".

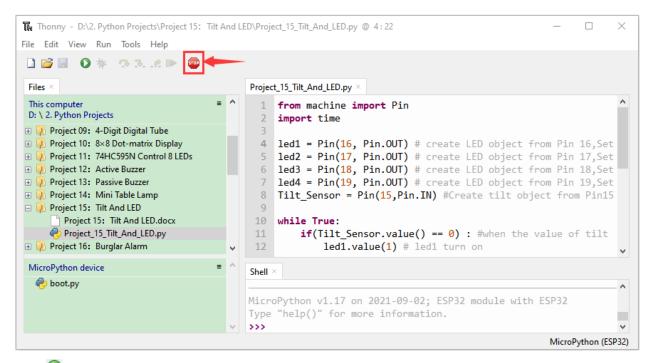


# from machine import Pin

```
import time
led1 = Pin(16, Pin.OUT) # create LED object from Pin 2,Set Pin 2 to output
led2 = Pin(17, Pin.OUT) # create LED object from Pin 0, Set Pin 0 to output
led3 = Pin(18, Pin.OUT) # create LED object from Pin 4,Set Pin 4 to output
led4 = Pin(19, Pin.OUT) # create LED object from Pin 16,Set Pin 16 to output
Tilt_Sensor = Pin(15, Pin. IN) #Create tilt object from Pin15, Set GP15 to input
while True:
    if(Tilt_Sensor.value() == 0) : #when the value of tilt sensor is 0
        led1.value(1) # led1 turn on
        time.sleep_ms(200)#delay
        led2.value(1) # led2 turn on
        time.sleep_ms(200)#delay
        led3.value(1) # led3 turn on
        time.sleep_ms(200)#delay
        led4.value(1) # led4 turn on
        time.sleep_ms(200)#delay
    else :
                     #when the value of tilt sensor is 1
        led4.value(0) # led4 turn off
        time.sleep_ms(200)#delay
        led3.value(0) # led3 turn off
        time.sleep_ms(200)#delay
        led2.value(0) # led2 turn off
        time.sleep_ms(200)#delay
        led1.value(0) # led1 turn off
        time.sleep_ms(200)#delay
```

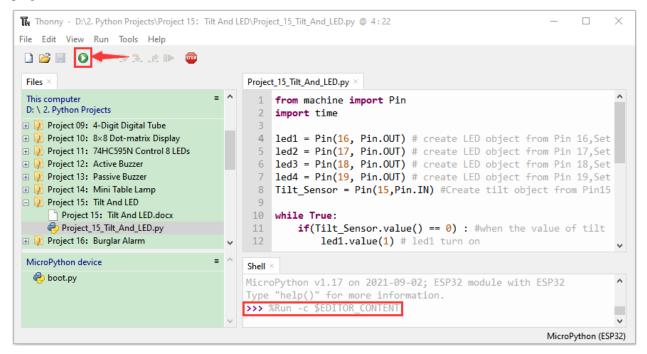
6.Project result

Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".



Click Circle Click Circle Click Circle Content and Starts to be executed and you'll see that when you tilt the breadboard to an angle, the LEDs will light up one by one. When you turn the breadboard to the original angle, the LEDs will turn off one by

one. Like the hourglass, the sand will leak out over time. Press"Ctrl+C"or click 🖤 "Stop/Restart backend" to exit the program.

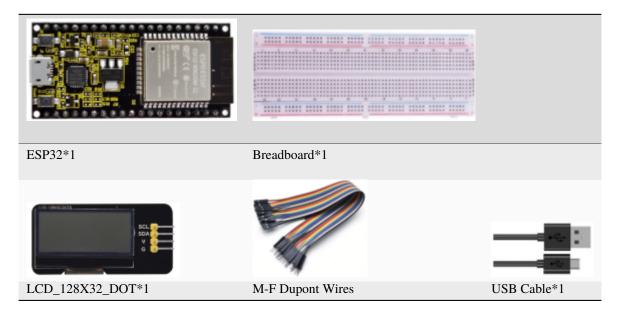


# 7.18 Project 16: I2C 128×32 LCD

1.Introduction

In everyday life, we can do all kinds of experiments with the display module and also DIY a variety of small objects. For example, you can make a temperature meter with a temperature sensor and display, or make a distance meter with an ultrasonic module and display. In this project, we will use the LCD\_128X32\_DOT module as the display and connect it to the ESP32, which will be used to control the LCD\_128X32\_DOT display to display various English words, common symbols and numbers.

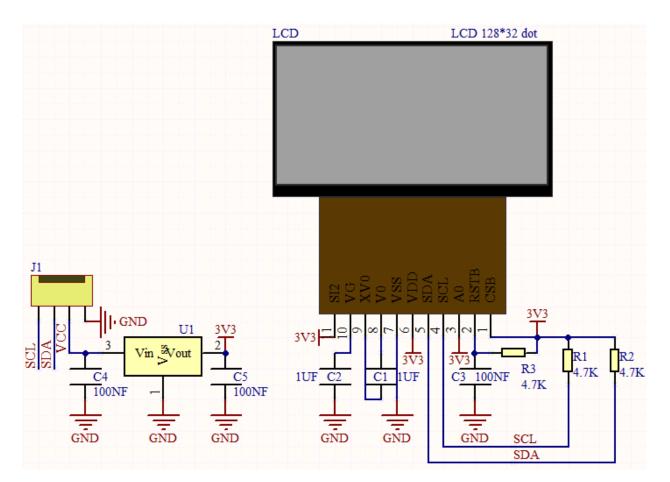
2.Components



3.Component knowledge

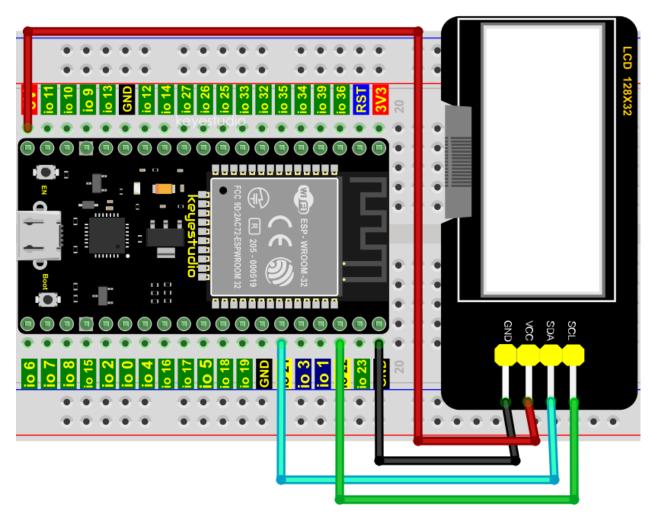
**LCD\_128X32\_DOT:** It is an LCD module with 128\*32 pixels and its driver chip is ST7567A. The module uses the IIC communication mode, while the code contains a library of all alphabets and common symbols that can be called directly. When using, we can also set it in the code so that the English letters and symbols show different text sizes. To make it easy to set up the pattern display, we also provide a mold capture software that converts a specific pattern into control code and then copies it directly into the test code for use.

Schematic diagram of LCD\_128X32\_DOT



# Features:

Pixel: 128\*32 character Operating voltage(chip)4.5V to 5.5V Operating current100mA (5.0V) Optimal operating voltage(module):5.0V 4.Wiring Diagram

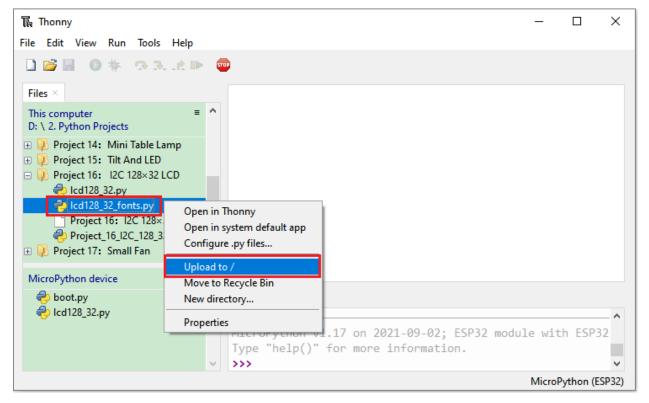


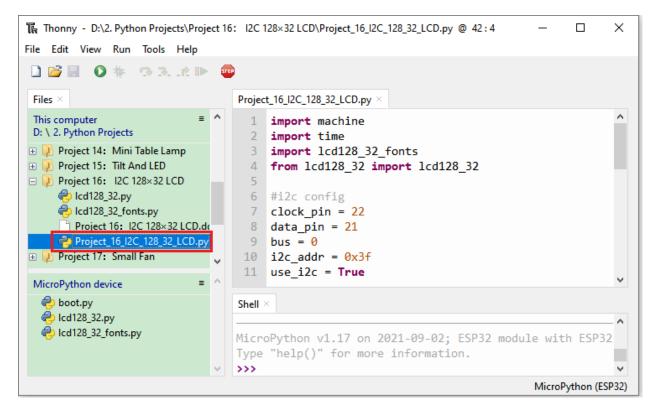
### 5.Project code

Codes used in this tutorial are saved in "2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects			$\Box$ $\times$
Share View			~ ?
→ This PC → Software (D:) → 2. Pyth	non Projects 🗸 🗸	)	2. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

🕼 Thonny	_		×
File Edit View Run Tools Help			
🗋 😂 📕 💿 🚸 🖙 R. Le 🕨 🥮			
Files ×			
This computer ≡ ↑ D: \ 2. Python Projects			
<ul> <li> <i>P</i> Project 14: Mini Table Lamp      </li> <li> <i>P</i> Project 15: Tilt And LED         </li> <li> <i>P</i> Project 16: 12C 128×32 LCD      </li> </ul>			
Image: Construction of the construc			
🗄 🕕 Project 17: Small F Upload to /			
MicroPython device Move to Recycle Bin New directory			
Properties			•
MicroPython v1.17 on 2021-09-02; ESP32 modul Type "help()" for more information. >>>		h ESP3	~





```
import machine
import time
import lcd128_32_fonts
from lcd128_32 import lcd128_32
#i2c config
clock_pin = 22
data_pin = 21
bus = 0
i2c_addr = 0x3f
use_i2c = True
def scan_for_devices():
   i2c = machine.I2C(bus,sda=machine.Pin(data_pin),scl=machine.Pin(clock_pin))
   devices = i2c.scan()
   if devices:
        for d in devices:
            print(hex(d))
   else:
       print('no i2c devices')
if use_i2c:
    scan_for_devices()
   lcd = lcd128_32(data_pin, clock_pin, bus, i2c_addr)
lcd.Clear()
lcd.Cursor(0, 4)
```

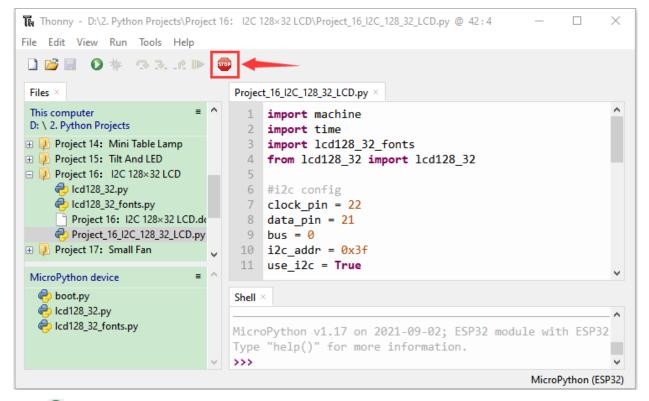
(continues on next page)

(continued from previous page)

```
lcd.Display("KEYESTUDIO")
lcd.Cursor(1, 0)
lcd.Display("ABCDEFGHIJKLMNOPQR")
lcd.Cursor(2, 0)
lcd.Display("123456789+-*/<>=$@")
lcd.Cursor(3, 0)
lcd.Display("%^&(){}:;'|?,.~\\[]")
"""
while True:
    scan_for_devices()
    time.sleep(0.5)
"""
```

#### 6.Project result

Make sure the ESP32 has been connected to the computer,  $click \stackrel{\text{\tiny WP}}{=}$  "Stop/Restart backend".



Click  $\mathbb{Q}$  "Run current script", the code starts to be executed and you'll see that the 128X32LCD module display will show "KEYESTUDIO" at the first line, "ABCDEFGHIJKLMNOPQR" will be displayed at the second line, "123456789±\*/<>=\$@"will be shown at the third line and "%^&(){}:;']?,.~\[]"will be displayed at the fourth line.

Press"Ctrl+C"or click <sup>22</sup> "Stop/Restart backend"to exit the program.

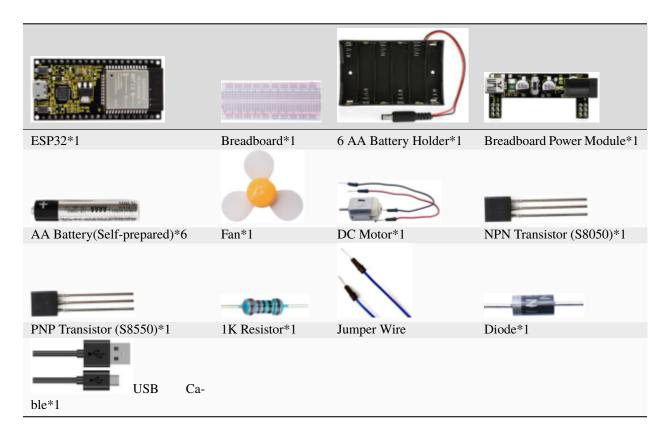
The Thonny - D:\2. Python Projects\Proje File Edit View Run Tools Help Project Projects Files ×	<b></b>	128×32 LCD\Project_16_I2C_128_32_LCD.py @ 42:4 — □	×
This computer       ■         D: \ 2. Python Projects         Image: Project 14: Mini Table Lamp         Image: Project 15: Tilt And LED         Image: Project 16: 12C 128×32 LCD         Image: Project 16: 12C 128×32 LCD         Image: Project 16: 12C 128×32 LCD.         Image: Project 17: Small Fan	2 3 4 5 6 7	<pre>#i2c config clock_pin = 22 data_pin = 21 bus = 0 i2c_addr = 0x3f</pre>	
MicroPython device ≡ ♦ boot.py ♦ lcd128_32.py ♦ lcd128_32_fonts.py	Shell Micr Type	<pre>use_i2c = True x oPython v1.17 on 2021-09-02; ESP32 module with ESP32 "help()" for more information. %Run -c \$EDITOR_CONTENT</pre>	~ 2 ~

# 7.19 Project 17Small Fan

1.Introduction

In hot summer, we need electric fans to cool us down, so in this project, we will use a ESP32 to control a DC motor and small fan blades to make a small electric fan.

2.Components



### Keyestudio Breadboard Power Supply Module



### Introduction:

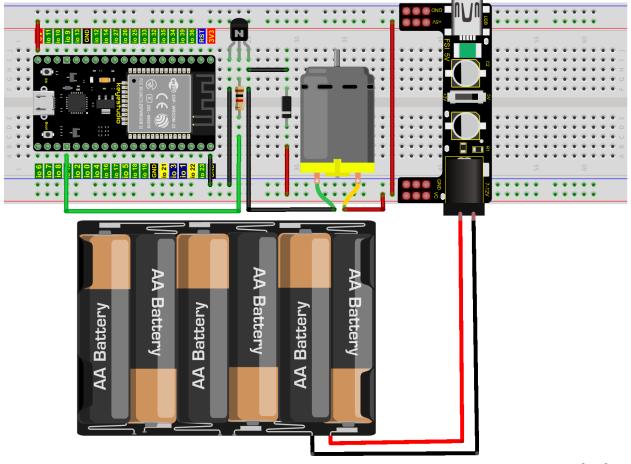
This breadboard power supply module is compatible with 5V and 3.3V, which can be applied to MB102 breadboard. The module contains two channels of independent control, powered by the USB all the way.

The output voltage is constant for the DC5V, and another way is powered by DC6.5-12V, output controlled by the slide switch, respectively for DC 5V and DC 3.3V.

If the other power supply is DC 6.5-12v, when the slide switch is switched to +5V, the output voltages of the left and right lines of the module are DC 5V. When the slide switch is switched to +3V, the output voltage of the USB power supply terminal of the module is DC 5V, and the output voltage of the DC 6.5-12V power supply terminal of the other power supply is DC3.3V.

3. Wiring Diagram 1

We use the S8050NPN transistor) to control the motor



fritzing

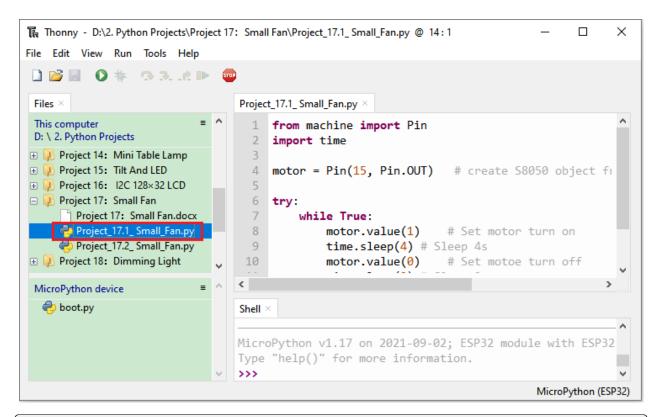
Wire up first, then connect a fan at the DC motor

### 5.Test Code 1

Codes used in this tutorial are saved in 2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

. Python Projects Share View		_	- ×
> This PC > Software (D:) > 2. Pyth	ion Projects 🗸 🗸 🗸	ر ک Search	n 2. Python
Name	Date modified	Туре	Size 🖍
Project 00: Boot	2/17/2022 10:21 AM	1 File folder	
Project 01: Hello World	2/17/2022 10:21 AN	1 File folder	
Project 02: Turn On LED	2/17/2022 11:10 AM	1 File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	1 File folder	
Project 04: Breathing LED	2/17/2022 10:21 AN	1 File folder	

Open "Thonny" click "This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 17: Small Fan", and then double left-click "Project\_17.1\_ Small\_Fan.py".

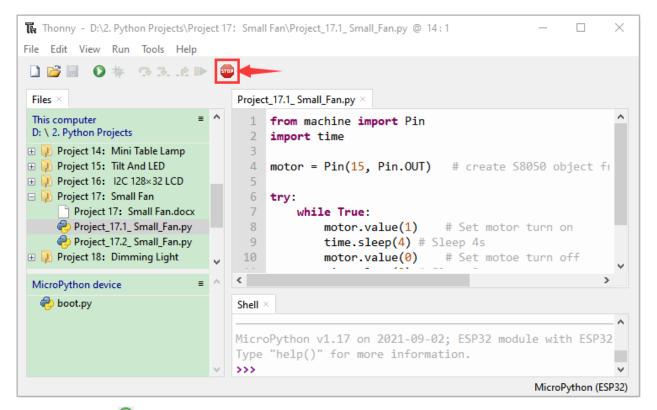


```
from machine import Pin
import time
motor = Pin(15, Pin.OUT)  # create S8050 object from Pin 15, Set Pin 15 to output
try:
    while True:
        motor.value(1)  # Set motor turn on
        time.sleep(4) # Sleep 4s
        motor.value(0)  # Set motoe turn off
        time.sleep(2) # Sleep 2s
except:
    pass
```

Ensure the ESP32 is connecteed to the computer and tap "Stop/Restart backend".

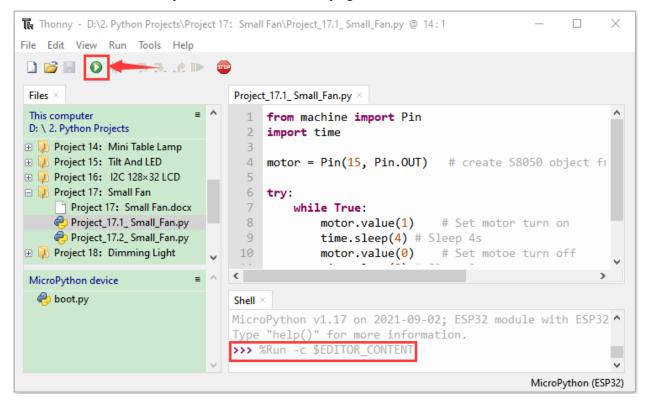
### 6.Project result

Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".



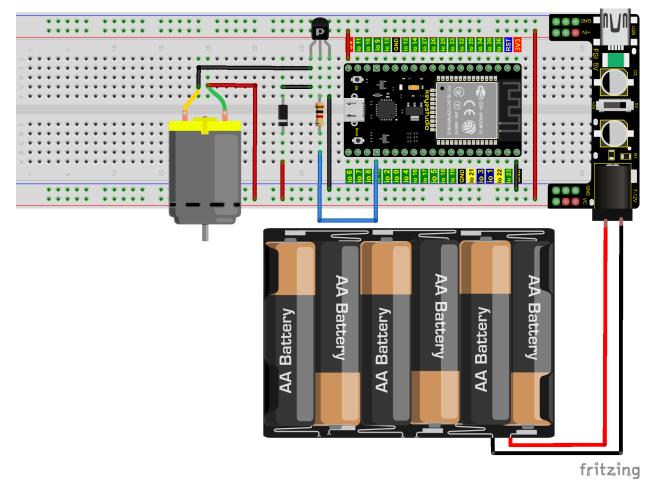
Power up and click  $\mathbb{Q}^{*}$  "Run current script", the code starts to be executed and you'll see that the small fan turn for 4s and stop for 2s.

Press"Ctrl+C"or click ""Stop/Restart backend" to exit the program.



# 7.Wiring Diagram 2

We use the S8050PNP transistor) to control the motor



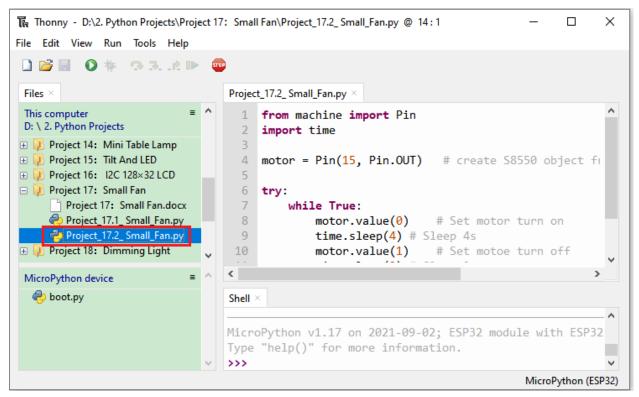
Wire up first, then connect a fan at the DC motor

## 8.Test Code 2

Codes used in this tutorial are saved in "2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

. Python Projects		_	
Share View			~ (
> This PC > Software (D:) > 2. Pyth	non Projects 🛛 🗸 🗸 🗸 🗸 🗸 🗸	ට 🔎 Sea	rch 2. Python
Name	Date modified	Туре	Size 1
Project 00: Boot	2/17/2022 10:21 AN	A File folder	
Project 01: Hello World	2/17/2022 10:21 AN	A File folder	
	2/17/2022 11:10 AM	A File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	A File folder	
Project 04: Breathing LED	2/17/2022 10:21 AN	A File folder	

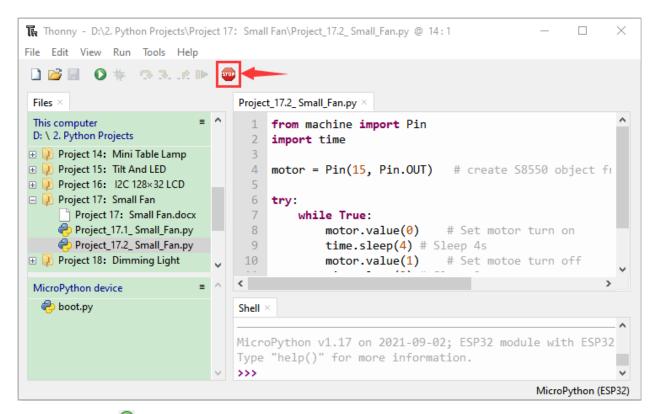
Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 17Small Fan", and then double left-click"Project\_17.2\_Small\_Fan.py".



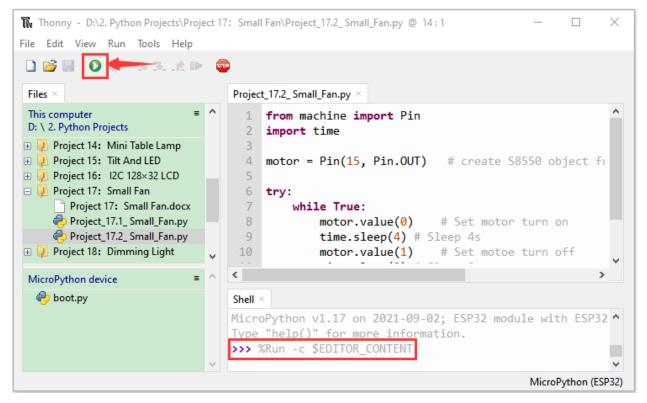
```
from machine import Pin
import time
motor = Pin(15, Pin.OUT)  # create S8550 object from Pin 15, Set Pin 15 to output
try:
    while True:
        motor.value(0)  # Set motor turn on
        time.sleep(4) # Sleep 4s
        motor.value(1)  # Set motoe turn off
        time.sleep(2) # Sleep 2s
except:
    pass
```

9.Test Result 2

Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".



Power up and click "Run current script", the code starts to be executed and you'll see that the small fan turn for 4s and stop for 2s. Press"Ctrl+C" or click "Stop/Restart backend" to exit the program.



# 7.20 Project 18Dimming Light

#### 1.Introduction

A potentiometer is a three-terminal resistor with sliding or rotating contacts that forms an adjustable voltage divider. It works by changing the position of the sliding contacts across a uniform resistance. In the potentiometer, the entire input voltage is applied across the whole length of the resistor, and the output voltage is the voltage drop between the fixed and sliding contact.

In this project, we will learn how to use ESP32 to read the values of the potentiometer, and make a dimming lamp with LED.

2.Components

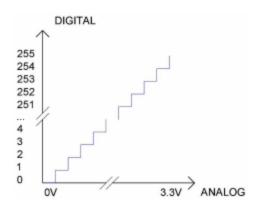
ESP32*1	Breadboard*1	Potentiometer*1	Red LED*1
220Resistor*1	Jumper Wires	USB Cable*1	

### 3.Component knowledge



Adjustable potentiometer: It is a kind of resistor and an analog electronic component, which has two states of 0 and 1(high level and low level). The analog quantity is different, its data state presents a linear state such as  $1 \sim 1024$ .

**ADC :** An ADC is an electronic integrated circuit used to convert analog signals such as voltages to digital or binary form consisting of 1s and 0s. The range of our ADC on ESP32 is 12 bits, that means the resolution is 2^12=4096, and it represents a range (at 3.3V) will be divided equally to 4096 parts. The rage of analog values corresponds to ADC values. So the more bits the ADC has, the denser the partition of analog will be and the greater the precision of the resulting conversion.



Subsection 1: the analog in rang of 0V—3.3/4095 V corresponds to digital 0;

Subsection 2: the analog in rang of 3.3/4095 V—2\*3.3 /4095V corresponds to digital 1;

• • •

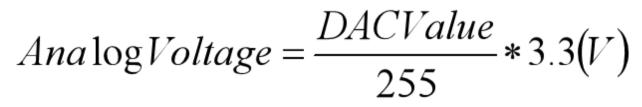
The following analog will be divided accordingly.

The conversion formula is as follows:

$$ADCValue = \frac{Ana \log Voltage}{3.3} * 4095$$

\*\*DAC\*\*The reversing of this process requires a DAC, Digital-to-Analog Converter. The digital I/O port can output high level and low level (0 or 1), but cannot output an intermediate voltage value. This is where a DAC is useful. ESP32 has two DAC output pins with 8-bit accuracy, GPIO25 and GPIO26, which can divide VCC(here is 3.3V) into 2^8=256 parts. For example, when the digital quantity is 1, the output voltage value is 3.3/256 \*1 V, and when the digital quantity is 128, the output voltage value is 3.3/256\*128=1.65V, the higher the accuracy of DAC, the higher the accuracy of output voltage value will be.

The conversion formula is as follows:



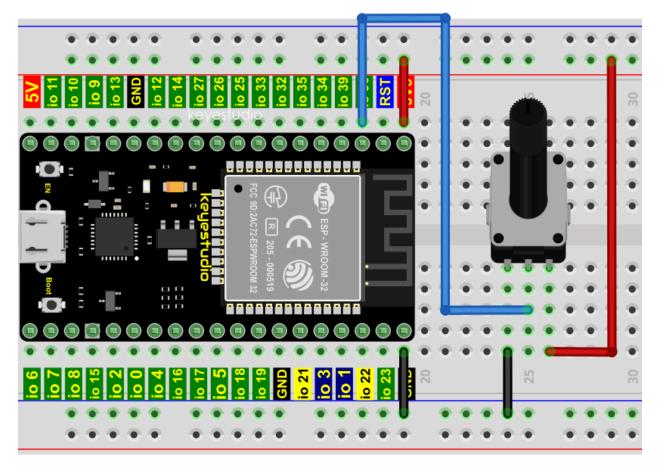
#### ADC on ESP32

ESP32 has 16 pins can be used to measure analog signals. GPIO pin sequence number and analog pin definition are shown in the following table

#### DAC on ESP32

ESP32 has two 8-bit digital analog converters to be connected to GPIO25 and GPIO26 pins, respectively, and it is immutable. As shown in the following table

We connect the potentiometer to the analog IO port of ESP32 to read the ADC value, DAC value and voltage value of the potentiometer, please refer to the wiring diagram below



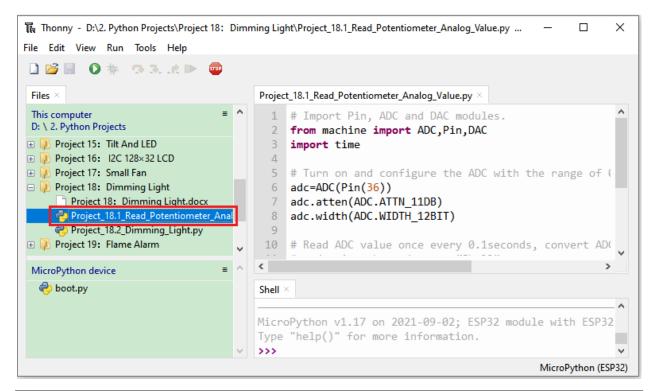
Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects

Share View			~ ?
→ This PC → Software (D:) → 2. Pyth	ion Projects 🗸 🗸 🗸	ට 🔎 Sear	ch 2. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	A File folder	
Project 01: Hello World	2/17/2022 10:21 AN	A File folder	
Project 02: Turn On LED	2/17/2022 11:10 AN	A File folder	
Project 03: LED Flashing	2/17/2022 11:12 AN	A File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	A File folder	

 $Open ``Thonny" click ``This computer" \rightarrow ``D:" \rightarrow ``2. Python Projects" \rightarrow ``Project 18Dimming Light" and then double left-click ``Project_18.1_Read_Potentiometer_Analog_Value.py".$ 

 $\times$ 



```
# Import Pin, ADC and DAC modules.
from machine import ADC,Pin,DAC
import time
# Turn on and configure the ADC with the range of 0-3.3V
adc=ADC(Pin(36))
adc.atten(ADC.ATTN_11DB)
adc.width(ADC.WIDTH_12BIT)
# Read ADC value once every 0.1seconds, convert ADC value to DAC value and output it,
# and print these data to "Shell".
try:
    while True:
        adcVal=adc.read()
        dacVal=adcVal//16
        voltage = adcVal / 4095.0 * 3.3
        print("ADC Val:",adcVal,"DACVal:",dacVal,"Voltage:",voltage,"V")
        time.sleep((0.1)
except:
   pass
```

Make sure the ESP32 has been connected to the computer,  $click^{22}$  "Stop/Restart backend".

iile Edit View Run Tools Help	+	
Files $\times$		Project_18.1_Read_Potentiometer_Analog_Value.py ×
This computer       =         D: \ 2. Python Projects	^	<pre>1 # Import Pin, ADC and DAC modules. 2 from machine import ADC,Pin,DAC 3 import time 4 5 # Turn on and configure the ADC with the range of (</pre>
<ul> <li>Project 18: Dimming Light         <ul> <li>Project 18: Dimming Light.docx</li> <li>Project_18.1_Read_Potentiometer_Ana</li> <li>Project_18.2_Dimming_Light.py</li> </ul> </li> <li>Project 19: Flame Alarm</li> </ul>	ı V	<pre>6 adc=ADC(Pin(36)) 7 adc.atten(ADC.ATTN_11DB) 8 adc.width(ADC.WIDTH_12BIT) 9 10 # Read ADC value once every 0.1seconds, convert ADC</pre>
MicroPython device =	~	<pre></pre>
😍 boot.py	~	Shell × MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32 Type "help()" for more information.

Click O"Run current script", the code starts to be executed and you'll see that the "Shell" window of Thonny IDE will print the ADC value, DAC value and voltage value of the potentiometer, turn the potentiometer handle, the ADC

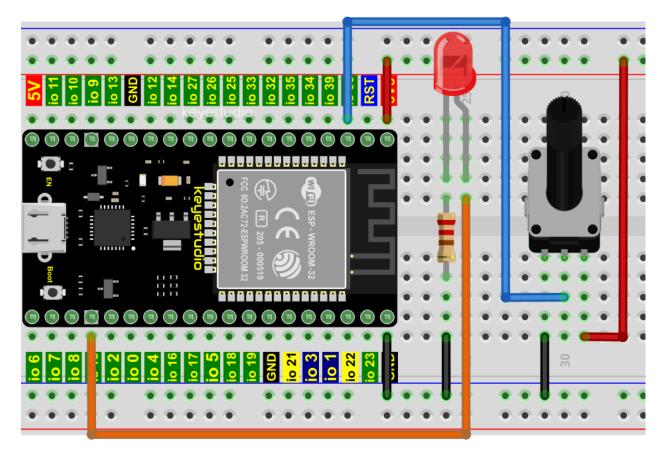
value and voltage value will change. Press "Ctrl+C" or click "" "Stop/Restart backend" to exit the program.

Thonny - D:\2. Python Projects\Project 18: [	Dimn	ning Light\Project_18.1_Read_Potentiometer_Analog_Value.py — 🛛 🛛 🗙
File Edit View Run Tools Help		
🗋 🗃 📓 💽 🌧 🖘e 🕨 👜		
Files $\times$		Project_18.1_Read_Potentiometer_Analog_Value.py $ imes$
This computer D: \ 2. Python Projects	^	1 # Import Pin, ADC and DAC modules. 2 from machine import ADC,Pin,DAC
		3 import time
Project 17: Small Fan		5 # Turn on and configure the ADC with the range of (
🖃 问 Project 18: Dimming Light		6 adc=ADC(Pin(36))
Project 18: Dimming Light.docx		7 adc.atten(ADC.ATTN_11DB)
Project_18.1_Read_Potentiometer_Anal		<pre>8 adc.width(ADC.WIDTH_12BIT)</pre>
Project_18.2_Dimming_Light.py Image: Project 19: Flame Alarm	J	9 10 # Read ADC value once every 0.1seconds, convert AD(
MicroPython device =	~	<pre></pre>
line of years a concerned with the second se		Shell ×
- boot.py		
		MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32
		Type "help()" for more information.
	$\sim$	>>> %Run -c \$EDITOR_CONTENT
		MicroPython (ESP32)

Shell ×		
ADC Val: 0	DACVal: 0 Voltage: 0.0 V	^
	3 DACVal: 1 Voltage: 0.0185348 V	
	8 DACVal: 3 Voltage: 0.03868132 V	
ADC Val: 20	68 DACVal: 16 Voltage: 0.2159707 V	
ADC Val: 55	59 DACVal: 34 Voltage: 0.4504762 V	
ADC Val: 0	DACVal: 0 Voltage: 0.0 V	
ADC Val: 55	53 DACVal: 34 Voltage: 0.445641 V	
ADC Val: 83	10 DACVal: 50 Voltage: 0.6527472 V	
ADC Val: 12	294 DACVal: 80 Voltage: 1.042784 V	
ADC Val: 12	280 DACVal: 80 Voltage: 1.031502 V	
ADC Val: 12	287 DACVal: 80 Voltage: 1.037143 V	
ADC Val: 1	514 DACVal: 94 Voltage: 1.220073 V	
ADC Val: 23	160 DACVal: 135 Voltage: 1.740659 V	
ADC Val: 23	162 DACVal: 135 Voltage: 1.742271 V	
ADC Val: 2	171 DACVal: 135 Voltage: 1.749524 V	
ADC Val: 24	467 DACVal: 154 Voltage: 1.988059 V	
ADC Val: 20	642 DACVal: 165 Voltage: 2.129084 V	
	640 DACVal: 165 Voltage: 2.127473 V	
	723 DACVal: 170 Voltage: 2.194359 V	
ADC Val: 29	911 DACVal: 181 Voltage: 2.345861 V	
	008 DACVal: 188 Voltage: 2.424029 V	
	029 DACVal: 189 Voltage: 2.440952 V	
	140 DACVal: 196 Voltage: 2.530403 V	
	271 DACVal: 204 Voltage: 2.635971 V	
	583 DACVal: 223 Voltage: 2.887399 V	
ADC Val: 3	664 DACVal: 229 Voltage: 2.952674 V	$\checkmark$

5. Wiring diagram of the dimming lamp

In the previous step, we read the ADC value, DAC value and voltage value of the potentiometer. Now we need to convert the ADC value of the potentiometer into the brightness of the LED to make a lamp that can adjust the brightness. The wiring diagram is as follows:

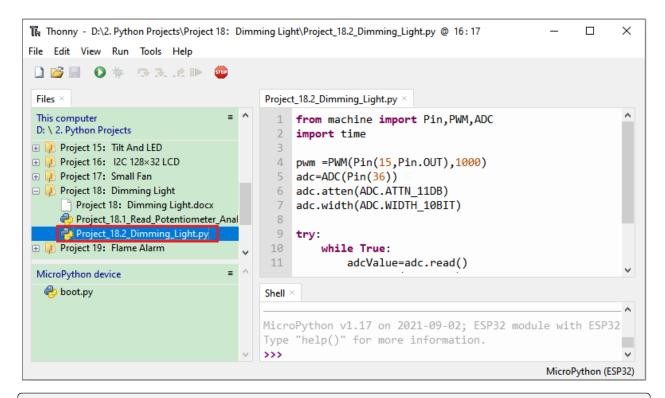


# 6.Project code

Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		—	⊐ ×
Share View			~ ?
→ This PC → Software (D:) → 2. Pyth	on Projects 🗸 진	,○ Search 2.	Python
Name	Date modified	Туре	Size ^
	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
Project 02: Turn On LED	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 18: Dimming Light" and then double left-click"Project\_18.2\_Dimming\_Light.py".



```
from machine import Pin,PWM,ADC
import time

pwm =PWM(Pin(15,Pin.OUT),1000)
adc=ADC(Pin(36))
adc.atten(ADC.ATTN_11DB)
adc.width(ADC.WIDTH_10BIT)

try:
```

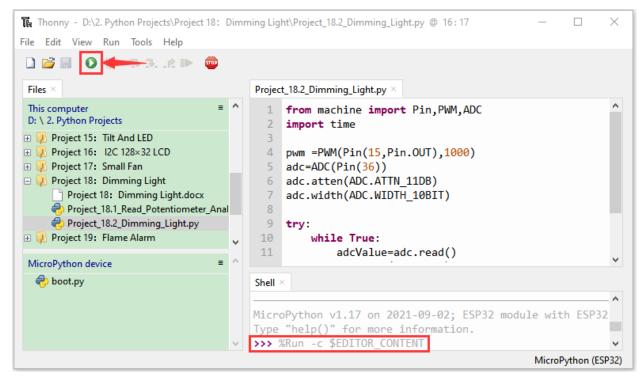
```
while True:
    adcValue=adc.read()
    pwm.duty(adcValue)
    print(adc.read())
    time.sleep_ms(100)
except:
    pwm.deinit()
```

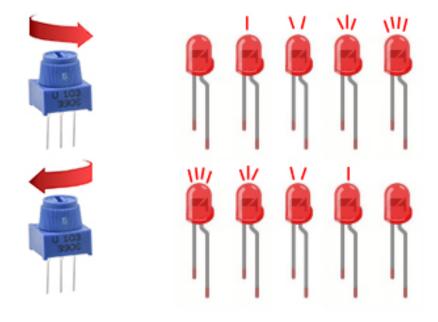
7.Project result

Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".

In Thonny - D:\2. Python Projects\Project 18: File Edit View Run Tools Help □ 😂 🖃 O 🎋 🕞 3 ৫ 🕪 🔤			ht\Project_18.2_Dimming_Light.py @ 16 : 17 — 🗆 🗙
Files $\times$		Projec	t_18.2_Dimming_Light.py $ imes$
This computer D: \ 2. Python Projects	^	1 2 3	<pre>from machine import Pin,PWM,ADC import time</pre>
		4 5	<pre>pwm =PWM(Pin(15,Pin.OUT),1000) adc=ADC(Pin(36))</pre>
Project 18: Dimming Light Project 18: Dimming Light.docx		6 7	adc.atten(ADC.ATTN_11DB) adc.width(ADC.WIDTH_10BIT)
Project_18.1_Read_Potentiometer_Ana Project_18.2_Dimming_Light.py	al	8 9	try:
🗄 🔑 Project 19: Flame Alarm	۲	10 11	<pre>while True: adcValue=adc.read()</pre>
MicroPython device =	$\sim$		· · · · · · · · · · · · · · · · · · ·
😓 boot.py		Shell >	^
			oPython v1.17 on 2021-09-02; ESP32 module with ESP32 "help()" for more information.
	Ŧ		MicroPython (ESP32

Click O"Run current script", the code starts to be executed and you'll see that turn the potentiometer handle and the brightness of the LED will change accordingly. Press"Ctrl+C" or click "Stop/Restart backend" to exit the program.



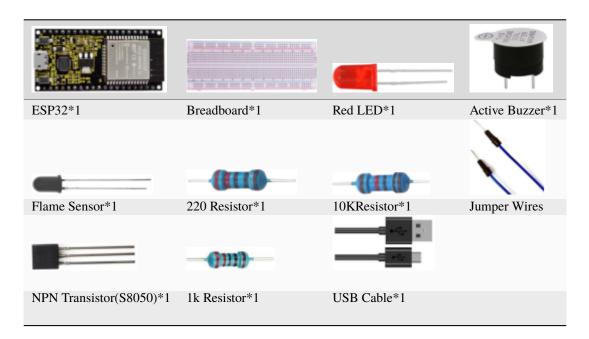


## 7.21 Project 19Flame Alarm

1.Introduction

Fire is a terrible disaster and fire alarm systems are very useful in housescommercial buildings and factories. In this project, we will use ESP32 to control a flame sensor, a buzzer and a LED to simulate fire alarm devices. This is a meaningful maker activity.

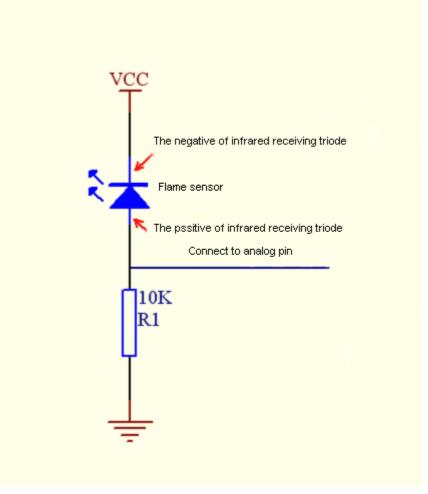
2.Components



3.Component knowledge



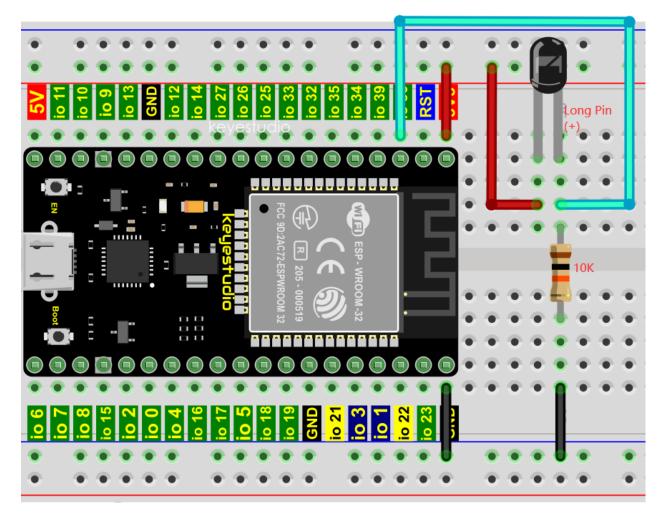
The flame emits a certain amount IR light that is invisible to the human eye, but our flame sensor can detect it and alert a microcontroller(such as ESP32) that a fire has been detected. It has a specially designed infrared receiver tube to detect the flame and then convert the flame brightness into a fluctuating level signal. The short pin of the receiving triode is negative pole and the other long pin is positive pole. We should connect the short pin (negative) to 5V and the long pin(positive) to the analog pin, a resistor and GND. As shown in the figure below



**Note:** Since vulnerable to radio frequency radiation and temperature changes, the flame sensor should be kept away from heat sources like radiators, heaters and air conditioners, as well as direct irradiation of sunlight, headlights and incandescent light.

4.Read the ADC value, DAC value and voltage value of the flame sensor

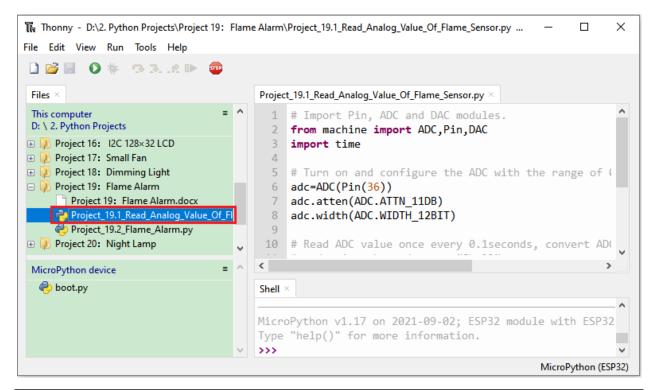
We first use a simple code to read the ADC value, DAC value and voltage value of the flame sensor and print them out. Please refer to the wiring diagram below



Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		—	$\Box$ $\times$
Share View			~ ?
→ This PC → Software (D:) → 2. Pyth	hon Projects 🗸 🗸 🗸	ල 🔎 Search 2	2. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 19: Flame Alarm", and then double left-click "Project\_19.1\_Read\_Analog\_Value\_Of\_Flame\_Sensor.py".



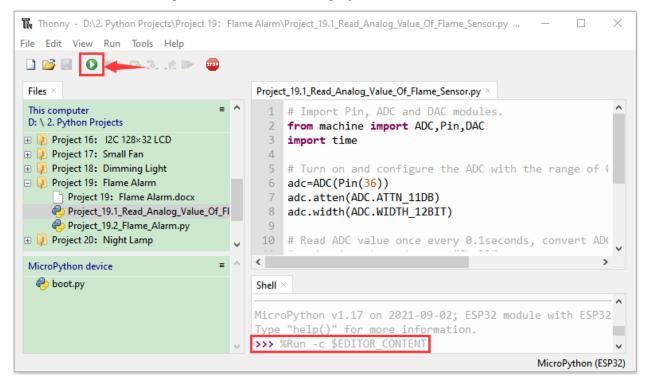
```
# Import Pin, ADC and DAC modules.
from machine import ADC,Pin,DAC
import time
# Turn on and configure the ADC with the range of 0-3.3V
adc=ADC(Pin(36))
adc.atten(ADC.ATTN_11DB)
adc.width(ADC.WIDTH_12BIT)
# Read ADC value once every 0.1seconds, convert ADC value to DAC value and output it,
# and print these data to "Shell".
try:
    while True:
        adcVal=adc.read()
        dacVal=adcVal//16
        voltage = adcVal / 4095.0 * 3.3
        print("ADC Val:",adcVal,"DACVal:",dacVal,"Voltage:",voltage,"V")
        time.sleep((0.1)
except:
   pass
```

Make sure the ESP32 has been connected to the computer,  $click \stackrel{\text{\tiny WP}}{=}$  "Stop/Restart backend" .

File Edit View Run Tools Help	Project 10.1 Read Analysis Value Of Flame Secondary V
Files ×         This computer       ■         D: \ 2. Python Projects         Image: Distribution of the state of the st	<pre>from machine import ADC,Pin,DAC import time f Turn on and configure the ADC with the range of ( adc=ADC(Pin(36)) adc.atten(ADC.ATTN_11DB)</pre>
MicroPython device ≡	<pre>Shell × MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32 Type "help()" for more information.</pre>

Click C "Run current script", the code starts to be executed and you'll see that the "Shell" window of Thonny IDE will print the ADC valueDAC value and voltage value of the flame sensor. When the flame is close to the flame sensor, the ADC value, DAC value and voltage value increase; Conversely, the ADC value, DAC value and voltage value decrease.

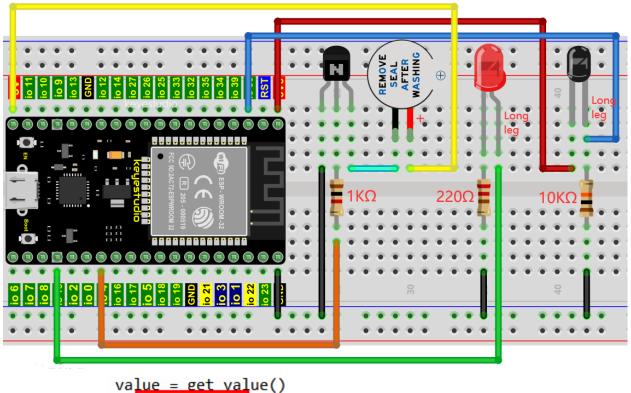
Press"Ctrl+C"or click <sup>22</sup> "Stop/Restart backend" to exit the program.



Shell ×		
ADC Val	0 DACVal: 0 Voltage: 0.0 V	$\mathbf{A}$
	23 DACVal: 1 Voltage: 0.0185348 V	
	48 DACVal: 3 Voltage: 0.03868132 V	
	268 DACVal: 16 Voltage: 0.2159707 V	
	559 DACVal: 34 Voltage: 0.4504762 V	
	0 DACVal: 0 Voltage: 0.0 V	
ADC Val:	553 DACVal: 34 Voltage: 0.445641 V	
ADC Val:	810 DACVal: 50 Voltage: 0.6527472 V	
	1294 DACVal: 80 Voltage: 1.042784 V	
ADC Val:	1280 DACVal: 80 Voltage: 1.031502 V	
ADC Val:	1287 DACVal: 80 Voltage: 1.037143 V	
ADC Val:	1514 DACVal: 94 Voltage: 1.220073 V	
ADC Val:	2160 DACVal: 135 Voltage: 1.740659 V	
ADC Val:	2162 DACVal: 135 Voltage: 1.742271 V	
ADC Val:	2171 DACVal: 135 Voltage: 1.749524 V	
ADC Val:	2467 DACVal: 154 Voltage: 1.988059 V	
ADC Val:	2642 DACVal: 165 Voltage: 2.129084 V	
ADC Val:	2640 DACVal: 165 Voltage: 2.127473 V	
ADC Val:	2723 DACVal: 170 Voltage: 2.194359 V	
ADC Val:	2911 DACVal: 181 Voltage: 2.345861 V	
ADC Val:	3008 DACVal: 188 Voltage: 2.424029 V	
ADC Val:	3029 DACVal: 189 Voltage: 2.440952 V	
ADC Val:	3140 DACVal: 196 Voltage: 2.530403 V	
ADC Val:	3271 DACVal: 204 Voltage: 2.635971 V	
ADC Val:	3583 DACVal: 223 Voltage: 2.887399 V	
ADC Val:	3664 DACVal: 229 Voltage: 2.952674 V	$\checkmark$

5.Wiring diagram of the flame alarm

Next, we will use a flame sensor, a buzzer, and a LED to make an interesting project, that is flame alarm. When flame is detected, the LED flashes and the buzzer alarms.



if value >500:

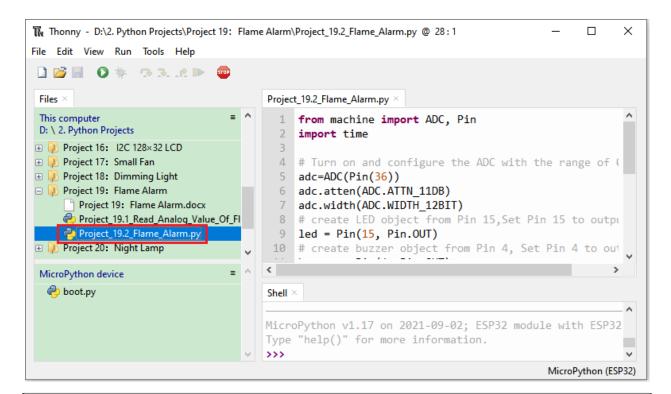
6.Project code:Note

buzzer.value(1) the threshold of 500 in the code can be reset itself as required)

Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

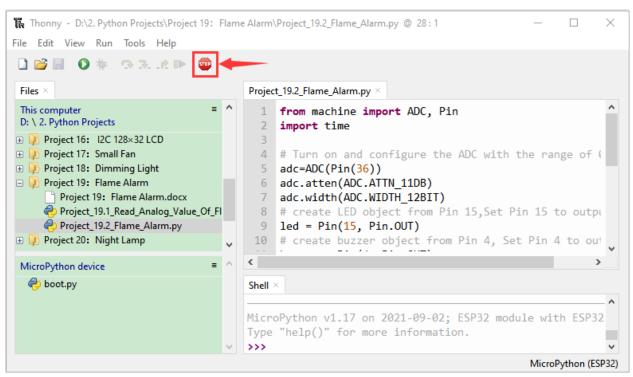
2. Python Projects			$\Box$ $\times$
Share View			~ ?
→ This PC → Software (D:) → 2. Pyth	on Projects 🛛 🗸 🗸	ට 🔎 Search	2. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
	2/17/2022 10:21 AM	File folder	
	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 19: Flame Alarm", and then double left-click"Project 19Flame Alarm".



```
from machine import ADC, Pin
import time
# Turn on and configure the ADC with the range of 0-3.3V
adc=ADC(Pin(36))
adc.atten(ADC.ATTN_11DB)
adc.width(ADC.WIDTH_12BIT)
# create LED object from Pin 15,Set Pin 15 to output
led = Pin(15, Pin.OUT)
# create buzzer object from Pin 4, Set Pin 4 to output
buzzer = Pin(4, Pin.OUT)
# If the flame sensor detects a flame, the buzzer will beep
# and the LED will blink when the analog value is greater than 500
# Otherwise, the buzzer does not sound and the LED goes off
while True:
   adcVal=adc.read()
   if adcVal >500:
        buzzer.value(1)
                           # Set buzzer turn on
        led.value(1)
                      # Set led turn on
        time.sleep(0.5) # Sleep 0.5s
        buzzer.value(0)
        led.value(♥)
                       # Set led turn off
        time.sleep(0.5) # Sleep 0.5s
   else:
        buzzer.value(0)
                           # Set buzzer turn off
                        # Set led turn off
        led.value(≬)
```

7.Project result



Click Click

click ""Stop/Restart backend" to exit the program.

Thonny - D:\2. Python Projects\Project 19: F	lame	e Alarm∖	Project_19.2_Flame_Alarm.py @ 28:1 — 🗆 🗙
File Edit View Run Tools Help			
Files ×		Project	:_19.2_Flame_Alarm.py ×
This computer D: \ 2. Python Projects ≡	^	1 2	from machine import ADC, Pin ^
<ul> <li>Project 16: 12C 128×32 LCD</li> <li>Project 17: Small Fan</li> <li>Project 18: Dimming Light</li> <li>Project 19: Flame Alarm</li> <li>Project 19: Flame Alarm.docx</li> <li>Project 19.1 Read_Analog_Value_Of_Fl</li> </ul>		5 6 7 8	<pre># Turn on and configure the ADC with the range of ( adc=ADC(Pin(36)) adc.atten(ADC.ATTN_11DB) adc.width(ADC.WIDTH_12BIT) # create LED object from Pin 15,Set Pin 15 to output</pre>
Project_19.2_Flame_Alarm.py Image: Project 20: Night Lamp	~		<pre>led = Pin(15, Pin.OUT) # create buzzer object from Pin 4, Set Pin 4 to out</pre>
MicroPython device =	$\sim$	<	>
🔶 boot.py	~	Type	<pre>Python v1.17 on 2021-09-02; ESP32 module with ESP32 "help()" for more information. Run -c \$EDITOR_CONTENT</pre>
			MicroPython (ESP32)

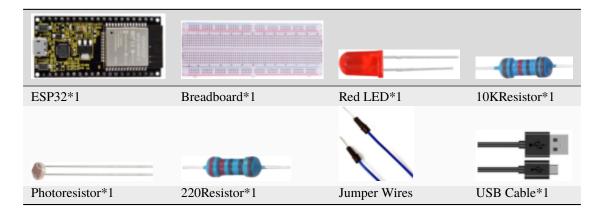
Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".

# 7.22 Project 20Night Lamp

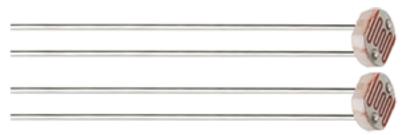
## 1.Introduction

Sensors or components are ubiquitous in our daily life. For example, some public street lamps will automatically turn on at night and turn off during the day. Why? In fact, this make use of a photosensitive element that senses the intensity of external ambient light. When the outdoor brightness decreases at night, the street lights will turn on automatically. In the daytime, the street lights will automatically turn off. the principle of which is very simple, In this Project, we use ESP32 to control a LED to achieve the effect of the street light.

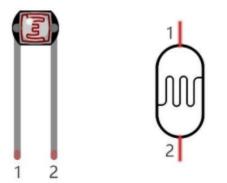
2.Components



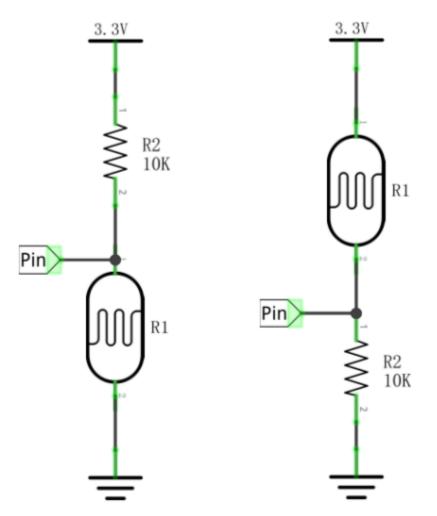
## 3.Component knowledge



**Photoresistor :** It is a kind of photosensitive resistance, its principle is that the photoresistor surface receives brightness (light) to reduce the resistance, the resistance value will change with the detected intensity of the ambient light . With this characteristic, we can use the photosensitive resistance to detect the light intensity. Photosensitive resistance and its electronic symbol are as follows



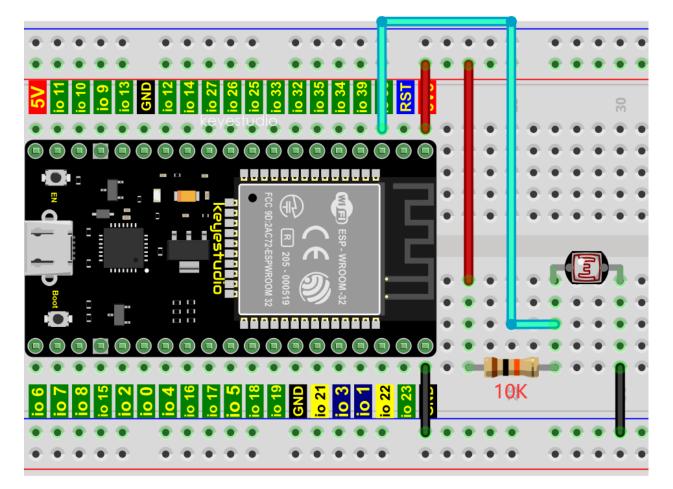
The following circuit is used to detect changes in resistance values of photoresistors



In the circuit above, when the resistance of the photoresistor changes due to the change of light intensity, the voltage between the photoresistor and resistance R2 will also change. Thus, the intensity of light can be obtained by measuring this voltage.

4.Read the ADC value, DAC value and voltage value of the photoresistor

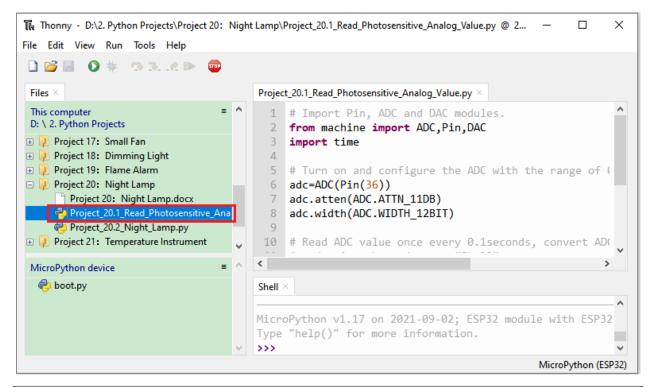
We first use a simple code to read the ADC value, DAC value and voltage value of the photoresistor and print them out. Please refer to the following wiring diagram



Codes used in this tutorial are saved in "2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

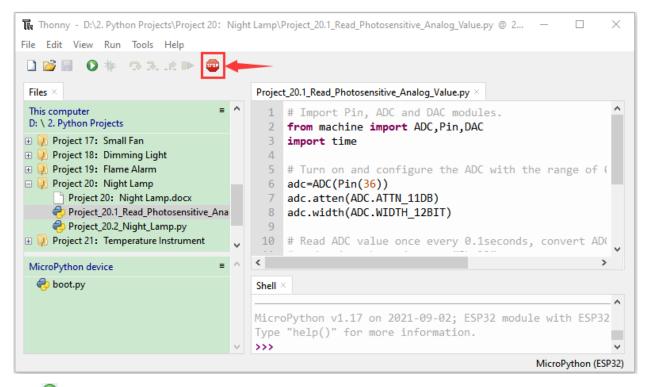
2. Python Projects			
Share View			~ 🕐
→ This PC → Software (D:) → 2. Pyt	hon Projects 🗸 🗸		. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
Project 02: Turn On LED	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

Open "Thonny" click "This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 20: Night Lamp" and then double left-click "Project\_20.1\_Read\_Photosensitive\_Analog\_Value.py"".



```
# Import Pin, ADC and DAC modules.
from machine import ADC,Pin,DAC
import time
# Turn on and configure the ADC with the range of 0-3.3V
adc=ADC(Pin(36))
adc.atten(ADC.ATTN_11DB)
adc.width(ADC.WIDTH_12BIT)
# Read ADC value once every 0.1seconds, convert ADC value to DAC value and output it,
# and print these data to "Shell".
try:
    while True:
        adcVal=adc.read()
        dacVal=adcVal//16
        voltage = adcVal / 4095.0 * 3.3
        print("ADC Val:",adcVal,"DACVal:",dacVal,"Voltage:",voltage,"V")
        time.sleep(0.1)
except:
   pass
```

Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".



Click C "Run current script", the code starts to be executed and you'll see that the "Shell" window of Thonny IDE will print the ADC valueDAC value and voltage value of the photoresistor. When the light intensity around the photoresistor is gradually reduced, the ADC valueDAC value and voltage value will gradually increase. On the contrary, the ADC

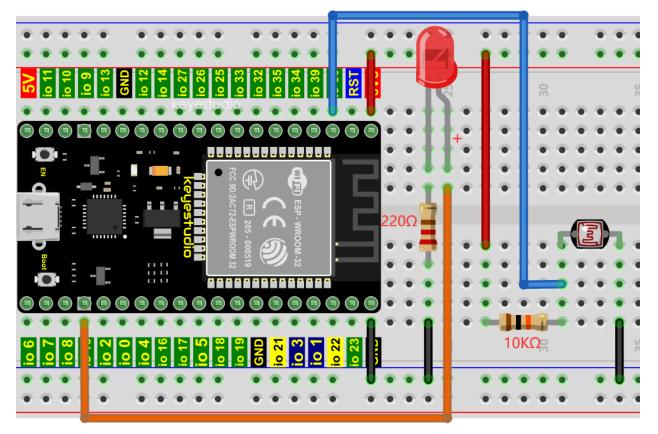
value, DAC value and voltage value decreases gradually. Press"Ctrl+C"or click <sup>229</sup>"Stop/Restart backend"to exit the program.

Thonny - D:\2. Python Projects\Project 20: 1	Night	t Lamp\Project_20.1_Read_Photosensitive_Analog_Value.py @ 2 – 🛛 🛛 🗙
File Edit View Run Tools Help		
Files $\times$		${\sf Project\_20.1\_Read\_Photosensitive\_Analog\_Value.py} \ \times$
This computer D: \ 2. Python Projects	^	<pre>1 # Import Pin, ADC and DAC modules. 2 from machine import ADC,Pin,DAC</pre>
Project 17: Small Fan		3 import time
Project 18: Dimming Light		4
		<pre>5 # Turn on and configure the ADC with the range of ( 6 adc=ADC(Pin(36))</pre>
Project 20: Night Lamp.docx		7 adc.atten(ADC.ATTN 11DB)
Project_20.1_Read_Photosensitive_Ana		8 adc.width(ADC.WIDTH_12BIT)
🚽 🌏 Project_20.2_Night_Lamp.py		9
Project 21: Temperature Instrument	¥	10 # Read ADC value once every 0.1seconds, convert AD(
MicroPython device =	^	٢ >
🌏 boot.py		Shell ×
		MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32 ^
		Type "help()" for more information.
		>>> %Run -c \$EDITOR_CONTENT
	Υ.	······································
		MicroPython (ESP32)

Shell ×			
ADC Val:	1472 DACVal:	92 Voltage: 1.186227 V	~
		102 Voltage: 1.325641 V	
ADC Val:	1847 DACVal:	115 Voltage: 1.488425 V	
ADC Val:	2043 DACVal:	127 Voltage: 1.646374 V	
ADC Val:	2254 DACVal:	140 Voltage: 1.81641 V	
ADC Val:	2442 DACVal:	152 Voltage: 1.967912 V	
ADC Val:	2625 DACVal:	164 Voltage: 2.115385 V	
ADC Val:	2752 DACVal:	172 Voltage: 2.217729 V	
ADC Val:	2832 DACVal:	177 Voltage: 2.282198 V	
		180 Voltage: 2.320879 V	
		180 Voltage: 2.32652 V	
		179 Voltage: 2.315238 V	
		182 Voltage: 2.354725 V	
		186 Voltage: 2.41033 V	
		190 Voltage: 2.458681 V	
		193 Voltage: 2.500586 V	
		196 Voltage: 2.534432 V	
		197 Voltage: 2.548938 V	
		198 Voltage: 2.563443 V	
		199 Voltage: 2.568278 V	
		200 Voltage: 2.590037 V	
ADC Val:	3314 DACVal:	207 Voltage: 2.670623 V	×

5. Wiring diagram of the light-controlled lamp

We made a small dimming lamp in the front, now we will make a light controlled lamp. The principle is the same, that is, the ESP32 takes the ADC value of the sensor, and then adjusts the brightness of the LED.



## 6.Project code

Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		—	) ×
Share View			~ ?
> This PC > Software (D:) > 2. Python Pro	ojects v ひ		Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
Project 02: Turn On LED	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

Open "Thonny" click "This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 20: Night Lamp" and then double left-click "Project\_20.2\_Night\_Lamp.py".

🕞 Thonny - D:\2. Python Projects\Project 20:	Night	t Lamp\	Project_20.2_Night_Lamp.py @ 16:17 — 🗆 🗙
File Edit View Run Tools Help			
🗋 🗃 🖥 🔹 🔅 🖘 🗷 🗈 📼			
Files ×		Projec	t_20.2_Night_Lamp.py $ imes$
This computer ≡ D: \ 2. Python Projects ⊕ ) Project 17: Small Fan	^	1 2 3	from machine import Pin,PWM,ADC ^
<ul> <li></li></ul>		4 5	<pre>pwm =PWM(Pin(15,Pin.OUT),1000) adc=ADC(Pin(36))</pre>
<ul> <li>Image: Project 20: Night Lamp</li> <li>Project 20: Night Lamp.docx</li> <li>Project_20.1_Read_Photosensitive_Anality</li> </ul>		6 7 8	adc.atten(ADC.ATTN_11DB) adc.width(ADC.WIDTH_10BIT)
Project_20.2_Night_Lamp.py Project 21: Temperature Instrument	¥	9 10	try: while True:
MicroPython device =		11	adcValue=adc.read()
net in the second secon		Shell	< A 1
			oPython v1.17 on 2021-09-02; ESP32 module with ESP32 "help()" for more information.
			MicroPython (ESP32)

from machine import Pin,PWM,ADC
import time

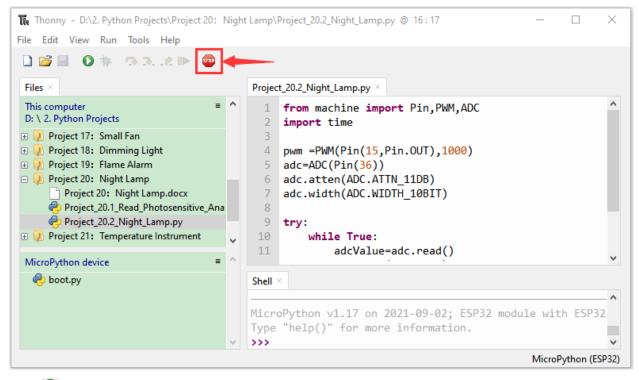
pwm =PWM(Pin(15,Pin.OUT),1000)
adc=ADC(Pin(36))
adc.atten(ADC.ATTN\_11DB)
adc.width(ADC.WIDTH\_10BIT)

(continues on next page)

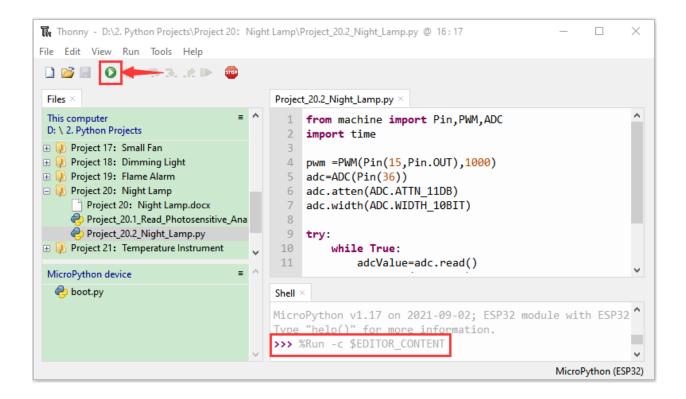
(continued from previous page)

7.Project result

Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".



Click O "Run current script", the code starts to be executed and you'll see that when the intensity of light around the photoresistor is reduced, the LED will be bright, on the contrary, the LED will be dim. Press"Ctrl+C"or click "Stop/Restart backend" to exit the program.

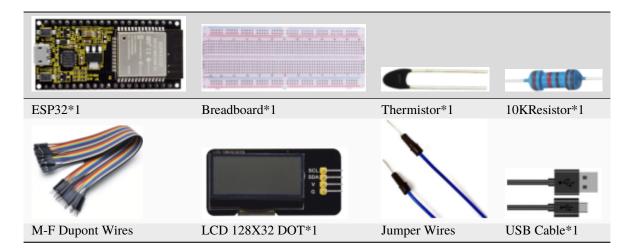


## 7.23 Project 21Temperature Instrument

## 1.Introduction

Thermistor is a kind of resistor whose resistance depends on temperature changes, which is widely used in gardening, home alarm system and other devices. Therefore, we can use the feature to make a temperature instrument.

## 2.Components



## 3.Component knowledge

**Thermistor:** A Thermistor is a temperature sensitive resistor. When it senses a change in temperature, the resistance of the Thermistor will change. We can take advantage of this characteristic by using a Thermistor to detect temperature intensity. A Thermistor and its electronic symbol are shown below:



The relationship between resistance value and temperature of a thermistor is

$$Rt = R * EXP[B * \left(\frac{1}{T2} - \frac{1}{T1}\right)]$$

## Where:

**Rt** is the thermistor resistance under T2 temperature;

**R** is the nominal resistance of thermistor under T1 temperature;

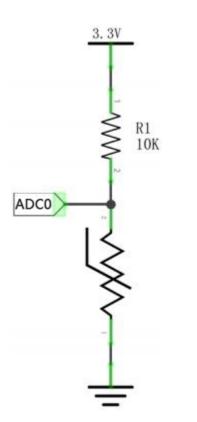
**EXP[n]** is nth power of e;

**B** is for thermal index;

T1, T2 is Kelvin temperature (absolute temperature). Kelvin temperature=273.15 + Celsius temperature.

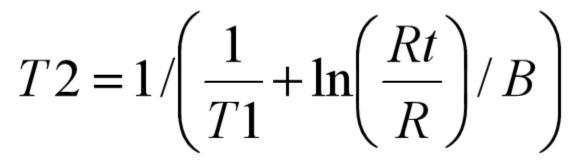
For the parameters of the Thermistor, we use: B=3950, R=10k, T1=25.

The circuit connection method of the Thermistor is similar to photoresistor, as the following



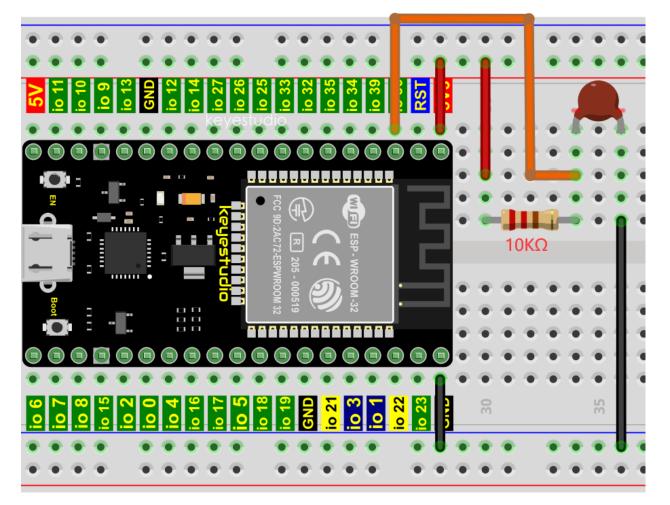
We can use the value measured by the ADC converter to obtain the resistance value of Thermistor, and then we can use the formula to obtain the temperature value.

Therefore, the temperature formula can be derived as:



4.Read the value of the Thermistor

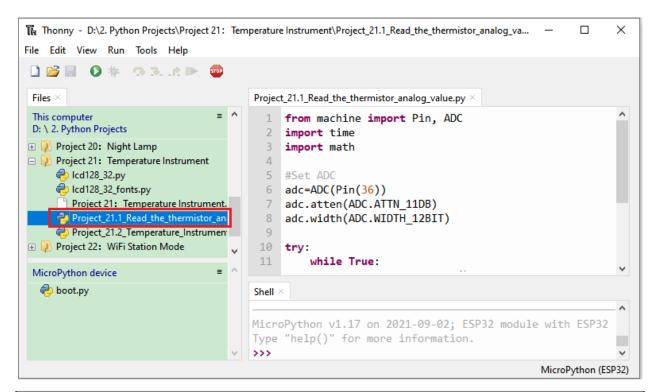
First we will learn the thermistor to read the current ADC value, voltage value and temperature value and print them out. Please connect the wires according to the wiring diagram below



Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

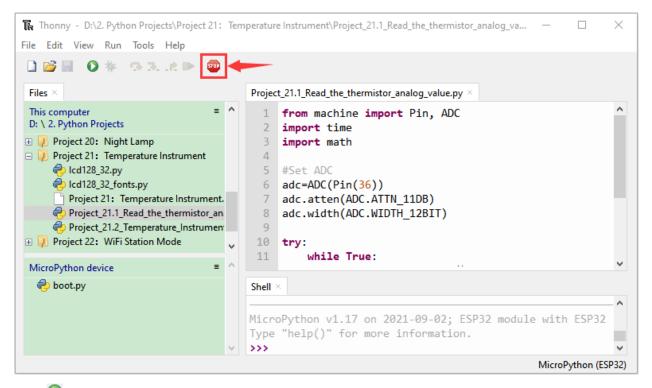
2. Python Projects		_	$\Box$ $\times$
Share View			~ 🕐
→ This PC → Software (D:) → 2. Pyth	non Projects 🗸 진	🔎 Search 2	2. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

Open "Thonny" click "This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 21: Temperature Instrument, and then double left-click "Project\_21.1\_Read\_the\_thermistor\_analog\_value.py".



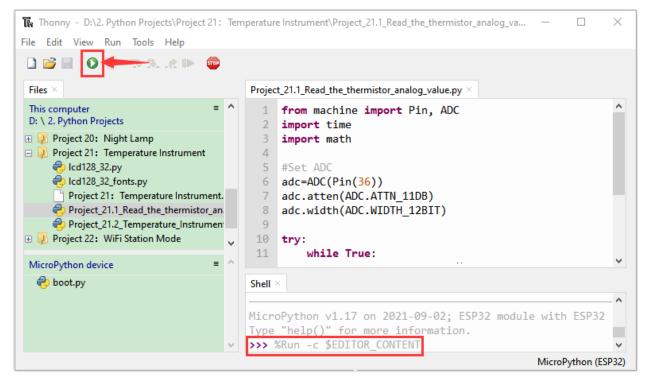
```
from machine import Pin, ADC
import time
import math
#Set ADC
adc=ADC(Pin(36))
adc.atten(ADC.ATTN_11DB)
adc.width(ADC.WIDTH_12BIT)
try:
    while True:
       adcValue = adc.read()
        voltage = adcValue / 4095 * 3.3
        Rt = 10 * voltage / (3.3-voltage)
        tempK = (1 / (1 / (273.15+25) + (math.log(Rt/10)) / 3950))
        tempC = (tempK - 273.15)
       print("ADC value:",adcValue," Voltage:",voltage,"V"," Temperature: ",tempC,"C
time.sleep(1)
except:
   pass
```

Make sure the ESP32 has been connected to the computer, click <sup>10</sup> "Stop/Restart backend".



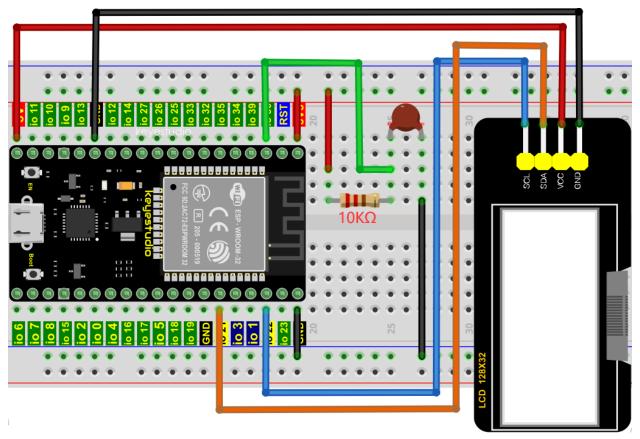
Click Circle Click

Press"Ctrl+C"or click "Stop/Restart backend" to exit the program.



Shell ×						
>>> %Run -c	\$EDI	TOR_CONTE	NT			^
ADC value:	2305	Voltage:	1.857509 V	Temperature:	19.41592 C	
ADC value:	2287	Voltage:	1.843004 V	Temperature:	19.80316 C	
ADC value:	2256	Voltage:	1.818022 V	Temperature:	20.47055 C	
ADC value:	2246	Voltage:	1.809963 V	Temperature:	20.68604 C	
ADC value:	2271	Voltage:	1.83011 V	Temperature:	20.14752 C	
ADC value:	2269	Voltage:	1.828498 V	Temperature:	20.19058 C	
ADC value:	2197	Voltage:	1.770476 V	Temperature:	21.74371 C	
ADC value:	2218	Voltage:	1.787399 V	Temperature:	21.29001 C	
ADC value:	2251	Voltage:	1.813993 V	Temperature:	20.57831 C	
ADC value:	2227	Voltage:	1.794652 V	Temperature:	21.0958 C	
ADC value:	2227	Voltage:	1.794652 V	Temperature:	21.0958 C	
ADC value:	2247	Voltage:	1.810769 V	Temperature:	20.66449 C	
ADC value:	2257	Voltage:	1.818828 V	Temperature:	20.44904 C	
						~

5. Wiring diagram of the temperature instrument



## 6.Project code

Codes used in this tutorial are saved in"2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

. Python Projects		_	
Share View			~ 🔮
→ This PC → Software (D:) → 2. Pytł	non Projects 🗸 ぐ	,	. Python
Name	Date modified	Туре	Size '
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

Open"Thonny"click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 21: Temperature Instrument". Select"lcd128\_32.py"and "lcd128\_32\_fonts.py"right-click your mouse to select"Upload to/"wait for"lcd128\_32.py"an "lcd128\_32\_fonts.py"to be uploaded to ESP32and double left-click"Project\_21.2\_Temperature\_Instrument.py".

ត្រី Thonny	_		×
File Edit View Run Tools Help			
🗋 😂 📕 💿 🌞 🕾 3et 🕨 🚭			
Files ×			
This computer = ^ D: \ 2. Python Projects			
<ul> <li></li></ul>			
Icd128_32.py       Open in Thonny         Icd128_32_fonts.p       Open in system default app         Project 21: Temp       Configure .py files			
Project_21.2_Tem Upload to /			
MicroPython device Move to Recycle Bin New directory			-
boot.py Properties			^
MicroPython v1.17 on 2021-09-02; ESP32 modul Type "help()" for more information.		thon (ES	*

🕞 Thonny				_		Х
File Edit View Run Tools	Help					
🗋 😂 🖩 🛛 🚸 👁 🦻	🗈 🕨 🚥					
Files ×						
This computer	= ^					
D: \ 2. Python Projects						
🕀 💫 Project 19: Flame Alarm						
Project 20: Night Lamp Project 21: Temperature	Instrument					
Cd128 32.pv						
lcd128_32_fonts.py Project 21: Tempera	Open in Thonny					
Project_21: Tempera	Open in system d		q			
Project_21.2_Tempera		s				
MicroPython device	Upload to / Move to Recycle I	Rin				
eboot.py	New directory	5111				
🎨 lcd128_32.py	Properties					^
			<pre>rychon v1.17 on 2021-09-02; ESP32 mod "help()" for more information.</pre>	ule wit	h ESP3	2
	v .	>>>	heip() for more information.			~
				Micro	ython (E	SP32)
					_	
-		erature	nstrument\Project_21.2_Temperature_Instrument.py @ .	. –		×
File Edit View Run Tools						
🛄 🚰 📕 💽 🏇 🕾 🍡						
	🖻 🕪 🚥					
Files ×	e 🕨 📟	Projec	t_21.2_Temperature_Instrument.py $ imes$			
Files × This computer	R 🕪 🚥	1	from machine import Pin, ADC, I2C			^
Files × This computer D: \ 2. Python Projects	-	1 2	<pre>from machine import Pin, ADC, I2C import machine</pre>			^
Files × This computer D: \ 2. Python Projects	= ^	1	from machine import Pin, ADC, I2C			^
Files × This computer D: \ 2. Python Projects ⊕ ↓ Project 20: Night Lamp ⊜ ↓ Project 21: Temperature ⊕ Icd128_32.py	= ^	1 2 3 4 5	<pre>from machine import Pin, ADC, I2C import machine import time import math import lcd128_32_fonts</pre>			^
Files × This computer D: \ 2. Python Projects	= ^	1 2 3 4	<pre>from machine import Pin, ADC, I2C import machine import time import math</pre>			^
Files × This computer D: \ 2. Python Projects Project 20: Night Lamp Project 21: Temperature Project 21: Temperature Project 21: Temperature Project 21: Temperature Project 21. Temperature Project 21.1_Read_th	■ ^ Instrument ture Instrument.d e_thermistor_anal	1 2 3 4 5 6 7 8	<pre>from machine import Pin, ADC, I2C import machine import time import math import lcd128_32_fonts from lcd128_32 import lcd128_32 #Set ADC</pre>			^
Files × This computer D: \ 2. Python Projects Project 20: Night Lamp Project 21: Temperature Cd128_32.py Cd128_32_fonts.py Project 21: Temperat Project 21: Temperat Project 21.1_Read_th	Instrument ture Instrument.d e_thermistor_anal ature_Instrument.	1 2 3 4 5 6 7 8 9	<pre>from machine import Pin, ADC, I2C import machine import time import math import lcd128_32_fonts from lcd128_32 import lcd128_32 #Set ADC adc=ADC(Pin(36))</pre>			^
Files × This computer D: \ 2. Python Projects Project 20: Night Lamp Project 21: Temperature Coll28_32.py Coll28_32_fonts.py Project 21: Temperat Project 21: Temperat Project 21.2_Temperat Project 22: WiFi Station	= ^ Instrument ture Instrument.d e_thermistor_anal ature_Instrument.J Mode	1 2 3 4 5 6 7 8	<pre>from machine import Pin, ADC, I2C import machine import time import math import lcd128_32_fonts from lcd128_32 import lcd128_32 #Set ADC</pre>			^
Files × This computer D: \ 2. Python Projects Project 20: Night Lamp Project 21: Temperature Coll 28_32.py Coll 28_32_fonts.py Project 21: Tempera Project 21.1_Read_th Project 21.2_Tempera Project 22: WiFi Station MicroPython device	Instrument ture Instrument.d e_thermistor_anal ature_Instrument.	1 2 3 4 5 6 7 8 9 10 11	<pre>from machine import Pin, ADC, I2C import machine import time import math import lcd128_32_fonts from lcd128_32 import lcd128_32 #Set ADC adc=ADC(Pin(36)) adc.atten(ADC.ATTN_11DB) adc.width(ADC.WIDTH_12BIT)</pre>			~
Files × This computer D: \ 2. Python Projects Project 20: Night Lamp Project 21: Temperature Coll 28_32.py Coll 28_32 fonts.py Project 21: Tempera Project 21: Tempera Project 21.1 Read_th Project 21.2 Tempera Project 21.2 Tempera Project 21.2 Tempera Project 21.2 Tempera Description device Description device	= ^ Instrument ture Instrument.d e_thermistor_anal ature_Instrument.J Mode	1 2 3 4 5 6 7 8 9 10	<pre>from machine import Pin, ADC, I2C import machine import time import math import lcd128_32_fonts from lcd128_32 import lcd128_32 #Set ADC adc=ADC(Pin(36)) adc.atten(ADC.ATTN_11DB) adc.width(ADC.WIDTH_12BIT)</pre>			~
Files × This computer D: \ 2. Python Projects Project 20: Night Lamp Project 21: Temperature Coll 28_32.py Coll 28_32_fonts.py Project 21: Tempera Project 21.1_Read_th Project 21.2_Tempera Project 22: WiFi Station MicroPython device	= ^ Instrument ture Instrument.d e_thermistor_anal ature_Instrument.J Mode	1 2 3 4 5 6 7 8 9 10 11 Shell	<pre>from machine import Pin, ADC, I2C import machine import time import math import lcd128_32_fonts from lcd128_32 import lcd128_32 #Set ADC adc=ADC(Pin(36)) adc.atten(ADC.ATTN_11DB) adc.width(ADC.WIDTH_12BIT)</pre>	ule wit	:h ESP3	^
Files × This computer D: \ 2. Python Projects Project 20: Night Lamp Project 21: Temperature Coll 28_32.py Coll 28_32_fonts.py Project 21: Tempera Project 21.1_Read_th Project 21.2_Tempera Project 21.2_Tempera Project 21.2_Tempera Project 21.2_Tempera Project 22: WiFi Station MicroPython device boot.py boot.py lcd128_32.py	<pre>= ^ Instrument ture Instrument.d e_thermistor_anal ature_Instrument.s Mode = ^</pre>	1 2 3 4 5 6 7 8 9 10 11 Shell 2 Micro	<pre>from machine import Pin, ADC, I2C import machine import time import math import lcd128_32_fonts from lcd128_32 import lcd128_32 #Set ADC adc=ADC(Pin(36)) adc.atten(ADC.ATTN_11DB) adc.width(ADC.WIDTH_12BIT)</pre>	ule wit	:h ESP3	2
Files × This computer D: \ 2. Python Projects Project 20: Night Lamp Project 21: Temperature Coll 28_32.py Coll 28_32_fonts.py Project 21: Tempera Project 21.1_Read_th Project 21.2_Tempera Project 21.2_Tempera Project 22: WiFi Station MicroPython device Coll 28_32.py Coll 28_32.py	= ^ Instrument ture Instrument.d e_thermistor_anal ature_Instrument.J Mode	1 2 3 4 5 6 7 8 9 10 11 Shell >	<pre>from machine import Pin, ADC, I2C import machine import time import nath import lcd128_32_fonts from lcd128_32 import lcd128_32 #Set ADC adc=ADC(Pin(36)) adc.atten(ADC.ATTN_11DB) adc.width(ADC.WIDTH_12BIT) #Python v1.17 on 2021-09-02; ESP32 model </pre>		h ESP3	2

from machine import Pin, ADC, I2C import machine import time import math import lcd128\_32\_fonts from lcd128\_32 import lcd128\_32

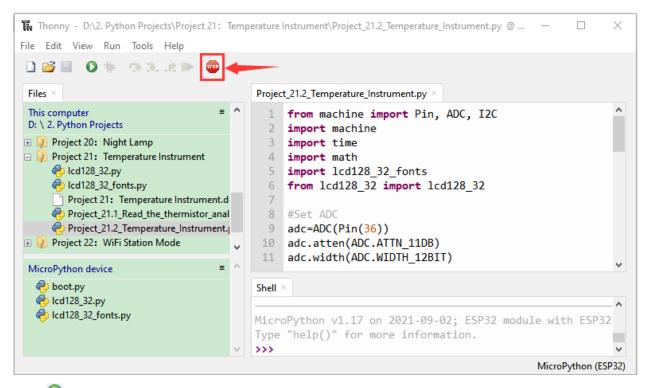
(continues on next page)

(continued from previous page)

```
#Set ADC
adc=ADC(Pin(36))
adc.atten(ADC.ATTN_11DB)
adc.width(ADC.WIDTH_12BIT)
#i2c config
clock_pin = 22
data_pin = 21
bus = 0
i2c_addr = 0x3f
use_i2c = True
def scan_for_devices():
   i2c = machine.I2C(bus,sda=machine.Pin(data_pin),scl=machine.Pin(clock_pin))
   devices = i2c.scan()
   if devices:
        for d in devices:
            print(hex(d))
   else:
        print('no i2c devices')
try:
    while True:
        adcValue = adc.read()
        voltage = adcValue / 4095 * 3.3
        Rt = 10 * voltage / (3.3-voltage)
        tempK = (1 / (1 / (273.15+25) + (math.log(Rt/10)) / 3950))
        tempC = int(tempK - 273.15)
        if use_i2c:
            scan_for_devices()
            lcd = lcd128_32(data_pin, clock_pin, bus, i2c_addr)
        lcd.Clear()
        lcd.Cursor(∅, ∅)
        lcd.Display("Voltage:")
        lcd.Cursor(0, 8)
        lcd.Display(str(voltage))
        lcd.Cursor(0, 20)
        lcd.Display("V")
        lcd.Cursor(2, ♥)
        lcd.Display("Temperature:")
        lcd.Cursor(2, 12)
        lcd.Display(str(tempC))
        lcd.Cursor(2, 15)
        lcd.Display("C")
        time.sleep(0.5)
except:
   pass
```

7.Project result

Make sure the ESP32 has been connected to the computer, click "Stop/Restart backend".



Click C"Run current script", the code starts to be executed and you'll see that the LCD 128X32 DOT displays the voltage value of the thermistor and the temperature value in the current environment. Press"Ctrl+C" or click

"Stop/Restart backend"to exit the program.

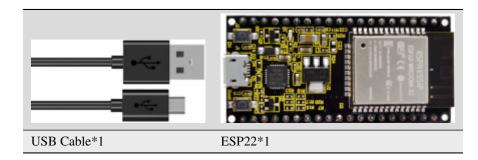
Image: Thonny - D:\2. Python Projects\Project 21: 1         File       Edit       View       Run       Tools       Help         Image: Imag	lemp	perature Instrument\Project_21.2_Temperature_Instrument.py @ — D >	×
This computer       ■         D: \ 2. Python Projects       ■         Image: Project 20: Night Lamp       ■         Image: Project 21: Temperature Instrument       Image: Project 21: Temperature Instrument         Image: Project 21: Temperature Instrument.       Image: Project 21: Temperature Instrument.         Image: Project 21: Temperature Instrument.       Image: Project 21: Temperature Instrument.         Image: Project 21: Temperature Instrument.       Image: Project 21: Temperature Instrument.         Image: Project 21: 2: Temperature Instrument.       Image: Project 21: 2: Temperature Instrument.         Image: Project 22: WiFi Station Mode       Image: Project 22: WiFi Station Mode		<pre>1 from machine import Pin, ADC, I2C 2 import machine 3 import time 4 import math 5 import lcd128_32_fonts 6 from lcd128_32 import lcd128_32 7 8 #Set ADC 9 adc=ADC(Pin(36)) 10 adc.atten(ADC.ATTN_11DB) 11 adc.atten(ADC.ATTN_11DB)</pre>	^
MicroPython device ≡ <pre>   boot.py   cld128_32.py   cld128_32_fonts.py </pre>	<	<pre>11 adc.width(ADC.WIDTH_12BIT) Shell × MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32 Type "help()" for more information. &gt;&gt;&gt; %Run -c \$EDITOR_CONTENT MicroPython (ESP3</pre>	~

# 7.24 Project 22WiFi Station Mode

## 1.Introduction

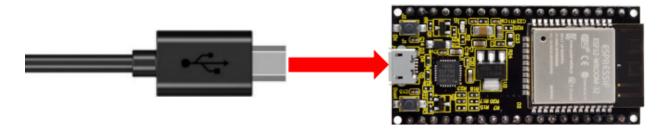
ESP32 has three different WiFi operating modes : Station modeAP mode and AP+Station mode. All WiFi programming projects must be configured with WiFi operating mode before using WiFi, otherwise WiFi cannot be used. In this project, we will learn about ESP32's WiFi Station mode.

#### 2.Components



## 3.Project wiring

Connect the ESP32 to the USB port on your computer using a USB cable.



## 4.Component knowledge

**Station mode:** When ESP32 selects Station mode, it acts as a WiFi client. It can connect to the router network and communicate with other devices on the router via WiFi connection. As shown below, the PC is connected to the router, and if ESP32 wants to communicate with the PC, it needs to be connected to the router.

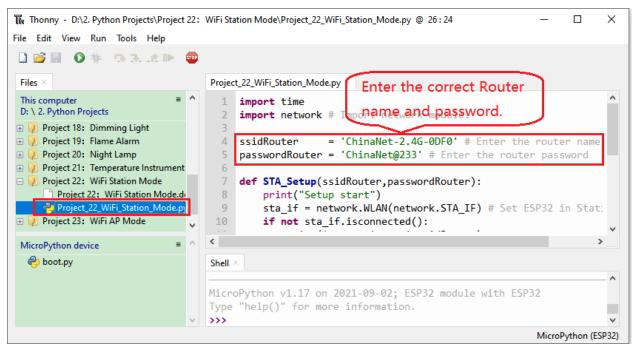


5.Project code

Codes used in this tutorial are saved in "2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		_	
Share View			~ ?
→ This PC → Software (D:) → 2. Pyth	oon Projects 🗸 こ		. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
	2/17/2022 10:21 AM	File folder	
	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

 $Open ``Thonny" click ``This computer" \rightarrow ``D:" \rightarrow ``2. Python Projects" \rightarrow ``Project 22WiFi Station Mode" and double left-click ``Project_22_WiFi_Station_Mode.py".$ 



```
import time
import network # Import network module.
ssidRouter = 'ChinaNet-2.4G-0DF0' # Enter the router name
passwordRouter = 'ChinaNet@233' # Enter the router password
def STA_Setup(ssidRouter,passwordRouter):
    print("Setup start")
    sta_if = network.WLAN(network.STA_IF) # Set ESP32 in Station mode.
    if not sta_if.isconnected():
        print('connecting to',ssidRouter)
    # Activate ESP32's Station mode, initiate a connection request to the router
    # and enter the password to connect.
        sta_if.active(True)
```

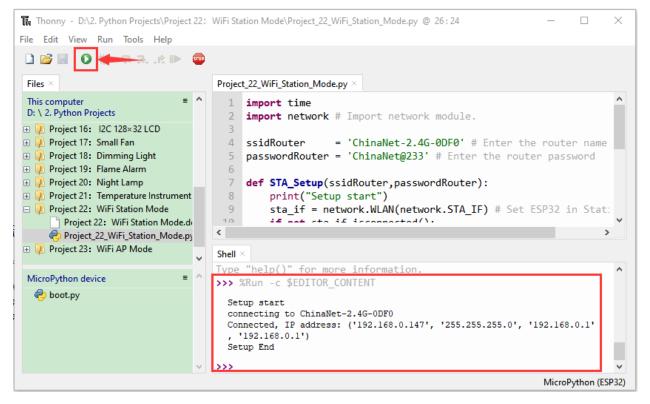
(continues on next page)

(continued from previous page)

```
sta_if.connect(ssidRouter,passwordRouter)
#Wait for ESP32 to connect to router until they connect to each other successfully.
while not sta_if.isconnected():
    pass
# Print the IP address assigned to ESP32-WROVER in "Shell".
    print('Connected, IP address:', sta_if.ifconfig())
    print("Setup End")
try:
    STA_Setup(ssidRouter,passwordRouter)
except:
    sta_if.disconnect()
```

Because the names and passwords of routers in various places are different, before the code runs, users need to enter the correct router's name and password in the box as shown in the illustration above.

After making sure the router name and password are entered correctly, click <sup>O</sup>"Run current script", the code starts to be executed and wait for ESP32 to connect to your router and print the IP address assigned by the router to ESP32 in the "Shell" window of Thonny IDE.



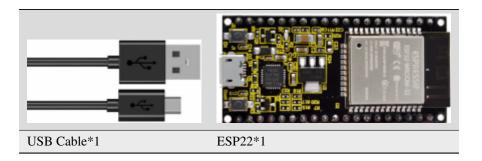
```
Shell ×
>>> %Run -c $EDITOR_CONTENT
Setup start
connecting to ChinaNet-2.4G-0DF0
Connected, IP address: ('192.168.0.147', '255.255.255.0', '192.168.0.1'
, '192.168.0.1')
Setup End
>>>
```

## 7.25 Project 23WiFi AP Mode

#### 1.Introduction

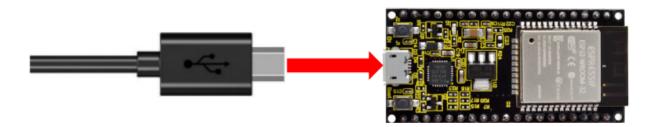
ESP32 has three different WiFi operating modes : Station modeAP mode and AP+Station mode. All WiFi programming projects must be configured with WiFi operating mode before using WiFi, otherwise WiFi cannot be used. In this project, we will learn about ESP32's WiFi AP mode.

2.Components



## 3.Project wiring

Connect the ESP32 to the USB port on your computer using a USB cable.



## 4.Component knowledge

**AP mode :** When ESP32 selects AP mode, it creates a hotspot network that is separated from the Internet and waits for other WiFi devices to connect. As shown in the figure below, ESP32 is used as a hotspot. If a mobile phone or PC wants to communicate with ESP32, it must be connected to the hotspot of ESP32. Only after a connection is established with ESP32 can they communicate.

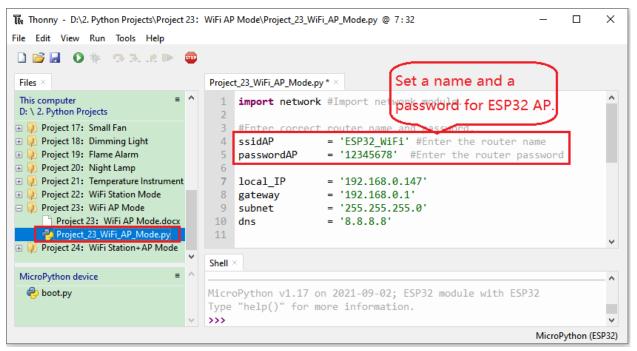


5.Project code

Codes used in this tutorial are saved in "2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		_	
Share View			~ 🕐
→ This PC → Software (D:) → 2. Pyth	on Projects 🗸 🗸	. Search 2	2. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AM	File folder	
Project 01: Hello World	2/17/2022 10:21 AM	File folder	
	2/17/2022 11:10 AM	File folder	
Project 03: LED Flashing	2/17/2022 11:12 AM	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AM	File folder	

 $Open ``Thonny" click ``This computer" \rightarrow ``D:" \rightarrow ``2. Python Projects" \rightarrow ``Project 23WiFi AP Mode", and double left-click ``Project_23_WiFi_AP_Mode.py".$ 



```
import network #Import network module.
#Enter correct router name and password.
           = 'ESP32_WiFi' #Enter the router name
ssidAP
passwordAP
             = '12345678' #Enter the router password
local_IP
              = '192.168.0.147'
gateway
              = '192.168.0.1'
subnet
              = '255.255.255.0'
              = '8.8.8.8'
dns
#Set ESP32 in AP mode.
ap_if = network.WLAN(network.AP_IF)
```

(continues on next page)

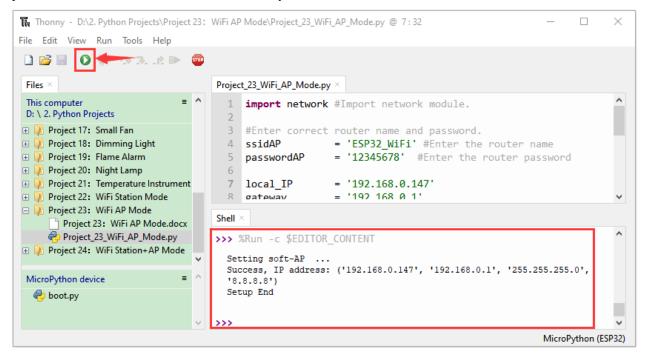
(continued from previous page)

```
def AP_Setup(ssidAP,passwordAP):
    ap_if.ifconfig([local_IP,gateway,subnet,dns])
    print("Setting soft-AP ... ")
    ap_if.config(essid=ssidAP,authmode=network.AUTH_WPA_WPA2_PSK, password=passwordAP)
    ap_if.active(True)
    print('Success, IP address:', ap_if.ifconfig())
    print("Setup End\n")
try:
    AP_Setup(ssidAP,passwordAP)
except:
    print("Failed, please disconnect the power and restart the operation.")
    ap_if.disconnect()
```

6.Project result

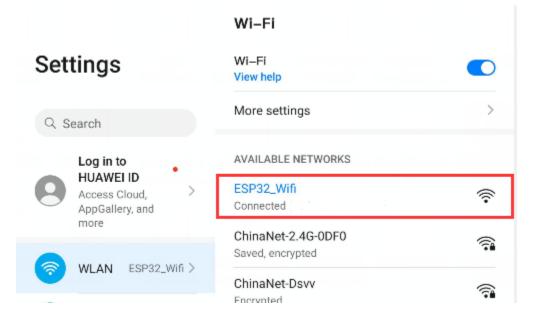
Before the code runs, you can make any changes to the AP name and password for ESP32 in the box as shown in the illustration above. Of course, you can leave it alone by default.

Click Circle Click



```
Shell ×
>>> %Run -c $EDITOR_CONTENT
Setting soft-AP ...
Success, IP address: ('192.168.0.147', '192.168.0.1', '255.255.255.0',
'8.8.8.8')
Setup End
>>>
```

Turn on the WiFi scanning function of your phone, and you can see the ssid\_AP on ESP32, which is called "ESP32\_Wifi" in this code. You can enter the password "12345678" to connect it or change its AP name and password by modifying ode.

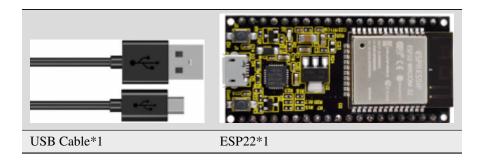


# 7.26 Project 24WiFi Station+AP Mode

1.Introduction

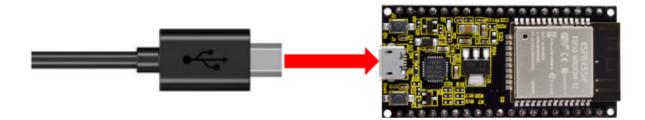
ESP32 has three different WiFi operating modes : Station modeAP mode and AP+Station mode. All WiFi programming projects must be configured with WiFi operating mode before using WiFi, otherwise WiFi cannot be used. In this project, we will learn ESP32's WiFi Station+AP mode.

2.Components



#### 3.Project wiring

Connect the ESP32 to the USB port on your computer using a USB cable.



4.Component knowledge

**AP+Station mode:** In addition to AP mode and Station mode, ESP32 can also use AP mode and Station mode at the same time. This mode contains the functions of the previous two modes. Turn on ESP32's Station mode, connect it to the router network, and it can communicate with the Internet via the router. At the same time, turn on its AP mode to create a hotspot network. Other WiFi devices can choose to connect to the router network or the hotspot network to communicate with ESP32.

#### 5.Project code

Codes used in this tutorial are saved in "2. Python Projects". If you haven't downloaded the code file yet, please click on the link to download it:Download Python Codes

2. Python Projects		—	$\Box$ $\times$
Share View			~ 🕐
→ This PC → Software (D:) → 2. Pytł	non Projects 🗸 🗸 🗸	ට 🔎 Sea	rch 2. Python
Name	Date modified	Туре	Size ^
Project 00: Boot	2/17/2022 10:21 AN	I File folder	
Project 01: Hello World	2/17/2022 10:21 AN	1 File folder	
Project 02: Turn On LED	2/17/2022 11:10 AN	1 File folder	
Project 03: LED Flashing	2/17/2022 11:12 AN	File folder	
Project 04: Breathing LED	2/17/2022 10:21 AN	File folder	

Open"Thonny" click"This computer"  $\rightarrow$  "D:"  $\rightarrow$  "2. Python Projects"  $\rightarrow$  "Project 24WiFi Station+AP Mode" and double left-click "Project\_24\_WiFi\_Station+AP\_Mode.py".

🙀 Thonny - D:\2. Python Projects\Project : File Edit View Run Tools Help	24: WiFi St	tation+AP Mode\Project_24_WiFi_Station+AP_Mode.py @ 9:32 - □ >
🗋 📸 🛃 💿 🐡 😳 3t 🕪 4 Files ×	_	ect_24_WiFi_Station+AP_Mode.py * ×
This computer ≡ D: \ 2. Python Projects	^ 1 2	import network #In Router and AP
<ul> <li>              Project 17: Small Fan Project 18: Dimming Light Project 19: Flame Alarm      </li> </ul>	3 4 5	<pre>ssidRouter = 'ChinaNet-2.4G-0DF0' #Enter the router name passwordRouter = 'ChinaNet@233' #Enter the router password</pre>
<ul> <li>Project 20: Night Lamp</li> <li>Project 21: Temperature Instrument</li> </ul>	6 7	<pre>ssidAP = 'ESP32_WiFi'#Enter the AP name passwordAP = '12345678' #Enter the AP password</pre>
<ul> <li></li></ul>	8 9 10	local_IP = '192.168.0.147' gateway = '192.168.0.1'
Project 24: WiFi Station+AP Mon		subnet = '255.255.0'
MicroPython device =	^ Shell	×
🔁 boot.py		roPython v1.17 on 2021-09-02; ESP32 module with ESP32 e "help()" for more information.
		MicroPython (ESP3

```
import network #Import network module.
               = 'ChinaNet-2.4G-0DF0' #Enter the router name
ssidRouter
passwordRouter = 'ChinaNet@233' #Enter the router password
ssidAP
              = 'ESP32_WiFi'#Enter the AP name
              = '12345678' #Enter the AP password
passwordAP
local_IP
              = '192.168.0.147'
gateway
              = '192.168.0.1'
subnet
              = '255.255.255.0'
dns
              = '8.8.8.8'
sta_if = network.WLAN(network.STA_IF)
ap_if = network.WLAN(network.AP_IF)
def STA_Setup(ssidRouter,passwordRouter):
   print("Setting soft-STA ... ")
   if not sta_if.isconnected():
        print('connecting to',ssidRouter)
        sta_if.active(True)
        sta_if.connect(ssidRouter,passwordRouter)
        while not sta_if.isconnected():
            pass
   print('Connected, IP address:', sta_if.ifconfig())
   print("Setup End")
def AP_Setup(ssidAP,passwordAP):
   ap_if.ifconfig([local_IP,gateway,subnet,dns])
   print("Setting soft-AP ... ")
    ap_if.config(essid=ssidAP,authmode=network.AUTH_WPA_WPA2_PSK, password=passwordAP)
                                                                           (continues on next page)
```

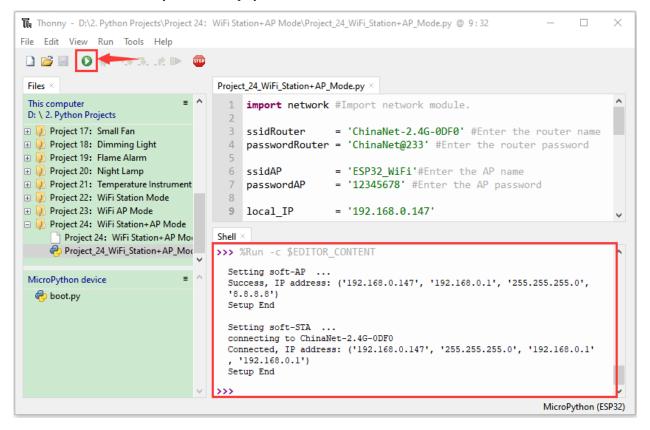
(continued from previous page)

```
ap_if.active(True)
print('Success, IP address:', ap_if.ifconfig())
print("Setup End\n")
try:
    AP_Setup(ssidAP,passwordAP)
    STA_Setup(ssidRouter,passwordRouter)
except:
    sta_if.disconnect()
    ap_if.idsconnect()
```

6.Project result

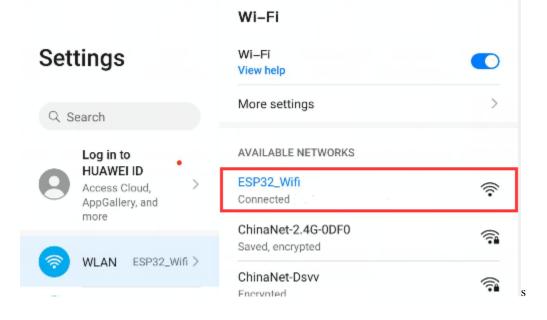
It is analogous to Project 35 and project 36. Before running the code, you need to modify ssidRouter, passwordRouter, ssidAP and passwordAP shown in the box of the illustration above.

After making sure that the code is modified correctly, click "Run current script" the code starts to be executed and the "Shell" window of Thonny IDE will display as follows:



```
Shell ×
>>> %Run -c $EDITOR_CONTENT
Setting soft-AP ...
Success, IP address: ('192.168.0.147', '192.168.0.1', '255.255.255.0',
'8.8.8.8')
Setup End
Setting soft-STA ...
connecting to ChinaNet-2.4G-0DF0
Connected, IP address: ('192.168.0.147', '255.255.255.0', '192.168.0.1'
, '192.168.0.1')
Setup End
>>>
```

Turn on the WiFi scanning function of your phone, and you can see the ssidAP on ESP32.



CHAPTER

EIGHT

# **GETTING STARTED WITH C LANGUAGE(RASPBERRY PI)**

Raspberry Pi is a card computer whose official system is Raspberry Pi OS, which can be installed on other systems, such as: ubuntu, Windows IoT. Raspberry Pi can be used as a personal server, a router camera monitoring and recognition, as well as voice interaction by connecting a camera and a voice interactive assistant.

Furthermore, Raspberry Pi leads out 40Pin pins that can be connected to various sensors and control LEDs, motors, etc, making it can be used to create a robot.

## 8.1 Install the Raspberry Pi OS System

## 8.1.1 1. Tools needed for the Raspberry Pi system

#### 1.1. Hardware Tool

1Raspberry Pi 4B/3B/2B 2Above 16G TFT Memory Card 3Card Reader 4Computer and other parts

#### 1.2. Software tools that need to be installed

Windows System 1Install putty Download linkhttps://www.chiark.greenend.org.uk/~sgtatham/putty/

PuTTY: a free	e SSH and Telnet clier × +	- L ,
$\leftrightarrow$ $\rightarrow$ G	chiark.greenend.org.uk/~sgtatham/putty/	☆ \varTheta

## **PuTTY: a free SSH and Telnet client**

Home | FAQ | Feedback | Licence | Updates | Mirrors | Keys | Links | Team Download: Stable · Snapshot | Docs | Changes | Wishlist

PuTTY is a free implementation of SSH and Telnet for Windows and Unix platforms, along with an xterm terminal emulator. It is written and maintained primarily by <u>Simon Tatham</u>.

The latest version is 0.74 **Download it here**.

**LEGAL WARNING**: Use of PuTTY, PSCP, PSFTP and Plink is illegal in countries where encryption is outlawed. We believe it is legal to use PuTTY, PSCP, PSFTP and Plink in England and Wales and in many other countries, but we are not lawyers, and so if in doubt you should seek legal advice before downloading it. You may find useful information at <u>cryptolaw.org</u>, which collects information on cryptography laws in many countries, but we can't vouch for its correctness.

Use of the Telnet-only binary (PuTTYtel) is unrestricted by any cryptography laws.

### Latest news

#### 2020-11-22 Primary git branch renamed

The primary branch in the PuTTY git repository is now called main, instead of git's default of master. For now, both branch names continue to exist, and are kept automatically in sync by a symbolic-ref on the server. In a few months' time, the alias master will be withdrawn.



## Download PuTTY: latest release (0.74)

<u>Home</u> | FAQ | Feedback | Licence | Updates | Mirrors | Keys | Links | Team Download: Stable - <u>Snapshot</u> | Docs | Changes | Wishlist

This page contains download links for the latest released version of PuTTY. Currently this is 0.74, released on 2020-06-27.

When new releases come out, this page will update to contain the latest, so this is a good page to bookmark or link to. Alternatively, here is a <u>permanent link to the 0.74 release</u>.

Release versions of PuTTY are versions we think are reasonably likely to work well. However, they are often not the most up-to-date version of the code available. If you have a problem with this release, then it might be worth trying out the <u>development snapshots</u>, to see if the problem has already been fixed in those versions.

Package file	25			
	want one of these. They include versions of ther you want the 32-bit or the 64-bit versions of the second second second second second second second second			
32-bit:	putty-0.74-installer.msi	(or by FTP)	(signature)	
64-bit:	<u>putty-64bit-0.74-installer.msi</u>	(or by FTP)	(signature)	
Unix source a .tar.gz:	rchive putty-0.74.tar.gz	( <u>or by FTP)</u>	(signature)	

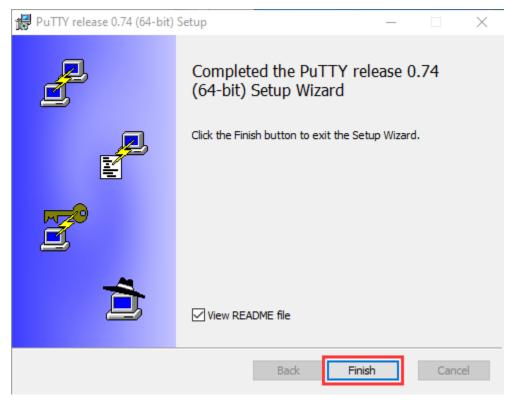
After downloading the package file putty-64bit-0.74-installer , double-click it and tap"Next".

🛃 PuTTY release 0.74 (64-bit)	Setup — 🗆 🗙
	Welcome to the PuTTY release 0.74 (64-bit) Setup Wizard
	The Setup Wizard allows you to change the way PuTTY release 0.74 (64-bit) features are installed on your computer or to remove it from your computer. Click Next to continue or Cancel to exit the Setup Wizard.
2	
Ô	
	Back Next Cancel
Click "Next".	
🛃 PuTTY release 0.74 (64-bit)	Setup — 🗆 🗙
Destination Folder	
Click Next to install to the d	efault folder or dick Change to choose another.
Install PuTTY release 0.74 (6	4-bit) to:
C:\Program Files\PuTTY\	
Change	
	Back Next Cancel

Select "Install Putty files", and click "Install"

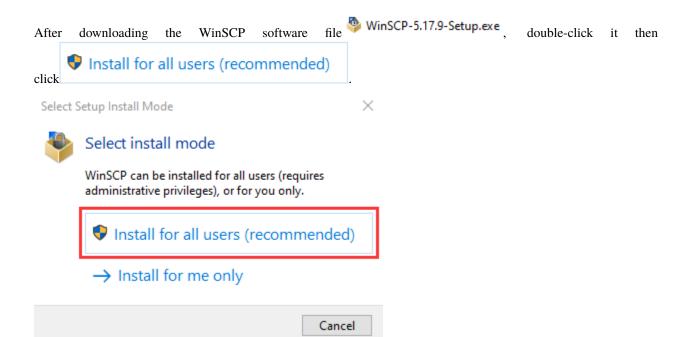
👘 Р	uTTY release 0.74 (64-bit) Setup			_		×
Pi	roduct Features Select the way you want features to	be installed.			Ē	ł
	Add shortcut to PuTTY Add shortcut to PuTTY Put install directory on Associate .PPK files wit	the PATH for	command prompts			
т	his feature requires 3914KB on your h	nard drive.				
	[	Back	🗣 Install		Cance	el l

After a few seconds, the installation is complete, click "Finish".



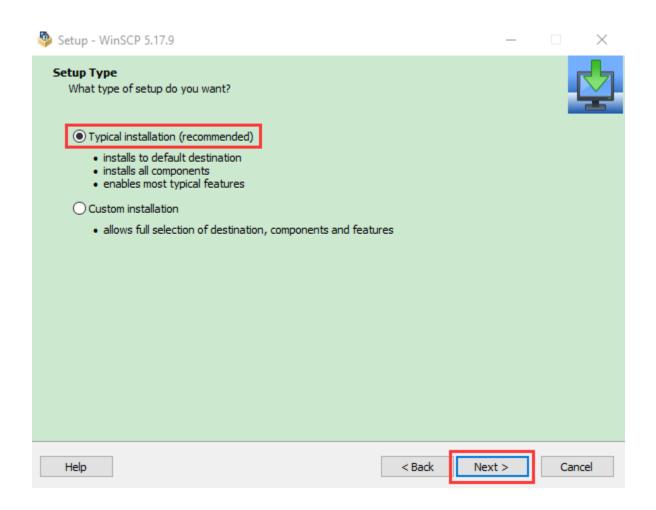
#### Remote Login software -WinSCP

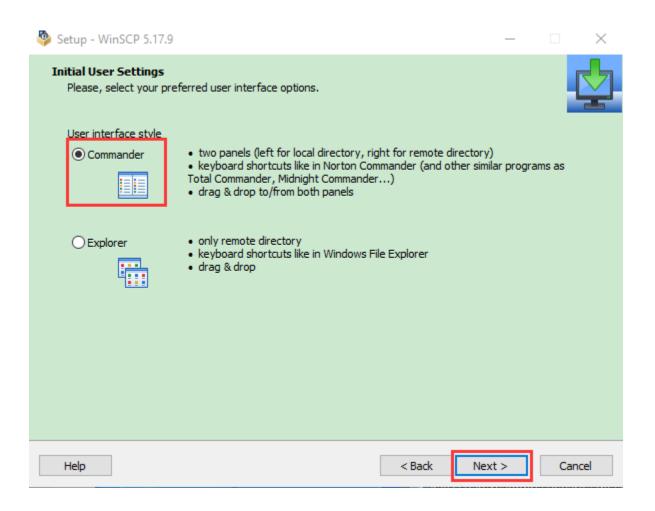
Download link: https://winscp.net/eng/download.php

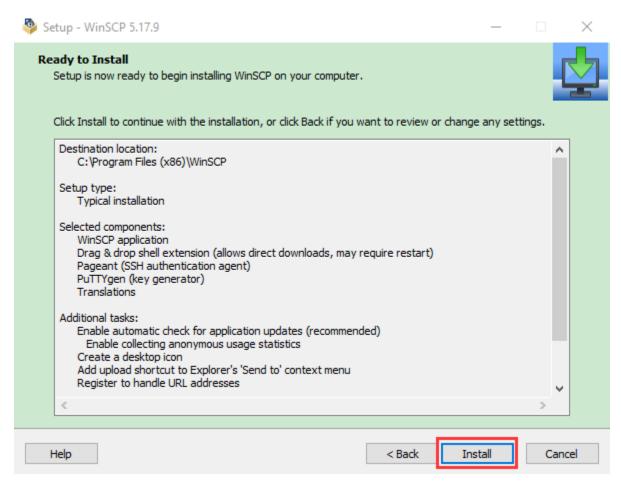


Click"Accept"then select the appropriate option and click"Next", then click"Install".

Setup - WinSCP 5.17.9		$\times$
License Agreement Please read the following important information before continuing.	[	
Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.		
You can also review this license and further details online at:	~	
https://winscp.net/eng/docs/license		
A. GNU General Public License B. License of WinSCP Icon Set C. Privacy Policy		
A. GNU GENERAL PUBLIC LICENSE		
Version 3, 29 June 2007		
Copyright (C) 2007 Free Software Foundation, Inc. < <u>https://www.fsf.org/</u> >		
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing	it	
is not allowed.	~	
Help Accept >	Can	cel







After a few seconds, the installation is complete, click "Finish".



#### Format TFT card tool- SD Card Formatter

Download link

http://www.canadiancontent.net/tech/download/SD\_Card\_Formatter.html

### ESP32 Learning Kit

SD Card Formatter	Free Formatter Download	Download File		
Software Downloads D Hardware Software D Hard Drive S SD Card Formatter 5.0.1 Update Submitted 12 May 2019 Software Review:	oftware 》 Hard Drive Formatters 》 ★★★★★★	Download SD Card Formatter 6 MB - Filesize		
		Details		
SD Card Formatter is a simple and basic formatted which is designed to be used with SD, SDHC and SDXC memory cards. The application itself isn't too different from the format utility included with Windows and includes two modes: Quick format and Overwrite format. CHS format size adjustment is the only other option. Once the appropriate card and volume label has been selected, the format can begin after hitting "Format".	Elle: Help Select.card Fil-SCNY_16G Card information Type SDHC Capacity 14.09 GB File File File Gaudi format Coss	-		
Version 5.0.1 is a freeware program which so it doesn't cost anything.	n does not have restrictions and it's free	tRank Based on many factors, we give this program a Trust rating of 5 / 10.		
CanadianContent	Register Account	•		
Software Downloads D Hardware Software D Hard Drive Download SD Card Formatter Download SD Card Formatter 5.0.	Software > Hard Drive Formatters > SD Card Formatter > 1 (x64 & x32) Free Have you tried the SD Card Formatter be recommending it by clicking the Faceboo			
Card Information Type SDHC Caracity 14.66.08	Download SD Card Formatter 5.0.1 from Hosted by Sd	card.org		
Capacity 14.69 G8	SD Card Formatter has been tested for viruses and malware         This download is 100% clean of viruses. It was tested with 24 different antivirus and anti-malware programs and was clean 100% of the time. View the full SD Card Formatter homepage for virus test results.         The file that was tested: SDCardFormatterv5_WinEN.zip.			
SD Card Formatter 5.0.1 Screenshot	Card Formatter you may be using. If you're receiving a 404 File Not Found error, this means the atter. Please do drop us a note in the event of a missing file.	publisher has taken the file offline and		

Unzip the SDCardFormatterv5\_WinEN package and double-click the SD Card Formatter file SD Card Formatter 5.0.1 Setup.exe to run it.

SD Card Formatter - InstallShield Wizard

	Preparing to Install SD Card Formatter Setup is preparing the InstallShield Wizard, which will guide you through the program setup process. Please wait. Extracting: SD Card Formatter Setup.msi
Click "Next"select "  I accept	Cancel the terms in the license agreement " and click "Next".
🕷 SD Card Formatter - InstallSi	
2	Welcome to the InstallShield Wizard for SD Card Formatter
	The InstallShield(R) Wizard will install SD Card Formatter on your computer. To continue, click Next.
	WARNING: This program is protected by copyright law and international treaties.
	< Back Next > Cancel

🕼 SD Card Formatter - InstallShield Wizard	$\times$
License Agreement Please read the following license agreement carefully.	と
END USER LICENSE AGREEMENT	^
NOTICE: BY DOWNLOADING, INSTALLING OR USING THE PRODUCT, THE ENTITY OR INDI ENTERING INTO THIS AGREEMENT AGREES TO BE BOUND BY THE FOLLOWING IF YOU DO NOT AGREE WITH ANY OF THESE TERMS, DO NOT DOWNLOAD, IN OR USE THE PRODUCT; PROMPTLY RETURN (IF APPLICABLE) THE PRODUCT SDA OR YOUR SDA DISTRIBUTOR. IF YOU REJECT THIS AGREEMENT, YOU WI ACQUIRE ANY LICENSE TO USE THE PRODUCT.	TERMS. NSTALL, TO THE
I accept the terms in the license agreement     I do not accept the terms in the license agreement InstallShield	Print
	Cancel
Click "Next" again, and then click "Install".	
🙀 SD Card Formatter - InstallShield Wizard	$\times$
Destination Folder Click Next to install to this folder, or click Change to install to a different folder.	と
Install SD Card Formatter to: C:\Program Files (x86)\SDA\SD Card Formatter\	hange

InstallShield			
ביים ביים ביים ביים ביים ביים ביים ביים			
	< Back	Next >	Cancel

🔀 SD Card Formatter - InstallShield Wizard	×
Ready to Install the Program	4.
The wizard is ready to begin installation.	
If you want to review or change any of your installation settings, click Bac exit the wizard.	:k. Click Cancel to
Current Settings:	
Setup Type:	
Typical	
Destination Folder:	
C:\Program Files (x86)\SDA\SD Card Formatter\	
User Information:	
Name:	
Company:	
InstallShield	
< Back	Cancel

After a few seconds, the installation is complete, click "Finish".

🛃 SD Card Formatter - InstallShield Wizard 🛛 🕹					
ی	InstallShield Wizard Completed The InstallShield Wizard has successfully installed SD Card	1			
	Formatter. Click Finish to exit the wizard.				
	< Back Finish Cancel				

#### 4Burn mirror system software tool— Win32DiskImager

Download linkhttps://sourceforge.net/projects/win32diskimager/

Home / Browse / System Administration / Storage / Win32 Disk Imager Win32 Disk Imager A Windows tool for writing images to USB sticks or SD/CF cards Brought to you by: gruemaster, tuxinator 2009									
★★★★★ 112 Reviews Downloads: 42,251 This Week Last Update: 2018-06-07							18-06-07		
Get Updates Share This									
Summary	Files	Reviews	Support	Wiki	Feature Requests	Bugs	Code	Mailing Lists	Blog

This program is designed to write a raw disk image to a removable device or backup a removable device to a raw image file. It is very useful for embedded development, namely Arm development projects (Android, Ubuntu on Arm, etc). Anyone is free to branch and modify this program. Patches are always welcome.

This release is for Windows 7/8.1/10. It will should also work on Windows Server 2008/2012/2016 (although not tested by the developmers). For Windows XP/Vista, please use v0.9 (in the files archive).

After downloading the software file win32diskimager-1.0.0-install.exe double-click it and then click "Run".

 $\times$ 

>SmartScreen can't be reached right now					
Check your Internet connection. Windows Defender SmartScreen is unreachable and can't help you decide if this app is ok to run.					
Publisher: Unknown Publisher App: win32diskimager-1.0.0-install.exe					
Run Don't Run					

After selecting I accept the agreement and click "Next".

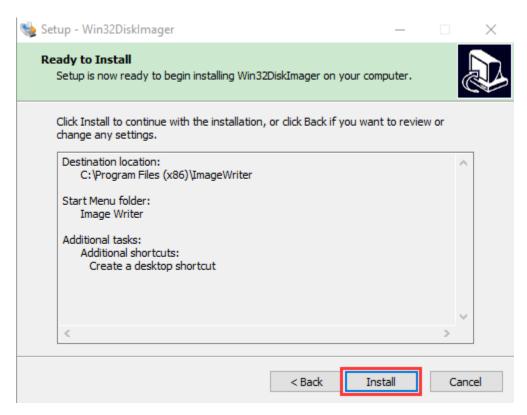
👒 Setup - Win32DiskImager			$\times$
License Agreement Please read the following important information before continuing.		Q	
Please read the following License Agreement. You must accept the ter agreement before continuing with the installation.	ms of th	is	_
This program is licensed under the GNU GPL Version 2 License. Includ	ed librari	ies \land	
are licensed under GPL v2 and LGPL v2.1 accordingly.			
GNU GENERAL PUBLIC LICENSE Version 2, June 1991			
Copyright (C) 1989, 1991 Free Software Foundation, Inc.,		~	
<ul> <li>I accept the agreement</li> <li>I do not accept the agreement</li> </ul>			
Next	>	Can	cel

Click "Browse..." select the location where Win32DiskImager is installed and click "Next".

📚 Setup - Win32DiskImager	_		$\times$
Select Destination Location Where should Win32DiskImager be installed?			
Setup will install Win32DiskImager into the following folder.			
To continue, click Next. If you would like to select a different folder,	click Bro	owse.	
C:\Program Files (x86)\ImageWriter	Br	rowse	
At least 44.2 MB of free disk space is required.			
< Back Nex	t >	Car	ncel

1. Click "Browse..." select the location where Win32DiskImager is installed and click "Next".

👒 Setup - Win32Disklmager	_		×
Select Start Menu Folder Where should Setup place the program's shortcuts?			
Setup will create the program's shortcuts in the following St	tart Meni	u folder.	
To continue, dick Next. If you would like to select a different folder,	click Bro	wse.	
Image Writer	Br	owse	
< Back Nex		Can	cel
<ol> <li>Select Create a desktop shortcut and click "Next" and then of Setup - Win32DiskImager</li> </ol>	lick"In	stall".	×
Select Additional Tasks Which additional tasks should be performed?			
Select the additional tasks you would like Setup to perform while inst Win32DiskImager, then click Next.	talling		
Additional shortcuts:			
Create a desktop shortcut			
< Back Nex	t >	Can	cel



d. After a few seconds, the installation is complete, click "Finish".

🍓 Setup - Win32Disklmager	- 🗆 ×
	Completing the Win32DiskImager on your computer. The application may be launched by selecting the installed shortcuts.         Click Finish to exit Setup.         View README.txt         Launch Win32DiskImager
	Finish

5Scan for IP address software tool-WNetWatcher

Download Linkhttp://www.nirsoft.net/utils/wnetwatcher.zip

#### 1.3. Raspberry PI mirror system

Download link for the latest version

https://www.raspberrypi.org/downloads/raspberry-pi-os/

Download link for the old version

- Raspbian
- https://downloads.raspberrypi.org/raspbian/images/
- Raspbian full
- https://downloads.raspberrypi.org/raspbian\_full/images/
- Raspbian lite
- https://downloads.raspberrypi.org/raspbian\_lite/images/

We use the 2020.05.28 version in the tutorial and recommend you to use this version(Please download this version as shown in the picture below.)

https://downloads.raspberrypi.org/raspios\_full\_armhf/images/raspios\_full\_armhf-2021-05-28/

## Index of /raspios\_full\_armhf/images/raspios\_full\_armhf-2021-05-28

Name	Last mod	ified	<u>Size</u>	<u>Description</u>
Parent Directory			-	
2021-05-07-raspios-buster-armhf-full.info	2021-05-07	16:23	288K	_
2021-05-07-raspios-buster-armhf-full.zip	2021-05-07	16:35	2.8G	
2021-05-07-raspios-buster-armhf-full zip.sha1	2021-05-28	15:49	83	
2021-05-07-raspios-buster-armhf-full.zip.sha256	2021-05-28	15:49	107	
2021-05-07-raspios-buster-armhf-full.zip.sig	2021-05-28	15:00	488	
2021-05-07-raspios-buster-armhf-full.zip.torrent	2021-05-28	15:50	28K	

## 8.1.2 2. Install Raspberry Pi OS system on Raspberry Pi 4B:

2.1.Connect the TFT memory card to a card reader, then plug the card reader into a computer's USB port.2.2.Use the SD Card Formatter to format a TFT memory card, as illustrated below

📔 SD Card Format	ter	×
File Help		
Select card		
E:\-boot		~
		Refresh
Card information		
Туре	SDXC	52
Capacity	59.48 GB	XC
Formatting options <ul> <li>Quick format</li> </ul>		
Overwrite forma	t	
CHS format size	adjustment	
Volume label		
boot		
		Format
SD Logo, SE	HC Logo and SDXC Logo a	re trademarks of SD-3C, LLC.

SD Card Fo	ormatter	$\times$
File Help		
Select card		
E:\-boot		~
		Refresh
SD Card Form	atter	
Do No ove cor	rmatting will erase all data on this card. you want to continue? te: As formatting can take some time (espe erwrite option is selected), please make su nputer is connected to a power supply and de is disabled.	re that your
	Yes	No
hast		
boot		
SD Lo	go, SDHC Logo and SDXC Logo are trademark	Format ks of SD-3C, LLC.
File Help		~
- Select card -		
E:\-boot		~
		0-6
SD Car	d Formatter	×
Card Type Capa Form Q Q O O C Volun	Formatting was successfully complete Volume information: File system: exFAT Capacity: 59.45 GB (63,831,015,424 byt Free space: 59.45 GB (63,830,622,208 b Cluster size: 128 kilobytes Volume label: boot	es)
boot		
		Format
SD Lo	go, SDHC Logo and SDXC Logo are trademarl	ks of SD-3C, LLC.

### 2.3.Burn System:

Use Win32DiskImager to burn the official Raspberry Pi OS mirror to the TFT memory card.

🐝 Win32 Disk Imager - 1.0	Choose the correct letter
Image File	Det a
inistrator/Desktop/2020-12-02-ras	pios-buster-armhf-full.img 📔 [E:\] 🔻
Hash	2
None 🔻 Generate Copy	Click, and then find the mirror
	system(".img"file)that you
Read Only Allocated Partitions	download and unzipped
Progress	
Click Write	to write the system
Cancel Read Writ	te Verify Only Exit
Waiting for a task.	
👒 Win32 Disk Imager - 1.0	- 🗆 X
Image File	Device
inistrator/Desktop/2020-12-02-rasp	pios-buster-armhf-full.img 📔 [E:\] 🔻
Hash 💊 Confirm overwrite - 1.0	×
None 🔥 Writing to a phys	ical device can corrupt the device.
(Target Device: [E	() "boot")
	• •
	want to continue?
Are you sure you	• •
Are you sure you	want to continue?
Are you sure you	want to continue?
Are you sure you	Yes No

👒 Win32 Disk Imager - 1.0		_	$\Box$ $\times$
Image File			Device
inistrator/Desktop/2020-	-12-02-raspios-buster-armhf-	full. img 📔	[E:\] ▼
Hash	🍓 Complete - 1.0 🛛 🗙		
None 🔻 Generate	Write Successful.		
Read Only Allocated ) Progress	ОК		
Cancel Read	Write Verify	0 Only	Exit
Done.			08:40/08:40

After the mirror system is burned, don't pull out the card reader, use a notepad to create a file named **SSH** and delete\*\*.txt\*\*, then copy it to the boot directory of the TFT card, so that you can open the SSH login function, as shown in the following figure:

← → × ↑ 🕳 → This PC → boot (	E:)			~ Č	,P Search boot
	^	Name	Date modified	Туре	Size
📌 Quick access		start.elf	11/26/2020 5:30 PM	ELF File	2,869 KB
Desktop	#	start_cd.elf	11/26/2020 5:30 PM	ELF File	771 KB
Downloads		start_db.elf	11/26/2020 5:30 PM	ELF File	4,674 KB
Documents		start_x.elf	11/26/2020 5:30 PM	ELF File	3,610 KB
Pictures		start4.elf	11/26/2020 5:30 PM	ELF File	2,162 KB
		start4cd.elf	11/26/2020 5:30 PM	ELF File	771 KB
This PC		start4db.elf	11/26/2020 5:30 PM	ELF File	3,627 KB
3D Objects		start4x.elf	11/26/2020 5:30 PM	ELF File	2,904 KB
E Desktop		bcm2708-rpi-b.dtb	11/26/2020 5:30 PM	DTB File	25 KB
Cocuments		bcm2708-rpi-b-plus.dtb	11/26/2020 5:30 PM	DTB File	25 KB
Downloads		bcm2708-rpi-b-rev1.dtb	11/26/2020 5:30 PM	DTB File	25 KB
b Music		bcm2708-rpi-cm.dtb	11/26/2020 5:30 PM	DTB File	25 KB
E Pictures		bcm2708-rpi-zero.dtb	11/26/2020 5:30 PM	DTB File	25 KB
Videos		bcm2708-rpi-zero-w.dtb	11/26/2020 5:30 PM	DTB File	26 KB
		bcm2709-rpi-2-b.dtb	11/26/2020 5:30 PM	DTB File	26 KB
L Windows10 1909 (C:)		bcm2710-rpi-2-b.dtb	11/26/2020 5:30 PM	DTB File	26 KB
🕳 新加卷(D:)		bcm2710-rpi-3-b.dtb	11/26/2020 5:30 PM	DTB File	28 KB
boot (E:)		bcm2710-rpi-3-b-plus.dtb	11/26/2020 5:30 PM	DTB File	28 KB
		bcm2710-rpi-cm3.dtb	11/26/2020 5:30 PM	DTB File	26 KB
🔫 New folder (\\desktop-eng) (Z:)		bcm2711-rpi-4-b.dtb	11/26/2020 5:30 PM	DTB File	47 KB
-		bcm2711-rpi-400.dtb	11/26/2020 5:30 PM	DTB File	47 KB
boot (E:)		bcm2711-rpi-cm4.dtb	11/26/2020 5:30 PM	DTB File	47 KB
overlays		bootcode.bin	11/26/2020 5:30 PM	BIN File	52 KB
- USB Drive (F:)		fixup.dat	11/26/2020 5:30 PM	DAT File	8 KB
Since (P.)		fixup_cd.dat	11/26/2020 5:30 PM	DAT File	4 KB
🥏 Network		fixup_db.dat	11/26/2020 5:30 PM	DAT File	11 KB
DESKTOP-1V3JQ2C		fixup_x.dat	11/26/2020 5:30 PM	DAT File	11 KB
DESKTOP-901C3HI		fixup4.dat	11/26/2020 5:30 PM	DAT File	6 KB
DESKTOP-BDC71VT		fixup4cd.dat	11/26/2020 5:30 PM	DAT File	4 KB
*		fixup4db.dat	11/26/2020 5:30 PM	DAT File	9 KB
DESKTOP-CMPIKJB		fixup4x.dat	11/26/2020 5:30 PM	DAT File	9 KB
DESKTOP-SMT1PCK		LICENCE.broadcom	9/30/2020 12:00 PM	BROADCOM File	2 KB
UYJ		COPYING.linux	5/27/2020 10:57 AM	LINUX File	19 KB
FTI69C9C26XOD05		overlays	12/2/2020 12:39 PM	File folder	
HK4KUMDY9PBVSC0		SSH 📄 SSH	12/8/2020 11:48 AM	Text Document	0 KB

Pull out the card reader.

#### 2.4.Log in system:

The following operations require raspberry to share the same LOCAL area network with the PC.

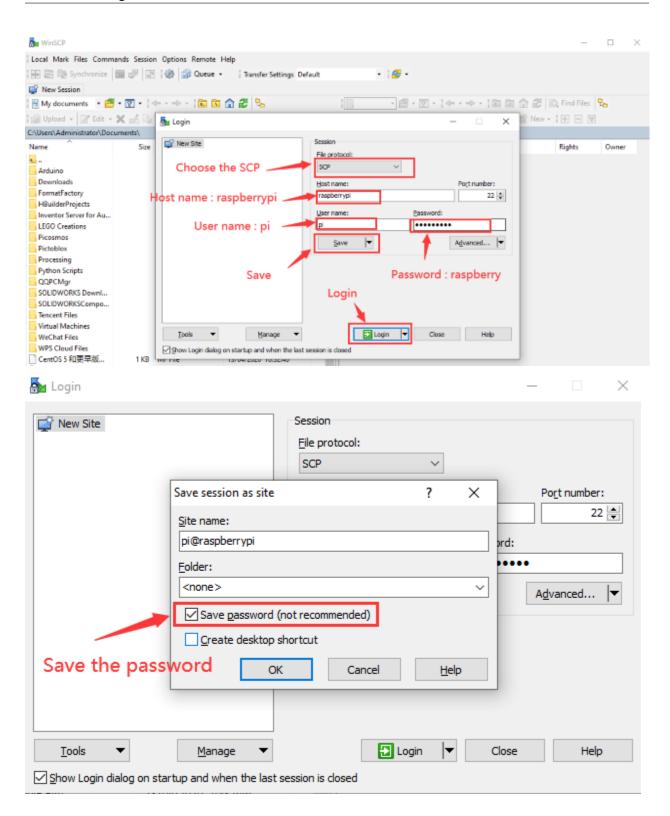
Insert the burned TFT memory card into the Raspberry Pi, connect internet cables and plug in power. If there is a screen and a HDMI cable of Raspberry Pi, connect the screen, and you can see the Raspberry Pi OS startup screen. If there is not a HDMI cable of Raspberry Pi, you can enter the desktop of Raspberry Pi via SSH remote login software—WinSCP and xrdp.

#### **Remote login**

Use WinSCP to log in using the default Raspberry Pi system namedefault user namedefault password.

Note that only a raspberry pi can be connected to a network.

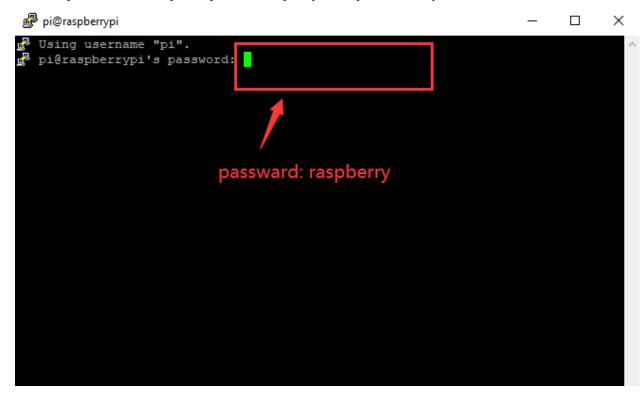
#### **ESP32 Learning Kit**



	🚰 / - pi@raspberry - Win	ICP .								
1	Local Mark Eiles Commands Session Options Bemote Help									
	🐨 🗃 😓 Synchronize 📷 🖉 😰 🛞 🚳 Queue - 🛛 Transfer Settings Default - 🛛 💋 -									
	pi@raipberry X and New Service									
	💶 De 🔹 🥌 • 🔯 • 🗠				<mark>  / <root> * 🚰 * 🛐 *</root></mark>   (+ + +)					
	🖟 Upload 🔹 📝 Edit 🔹	× 15 B	Properties 📑 Ne	w - 🕂 🗕 🕅	📔 Download - 📝 Edit - 💥 🛃 🕞 Pro	perties 📑 New - 🛛 🕂 🖃 😾				
(	:\Users\Administrator\Des	ktop\			1					
	Name	Size	Ђре	Changed ^	Name	Size Changed				
	<b>.</b> .		Parent directory	19/10/2020 09:3	<b>t</b>	20/08/2020 12:09:38				
	3D8S_CSharp_Control		File folder	07/08/2020 14:1	bin	20/08/2020 11:36:31				
	4wd_motor_test		File folder	17/08/2020 17:5	boot	01/01/1970 01:00:00				
	7大课程收获		File folder	29/09/2020 17:5	dev	20/08/2020 12:09:38				
	15 ble all		File folder	15/09/2020 11:5	etc	20/08/2020 12:09:48				
	16and8Game		File folder	16/09/2020 11:5	home	20/08/2020 11:31:10				
	alienzhangyw-BlockP		File folder	11/08/2020 11:4	lib	20/08/2020 11:45:50				
	arduino-esp32-esp32s2		File folder	31/08/2020 15:1	lost+found	20/08/2020 12:08:08				
	Arm_car		File folder	31/07/2020 15:	media	20/08/2020 11:26:08				
	AutoCAD		File folder	21/09/2020 18:0	met	20/08/2020 11:26:08				
	bluetooth test		File folder	24/08/2020 16:1	opt	20/08/2020 11:43:02				
	car_test		File folder	04/09/2020 13:5	proc	01/01/1970 01:00:00				
	EB0028 原理图		File folder	15/10/2020 17:1	root	20/08/2020 12:09:43				
	esp		File folder	12/09/2020 10:0	run	20/08/2020 12:10:05				
	esp32_test		File folder	14/09/2020 10:2	sbin	20/08/2020 11:45:50				
	esp8266 arduino		File folder	09/10/2020 154	stv	20/08/2020 11:26:08				
	esp8266_test		File folder	09/10/2020 15:5	595	01/01/1970 01:00:01				
	esp-idf		File folder	03/09/2020 16:5	_ tmp	20/08/2020 12:09:45				
	Grove-Beginner-Kit-f		File folder	09/10/2020 09:1	usr	20/08/2020 11:38:05				
	HT16K33_8x16dot_ma		File folder	28/08/2020 14:5	var	20/08/2020 12:09:38				
	keyestudio-e-Paper		File folder	19/06/2020 11:2						
	M_car		File folder	28/08/2020 16:2	L					
	ottol		File folder	16/10/2020 08:4						
	processing_test		File folder	10/10/2020 15:5						
	ps2_arm_car		File folder	31/07/2020 15:5	The system file for the r	aspberry PI system				
	RGB点阵		File folder	24/08/2020 17:0						
	TS1693 CQRobot PAJ		File folder	07/09/2020 17:5						
	TS1695 CQRobot BM		File folder	07/09/2020 144						
	TS1727 CQROBOT AD		File folder	27/09/2020 10:4						
and mac address	turtle car code		File folder	21/08/2020 11-3 M	<i>i</i>					

#### View the ip address and mac address

Click to open terminal and input the password: raspberry, and tap"Enter" on keyboard.



🚰 pi@raspberrypi: ~  $\times$ Using username "pi". pi@raspberrypi's password: Linux raspberrypi 5.4.51-v71+ #1333 SMP Mon Aug 10 16:51:40 BST 2020 armv71 The programs included with the Debian GNU/Linux system are free software; the exact distribution terms for each program are described in the individual files in /usr/share/doc/\*/copyright. Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law. Last login: Mon Oct 19 03:54:47 2020 SSH is enabled and the default password for the 'pi' user has not been changed. This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password. Wi-Fi is currently blocked by rfkill. Use raspi-config to set the country before use. pi@raspberrypi:~ \$

After successfully login, open the terminal, input ip a and tap"Enter" keyboard to view the ip address and mac address.

```
\times
🧬 pi@raspberrypi: ~
                                                                          Wi-Fi is currently blocked by rfkill.
Use raspi-config to set the country before use.
pi@raspberrypi:~ $ ip a
l: lo: <LOOPBACK,UP,LOWER UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t glen 1000
   link/loopback 00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
      valid lft forever preferred lft forever
    inet6 ::1/128 scope host
       valid lft forever preferred lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER UP> mtu 1500 qdisc mq state UP group defa
ult qlen 1000
    link/ether dc:a6:32:17:61:9c brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.128/24 prd 192.168.1.255 scope global dynamic noprefixroute e
th0
       valid lft 1357sec preferred lft 1132sec
    inet6 fe80::le7d:5653:59e9:3262/64 scope link
      valid lft forever preferred lft forever
3: wlan0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qle
n 1000
    link/ether dc:a6:32:17:61:9d brd ff:ff:ff:ff:ff:ff
pi@raspberrypi:~ $
```

From the above figure, mac address of this Raspberry Pi is a6:32:17:61:9c, and ip address is 192.168.1.128(use ip address to finish xrdp remote login).

Since mac address never changes, you could confirm ip via mac address when not sure which ip it is.

#### Fix the IP address of Raspberry Pi

IP address is changeable, therefore, we need to make IP address fixed for convenient use.

Follow the below steps: Switch to root user If without root user's password Set root password Input password in the terminal: sudo passwd root to set password. Switch to root user su root Fix the configuration file of IP address Firstly change IP address of the following configuration file. #New IP address:address 192.168.1.99 Copy the above new address to terminal and tap"Enter"keyboard. **Configuration File:** echo -e ' auto eth0 iface eth0 inet static #Change IP address address 192.168.1.99 netmask 255.255.255.0 gateway 192.168.1.1 network 192.168.1.0 broadcast 192.168.1.255 dns-domain 119.29.29.29 dns-nameservers 119.29.29.29 metric 0 mtu 1492 '>/etc/network/interfaces.d/eth0

pi@raspberrypi:~ \$ su root
Password:
root@raspberrypi:/home/pi# echo -e '
> auto eth0
> iface eth0 inet static
> #Change IP address
> address 192.168.1.99
> netmask 255.255.255.0
> gateway 192.168.1.1
> network 192.168.1.0
> broadcast 192.168.1.255
> dns-domain 119.29.29.29
> dns-nameservers 119.29.29.29
> metric 0
> mtu 1492
> '>/etc/network/interfaces.d/eth0
root@raspberrypi:/home/pi#

Reboot the system to activate the configuration file.

Input the restart command in the terminal: sudo reboot

You could log in via fixed IP afterwards.

Check IP to ensure IP address fixed well.

```
pi@raspberrypi:~ $ ip a
1: 10: <LOOPBACK, UP, LOWER UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid lft forever preferred lft forever
    inet6 ::1/128 scope host
       valid lft forever preferred lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1492 qdisc mq state UP group defa
ult qlen 1000
    link/ether dc:a6:32:17:61:9c brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.99/24 brd 192.168.1.255 scope global eth0
       valid lft forever preferred lft forever
    inet 192.168.1.128/24 brd 192.168.1.255 scope global secondary dynamic nopre
 fixroute eth0
       valid 1ft 1730sec preferred 1ft 1505sec
    inet6 fe80::le7d:5653:59e9:3262/64 scope link
       valid lft forever preferred lft forever
3: wlan0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qle
 n 1000
    link/ether dc:a6:32:17:61:9d brd ff:ff:ff:ff:ff:ff
pi@raspberrypi:~ $
```

Log in desktop on Raspberry Pi wirelessly

If we don't have an HDMI cable to connect to the display, can we wirelessly log in to the Raspberry Pi desktop from the Windows desktop? Yes, there are many methods, VNC and Xrdp are commonly used to log in desktop of Raspberry Pi wirelessly.

Let's take an example of Xrdp.

Install Xrdp Service in the terminal

Installation commands:

Switch to root User: su root

Installation commands: apt-get install xrdp

Enter y and tap"Enter"keyboard...

As shown below:

🛃 pi@raspberrypi: ~	_	×
the exact distribution terms for each program are described in the individual files in /usr/share/doc/*/copyright.		^
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.		
SSH is enabled and the default password for the 'pi' user has not b This is a security risk - please login as the 'pi' user and type 'p a new password.		
<pre>pi@raspberrypi:~ \$ sudo apt-get install xrdp Reading package lists Done Building dependency tree Reading state information Done The following additional packages will be installed: ssl-cert</pre>		
Suggested packages: openssl-blacklist guacamole xrdp-pulseaudio-installer The following NEW packages will be installed: ssl-cert xrdp 0 upgraded, 2 newly installed, 0 to remove and 373 not upgraded.		
Need to get 415 kB of archives. After this operation, 2,722 kB of additional disk space will be use Do you want to continue? [Y/n]	d.	~

Open the remote desktop connection on Windows

Press WIN+R on keyboard and enter mstsc.exe.

As shown below:

🖅 Run		$\times$
	Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.	
<u>O</u> pen:	mstsc.exe	~
	OK Cancel <u>B</u> rowse	

Enter the IP address of the Raspberry Pi, as shown below. Click "Connect" and then click "Connect" again. 192.168.1.99 is the ip address we use, you could change it into your IP address.

Nemote Desktop Connection		
Do you trust this remote connection?		
This remote connection could harm your local or remote computer. Make sure that you trust the remote computer before you connect.		
Type: Remote Desktop Connection Remote computer: 192.168.1.99		
Don't ask me again for connections to this computer		
Show Details Connect Canc	el	

A prompt will appear and you can click"Yes".

Remote Desktop Connection	<	
The identity of the remote computer cannot be verified. Do you want to connect anyway?	D	
This problem can occur if the remote computer is running a version of Windows that is earlier than Windows Vista, or if the remote computer is not configured to support server authentication.		
For assistance, contact your network administrator or the owner of the remote computer.		
Don't ask me again for connections to this computer		
Yes No		

Then enter the user name: pi ,and the default password: raspberry, as shown below:

192.168.1.253 - 远程桌面连接	_	×
Login to raspberrypi     Just   Just   connecting     Session   vsg   username   pi   password     WK密码:   raspberry     OK		
<		>:

Click"OK" or tap"Enter" keyboard, you will view the desktop of Raspberry Pi OS, as shown below:



Now, we finish the basic configuration of the Raspberry Pi OS system.

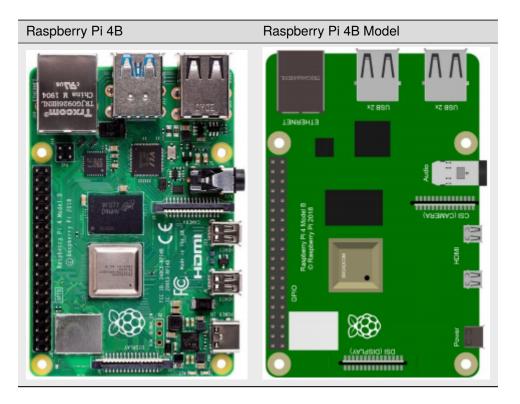
## 8.2 Preparation of C language control basic hardware:

C language is a programming language with a considerably fast running speed. There are numerous software and system core code written in it, such as Linux system. Notably, hardware MCU and embedded class are not exception. Thereby, it makes sense to learn the C language to control hardware.

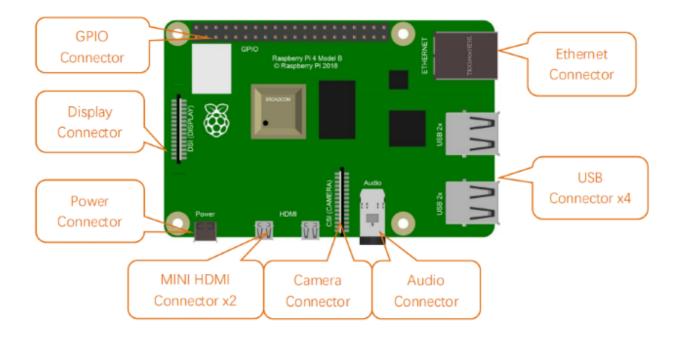
### 8.2.1 (1)Description of basic raspberry pi accessories

#### Raspberry Pi 4B

Below are the raspberry pi pictures and model pictures supported by this learning kit. There are 40 pins.



**Hardware Interfaces** 

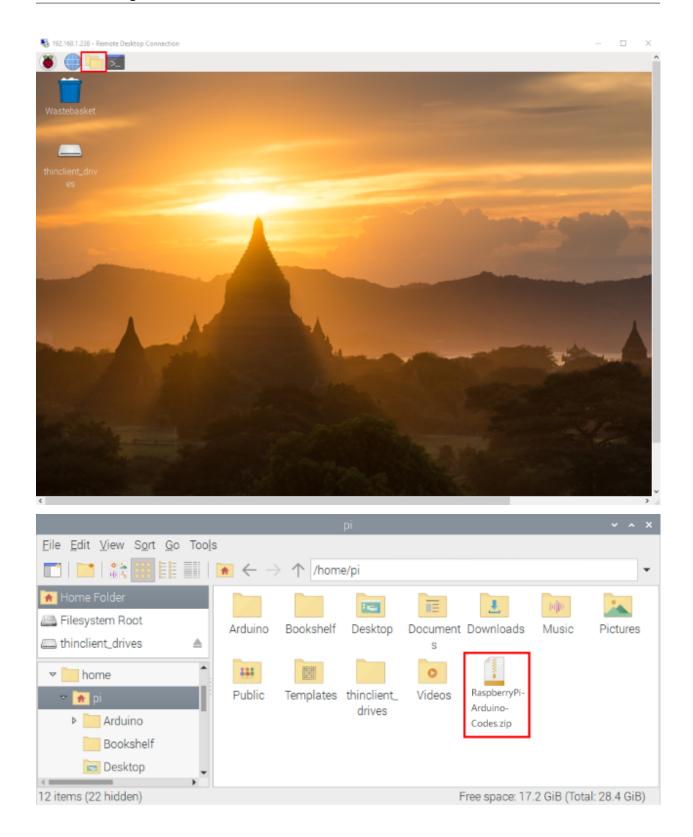


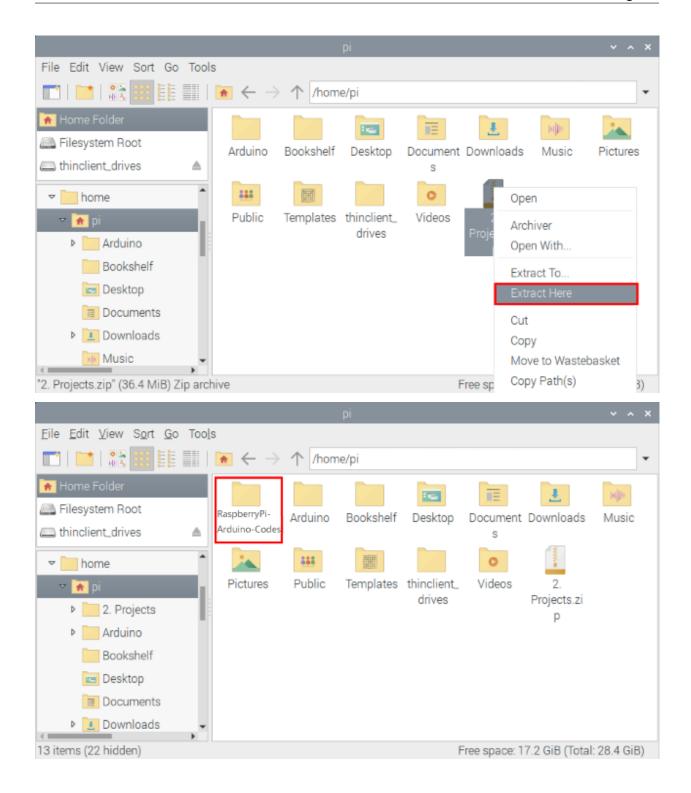
8.2.2 Raspberry Pi +ESP32 main board + breadboard +USB cable, as shown below



(3)Copy Example Code Folder to Raspberry Pi

Place example code folder to the pi folder of Raspberry Pi. Just copy and paste the **2. Projects.zip** file (the default is **ZIP** file)that we provided into user pi and unzip it, as shown below:





		2.	Projects				~ ~	×
file <u>E</u> dit <u>V</u> iew S <u>o</u> rt <u>G</u> o Too <u>l</u> e	S							
🗂   🔛   👯 🧱 📑	$\blacktriangleright$ $\leftarrow$ $\rightarrow$	↑ /hom	e/pi/Raspbe	rry Pi-Arduin	o-Codes			•
附 Home Folder								Î
🔜 Filesystem Root	Project	Project	Project	Project	Project	Project	Project	
🗅 thinclient_drives 🛛 🔺	01 : Hello	02 : Turn	03 : LED	04 :	05 :	06 : RGB	07:	
	World	On LED	Flashing	Breathing	Traffic Lig	LED	Flowing	
▶ etc								
🗢 📃 home	Project	Project	Project	Project	Project	Project	Project	
🗢 📻 pi	08:1-	09:4-	10:8×8	11:	12:	13:	14: Mini	
🝷 📜 2. Projects	Digit Digit	Digit Digit	Dot-matri	74HC595	Active Bu	Passive B	Table La	
Project 01 : Hello								
	Project	Project	Project	Project	Project	Project	Project	-
D items					Free spac	e: 17.2 GiB (T	otal: 28.4 Gi	B)

Linux SystemRaspberry Pi

## 8.2.3 (2) Download and install Arduino IDE

1First, click on Raspberry Pi's browser.

b 192.168.0.167 - Remote Desktop Connection	_	$\times$
🛞 💮 🛅 🔟		^
		ŀ
Wastebasket		
thinclient_driv		
es		~
<		 >:

2. Enter the Official Arduino website in your browser: www.arduino.cc/en/software , as shown below:

퉣 192.168.0.167 - Remote	Desktop Connection			-		$\times$
👅 🛑 🔽	💽 New Tab - Chromium					^
		New Tab - Chromiu	m			11
New Tab	× +					
	ww.arduino.cc/en/software					
						~
٢.						> ,i
No. 192.168.0.167 - Remote D				-		×
🗾 🚽 🌐 ě	Software   Arduino - C					Î
		re   Arduino - Chromiu	im			۰,
💿 Software   Arduin						_
$\leftarrow \rightarrow \mathbf{G}$ $\square$ ard	ino.cc			0 ☆	6	U
00	HARDWARE SOFTWARE CLOUD C	OCUMENTATION - CO	MMUNITY - BLOG ABOUT			
[	Downloads					
			DOWNLOAD OPTIONS			
	Arduino IDE 1.8.19		Windows Win 7 and newer Windows ZIP file			
	The open-source Arduino Software (IDE) makes	it easy to write code	Windows app Win 8.1 or 10 Ge			
	and upload it to the board. This software can be Arduino board.		Linux 32 bits			
	Refer to the Getting Started page for Installation	on instructions.	Linux 64 bits Linux ARM 32 bits			
	SOURCE CODE		Linux ARM 64 bits			
	Active development of the Arduino software is I		Mac OS X 10.10 or newer			
	See the instructions for <b>building the code</b> . Late code archives are available <b>here</b> . The archives a					
	they can be verified using this gpg key.					
(						> .:

There are various versions of IDE for Arduino. Just download a version compatible with your system (install the lasted Arduino IDE) and click"Linux ARM 32 bits".

# Downloads



## Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the Getting Started page for Installation instructions.

#### SOURCE CODE

Active development of the Arduino software is **hosted by GitHub**. See the instructions for **building the code**. Latest release source code archives are available **here**. The archives are PGP-signed so they can be verified using **this** gpg key.

#### Just click JUST DOWNLOAD to download



Windows Win 7 and newer Windows ZIP file

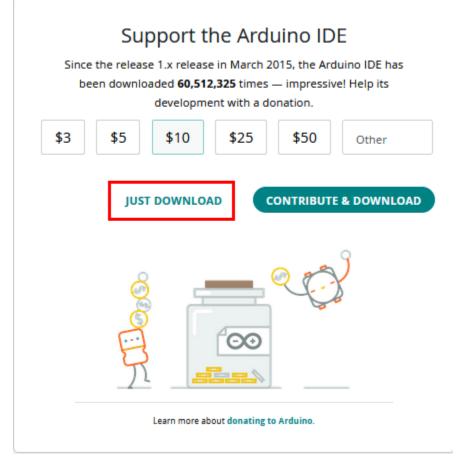
Windows app Win 8.1 or 10 Get 👫

Linux 32 bits Linux 64 bits Linux ARM 32 bits Linux ARM 64 bits

Mac OS X 10.10 or newer

Release Notes

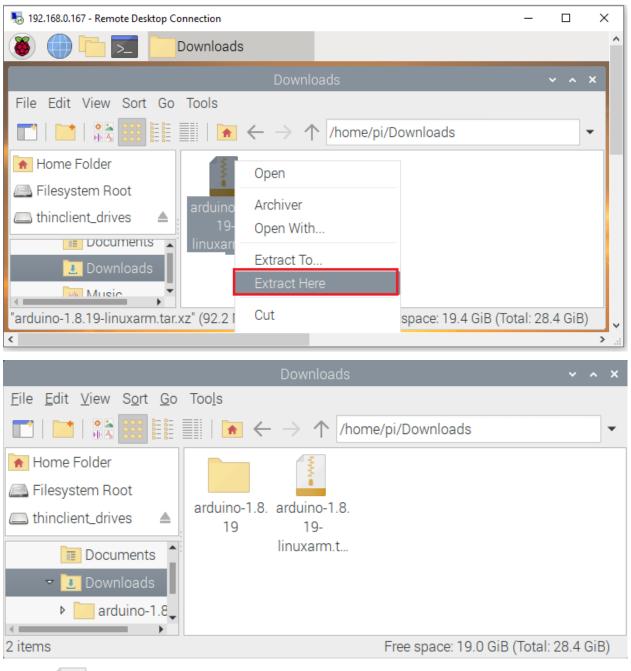
Checksums (sha512)



After a few seconds, the lastest Arduino IDEArduino 1.8.19 versionzip file can be directly downloaded.

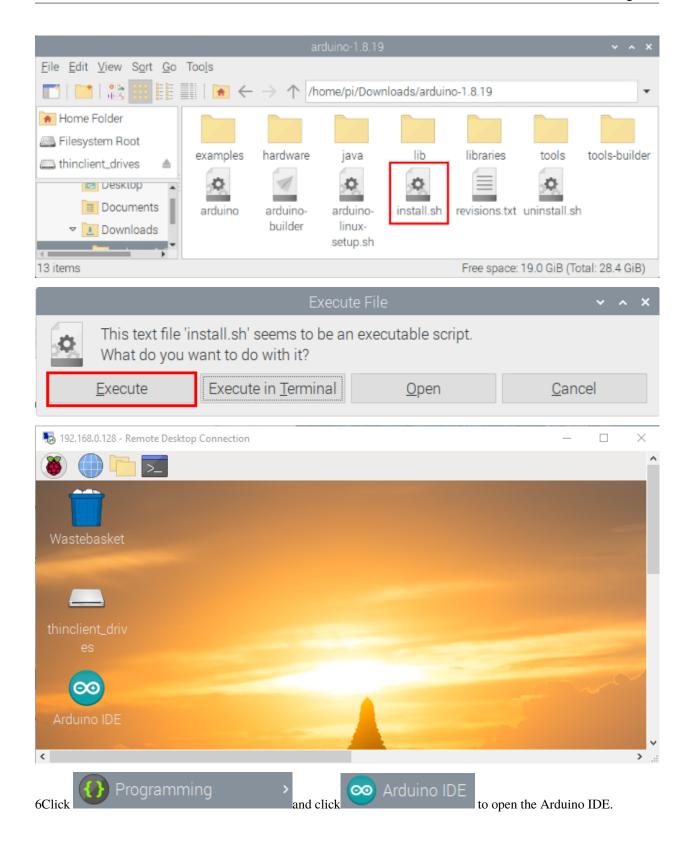
(4) Click , then find the **Downloads** file from the pi folder and click it. In the **Downloads** folder, you can see the package"arduino-1.8.19-linuxarm.tar.xz"that you just downloaded. Then unzip the package"arduino-1.8.19-linuxarm.tar.xz", after a while, the package is unzipped.

No. 192.168.0.167 - Remote Desktop Connection	- 🗆 X
🗕 🛑 🗾	^
Wastebasket	
and the second	
thinclient_driv	
<	· 
No. 192.168.0.167 - Remote Desktop Connection	- 🗆 X
🥘 🛑 🗾 📄 pi	^
pi	~ ^ X
<u>F</u> ile <u>E</u> dit <u>V</u> iew S <u>o</u> rt <u>G</u> o Too <u>l</u> s	
📰   🔛   👬 🧱 🏭   🖿 🔶 🛧 /	′home/pi
A Home Folder	
🖂 Filesystem Root	
☐ thinclient_drives ▲ Bookshelf Desktop	Documents Downloads Music
▶ etc	
▼ home Pictures Public	Templates thinclient_dr Videos
	ives
10 items (21 hidden)	Free space: 19.4 GiB (Total: 28.4 GiB)

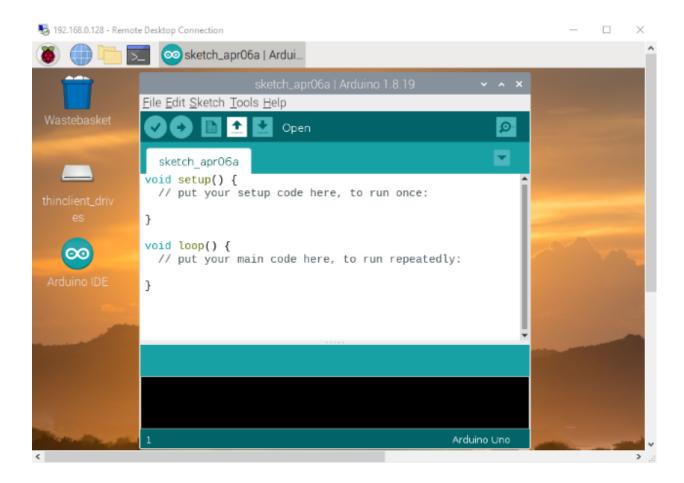


\$

5Click install.sh file and tap itclick "Execute" in the dialog that appears to install the Arduino IDE. Once installed, an Arduino software shortcut is generated in the desktop.



November 2015 - Remote Deskto 💀	p Connection	– 🗆 ×
👅 🛑 🌑		^
Programming	> 🐼 Arduino IDE	
🗒 Education	> 🐔 BlueJ Java IDE	
🥠 Office	> 😳 Geany Programmer's Editor	
Internet	> 🐔 Greenfoot Java IDE	
Sound & Video	> 🔆 Mathematica	
🚏 Graphics	> 🕐 mu	
🕌 Games	> Rode-RED	- And
Accessories	> 🚯 Scratch	
🔁 Help	Scratch 3	
	Sense HAT Emulator	
📰 Preferences	n))) Sonic Pi	
Run	Thonny Python IDE	
Shutdown	😣 Wolfram	v 2



### 8.2.4 (3) Install the ESP32 on Arduino IDE

Note: You need to download Arduino IDE 1.8.5 or advanced version to install the ESP32.



sketch_apr06a   Arduino 1.8.19
<u>File Edit Sketch Tools H</u> elp
sketch_apr06a
<pre>void setup() {     // put your setup code here, to run once:</pre>
}
<pre>void loop() {     // put your main code here, to run repeatedly:</pre>
}
1 Arduino Uno

2) Click **"File**"→\*\*"Preferences"\*\*, copy the website address https://dl.espressif.com/dl/package\_esp32\_index. json in the "Additional Boards Manager URLs:" and click "OK" to save the address.

	sketch_a	apr06a   Arduino 1.8.19 🛛 🗸 🗙 🗙
<u>File</u> Edit <u>S</u> ketch	n <u>T</u> ools <u>H</u> elp	
New	Ctrl+N	n an
Open	Ctrl+O	
Open Recent	>	
Sketchbook	>	e here, to run once:
Examples	>	· ·
Close	Ctrl+W	
Save	Ctrl+S	here, to run repeatedly:
Save As	Ctrl+Shift+S	nere, to run repeatedty.
Page Setup	Ctrl+Shift+P	
Print	Ctrl+P	
Preferences	Ctrl+Comma	
Quit	Ctrl+Q	
1		Arduino Uno

	Preferences 🗸 🗸 👌
Settings Network	
Sketchbook location:	
/home/pi/Arduino	Browse
Editor language:	System Default  v (requires restart of Arduino)
Editor font size:	12
Interface scale:	Automatic 100 💭% (requires restart of Arduino)
Theme:	Default theme  • (requires restart of Arduino)
Show verbose output during:	compilation upload
Compiler warnings:	None 🔻
<ul> <li>Display line numbers</li> <li>Verify code after upload</li> <li>Check for updates on start</li> <li>Use accessibility features</li> </ul>	<ul> <li>Enable Code Folding</li> <li>Use external editor</li> <li>up</li> <li>✓ Save when verifying or uploading</li> </ul>
Additional Boards Manager U	RLs: https://dl.espressif.com/dl/package_esp32_index.json 0
More preferences can be edite	d directly in the file
/home/pi/.arduino15/preferen	ces.txt
(edit only when Arduino is not	running) (2)
	OK Cancel

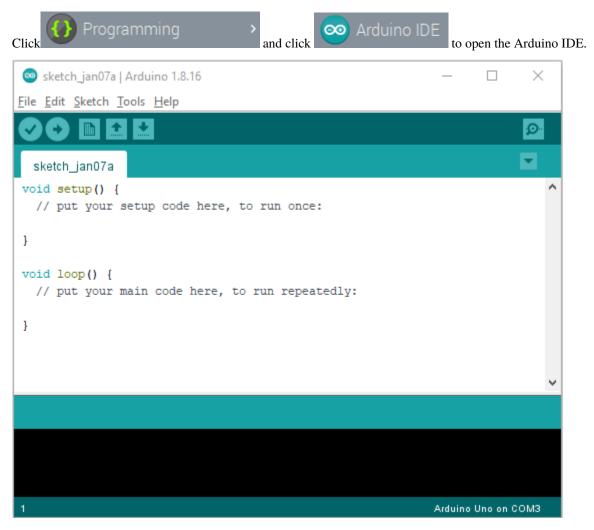
3. Click "Tools"→"Board:", then click "Boards Manager…" to enter "Boards Manager" page . Enter "esp32" as follows and select the latest version to Install. The installation package is not large, click "Install" to start the installation.

	sketch_apr06a   Arduino 1.8.19	~ ^	×
<u>F</u> ile <u>E</u> dit <u>S</u> ketch			
	Auto Format	Ctrl+T	
	Archive Sketch		and the second s
sketch_apr06	Fix Encoding & Reload		Boards Manager
<pre>void setup()    // put your</pre>	Manage Libraries	Ctrl+Shift+I	Δ
	Serial Monitor	Ctrl+Shift+M	Arduino Yún
}	Serial Plotter	Ctrl+Shift+L	<ul> <li>Arduino Uno</li> </ul>
<pre>void loop() {     // put your</pre>	WiFi101 / WiFiNINA Firmware Updater		Arduino Duemilanove or Diecimila Arduino Nano
}	Board: "Arduino Uno"	>	Arduino Mega or Mega 2560
	Port	>	Arduino Mega ADK
	Get Board Info		Arduino Leonardo
	Programmer: "AVRISP mkII"	>	Arduino Leonardo ETH
	Burn Bootloader		Arduino Micro
			Arduino Esplora
1		Arduino Uno	Arduino Mini
-	Boards Mar		· · ·
Type All	esp32	nager	~ ~ ~
esp32	Cabor		-
by Espressif System Boards included in this			106 • Install
			1.0.0 mistai

Boards Manager	~ ^ ×
Type All 🔹 lesp32	
esp32 by Espressif Systems Boards included in this package: ESP32 Dev Module, WEMOS LoLin32, WEMOS D1 MINI ESP32. More Info	alling
	ļ
Downloading boards definitions. Downloaded 23,526kb of 51,126kb.	Cancel

4. After a while, the ESP32 installation package is installed. Then click "Close".

		Boards Manager	*	^	×
Type All	*	esp32			
esp32			_	_	î
Boards included in this	s packa	ian 1.0.6 INSTALLED ge: .oLin32, WEMOS D1 MINI ESP32.			
					l
					ļ
			0	Clos	e.

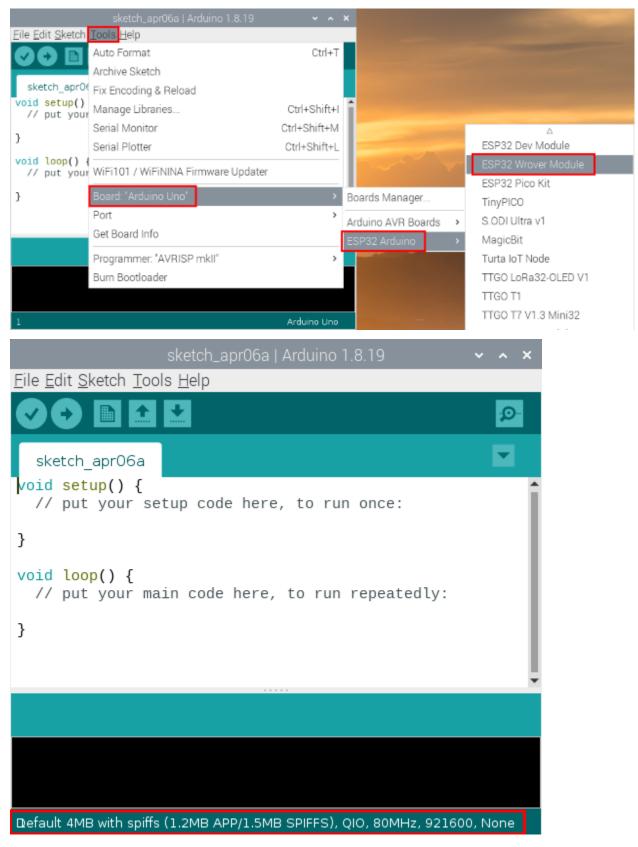


## 8.2.5 3. Arduino IDE Settings and Toolbars:

When downloading the code to the board, you must select the correct name of Arduino board that matches the board connected to the Raspberry Pi, click"**Tools**" $\rightarrow$ "**Board:**", as shown below ;

(Note: We use the ESP32 board in this tutorial; therefore, we select ESP32 Arduino\*\*)\*\*

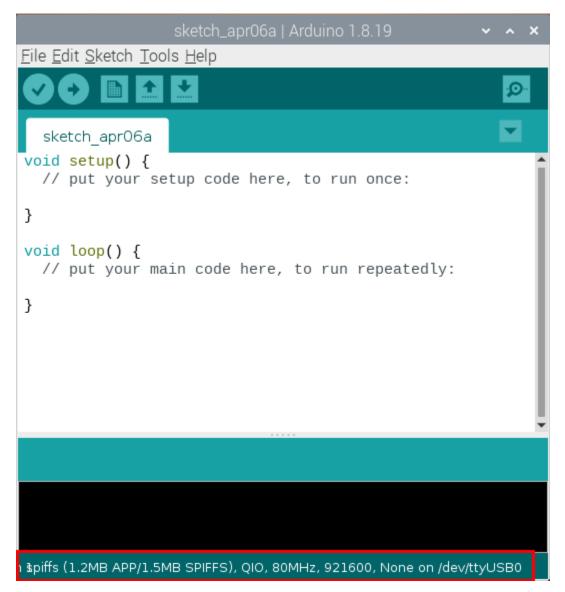
#### **ESP32 Learning Kit**



Then select the correct COM port (After connecting the ESP32 mainboard to the Raspberry Pi via a USB cable, you

### can see the corresponding COM port).

	sketch_apr06a   Arduino 1.8.19 🛛 🗸 🔺 🗙		
<u>F</u> ile <u>E</u> dit <u>S</u> ketch	<u>Tools H</u> elp		
	Auto Format	Ctrl+T	
	Archive Sketch		
sketch_apr06	Fix Encoding & Reload		
<pre>void setup()   // put your</pre>	Manage Libraries	Ctrl+Shift+I	
1	Serial Monitor	Ctrl+Shift+M	
}	Serial Plotter	Ctrl+Shift+L	-
<pre>void loop() {     // put your</pre>	WiFi101 / WiFiNINA Firmware Updater		and the second
}	Board: "ESP32 Wrover Module"	>	
	Upload Speed: '921600'	>	
	Flash Frequency: "80MHz"	>	
	Flash Mode: "QIO"	>	
	Partition Scheme: 'Default 4MB with spiffs (1.2MB APP/1.5MB SPIFF	S)' >	
	Core Debug Level: "None"	>	
	Port	>	Serial ports
	Get Board Info		/dev/ttyAMA0
	Programmer	>	/dev/ttyUSB0
Default 4MB with :	Burn Bootloader		a second and



Before uploading the code to the ESP32 mainboard, we have to demonstrate the function of each symbol.

sketch_apr06a   Arduino 1.8.19 🛛 🗸 🗙 🗙
<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp
skeich_apr06a
<pre>void setup() {     // put your setup code here, to run once:     F</pre>
A B C D E
<pre>void loop() {     // put your main code here, to run repeatedly:</pre>
}
·····
n \$piffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0

- A- Used to verify whether there is any compiling mistakes or not.
- B- Used to upload the sketch to your Arduino board.
- C- Used to create shortcut window of a new sketch.
- D- Used to directly open an example sketch.
- E- Used to save the sketch.
- F- Used to send the serial data received from board to the serial monitor.

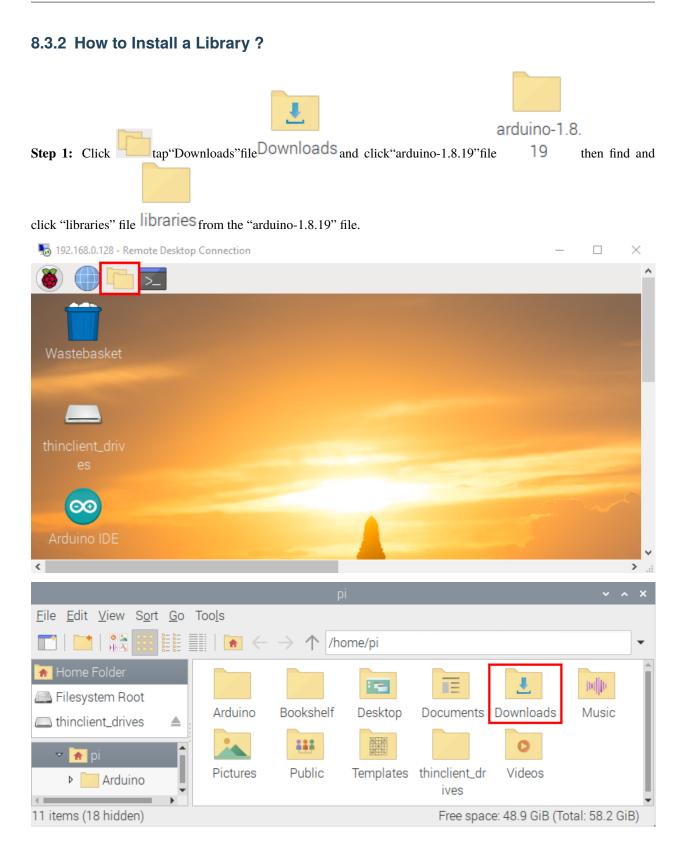
## 8.3 Import the Arduino C library

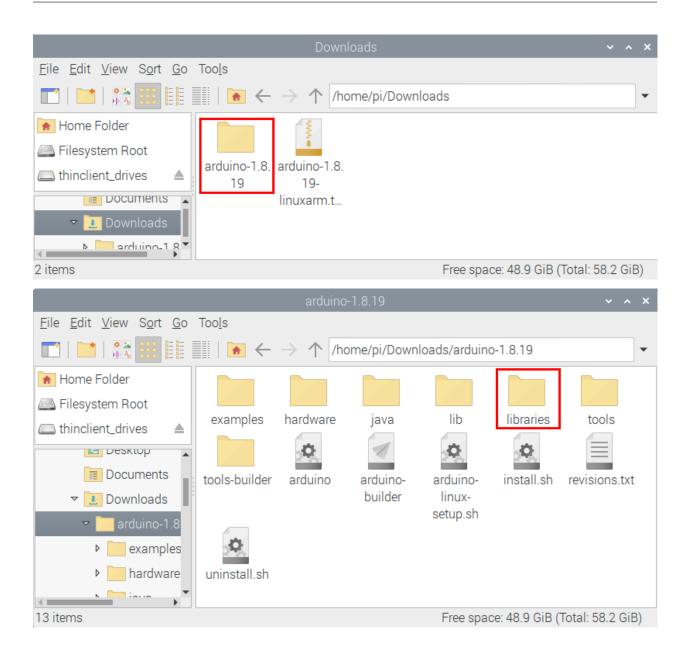
### 8.3.1 What are Libraries ?

Librariesare a collection of code that make it easy for you to connect sensors, displays and modules, etc.

For example, the built-in LiquidCrystal library helps talk to LCD displays. There are hundreds of additional libraries available on the Internet for download.

The built-in libraries and some of these additional libraries are listed in the reference. (https://www.arduino.cc/en/Reference/Libraries)

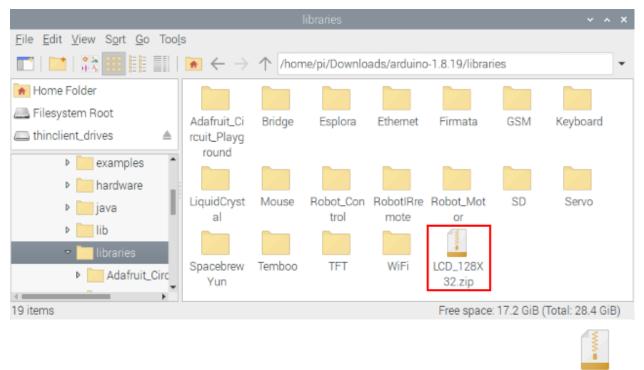




		libr	aries			~ ^ X
<u>F</u> ile <u>E</u> dit <u>V</u> iew S <u>o</u> rt <u>G</u> o	Too <u>l</u> s					
	- 🖌	ightarrow $ ightarrow$ /h	ome/pi/Down	loads/arduin	o-1.8.19/librar	ries 🗸 🗸
Home Folder						
☐ thinclient_drives ▲	Adafruit_Cir cuit_Playgro	Bridge	Esplora	Ethernet	examples	Firmata
▶ <mark>i</mark> java ▶ <mark>i</mark> lib	und					
✓ libraries Adafrui	GSM	Keyboard	LiquidCrysta I	Mouse	Robot_Contr ol	RobotIRrem ote
▶ Bridge						
▶ Esplora	Robot_Moto	SD	Servo	SpacebrewY	Stepper	Temboo
21 items Free space: 48.9 GiB (Total: 58.2 GiB)						

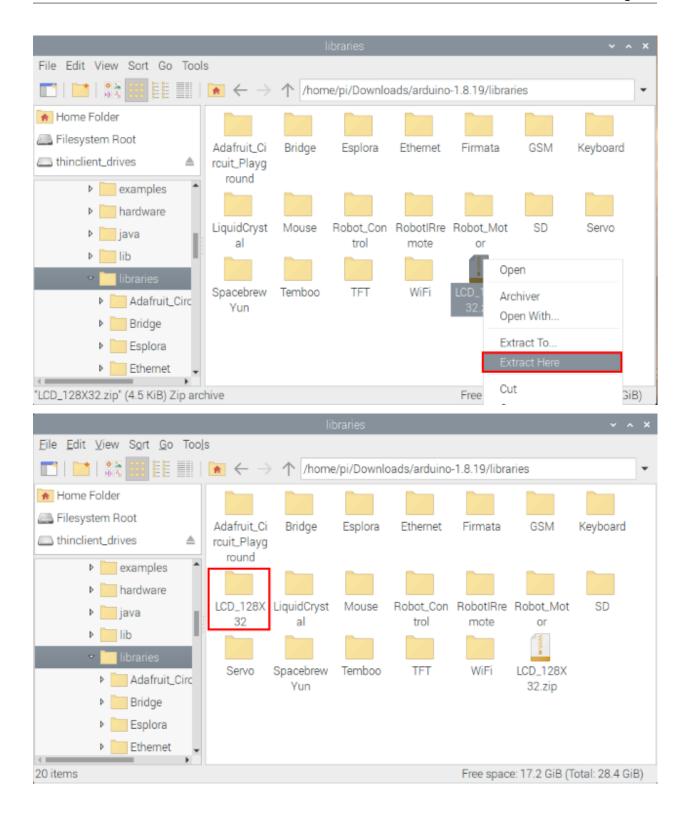
**Step 2 :** Copy and paste the Arduino C library ZIP file (the default is ZIP file) from the provided Arduino Libraries folder into the libraries file opened in the first stepthe route is/home/pi/Downloads/arduino-1.8.19/libraries.

Click on the link to download the library fileArduino C Librarie



**Step 3:** Unzip the Arduino C package in the libraries folderfor exampleright-click"LCD\_128X32.zip"file 32.zip select and tap "Extract Here"to unzip the "LCD\_128X32.zip"file. Similarly, unzip the remaining library files in the same way.). So you can see all the decompressed Arduino C library files.

LCD\_128X



### CHAPTER

NINE

## C LANGUAGE (RASPBERRY PI) TUTORIAL

Development Environment Configuration

RaspberryPi—Arduino Development Environment ConfigurationRaspberryPi—Arduino

## 9.1 Project 01: Hello World

#### 1. Introduction

For ESP32 beginners, we'll start with some simple things. In this project, you just need an ESP32 mainboard, USB cable and Raspberry Pi to complete "Hello World!" Project. It is not only a communication test for ESP32 mainboard and Raspberry Pi, but also a primary project for ESP32.

2. Components

ESP32*1	USB Cable*1

3. Wiring Diagram

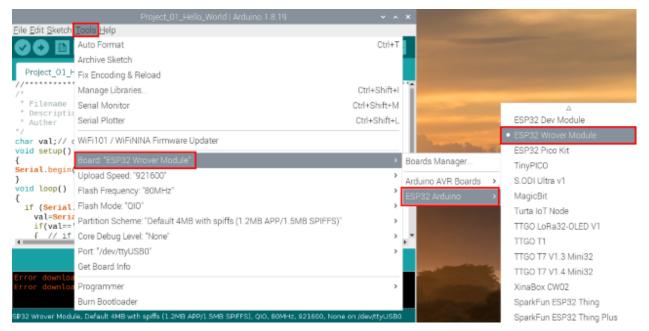
In this project, we will use a USB cable to connect the ESP32 to Raspberry Pi.

(continues on next page)

(continued from previous page)

```
{
    if (Serial.available() > 0) {
    val=Serial.read();// reads symbols assigns to "val"
    if(val=='R')// checks input for the letter "R"
    {        // if so,
        Serial.println("Hello World!");// shows "Hello World !".
    }
    }
    //
}
```

Before uploading the project code to ESP32click "Tools"→"Board" and select"ESP32 Wrover Module".

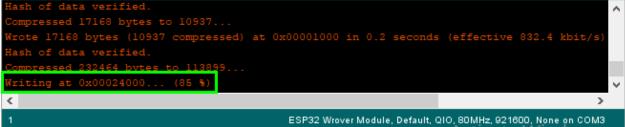


Select the serial port.

	Project_01_Hello_World   Arduino 1.8.19	~ ^	×
<u>F</u> ile <u>E</u> dit <u>S</u> ketch	Tools Help		
	Auto Format	Ctrl+T	
	Archive Sketch		
Project_01_H	Fix Encoding & Reload		
//***********	Manage Libraries	Ctrl+Shift+I	1
* Filename	Serial Monitor	Ctrl+Shift+M	-
* Descriptio * Auther	Serial Plotter	Ctrl+Shift+L	-
	WiFi101 / WiFiNINA Firmware Updater		and the second second
<pre>void setup() {</pre>	Board: "ESP32 Wrover Module"	>	
Serial.begin	Upload Speed: "921600"	>	and the second
void loop()	Flash Frequency: "80MHz"	>	-
{ if (Serial.	Flash Mode: "QIO"	>	
val= <mark>Seria</mark> if(val==	Partition Scheme: "Default AMB with eniffe (1 2MB APD/1 5MB SDIEES)"	>	
	Core Debug Level: "None"	>	•
	Port: "/dev/ttyUSB0"	>	Serial ports
	Get Board Info		/dev/ttyAMA
Error downloa Error downloa	Programmer	>	✔ /dev/ttyUSB
	Burn Bootloader	1	and the second
1932 Wrover Modu	le, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None	on /dev/ttyUSB0	

Click to download the code to ESP32.

Project_01_Hello_World         /*         * Filename : Hello World         * Description : Enter the letter R, and the serial port displays"Hello World".         * Auther : http://www.keyestudio.com         */         * Auther : http://www.keyestudio.com         */         */         char val;// defines variable "val"         void setup()         §         Serial.begin(115200);// sets baudrate to 115200         */         */         */         f(serial.available() > 0) {         */ <th></th> <th>Project_01_Hello_World   Arduino 1.8.19</th> <th>~ ^</th> <th>×</th>		Project_01_Hello_World   Arduino 1.8.19	~ ^	×
<pre>/* * Filename : Hello World * Description : Enter the letter R, and the serial port displays"Hello World". * Auther : http://www.keyestudio.com */ char val;// defines variable "val" void setup() { Serial.begin(115200);// sets baudrate to 115200 } void loop() {     if (Serial.available() &gt; 0) {         val=Serial.read();// read();// read();</pre>	<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp			
<pre>/* * Filename : Hello World * Description : Enter the letter R, and the serial port displays"Hello World". * Auther : http://www.keyestudio.com */ char val;// defines variable "val" void setup() { Serial.begin(115200);// sets baudrate to 115200 } Serial.begin(115200);// sets baudrate to 115200 } void loop() {     (</pre>			Q	
<pre>/* * Filename : Hello World Description : Enter the letter R, and the serial port displays"Hello World". Auther :http//www.keyestudio.com */ char val;// defines variable "val" void setup() { serial.begin(115200);// sets baudrate to 115200 } void loop() {     if (Serial.available() &gt; 0) {         val=Serial.read();// reads symbols assigns to "val"         if(val=='R')/( checks input for the letter "R"         {         // if so,         // Compiling sketch  SEP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0 Note: If uploading the code fails, you can press the Boot button on ESP32 after click, and release the Boot</pre>	Project_01_Hello_World			
<pre>* Description : Enter the letter R, and the serial port displays"Hello World". * Auther :http://www.keyestudio.com */ char val;// defines variable "val" void setup() { Serial.begin(115200);// sets baudrate to 115200 } void loop() {     if (Serial.available() &gt; 0) {         val=Serial.read();// reads symbols assigns to "val"         if(val=='R')// checks input for the letter "R"         { // if so. Compiling sketch  SEP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0 Note: If uploading the code fails, you can press the Boot button on ESP32 after click , and release the Boot</pre>	//*************************************	* * * * * * * * * * * * * * * * * * * *	* * * * * *	K 3
<pre>void setup() { Serial.begin(115200);// sets baudrate to 115200 } void loop() {     if (Serial.available() &gt; 0) {         val=Serial.read();// reads symbols assigns to "val"         if(val=='R')// checks input for the letter "R"         { // if so,         Compiling sketch  SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, BOMHz, 921600, None on /dev/ttyUSB0 Note: If uploading the code fails, you can press the Boot button on ESP32 after click , and release the Boot         RESET</pre>	* Description : Enter the	letter R, and the serial port displays"Hello World"		l
Serial.begin(115200);// sets baudrate to 115200   void loop()   {   if (Serial.available() > 0) {   val-Serial.read();// reads symbols assigns to "val"   if(val=='R')// checks input for the letter "R"   {   {   (    Compliing sketch    SEP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0 Note: If uploading the code fails, you can press the Boot button on ESP32 after click and release the Boot after the percentage of aploading progress appears, as shown below: Uploading	<pre>void setup()</pre>	le "val"		I
<pre>void loop() {     if (Serial.available() &gt; 0) {         val=Serial.read();// reads symbols assigns to "val"         if(val=='R')// checks input for the letter "R"         { // if so,         Compiling sketch  SEP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0 Note: If uploading the code fails, you can press the Boot button on ESP32 after click , and release the Boot         RESET</pre>		ts baudrate to 115200		1
<pre>{     if (Serial.available() &gt; 0) {       val=Serial.aread();// reads symbols assigns to "val"       if(val=='R')// checks input for the letter "R"       { // if so,       Compiling sketch  SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0 Note: If uploading the code fails, you can press the Boot button on ESP32 after click and release the Boot       <b>RESET BOOT</b>       after the percentage of       after the percentage of       aploading progress appears, as shown below:  Uploading</pre>				1
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0 Note: If uploading the code fails, you can press the Boot button on ESP32 after click, and release the Boot RESET BOOT outton after the percentage of uploading progress appears, as shown below:	<pre>val=Serial.read();// r if(val=='R')// checks</pre>	eads symbols assigns to "val"	_	•
Note: If uploading the code fails, you can press the Boot button on ESP32 after click, and release the Boot RESET	Compiling sketch			
Note: If uploading the code fails, you can press the Boot button on ESP32 after click, and release the Boot RESETBOOT button uploading progress appears, as shown below: Uploading				
outton after the percentage of after the percentage of Uploading				
outton after the percentage of after the percentage of Uploading	SP32 Wrover Module, Default 4MB wit	h spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/tt	yUSB0	
Uploading				
	Note: If uploading the code fails, <b>RESET</b> button	you can press the Boot button on ESP32 after click, and releas	e the B	Boot
Hash of data verified.	Note: If uploading the code fails, <b>RESET</b> button	you can press the Boot button on ESP32 after click, and releas	e the B	Boot



The Project code is uploaded successfully

Project_01_Hello_World   Arduino 1.8.19 🗸 🗸 :	ĸ
<u>Eile E</u> dit <u>Sketch T</u> ools <u>H</u> elp	
Project_01_Hello_World	
V/************************************	
<pre>/*  * Filename : Hello World  * Description : Enter the letter R,and the serial port displays"Hello World".  * Auther :http//www.keyestudio.com  */</pre>	l
<pre>char val;// defines variable "val"</pre>	I
<pre>void setup() {</pre>	I
Serial.begin(115200);// sets baudrate to 115200	I
<pre>} void loop()</pre>	I
{	1
<pre>if (Serial.available() &gt; 0) {    val=Serial.read();// reads symbols assigns to "val"    if(val=='R')// checks input for the letter "R"    { // if so,</pre>	•
· · · · · · · · · · · · · · · · · · ·	
Done uploading.	
Leaving Hard resetting via RTS pin	
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0	

5. Project result

After the project code is uploaded successfully, power up with a USB cable and click the icon to enter the serial monitor.

Set baud rate to 115200 and type "R" in the text box. Click "Send", and the serial monitor will display "Hello World!".

(Note: If you enter"R" in the text box and click"Send", the serial monitor does not print"Hello World!", you need to press the RESET button on the ESP32 main board and repeat the above operation.)

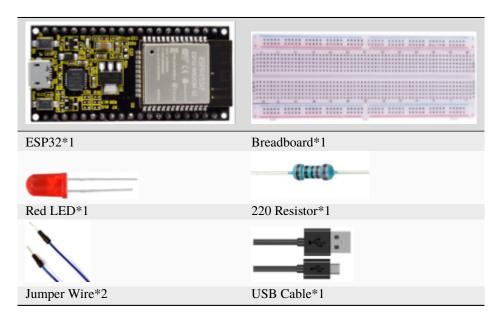
	/dev/ttyUSB0	~ ^ X
R 2		Send
Hello World! Hello World!		3
✓ Autoscroll	Newline • 115200	baud 🔻 Clear output

## 9.2 Project 02: Turn On LED

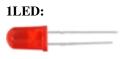
#### 1.Introduction

In this project, we will show you how to light up the LED. We use the ESP32's digital pin to turn on the LED so that the LED is lit up.

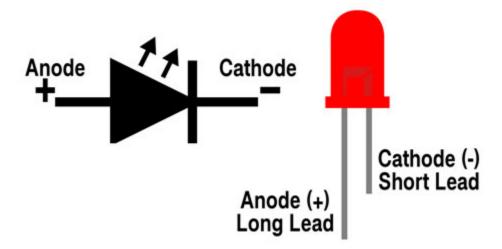
#### 2.Components



3.Component knowledge

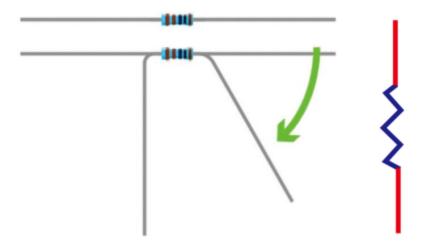


The LED is a semiconductor known as "light-emitting diode", which is an electronic device made from semiconducting materials(silicon, selenium, germanium, etc.). It has an anode and a cathode, the short lead is cathode, which connects to GND; the long lead is anode, which connects to 3.3V or 5V.



#### 2Five-color ring resistor

A resistor is an electronic component in a circuit that restricts or regulates the flow current flow. On the left is the appearance of the resistor and on the right is the symbol for the resistance in the circuit. Its unit is(). 1 m = 1000 k1k = 1000.



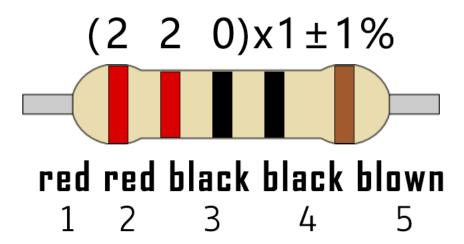
We can use resistors to protect sensitive components, such as LED. The strength of the resistance is marked on the body of the resistor with an electronic color code. Each color code represents a number, and you can refer to it in a resistance card.

- -Color 1 1st Digit.
- -Color 2 2nd Digit.
- -Color 3 3rd Digit.
- -Color 4 Multiplier.
- -Color 5 Tolerance.

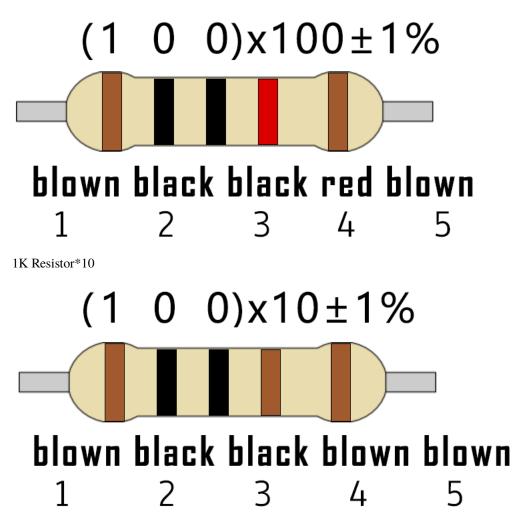
	1st Digit	2nd Digit	3rd Digit	Multiplier	Tolerance
Black		0	0	x1	
Brown	1	1	1	x10	± 1%
Red	2	2	2	x100	± 2%
Orange	3	3	3	x1K	± 3%
Yellow	4	4	4	x10K	± 4%
Green	5	5	5	x100K	± 0.5%
Blue	6	6	6	x1M	±0.25%
Violet	7	7	7	x10M	±0.10%
Grey	8	8	8	x100M	±0.05%
White	9	9	9	x1G	
Gold				÷ 10	± 5%
Silver				÷ 100	± 10%

In this kit, we provide three Five-color ring resistor with different resistance values. Take three Five-color ring resistor as an example.

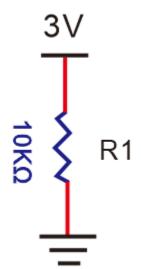
220 Resistor\*10



10K Resistor\*10



In the same voltage, there will be less current and more resistance. The connection between current(I), voltage(V), and resistance (a) can be expressed by the formula: I=U/R. In the figure below, if the voltage is 3V, the current through R1 is: I = U / R = 3 V / 10 K = 0.0003 A = 0.3 mA.

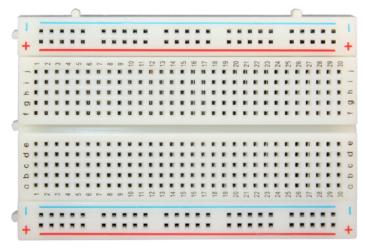


Don't connect a low resistance directly to the two poles of the power supply. as this will cause excessive current to

damage the electronic components. Resistors do not have positive and negative poles.

#### **3Bread board**

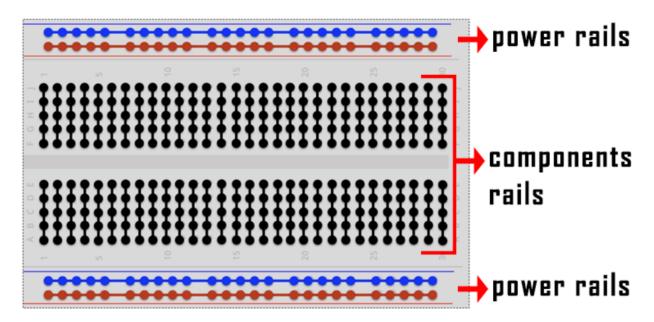
Breadboards are used to build and test circuits quickly before completing any circuit design. There are many holes in the breadboard that can be inserted into circuit components such as integrated circuits and resistors. A typical breadboard is shown below



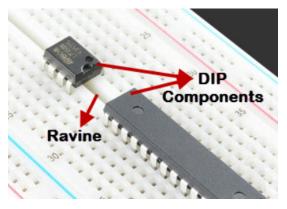
The breadboard has strips of metal, which run underneath the board and connect the holes on the top of the board. The metal strips are laid out as shown below. Note that the top and bottom rows of holes are connected horizontallywhile the remaining holes are connected vertically.

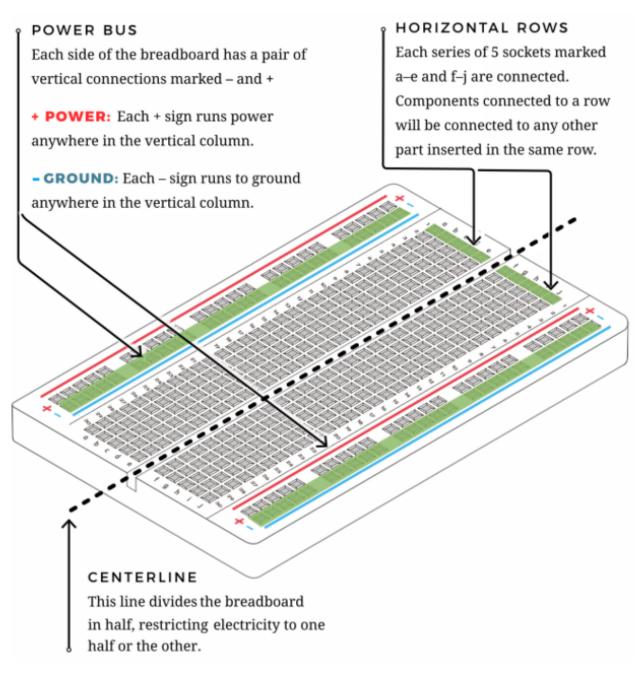


The first two rows (top) and the last two rows (bottom) of the breadboard are used for the positive pole (+) and negative pole (-) of the power supply respectively. The conductive layout of the breadboard is shown in the figure below:



When we connect DIP (Dual In-line Packages) components, such as integrated circuits, microcontrollers, chips and so on, we can see that a groove in the middle isolates the middle part, so the top and bottom of the groove is not connected. DIP components can be connected as shown in the following diagram:





## **4Power Supply**

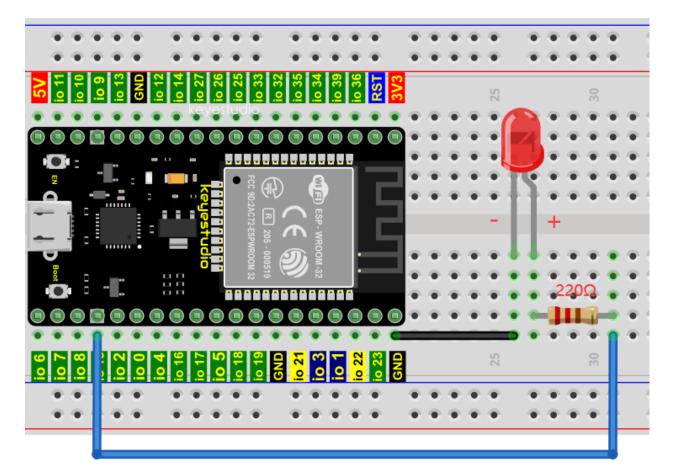
In this project, we connected the ESP32 to the Raspberry Pi by using USB cable.

#### 4.Wiring diagram

First, disconnect all power from the ESP32. Then build the circuit according to the wiring diagram. After the circuit is built and verified correct, connect the ESP32 to the Raspberry Pi by using a USB cable.

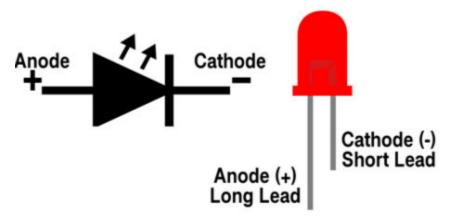
Note: Avoid any possible short circuits (especially connecting 3.3V and GND)!

**WARNING:** A short circuit can cause high current in your circuit, create excessive component heat and cause permanent damage to your hardware!

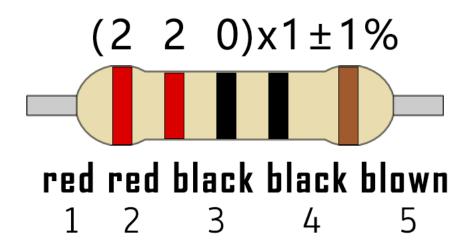


Note:

How to connect a LED



How to identify the 220 Five-color ring resistor



5.Project code

/\* \* Filename : Turn On LED \* Description : Make an led on. \* Auther : http//www.keyestudio.com \*/ #define LED\_BUILTIN 15 // the setup function runs once when you press reset or power the board void setup() { // initialize digital pin LED\_BUILTIN as an output. pinMode(LED\_BUILTIN, OUTPUT); } void loop() { digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level) } 

Before uploading the project code to ESP32click "Tools"→"Board" and select"ESP32 Wrover Module".

	Project_02_Turn_On_LED   Arduino 1.8.19	~ ^	×		
le <u>E</u> dit <u>S</u> ketch	<u>Tools H</u> elp				
20 🖬	Auto Format	Ctrl+T			
	Archive Sketch				
Project_02_T	Fix Encoding & Reload				
	Manage Libraries	Ctrl+Shift+I	î		
	Serial Monitor	Ctrl+Shift+M	1000		Δ
Descriptic Auther	Serial Plotter	Ctrl+Shift+L	-		ESP32 Dev Module
efine LED C	WiFi101 / WiFiNINA Firmware Updater		-		<ul> <li>ESP32 Wrover Module</li> </ul>
				and a	ESP32 Pico Kit
	Board: "ESP32 Wrover Module"	>	Boards Mana	ger	TinyPICO
	Upload Speed: "921600"	,	Arduino AVR 8	Boards >	S.ODI Ultra v1
	Flash Frequency: "80MHz"	,	ESP32 Arduin	o >	MagicBit
id loop() {	Flash Mode: "QIO"	,			Turta IoT Node
digitalWrit	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"	>	' I		TTGO LoRa32-OLED V1
*********	Core Debug Level: "None"	>	17. M		TTGO T1
	Port: "/dev/ttyUSB0"	>			TTGD T7 V1.3 Mini32
	Get Board Info				TTGO T7 V1.4 Mini32
	Programmer	>		1000	XinaBox CWD2
	Bum Bootloader				SparkFun ESP32 Thing
	e. Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None	on Meditud ISBN			SparkFun ESP32 Thing Plus

# Select the serial port.

	Project_02_Turn_On_LED   Arduino 1.8.19		
<u>F</u> ile <u>E</u> dit <u>S</u> ketch	Tools Help		_
	Auto Format	Ctrl+T	
	Archive Sketch		
Project_02_T	Fix Encoding & Reload		
/************ /*	Manage Libraries	Ctrl+Shift+I	î
* Filename	Serial Monitor	Ctrl+Shift+M	
* Descriptio * Auther	Serial Plotter	Ctrl+Shift+L	1
/ /define LED_0	WiFi101 / WiFiNINA Firmware Updater		1
	Board: "ESP32 Wrover Module"	>	1
<pre>/oid setup()    // initial:</pre>	Upload Speed: "921600"	>	1
pinMode(LE	Flash Frequency: "80MHz"	>	1
oid loop()		>	1
digitalWri	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"	>	2
/********	Core Debug Level: "None"	>	
	Port: "/dev/ttyUSB0"	>	
	Get Board Info		
	Programmer	>	~.
	Burn Bootloader		

Click to download the code to ESP32.

Project_02_Turn_On_LED   Arduino 1.8.19 🛛 🗸 🔺 🗙
<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp
Project_02_Turn_On_LED
<pre>//***********************************</pre>
#define LED_BUILTIN 15
<pre>// the setup function runs once when you press reset or power the board void setup() {     // initialize digital pin LED_BUILTIN as an output.     pinMode(LED_BUILTIN, OUTPUT);</pre>
<pre>} void loop() {     digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level) } //**********************************</pre>
4
Compiling sketch
Compiling sketch SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0 Note: If uploading the code fails, you can press the Boot button on ESP32 after click , and release the Boo
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0 Note: If uploading the code fails, you can press the Boot button on ESP32 after click, and release the Boo RESET BOOT
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0 Note: If uploading the code fails, you can press the Boot button on ESP32 after click , and release the Boo
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0 Note: If uploading the code fails, you can press the Boot button on ESP32 after click, and release the Boo RESET BOOT after the percentage o

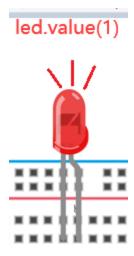


The Project code is uploaded successfully

Proj	ect_02_Turn_On_LED   Arduino 1.8.19	~ ^ X
<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp		
		<mark>,⊘</mark> ,
Project_02_Turn_On_LED		
//************************************	on.	ĺ
<pre>void setup() {    // initialize digital pin LI    pinMode(LED_BUILTIN, OUTPUT) } void loop() {</pre>		level)
4		• •
Done uploading.		
Leaving Hard resetting via RTS pin		Ĵ
▲ SP32 Wrover Module, Default 4 <u>MB with spil</u>	ffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/tty	yUSB0
SES2 wrover Module, Derault 4MB with spir	15 (1.2MB APP)1.3MB SPIPES), QIO, 80MH2, 921000, None on Jaev/(L)	10560

# 6.Project result

After the project code was uploaded successfully, power up with a USB cable and the LED is lit up.

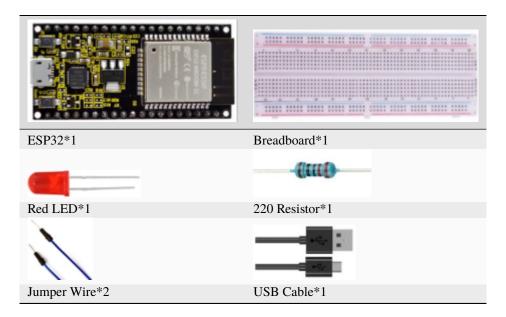


# 9.3 Project 03LED Flashing

#### 1.Introduction

In this project, we will show you the LED flashing effect .We use the ESP32's digital pin to turn on the LED and make it flashing.

#### 2.Components

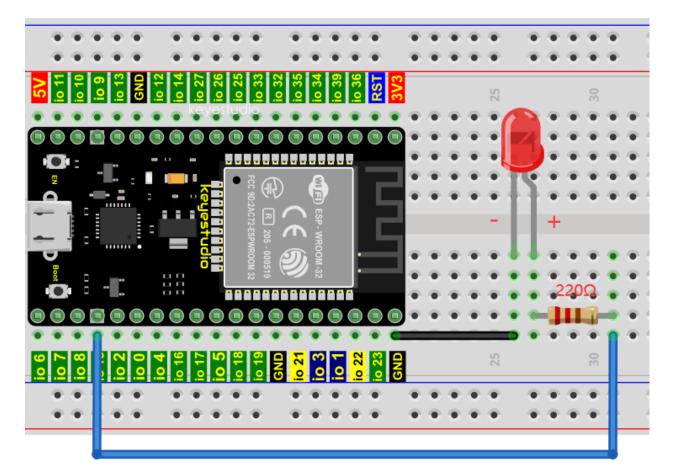


#### 3.Wiring diagram

First, disconnect all power from the ESP32. Then build the circuit according to the wiring diagram. After the circuit is built and verified correct, connect the ESP32 to your computer using a USB cable.

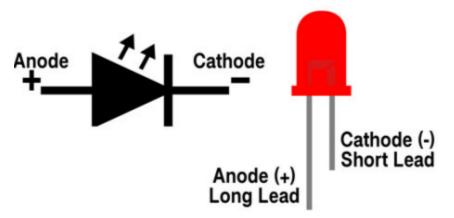
Note: Avoid any possible short circuits (especially connecting 3.3V and GND)!

**WARNING:** A short circuit can cause high current in your circuit, create excessive component heat and cause permanent damage to your hardware!

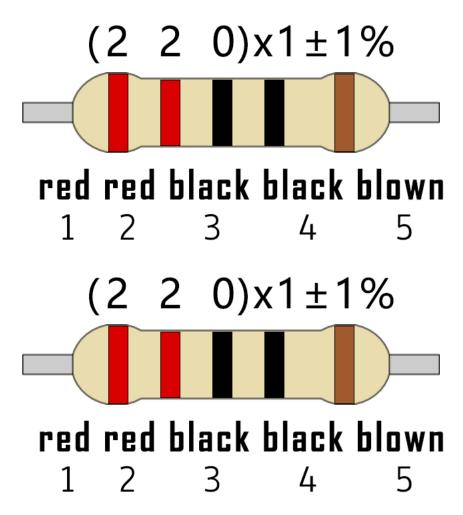


## Note:

How to connect a LED



How to identify the 220 Five-color ring resistor



```
4.Test Code
```

```
/*
* Filename
           : External LED flashing
* Description : Make an led blinking.
* Auther : http//www.keyestudio.com
*/
#define PIN_LED 15 //define the led pin
// the setup function runs once when you press reset or power the board
void setup() {
 // initialize digital pin LED as an output.
 pinMode(PIN_LED, OUTPUT);
}
// the loop function runs over and over again forever
void loop() {
 digitalWrite(PIN_LED, HIGH); // turn the LED on (HIGH is the voltage level)
 delay(500);
                               // wait for 0.5s
 digitalWrite(PIN_LED, LOW);
                            // turn the LED off by making the voltage LOW
 delay(500);
                               // wait for 0.5s
```

(continues on next page)

(continued from previous page)

Before uploading Project Code to ESP32, please check the configuration of Arduino IDE.

Click "Tools" to confirm the board type and port as shown below:

	Project_03_LED_Flashing   Arduino 1.8.19	~ ^	×
<u>F</u> ile <u>E</u> dit <u>S</u> ketch	<u>T</u> ools <u>H</u> elp		
	Auto Format	Ctrl+T	
	Archive Sketch		
Project_03_L	Fix Encoding & Reload		
/*	Manage Libraries	Ctrl+Shift+I	Î
* Filename	Serial Monitor	Ctrl+Shift+M	
* Description	Serial Plotter	Ctrl+Shift+L	
*/ #define PIN_L	WiFi101 / WiFiNINA Firmware Updater		
// the setup	Board: "ESP32 Wrover Module"	>	11
<pre>void setup()     // initial:</pre>	Upload Speed: "921600"	>	Ц
<pre>pinMode(PII }</pre>	Flash Frequency: "80MHz"	>	U
ſ	Flash Mode: "QIO"	>	
<pre>// the loop ' void loop() ·</pre>	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"	>	
digitalWri	Core Debug Level: "None"	>	
	Port: "/dev/ttyUSB0"	>	
	Get Board Info		
	Programmer	>	
	Burn Bootloader		
SP32 Wrover Modu	, lle, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None	on /dev/ttyUSB0	)

Click to download the project code to ESP32.

Project_03_LED_Flashing   Arduino 1.8.19		~	~ >	ĸ
<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp				
			<b>.</b> Q.	
Project_03_LED_Flashing	* * *		•	1
<pre>/*  * Filename : External LED flashing  * Description : Make an led blinking.  * Auther : http//<u>www.keyestudio.com</u> */</pre>				l
<pre>#define PIN_LED 15 //define the led pin // the setup function runs once when you press reset or power the boar void setup() {     // initialize digital pin LED as an output.     pinMode(PIN_LED, OUTPUT); }</pre>	rd			
<pre>// the loop function runs over and over again forever void loop() {     digitalWrite(PIN LED, HIGH); // turn the LED on (HIGH is the volta    </pre>	aqe lev	vel)	•	Ŧ
Compiling sketch				
S <b>P</b> 32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None	e on /de	v/ttyUs	5B0	
Note: If uploading the code fails, you can press the Boot button on ESP32 after click, RESET BOOT button progress appears, as shown below:	and rele			
Uploading				
Hash of data verified. Compressed 17168 bytes to 10937 Wrote 17168 bytes (10937 compressed) at 0x00001000 in 0.2 seconds (effective Hash of data verified. Compressed 232464 bytes to 113899 Writing at 0x00024000 (85 %)	832.4	kbit	/s)	^

The Project code is uploaded successfully

ESP32 Wrover Module, Default, QIO, 80MHz, 921600, None on COM3

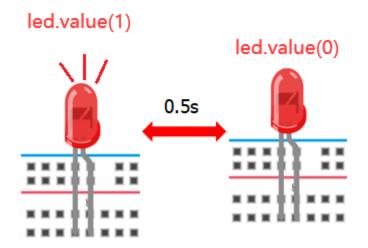
э

<

Project_03_LED_Flashing   Arduino 1.8.19 🗸 🗸	×
<u>File Edit Sketch T</u> ools <u>H</u> elp	
	<b>D</b> -
Project_03_LED_Flashing	
//*************************************	<b>A</b>
<pre>/*  * Filename : External LED flashing  * Description : Make an led blinking.  * Auther : http//www.keyestudio.com  */</pre>	
#define PIN_LED 15 //define the led pin	- 1
<pre>// the setup function runs once when you press reset or power the board void setup() {     // initialize digital pin LED as an output.     pinMode(PIN_LED, OUTPUT); }</pre>	
// the loop function runs over and over again forever	
<pre>void loop() {     digitalWrite(PIN LED, HIGH); // turn the LED on (HIGH is the voltage level) </pre>	•
Done uploading.	
Leaving Hard resetting via RTS pin	Ĵ
•	•
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB	0

## 5.Project result

After the project code was uploaded successfully, power up with a USB cable and the LED start flashing.



# 9.4 Project 04: Breathing Led

#### 1.Introduction

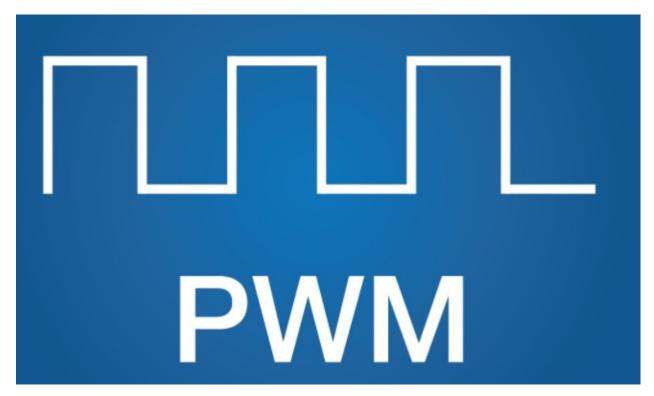
In previous studies, we know that LEDs have on/off state, so how to enter the intermediate state? How to output an intermediate state to make the LED half bright? That's what we're going to learn.

Breathing light, that is, LED is turned from off to on gradually, and gradually from on to off, just like "breathing". So, how to control the brightness of a LED? We will use ESP32's PWM to achieve this target.

2.Components

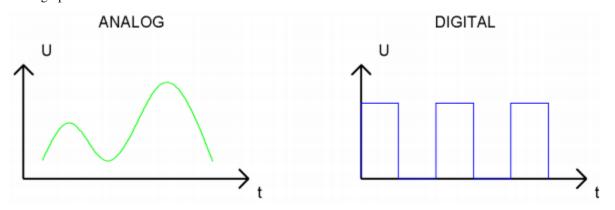
ESP32*1	Breadboard*1
	-(1111)
Red LED*1	220 Resistor*1
Jumper Wire*2	USB Cable*1

2.Component knowledge



### Analog & Digital

An Analog Signal is a continuous signal in both time and value. On the contrary, a Digital Signal or discrete time signal is a time series consisting of a sequence of quantities. Most signals in life are analog signals. A familiar example of an Analog Signal would be how the temperature throughout the day is continuously changing and could not suddenly change instantaneously from  $0^{\circ}$ C to  $10^{\circ}$ C. However, Digital Signals can instantaneously change in value. This change is expressed in numbers as 1 and 0 (the basis of binary code). Their differences can more easily be seen when compared when graphed as below.



In practical application, we often use binary as the digital signal, that is a series of 0's and 1's. Since a binary signal only has two values (0 or 1), it has great stability and reliability. Lastly, both analog and digital signals can be converted into the other.

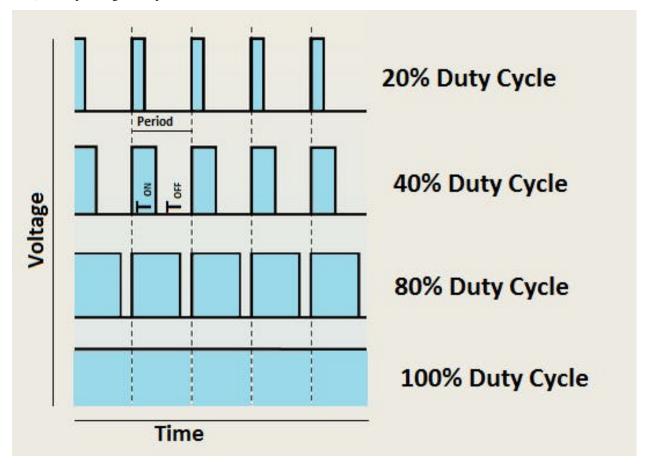
### PWM

PWM, Pulse-Width Modulation, is a very effective method for using digital signals to control analog circuits. Common processors cannot directly output analog signals. PWM technology makes it very convenient to achieve this conversion (translation of digital to analog signals).

PWM technology uses digital pins to send certain frequencies of square waves, that is, the output of high levels and

low levels, which alternately last for a while. The total time for each set of high levels and low levels is generally fixed, which is called the period (Note: the reciprocal of the period is frequency). The time of high level outputs are generally called "pulse width", and the duty cycle is the percentage of the ratio of pulse duration, or pulse width (PW) to the total period(T) of the waveform.

The longer the output of high levels last, the longer the duty cycle and the higher the corresponding voltage in the analog signal will be. The following figures show how the analog signal voltages vary between 0V-3V3 (high level is 3V3) corresponding to the pulse width 0%-100%:

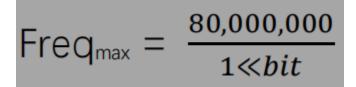


The longer the PWM duty cycle is, the higher the output power will be. Now that we understand this relationship, we can use PWM to control the brightness of an LED or the speed of DC motor and so on. It is evident from the above that PWM is not real analog, and the effective value of the voltage is equivalent to the corresponding analog. so, we can control the output power of the LED and other output modules to achieve different effects.

#### ESP32 and PWM:

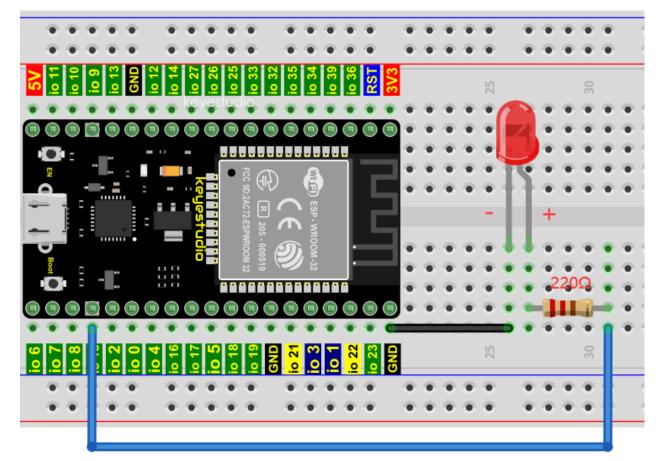
On ESP32, the LEDC(PWM) controller has 16 separate channels, each of which can independently control frequency, duty cycle, and even accuracy. Unlike traditional PWM pins, the PWM output pins of ESP32 are configurable, with one or more PWM output pins per channel. The relationship between the maximum

frequency and bit precision is shown in the following formula, where the maximum value of bit is 31.



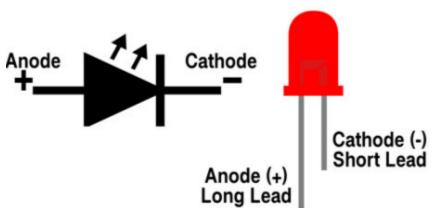
For example, generate a PWM with an 8-bit precision (28=256. Values range from 0 to 255) with a maximum frequency of 80,000,000/255 = 312,500 Hz.

3.Wiring diagram

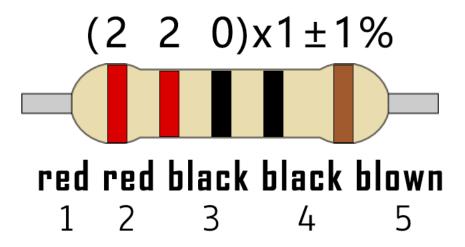


Note:

How to connect a LED



How to identify the 220 Five-color ring resistor



4.Project code

The design of this project makes the GP15 output PWM, and the pulse width gradually increases from 0% to 100%, and then gradually decreases from 100% to 0%.

```
/*
* Filename : Breathing Led
* Description : Make led light fade in and out, just like breathing.
* Auther : http//www.keyestudio.com
*/
#define PIN_LED 15 //define the led pin
#define CHN
             0 //define the pwm channel
#define FRQ 1000 //define the pwm frequency
#define PWM_BIT 8
                  //define the pwm precision
void setup() {
 ledcSetup(CHN, FRQ, PWM_BIT); //setup pwm channel
 ledcAttachPin(PIN_LED, CHN); //attach the led pin to pwm channel
}
void loop() {
 for (int i = 0; i < 255; i++) { //make light fade in</pre>
   ledcWrite(CHN, i);
   delay(10);
 }
 for (int i = 255; i > -1; i--) { //make light fade out
   ledcWrite(CHN, i);
   delay(10);
 }
}
```

Before uploading Project Code to ESP32, please check the configuration of Arduino IDE.

Click "**Tools**" to confirm the board type and port as shown below:

	Project_04_Breathing_Led   Arduino 1.8.19	~ ^ X
<u>F</u> ile <u>E</u> dit <u>S</u> ketch	<u>Tools</u> <u>H</u> elp	
	Auto Format	Ctrl+T .
	Archive Sketch	
Project_04_E	Fix Encoding & Reload	
/*	Manage Libraries	Ctrl+Shift+I
* Filename	Serial Monitor	Ctrl+Shift+M
* Description * Auther	Serial Plotter	Ctrl+Shift+L
*/ #define PIN_L	WiFi101 / WiFiNINA Firmware Updater	
#define CHN #define FRQ	Board: "ESP32 Wrover Module"	>
<pre>#define PWM_E void setup()</pre>	Upload Speed: "921600"	>
ledcSetup((		>
}	Flash Mode: "QIO"	>
<pre>void loop()</pre>	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"	>
for (int i	Core Debug Level: "None"	>
	Port: "/dev/ttyUSB0"	>
	Get Board Info	
	Programmer	>
	Burn Bootloader	
SP32 Wrover Modu	le, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None	on /dev/ttyUSB0

Click to download the project code to ESP32.

Pro	ject_04_Breathing_Led   Arduino 1.8.19	~ ^ X
<u>File Edit Sketch Tools H</u> elp		
		<mark>.ې</mark>
Project 04 Breathing Led		
<pre>//***********************************</pre>	ght fade in and out, just like breathin eyestudio.com ine the led pin ne the pwm channel fine the pwm frequency fine the pwm precision T); //setup pwm channel ); //attach the led pin to pwm channel	
<pre>for (int i = 0; i &lt; 255; i </pre>	++) { //make light fade in	*
Compiling sketch		
SP32 Wrover Module, Default 4MB with s	piffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, I	None on /dev/ttyUSB0
Note: If uploading the code fails, you <b>RESET</b> button uploading progress appears, as shown b	elow:	
Uploading		



The Project code is uploaded successfully

Project_04_Breathing_Led   Arduino 1.8.19	~ ^ X
<u>File E</u> dit <u>Sketch</u> <u>T</u> ools <u>H</u> elp	
	<mark>,⊘</mark> ,
Project_04_Breathing_Led	
//*************************************	<b></b>
/*     * Filename : Breathing Led     * Description : Make led light fade in and out, just like breathing.     * Auther : http// <u>www.keyestudio.com</u> */	
<pre>#define PIN_LED 15 //define the led pin #define CHN 0 //define the pwm channel #define FRQ 1000 //define the pwm frequency #define PWM_BIT 8 //define the pwm precision void setup() {</pre>	
<pre>ledcSetup(CHN, FRQ, PWM_BIT); //setup pwm channel ledcAttachPin(PIN_LED, CHN); //attach the led pin to pwm channel }</pre>	
<pre>void loop() {    for (int i = 0; i &lt; 255; i++) { //make light fade in</pre>	,*
Done uploading.	
Leaving Hard resetting via RTS pin	Î
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev	/ttyUSB0

#### 5.Project result

After the project code was uploaded successfully, power up with a USB cable and the LED is turned from ON to OFF and then back from OFF to ON gradually like breathing.



# 9.5 Project 05Traffic Lights

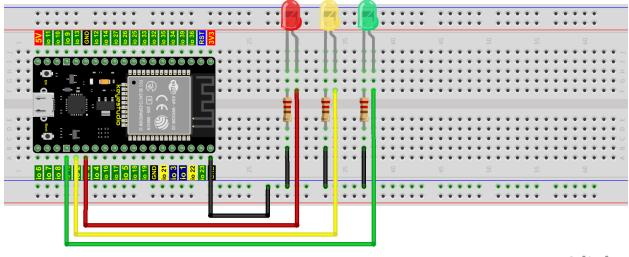
#### 1.Introduction

Traffic lights are closely related to people's daily lives, which generally show red, yellow, and green. Everyone should obey the traffic rules, which can avoid many traffic accidents. In this project, we will use ESP32 and some LEDs (red, green and yellow) to simulate the traffic lights.

#### 2.Components

ESP32*1	Bread board*1	
<b></b>		
Red LED*1	Yellow LED*1	Green LED*1
	-()111))()111))	$\mathbf{X}$
USB Cable*1	220 Resistor*3	Jumper Wires

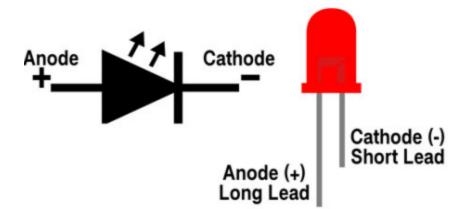
#### 3.Wiring diagram



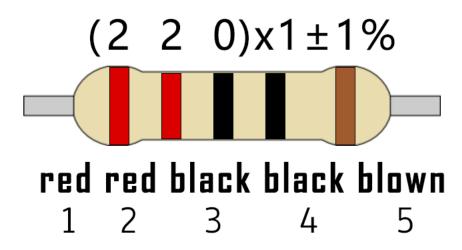
fritzing

#### Note:

How to connect a LED



How to identify the 220 Five-color ring resistor

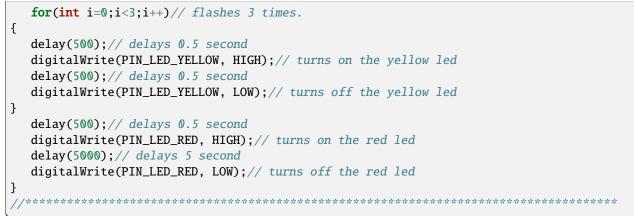


4.Test Code

```
/*
* Filename : Traffic Lights
* Description : Simulated traffic lights.
* Auther : http//www.keyestudio.com
*/
#define PIN_LED_RED 0 //define the red led pin
#define PIN_LED_YELLOW 2 //define the yellow led pin
#define PIN_LED_GREEN 15 //define the green led pin
void setup() {
 pinMode(PIN_LED_RED, OUTPUT);
 pinMode(PIN_LED_YELLOW, OUTPUT);
 pinMode(PIN_LED_GREEN, OUTPUT);
}
void loop() {
  digitalWrite(PIN_LED_GREEN, HIGH);// turns on the green led
  delay(5000);// delays 5 seconds
  digitalWrite(PIN_LED_GREEN, LOW); // turns off the green led
```

(continues on next page)

(continued from previous page)



Before uploading Project Code to ESP32, please check the configuration of Arduino IDE.

Click "Tools" to confirm the board type and port as shown below:

	Project_05_Traffic_Lights   Arduino 1.8.19	~ ^ >
<u>F</u> ile <u>E</u> dit <u>S</u> ketch	<u>Tools</u> <u>H</u> elp	
	Auto Format	Ctrl+T .
	Archive Sketch	
Project_05_1	Fix Encoding & Reload	
//*********	Manage Libraries	Ctrl+Shift+I
* Filename	Serial Monitor	Ctrl+Shift+M
* Description	Serial Plotter	Ctrl+Shift+L
_	WiFi101 / WiFiNINA Firmware Updater	
#define PIN_I #define PIN_I	Board: "ESP32 Wrover Module"	>
<pre>void setup()</pre>	Upload Speed: "921600"	>
pinMode(PI	Flash Frequency: "80MHz"	>
pinMode(PI pinMode(PI	Flash Mode: "QIO"	>
}	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"	>
void loop()	Core Debug Level: "None"	>
	Port: "/dev/ttyUSB0"	>
	Get Board Info	
	Programmer	>
	Burn Bootloader	
SP32 Wrover Modu	le, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None	on /dev/ttyUSB0

Click to download the project code to ESP32.

Project_05_Traffic_Lights   Arduino 1.8.19	~ ^ X
<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp	
	<b>₽</b>
Project_05_Traffic_Lights	
<pre>//***********************************</pre>	ĺ
<pre>void setup() {     pinMode(PIN_LED_RED, OUTPUT);     pinMode(PIN_LED_YELLOW, OUTPUT);     pinMode(PIN_LED_GREEN, OUTPUT); }</pre>	
<pre>void loop() {</pre>	-
Compiling sketch	
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/tty	USB0
Note: If uploading the code fails, you can press the Boot button on ESP32 after click, and release	the Boot
RESET BOOT	
button after the perception af	centage of
Uploading	
Hash of data verified. Compressed 17168 bytes to 10937 Wrote 17168 bytes (10937 compressed) at 0x00001000 in 0.2 seconds (effective 832.4 kb Hash of data verified.	^ it/s)

Compressed 232464 bytes to 113899... Writing at 0x00024000... (85 %)

ESP32 Wrover Module, Default, QIO, 80MHz, 921600, None on COM3

The Project code is uploaded successfully

v

>

Project_05_Traffic_Lights   Arduino 1.8.19	~ ^ X
<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp	
	<mark>,©</mark> ,
Project_05_Traffic_Lights	
//*************************************	<b>^</b>
/*     * Filename : Traffic Lights     * Description : Simulated traffic lights.     * Auther : http// <u>www.keyestudio.com</u> */	
<pre>#define PIN_LED_RED 0 //define the red led pin #define PIN_LED_YELLOW 2 //define the yellow led pin #define PIN_LED_GREEN 15 //define the green led pin</pre>	- 1
<pre>void setup() {     pinMode(PIN_LED_RED, OUTPUT);     pinMode(PIN_LED_YELLOW, OUTPUT);     pinMode(PIN_LED_GREEN, OUTPUT); }</pre>	
<pre>void loop() {</pre>	*
Done uploading.	
Leaving Hard resetting via RTS pin	Ĵ
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev	//ttyUSB0

### 5.Project result

After the project code was uploaded successfully, power up with a USB cable and you'll see are below:

First, the green light will be on for five seconds and then off;

Next, the yellow light blinks three times and then goes off;

Then, the red light goes on for five seconds and then goes off;

Repeat steps 1 to 3 above.

# 9.6 Project 06: RGB LED

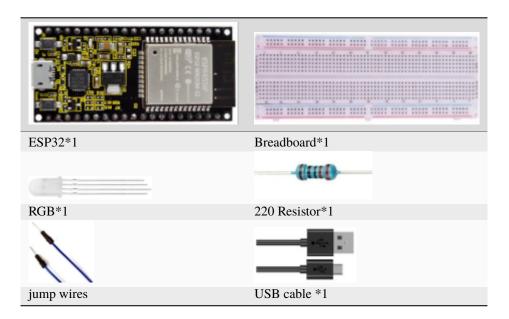
#### 1.Introduction



RGB is composed of three colors (red, green and blue), which can emit different colors of light by mixing these three basic colors.

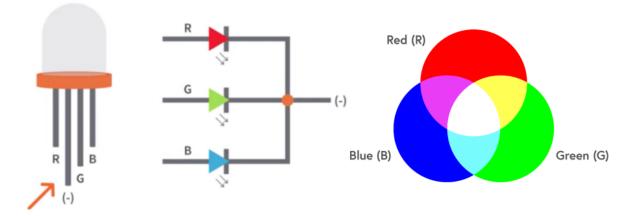
In this project, we will introduce the RGB and show you how to use ESP32 to control the RGB to emit different color light .RGB is pretty basic, but it's also a great way to learn the fundamentals of electronics and coding.

2.Components



#### 3.Component knowledge

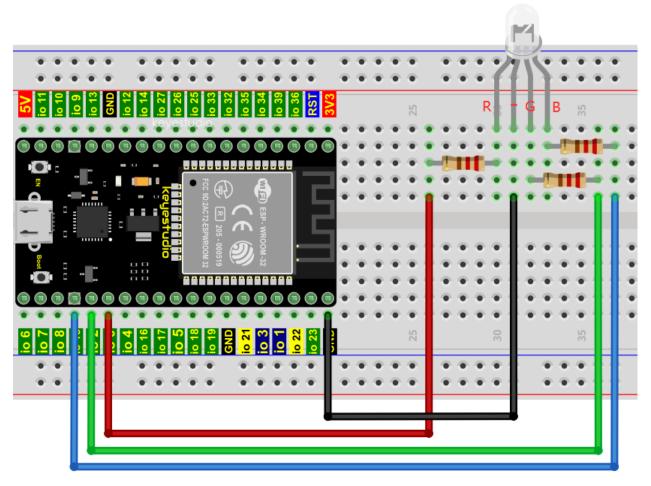
Most monitors adopt the RGB color standard, and all colors on a computer screen are a mixture of red, green and blue in varying proportions.



This RGB LED has 4 pins, each color (red, green, blue) and a common cathode, To change its brightness, we can use the PWM of the ESP32 pins, which can give different duty cycle signals to the RGB to produce different colors of light.

If we use three 10-bit PWM to control the RGB, in theory, we can create 2 10\*210\*210=1,073,741,824(1 billion) colors through different combinations.

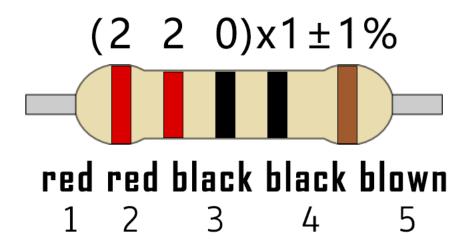
4.Wiring diagram



Notice: The longest pin (common cathode) of the RGB LED is connected to GND.



How to identify the 220 Five-color ring resistor



5.Project code

```
/*
* Filename : RGB LED
* Description : Use RGBLED to show random color.
* Auther : http//www.keyestudio.com
*/
int ledPins[] = {0, 2, 15}; //define red, green, blue led pins
const byte chns[] = {0, 1, 2}; //define the pwm channels
int red, green, blue;
void setup() {
 for (int i = 0; i < 3; i++) { //setup the pwm channels,1KHz,8bit
   ledcSetup(chns[i], 1000, 8);
   ledcAttachPin(ledPins[i], chns[i]);
 }
}
void loop() {
 red = random(0, 256);
 green = random(0, 256);
 blue = random(\emptyset, 256);
 setColor(red, green, blue);
 delay(200);
}
void setColor(byte r, byte g, byte b) {
 ledcWrite(chns[0], 255 - r); //Common anode LED, low level to turn on the led.
 ledcWrite(chns[1], 255 - g);
 ledcWrite(chns[2], 255 - b);
}
```

#### 6.Project result

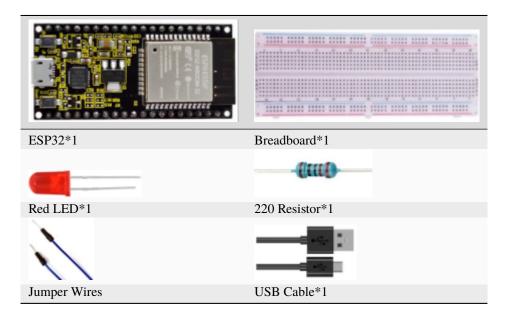
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the RGB LED starts to display random colors.

# 9.7 Project 07: Flowing Water Light

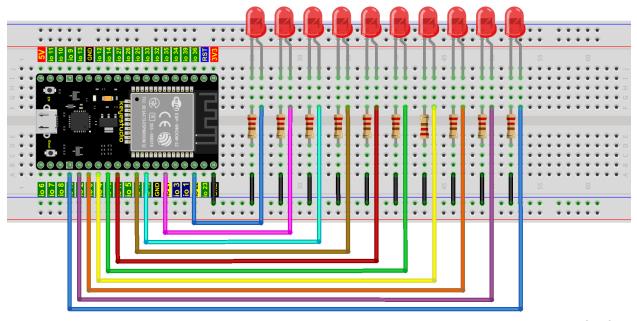
#### 1.Introduction

In our daily life, we can see many billboards composed of different colors of LED. They constantly change the light (like water) to attract customers' attention. In this project, we will use ESP32 to control 10 leds to achieve the effect of flowing water.

2.Components

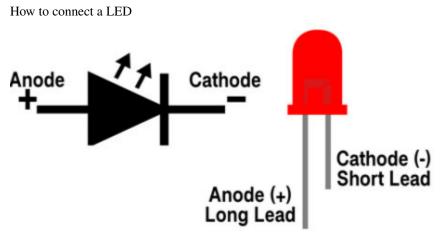


### 3. Wiring diagram:

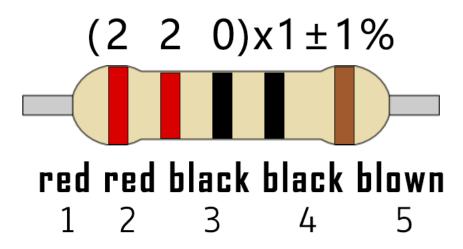


fritzing

Note:



How to identify the 220 Five-color ring resistor



#### 4. Test Code

This project is designed to make a flowing water lamp. Which are these actions: First turn LED #1 ON, then turn it OFF. Then turn LED #2 ON, and then turn it OFF... and repeat the same to all 10 LEDs until the last LED is turns OFF. This process is repeated to achieve the "movements" of flowing water.

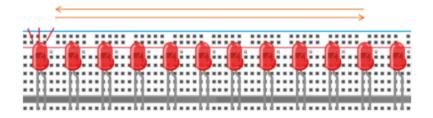
```
/*
* Filename
            : Flowing Water Light
* Description : Using ten leds to demonstrate flowing lamp.
         : http//www.keyestudio.com
* Auther
*/
byte ledPins[] = {22, 21, 19, 18, 17, 16, 4, 0, 2, 15};
int ledCounts;
void setup() {
 ledCounts = sizeof(ledPins);
 for (int i = 0; i < ledCounts; i++) {
   pinMode(ledPins[i], OUTPUT);
 }
}
```

(continues on next page)

```
void loop() {
  for (int i = 0; i < ledCounts; i++) {
    digitalWrite(ledPins[i], HIGH);
    delay(100);
    digitalWrite(ledPins[i], LOW);
  }
  for (int i = ledCounts - 1; i > -1; i--) {
    digitalWrite(ledPins[i], HIGH);
    delay(100);
    digitalWrite(ledPins[i], LOW);
  }
}
```

4.Project result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that 10 LEDs will light up from left to right and then back from right to left.



# 9.8 Project 081-Digit Digital Tube

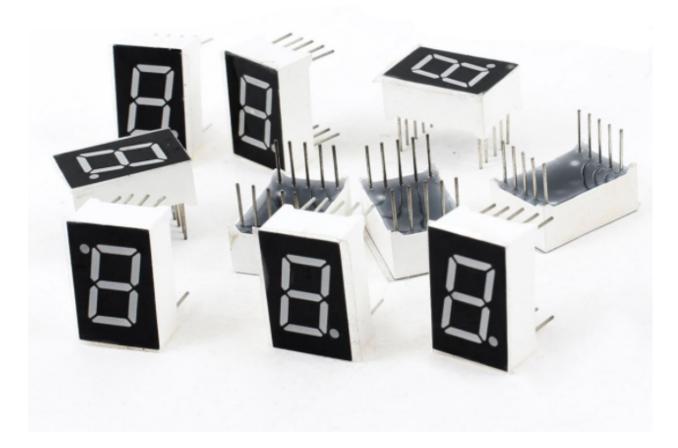
1.Introduction

The 1-Digit 7-Segment display is an device that displays decimal numbers, which is widely used in digital clocks, electronic meters, basic calculators and other electronic devices that display digital information. In this project, we will use ESP32 to control 1-Digit 7-segment display to display numbers.

2.Components

ESP32*1	Breadboard*1
	-()610()
1-Digit 7-Segment Display*1	220 Resistor*8
Jumper Wires	USB Cable*1

3.Component Knowledge

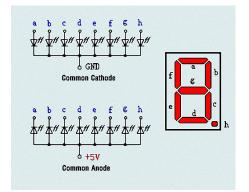


**1-Digit 7-Segment Display:** It is a semiconductor light emitting device, and its basic unit is a light-emitting diode (LED). The digital tube display can be divided into 7-segment display and 8-segment display according to the number of segments.

The 8-segment display has one more LED unit than the 7-segment display(used for decimal point display). Each segment of the 7-segment display is a separate LED. According to the connection mode of the LED unit, the digital tube can be divided into a common anode digital tube and a common cathode digital tube.

In the common cathode 7-segment display, all the cathodes (or negative pole) of the segmented LEDs are connected together, so you should connect the common cathode to GND. If you are about to light up a segmented LED, you can set its associated pin to "HIGH".

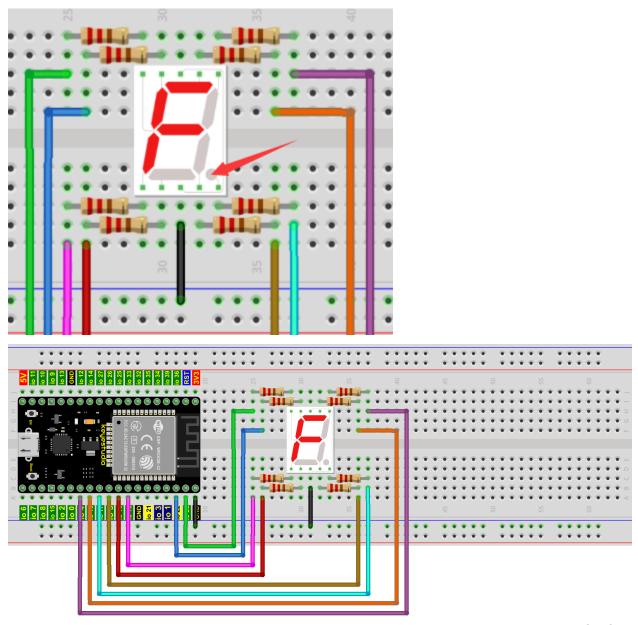
In the common anode 7-segment display, the LED anodes (positive pole) of all segments are connected together, so you should connect the common anode to "+5V". If you are about to light up a segmented LED, you can set its associated pin to "LOW".



Each part of the digital tube is composed of an LED. So when you use it, you also need to use a current limiting resistor. Otherwise, the LED will be damaged. In this experiment, we will use an ordinary common cathode one-digit digital tube. As we mentioned above, you should connect the common cathode to GND. If you are about to light up a segmented LED, you can set its associated pin to "HIGH".

## 4.Wiring Diagram

Note: The direction of the 7-segment display inserted into the breadboard is consistent with the wiring diagram, with one more point in the lower right corner.



fritzing

5.Test Code

The digital display is divided into 7 segments, and the decimal point display is divided into 1 segment. When certain numbers are displayed, the corresponding segment will be lit. For example, when the number 1 is displayed, segments b and c will be turned on.

```
int a=16; // digital PIN 16 for segment a
int b=4; // digital PIN 4 for segment b
int c=5; // digital PIN 5 for segment c
int d=18; // digital PIN 18 for segment d
int e=19; // digital PIN 19 for segment e
int f=22; // digital PIN 22 for segment f
int g=23; // digital PIN 23 for segment g
int dp=17; // digital PIN 17 for segment dp
void digital_0(void) // displays number 0
{
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,HIGH);
digitalWrite(f,HIGH);
digitalWrite(g,LOW);
digitalWrite(dp,LOW);
}
void digital_1(void) // displays number 1
{
digitalWrite(a,LOW);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,LOW);
digitalWrite(e,LOW);
digitalWrite(f,LOW);
digitalWrite(g,LOW);
digitalWrite(dp,LOW);
}
void digital_2(void) // displays number 2
{
digitalWrite(a.HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,LOW);
digitalWrite(d,HIGH);
digitalWrite(e,HIGH);
digitalWrite(f,LOW);
digitalWrite(g,HIGH);
digitalWrite(dp,LOW);
}
void digital_3(void) // displays number 3
ł
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(f,LOW);
digitalWrite(e,LOW);
digitalWrite(dp,LOW);
digitalWrite(g,HIGH);
}
```

```
void digital_4(void) // displays number 4
{
digitalWrite(a,LOW);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,LOW);
digitalWrite(e,LOW);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);
digitalWrite(dp,LOW);
}
void digital_5(void) // displays number 5
{
digitalWrite(a,HIGH);
digitalWrite(b,LOW);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,LOW);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);
digitalWrite(dp,LOW);
}
void digital_6(void) // displays number 6
{
digitalWrite(a,HIGH);
digitalWrite(b,LOW);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,HIGH);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);
digitalWrite(dp,LOW);
}
void digital_7(void) // displays number 7
{
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,LOW);
digitalWrite(e,LOW);
digitalWrite(f,LOW);
digitalWrite(g,LOW);
digitalWrite(dp,LOW);
}
void digital_8(void) // displays number 8
{
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,HIGH);
digitalWrite(f,HIGH);
```

```
digitalWrite(g,HIGH);
digitalWrite(dp,LOW);
}
void digital_9(void) // displays number 9
{
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,LOW);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);
digitalWrite(dp,LOW);
}
void setup()
{
 // initialize digital pin LED as an output.
 pinMode(a, OUTPUT);
 pinMode(b, OUTPUT);
 pinMode(c, OUTPUT);
 pinMode(d, OUTPUT);
 pinMode(e, OUTPUT);
 pinMode(f, OUTPUT);
 pinMode(g, OUTPUT);
 pinMode(dp, OUTPUT);
}
void loop()
{
while(1)
{
digital_9();// displays number 9
delay(1000); // waits a sencond
digital_8();// displays number 8
delay(1000); // waits a sencond
digital_7();// displays number 7
delay(1000); // waits a sencond
digital_6();// displays number 6
delay(1000); // waits a sencond
digital_5();// displays number 5
delay(1000); // waits a sencond
digital_4();// displays number 4
delay(1000); // waits a sencond
digital_3();// displays number 3
delay(1000); // waits a sencond
digital_2();// displays number 2
delay(1000); // waits a sencond
digital_1();// displays number 1
delay(1000);// waits a sencond
digital_0();// displays number 0
delay(1000); // waits a sencond
}}
  ******
```

6.Test Result

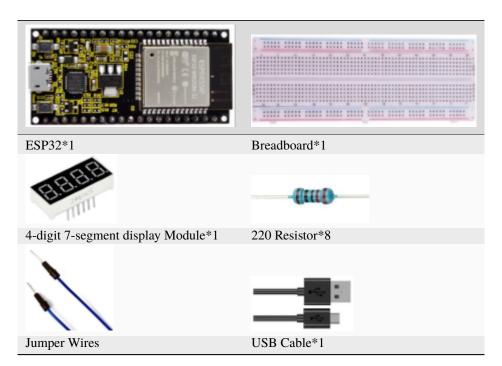
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the 1-Digit 7-Segment display will display numbers from 9 to 0.

## 9.9 Project 094-Digit Digital Tube

### 1.Introduction

The 4-digit 7-segment display is a very practical display device and it is used for devices such as electronic clocks, score counters and the number of people in the park. Because of the low price, easy to use, more and more projects will use the 4 Digit 7-segment display. In this project, we use ESP32 to control the 4-digit 7-segment display to display digits.

2.Components

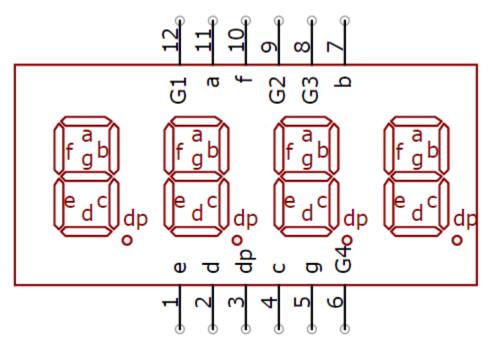


3.Component Knowledge

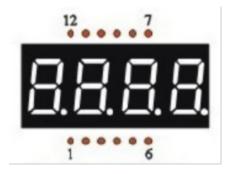


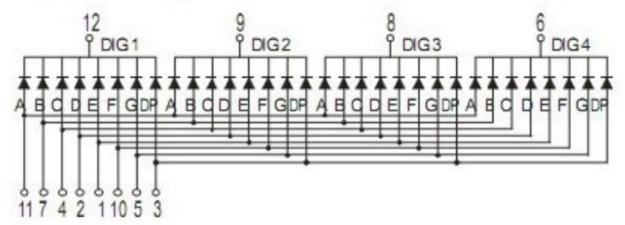
\*\*4-digit 7-segment display\*\*It is a device with common cathode and anode, its display principle is similar to the 1-Digit digital tube display. Both of them have eight GPIO ports to control the digital tube display, that is 8 leds. However, here is 4-digit, so it needs four GPIO ports to control the bit selection end. Our 4 - digit digital tube is common cathode.

The following figure shows the pin diagram of the 4-digit digital tube. G1, G2, G3 and G4 are the control pins.

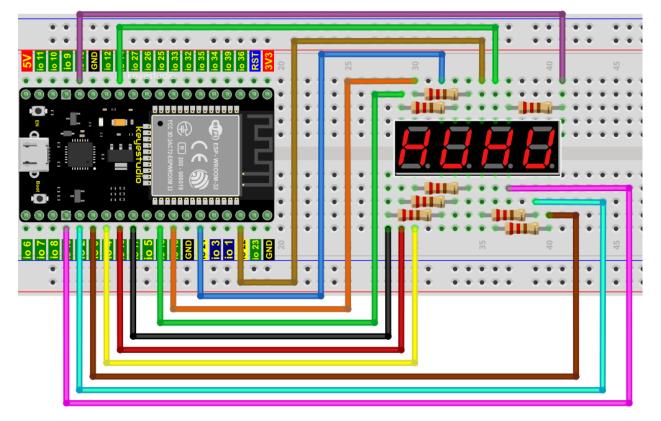


Schematic Diagram





4.Wiring Diagram



## 5.Test Code

```
/*
* Filename : 4-Digit Digital Tube
* Description : Four Digit Tube displays numbers from 0 to 9999.
* Auther : http//www.keyestudio.com
*/
#define d_a 18 //Define nixie tube a to pin 18
#define d_b 13
#define d_c 2
#define d_d 16
#define d_e 17
#define d_f 19
#define d_g 0
#define d_dp 4
#define G1 21 //Define the first set of nixtube G1 to pin 21
#define G2 22
#define G3 14
#define G4 15
//Nixie tube 0-F code value
unsigned char num[17][8] =
{
//abcdefgdp
\{1, 1, 1, 1, 1, 1, 1, 0, 0\},\
                          //0
\{0, 1, 1, 0, 0, 0, 0, 0\},\
                          //1
```

```
\{1, 1, 0, 1, 1, 0, 1, 0\},\
                                   //2
                                   //3
  \{1, 1, 1, 1, 1, 0, 0, 1, 0\},\
  \{0, 1, 1, 0, 0, 1, 1, 0\},\
                                   //4
                                   //5
  \{1, 0, 1, 1, 0, 1, 1, 0\},\
  \{1, 0, 1, 1, 1, 1, 1, 0\},\
                                   //6
                                   //7
  \{1, 1, 1, 0, 0, 0, 0, 0\},\
  \{1, 1, 1, 1, 1, 1, 1, 1, 0\},\
                                   //8
                                   //9
  \{1, 1, 1, 1, 1, 0, 1, 1, 0\},\
  \{1, 1, 1, 0, 1, 1, 1, 1\},\
                                   //A
  \{1, 1, 1, 1, 1, 1, 1, 1, 1\},\
                                   //B
                                   //C
  \{1, 0, 0, 1, 1, 1, 0, 1\},\
  \{1, 1, 1, 1, 1, 1, 1, 0, 1\},\
                                   //D
  \{1, 0, 0, 1, 1, 1, 1, 1\},\
                                   //E
  \{1, 0, 0, 0, 1, 1, 1, 1\},\
                                   //F
  \{0, 0, 0, 0, 0, 0, 0, 0, 1\},\
                                   //.
}:
void setup()
{
  pinMode(d_a,OUTPUT);
                           //Set to output pin
  pinMode(d_b,OUTPUT);
  pinMode(d_c,OUTPUT);
  pinMode(d_d,OUTPUT);
  pinMode(d_e,OUTPUT);
  pinMode(d_f,OUTPUT);
  pinMode(d_g,OUTPUT);
  pinMode(d_dp,OUTPUT);
  pinMode(G1,OUTPUT);
  pinMode(G2,OUTPUT);
  pinMode(G3,OUTPUT);
  pinMode(G4,OUTPUT);
}
void loop()
{
  //Start counting from 0 and gradually increase by 1 to 9999, repeating.
  for(int 1 = 0; 1 < 10; 1++)
  {
    for(int k = 0; k < 10; k++)
    {
      for(int j = 0; j < 10; j++)
      {
        for(int i = 0;i < 10;i++)</pre>
        {
          //125 flashes a second equals one second.
           //1000/8=125
          for(int q = 0;q<125;q++)
           ł
             Display(1,1);//The first nixie tube shows the value of L.
             delay(2);
                                                                                  (continues on next page)
```

```
Display(2,k);
            delay(2);
            Display(3,j);
            delay(2);
            Display(4,i);
            delay(2);
          }
        }
      }
   }
 }
}
//Display functions: g ranges from 1 to 4,num ranges from 0 to 9
void Display(unsigned char g, unsigned char n)
{
                              //Remove the light
  digitalWrite(d_a,LOW);
  digitalWrite(d_b,LOW);
  digitalWrite(d_c,LOW);
  digitalWrite(d_d,LOW);
  digitalWrite(d_e,LOW);
  digitalWrite(d_f,LOW);
  digitalWrite(d_g,LOW);
  digitalWrite(d_dp,LOW);
                      //Gate a choice
  switch(g)
  {
   case 1:
      digitalWrite(G1,LOW);
                              //Choose the first digit
      digitalWrite(G2,HIGH);
      digitalWrite(G3,HIGH);
      digitalWrite(G4,HIGH);
      break:
   case 2:
      digitalWrite(G1,HIGH);
                              //Choose the second bit
      digitalWrite(G2,LOW);
      digitalWrite(G3,HIGH);
      digitalWrite(G4,HIGH);
      break:
   case 3:
      digitalWrite(G1,HIGH);
      digitalWrite(G2,HIGH);
                              //Choose the third bit
      digitalWrite(G3,LOW);
      digitalWrite(G4,HIGH);
      break:
   case 4:
      digitalWrite(G1,HIGH);
      digitalWrite(G2,HIGH);
      digitalWrite(G3,HIGH);
      digitalWrite(G4,LOW);
                              //Choose the fourth bit
      break;
    default:break;
```

}	
<pre>digitalWrite(d_a,num[n][0]);</pre>	<pre>//a Queries the code value table</pre>
<pre>digitalWrite(d_b,num[n][1]);</pre>	
<pre>digitalWrite(d_c,num[n][2]);</pre>	
<pre>digitalWrite(d_d,num[n][3]);</pre>	
<pre>digitalWrite(d_e,num[n][4]);</pre>	
<pre>digitalWrite(d_f,num[n][5]);</pre>	
<pre>digitalWrite(d_g,num[n][6]);</pre>	
<pre>digitalWrite(d_dp,num[n][7]);</pre>	
}	
//*********	***************************

6.Test Result

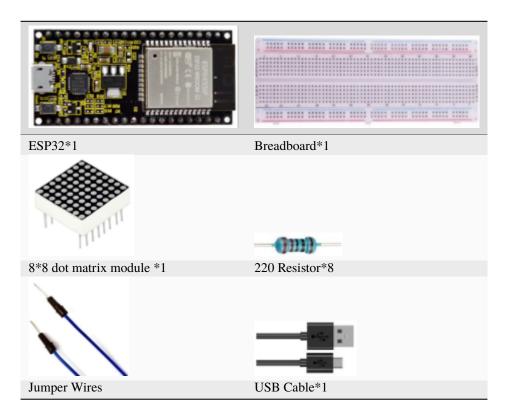
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the 4-digit 7-segment display 0-9999and repeat these actions in an infinite loop.

## 9.10 Project 108×8 Dot-matrix Display

#### 1.Introduction

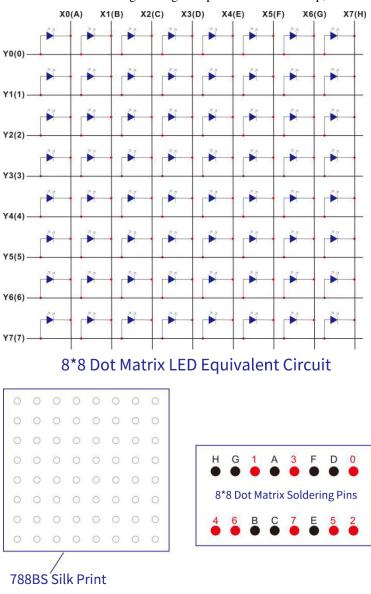
Dot matrix display is an electronic digital display device that can display information on machine, clocks, public transport departure indicators and many other devices. In this project, we will use ESP32 to control 8x8 LED dot matrix to display digits.

2.Components



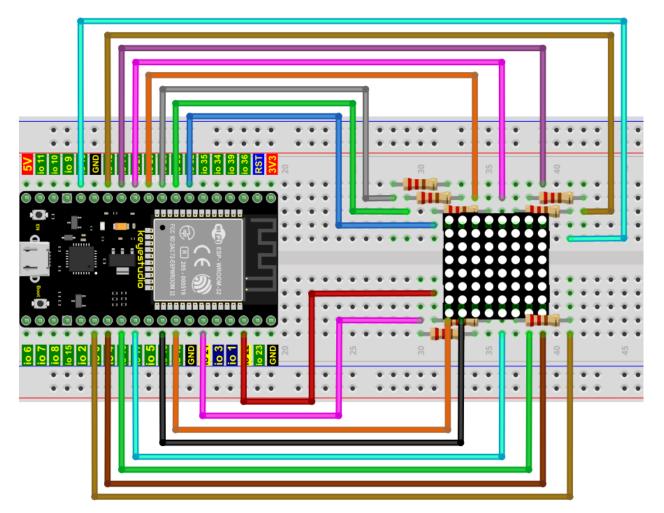
## 3.Component Knowledge

\*\*8\*8 dot matrix module\*\*The 8\*8 dot matrix is composed of 64 LEDs, including row common anode and row common cathode. Our module is row common anode, each row has a line connecting the positive pole of the LED, and the column is connecting the negative pole of the LED lamp, as shown in the following figure :



## 8\*8 Dot Matrix Outlook and Pinouts

4.Wiring Diagram



## 5.Test Code

```
/*
* Filename : 8×8 Dot-matrix Display
* Description : 8×8 Dot-matrix displays numbers from 0 to 9.
* Auther : http//www.keyestudio.com
*/
int R[] = {14,26,4,27,19,16,18,17};
int C[] = {32,21,22,12,0,13,33,25};
unsigned char data_0[8][8] =
{
\{0, 0, 1, 1, 1, 0, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 1, 1, 1, 0, 0, 0\}
};
```

```
unsigned char data_1[8][8] =
{
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 1, 1, 0, 0, 0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0,0,0,0,1,0,0,0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 1, 1, 1, 0, 0\}
};
unsigned char data_2[8][8] =
{
\{0, 0, 1, 1, 1, 0, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0,0,0,0,1,0,0,0\},\
\{0, 0, 0, 1, 0, 0, 0, 0\},\
\{0, 0, 1, 0, 0, 0, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 0, 0, 0\}
};
unsigned char data_3[8][8] =
{
\{0, 0, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 0, 0, 0\}
};
unsigned char data_4[8][8] =
{
\{0, 1, 0, 0, 0, 0, 0, 0\},\
\{0, 1, 0, 0, 1, 0, 0, 0\},\
\{0, 1, 0, 0, 1, 0, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 1, 1, 0\},\
\{0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 0, 1, 0, 0, 0\},\
\{0,0,0,0,0,0,0,0,0\}
};
unsigned char data_5[8][8] =
{
\{0, 1, 0, 0, 0, 0, 0, 0\},\
```

```
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 0, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0,0,0,0,0,0,0,0,0\}
};
unsigned char data_6[8][8] =
{
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 0, 0, 0\},\
\{0, 1, 0, 0, 0, 0, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0,0,0,0,0,0,0,0,0\}
};
unsigned char data_7[8][8] =
{
\{0, 0, 0, 0, 0, 0, 0, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
\{0, 0, 0, 0, 1, 0, 0, 0\},\
\{0, 0, 0, 1, 0, 0, 0, 0\},\
\{0, 0, 1, 0, 0, 0, 0, 0\},\
\{0, 1, 0, 0, 0, 0, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 0, 0, 0\}
};
unsigned char data_8[8][8] =
{
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0,0,0,0,0,0,0,0,0\}
};
unsigned char data_9[8][8] =
{
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 0, 0, 0, 1, 0, 0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 1, 0, 0\},\
```

```
\{0,0,0,0,0,0,1,0,0\},\
\{0, 1, 1, 1, 1, 1, 0, 0\},\
\{0, 0, 0, 0, 0, 0, 0, 0, 0\}
};
void Display(unsigned char dat[8][8])
{
for(int c = 0; c<8;c++)
{
digitalWrite(C[c],LOW);
for(int r = 0;r<8;r++)
{
digitalWrite(R[r],dat[r][c]);
}
delay(1);
Clear();
}
}
void Clear()
{
for(int i = 0;i<8;i++)</pre>
{
digitalWrite(R[i],LOW);
digitalWrite(C[i],HIGH);
}
}
void setup(){
  for(int i = 0;i<8;i++)</pre>
  {
    pinMode(R[i],OUTPUT);
    pinMode(C[i],OUTPUT);
  }
}
void loop(){
  for (int i = 1; i <= 100; i = i + (1)) {
    Display(data_0);
  }
  for (int i = 1; i <= 100; i = i + (1)) {</pre>
    Display(data_1);
  }
  for (int i = 1; i <= 100; i = i + (1)) {</pre>
    Display(data_2);
  }
  for (int i = 1; i <= 100; i = i + (1)) {</pre>
    Display(data_3);
  }
  for (int i = 1; i <= 100; i = i + (1)) {</pre>
    Display(data_4);
```

```
}
 for (int i = 1; i <= 100; i = i + (1)) {</pre>
   Display(data_5);
 }
 for (int i = 1; i \le 100; i = i + (1)) {
   Display(data_6);
 }
 for (int i = 1; i <= 100; i = i + (1)) {</pre>
   Display(data_7);
 }
 for (int i = 1; i <= 100; i = i + (1)) {</pre>
   Display(data_8);
 }
 for (int i = 1; i <= 100; i = i + (1)) {
   Display(data_9);
 }
}
```

## 6.Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the 8\*8 dot matrix displays the numbers 0~9 respectively.

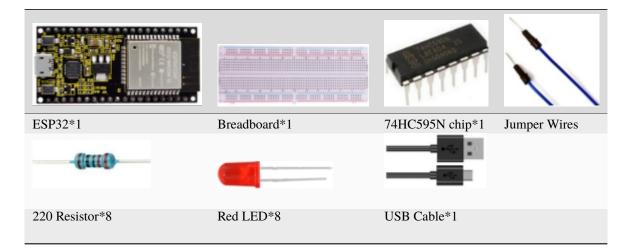
## 9.11 Project 1174HC595N Control 8 LEDs

## 1.Introduction

In previous projects, we learned how to light up an LED. With only 32 IO ports on ESP32, how do we light up a lot of leds? Sometimes it is possible to run out of pins on the ESP32, and you need to extend it with the shift register. You can use the 74HC595N chip to control 8 outputs at a time, taking up only a few pins on your microcontroller. In addition, you can also connect multiple registers together to further expand the output.

In this project, we will use a ESP32a 74HC595 chip and LEDs to make a flowing water light to understand the function of the 74HC595 chip.

2.Components

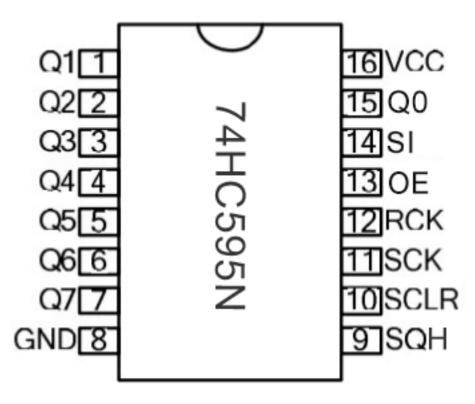


3.Component Knowledge



**74HC595N Chip:** To put it simply, 74HC595N chip is a combination of 8-digit shifting register, memorizer and equipped with tri-state output. The shift register and the memorizer are synchronized to different clocks, and the data is input on the rising edge of the shift register clock SCK and goes into the memory register on the rising edge of the memory register clock RCK.

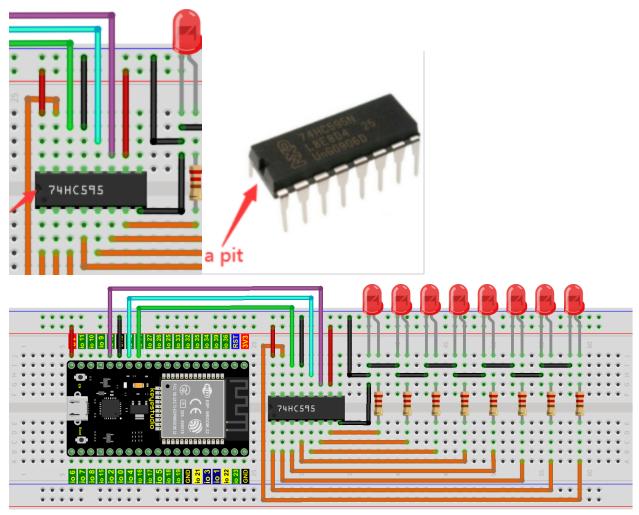
If the two clocks are connected together, the shift register is always one pulse earlier than the storage register. The shift register has a serial shift input (SI) and a serial output (SQH) for cascading. The 8-bit shift register can be reset asynchronously (low-level reset), and the storage register has an 8-bit Three-state parallel bus output, when the output enable (OE) is enabled (active low), the storage register is output to the 74HC595N pin (bus).



## Pins

4.Wiring Diagram

Note: Pay attention to the direction in which the 74HC595N chip is inserted.



fritzing

5.Test Code

```
/*
* Filename : 74HC595N Control 8 LEDs
* Description : Use 74HC575N to drive ten leds to display the flowing light.
* Auther : http//www.keyestudio.com
*/
int dataPin = 14; // Pin connected to DS of 74HC595(Pin14)
int latchPin = 12; // Pin connected to ST_CP of 74HC595(Pin12)
int clockPin = 13; // Pin connected to SH_CP of 74HC595(Pin11)
   void setup() {
     // set pins to output
     pinMode(latchPin, OUTPUT);
     pinMode(clockPin, OUTPUT);
     pinMode(dataPin, OUTPUT);
   }
   void loop() {
 // Define a one-byte variable to use the 8 bits to represent the state of 8 LEDs of
\rightarrow LED bar graph.
```

```
(continued from previous page)
```

```
// This variable is assigned to 0x01, that is binary 00000001, which indicates only.
→one LED light on.
 byte x = 0x01;
                  // 0b 0000 0001
 for (int j = 0; j < 8; j++) { // Let led light up from right to left
   writeTo595(LSBFIRST, x);
   \mathbf{x} <<= 1; // make the variable move one bit to left once, then the bright LED move.
\rightarrow one step to the left once.
   delay(50);
 }
 delay(100);
                 //0b 1000 0000
 x = 0 \times 80;
 for (int j = 0; j < 8; j++) { // Let led light up from left to right
   writeTo595(LSBFIRST, x);
   x >>= 1;
   delay(50);
 }
 delay(100);
}
void writeTo595(int order, byte _data ) {
 // Output low level to latchPin
 digitalWrite(latchPin, LOW);
 // Send serial data to 74HC595
 shiftOut(dataPin, clockPin, order, _data);
 // Output high level to latchPin, and 74HC595 will update the data to the parallel.
\rightarrowoutput port.
 digitalWrite(latchPin, HIGH);
}
```

## 6.Test Result

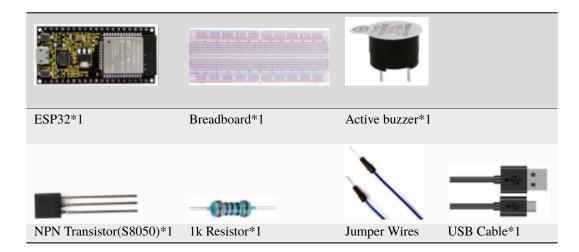
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the 8 LEDs start flashing in flowing water mode.

## 9.12 Project 12Active Buzzer

## 1.Introduction

Active buzzer is a sound component that is widely used as a sound component for computers, printers, alarms, electronic toys and phones, timers etc. It has an internal vibration source, just by connecting to a 5V power supply, it can continuously buzz. In this project, we will use ESP32 to control the active buzzer to beep.

2.Components

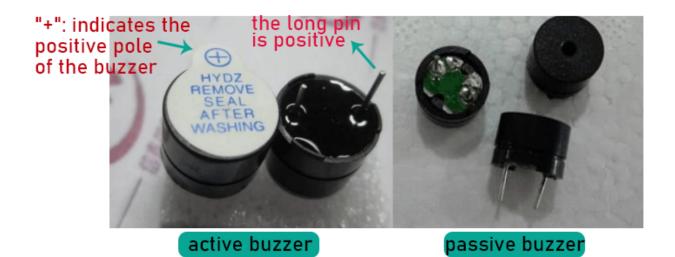


3.Component Knowledge



The active buzzer inside has a simple oscillator circuit, which can convert constant direct current into a certain frequency pulse signal. Once active buzzer receives a high level, it will sound. The passive buzzer is an integrated electronic buzzer with no internal vibration source. It must be driven by 2K to 5K square wave instead of a DC signal.

The appearance of the two buzzers is very similar, but passive buzzers come with a green circuit board, and active buzzers come with a black tape. Passive buzzers don't have positive pole, but active buzzers have. As shown below:



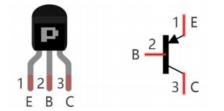
#### **Transistor:**

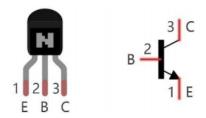


Since the buzzer requires such large current that GPIO of ESP32 output capability cannot meet the requirement, a transistor of NPN type is needed here to amplify the current.

Transistor, the full name: semiconductor transistor, is a semiconductor device that controls current. Transistor can be used to amplify weak signal, or work as a switch. It has three electrodes(PINs): base (b), collector © and emitter (e).

When there is current passing between "be", "ce", which will allow several-fold current (transistor magnification) pass, at this point, transistor works in the amplifying area. When current between "be" exceeds a certain value, "ce", which will not allow current to increase any longer, at this point, transistor works in the saturation area. Transistor has two types as shown below: PNP and NPN,





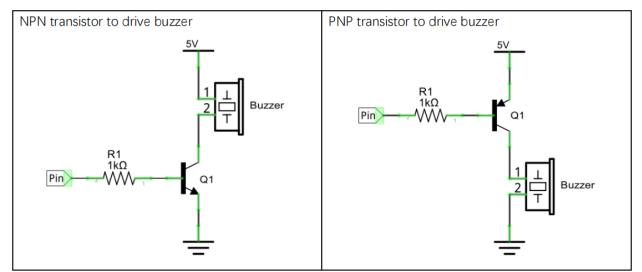
PNP transistor NPN transistor

In our kit, the PNP transistor is marked with 8550, and the NPN transistor is marked with 8050.

Based on the transistor's characteristics, it is often used as a switch in digital circuits. As micro-controller's capacity to output current is very weak, we will use a transistor to amplify current and drive large-current components.

When using the NPN transistor to drive a buzzer, we often adopt the following method. If GPIO outputs high level, current will flow through R1, the transistor will get conducted, and the buzzer will sound. If GPIO outputs low level, no current flows through R1, the transistor will not be conducted, and buzzer will not sound.

When using the PNP transistor to drive a buzzer, we often adopt the following method. If GPIO outputs low level, current will flow through R1, the transistor will get conducted, and the buzzer will sound. If GPIO outputs high level, no current flows through R1, the transistor will not be conducted, and buzzer will not sound.



#### 4.Wiring Diagram

		(	REMOVE SEAL AFTER	IING	-																		
•••••			REM SE		Ð	••	••					••			-			-		•••		•	
	7	1			1	35			40			45			00			22			09		
		••	••	• • •		••	••	••	• •			••	• •		••	•		• •		• •	•	• •	•
		••	•••		• •	••	••	**	**	•••	•	••	* 4	• • •	••	•	• •	* *	•	• •	•	**	• 0 • L
	. 15		• • •				• •																• •
		**	•••			• •	••	**	**	•••		••	**	• • •	••	• •	• •	* *	•	• •	•	• •	• 0
	1	•••	•••	•••		••	••	* *	**	•••		••	* *		••	•	••	• •	•	••	•	••	• 0 • <
6002:0002:0002 0002:00000000000000000000				9 9		35	-		40			45			20			55			60		
· · · · · · · · · · · · · · · · · · ·		•	•••	••	•	••	**	•	+ 4			••	**	••	•	•	••	•	•	••	•	•	

## fritzing

Note: The buzzer power supply in this circuit is 5V. On a 3.3V power supply, the buzzer can work, but it will reduce the loudness.

5.Test Code

```
(continued from previous page)
```

```
* Auther
           : http//www.keyestudio.com
*/
#define buzzerPin 15 //define buzzer pins
void setup ()
{
 pinMode (buzzerPin, OUTPUT);
}
void loop ()
{
 digitalWrite (buzzerPin, HIGH);
 delay (500);
 digitalWrite (buzzerPin, LOW);
 delay (500);
}
```

#### 6.Test Result

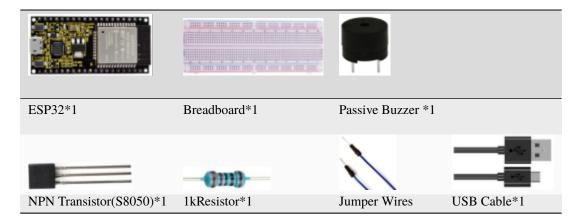
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the active buzzer beeps.

## 9.13 Project 13Passive Buzzer

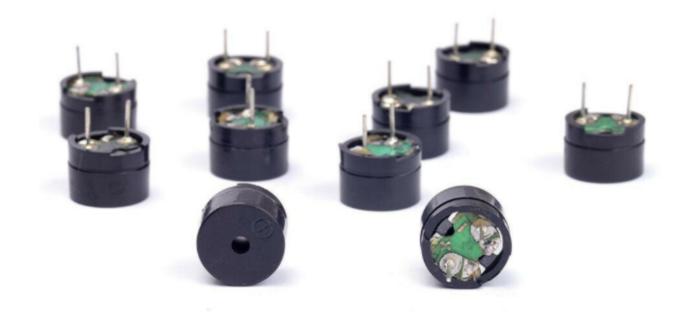
1.Introduction

In a previous project, we studied an active buzzer, which can only make a sound and may make you feel very monotonous. In this project, we will learn a passive buzzer and use the ESP32 control it to work. Unlike the active buzzer, the passive buzzer can emit sounds of different frequencies.

2. Components



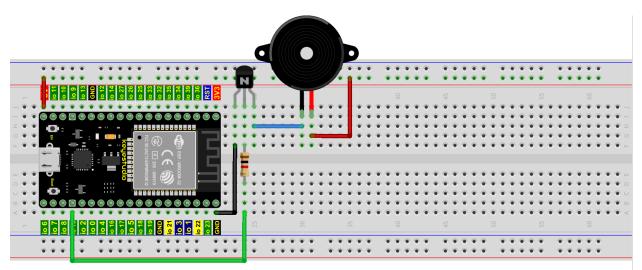
3.Component Knowledge



**Passive buzzer:** A passive buzzer is an integrated electronic buzzer with no internal vibration source and it has to be driven by 2K-5K square waves, not DC signals. The two buzzers are very similar in appearance, but one buzzer with a green circuit board is a passive buzzer and the other buzzer with black tape is an active buzzer. Passive buzzers cannot distinguish between positive polarity while active buzzers can.



Transistor: Please refer to project 12.4.Wiring Diagram



fritzing

5.Test Code

/\* \* Filename : Passive Buzzer \* Description : Passive Buzzer sounds the alarm. \* Auther : http//www.keyestudio.com \*/ #define LEDC\_CHANNEL\_0 0 // LEDC timer uses 13 bit accuracy #define LEDC\_TIMER\_13\_BIT 13 // Define tool I/O ports #define BUZZER\_PIN 15 //Create a musical melody list, Super Mario int melody[] = {330, 330, 330, 262, 330, 392, 196, 262, 196, 165, 220, 247, 233, 220, →196, 330, 392, 440, 349, 392, 330, 262, 294, 247, 262, 196, 165, 220, 247, 233, 220, **.** →196, 330, 392,440, 349, 392, 330, 262, 294, 247, 392, 370, 330, 311, 330, 208, 220, **1**  $\leftrightarrow$  262, 220, 262, 294, 392, 370, 330, 311, 330, 523, 523, 523, 392, 370, 330, 311, 330, 208, 220, 262,220, →262, 294, 330, 262, 262, 262, 262, 294, 330, 262, 220, 196}; //Create a list of tone durations  $\rightarrow$ 4,8,4,8,4,8,2,8,4,4,8,4,1,8,4,4,8,4,8,4,8,2}; void setup() { pinMode(BUZZER\_PIN, OUTPUT); // Set the buzzer to output mode



## 6.Test Result

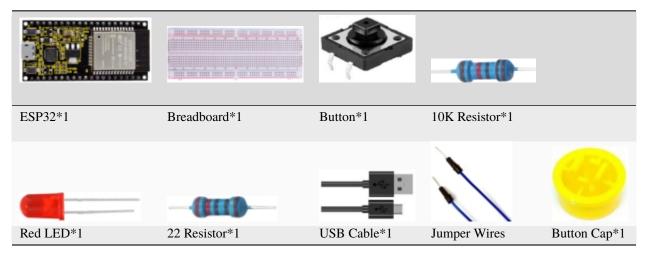
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the passive buzzer plays music.

## 9.14 Project 14: Mini Table Lamp

#### 1. Introduction

Do you know that the ESP32 can light up an LED when you press a button? In this project, we will use a ESP32, a button switch and a LED to make a mini table lamp.

2. Components



3. Component Knowledge



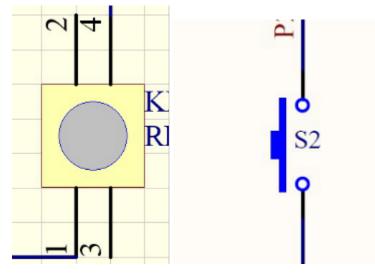
Button: A button can control the circuit on and off, the button is plugged into a circuit, the circuit is disconnected when the button is not pressed. The circuit works when you press the button, but breaks again when you release it. Why does it only work when you press it? It starts from the internal structure of the button, which don't allow current to travel from one end of the button to the other before it is pressed. When pressed, a metal strip inside the button connects the two sides to allow electricity to pass through.

The internal structure of the button is shown in the figure

. Before the button is pressed, 1 and 2 are on, 3 and 4 are also on, but 1, 3 or 1, 4 or 2, 3 or 2, 4 are off(not working). Only when the button is pressed, 1, 3 or 1, 4 or 2, 3 or 2, 4 are on.

The button switch is one of the most commonly used components in circuit design.

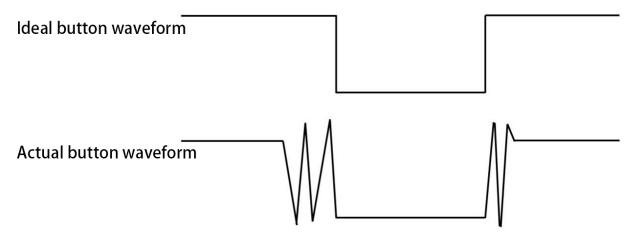
## Schematic diagram of the button:



## What is button shake?

We think of the switch circuit as "press the button and turn it on immediately", press it again and turn it off immediately". In fact, this is not the case.

The button usually uses a mechanical elastic switch, and the mechanical elastic switch will produce a series of [shake](javascript:;) due to the elastic action at the moment when the mechanical contact is opened and closed (usually about 10ms). As a result, the button switch will not immediately and stably turn on the circuit when it is closed, and it will not be completely and instantaneously disconnected when it is turned off.



## How to eliminate the shake?

There are two common methods, namely fix shake in the software and hardware. We only discuss the shake removal in the software.

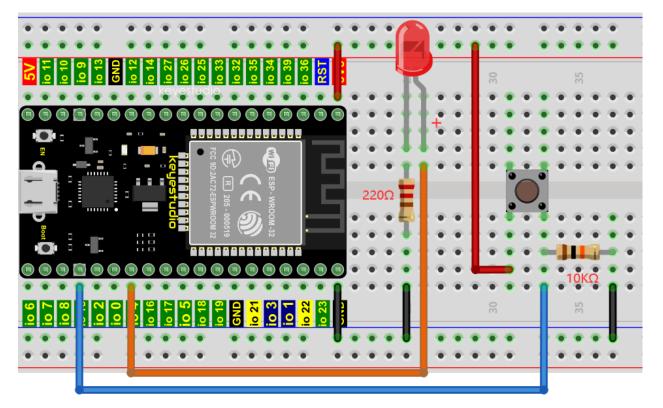
We already know that the shake time generated by elasticity is about 10ms, and the delay command can be used to delay the execution time of the command to achieve the effect of shake removal.

Therefore, we delay 0.02s in the code to achieve the key anti-shake function.

## Effect excluding jitter

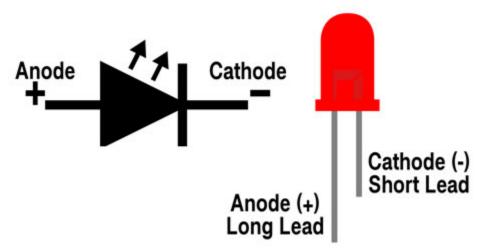


4. Wiring Diagram

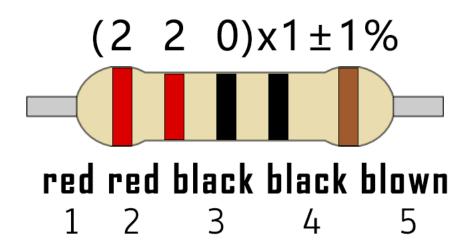


## Note:

How to connect the LED



How to identify the 220 5-band resistor and 10K 5-band resistor



5. Test code

```
/*
* Filename : Mini Table Lamp
* Description : Make a table lamp.
* Auther : http//www.keyestudio.com
  */
  #define PIN_LED
                4
  #define PIN_BUTTON 15
  bool ledState = false;
void setup() {
 // initialize digital pin PIN_LED as an output.
 pinMode(PIN_LED, OUTPUT);
 pinMode(PIN_BUTTON, INPUT);
}
// the loop function runs over and over again forever
void loop() {
 if (digitalRead(PIN_BUTTON) == LOW) {
   delay(20);
   if (digitalRead(PIN_BUTTON) == LOW) {
    reverseGPIO(PIN_LED);
   }
   while (digitalRead(PIN_BUTTON) == LOW);
 }
}
void reverseGPIO(int pin) {
 ledState = !ledState;
 digitalWrite(pin, ledState);
}
```

## 6. Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you

will see that press the push button switch, the LED turns on; When it is released, the LED is still on. Press it again, and the LED turns off. When it is released, the LED stays off. Doesn't it look like a mini table lamp?

## 9.15 Project 15Tilt And LED

1. Introduction

The ancients without electronic clock, so the hourglass are invented to measure time. The hourglass has a large capacity on both sides, and which is filled with fine sand on one side. What's more, there is a small channel in the middle, which can make the hourglass stand upright, the side with fine sand is on the top. due to the effect of gravity, the fine sand will flow down through the channel to the other side of the hourglass.

When the sand reaches the bottom, turn it upside down and record the number of times it has gone through the hourglass, therefore, the next day we can know the approximate time of the day by it.

In this project, we will use ESP32 to control the tilt switch and LED lights to simulate an hourglass to make an electronic hourglass.

- ESP32\*1Tilt Switch\*1Red LED\*410K Resistor\*1Image: Breadboard\*1220 Resistor\*4USB Cable\*1Jumper Wires
- 2. Components

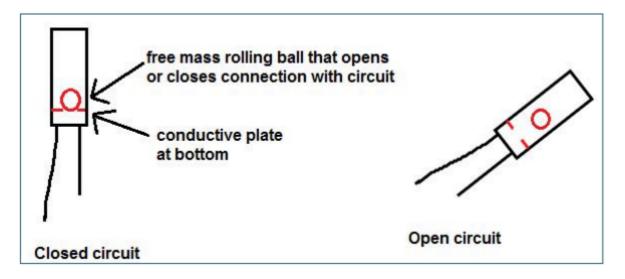
3. Component Knowledge



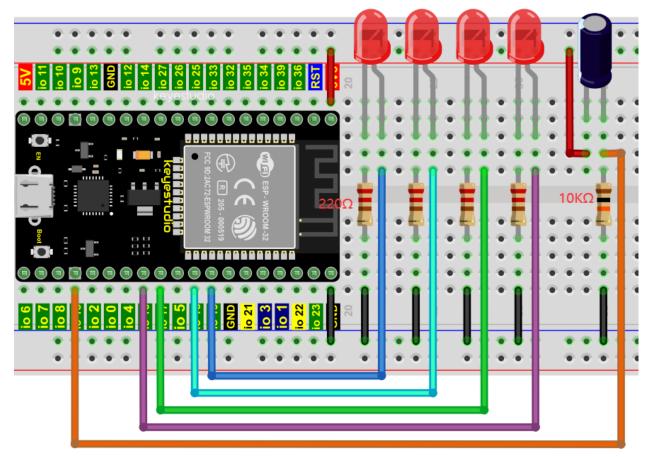
Tilt switch is also called digital switch. Inside is a metal ball that can roll. The principle of rolling the metal ball to contact with the conductive plate at the bottom, which is used to control the on and off of the circuit. When it is a rolling ball tilt sensing switch with single directional trigger, the tilt sensor is tilted toward the trigger end (two gold-plated pin ends), the tilt switch is in a closed circuit and the voltage at the analog port is about 5V(binary number is 1023).

In this way, the LED will light up. When the tilting switch is in horizontal position or tilting to the other end, the tilting switch is in open state the voltage of the analog port is about 0V (binary number is 0), the LED will turn off. In the program, we judge the state of the switch based on whether the voltage value of the analog port is greater than 2.5V (binary number is 512).

The internal structure of the tilt switch is used here to illustrate how it works, as shown below:

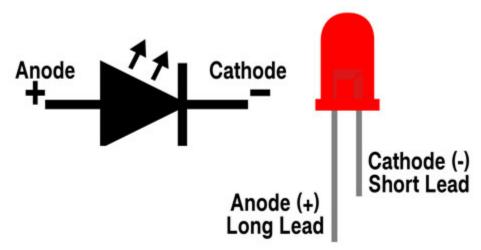


4. Wiring Diagram

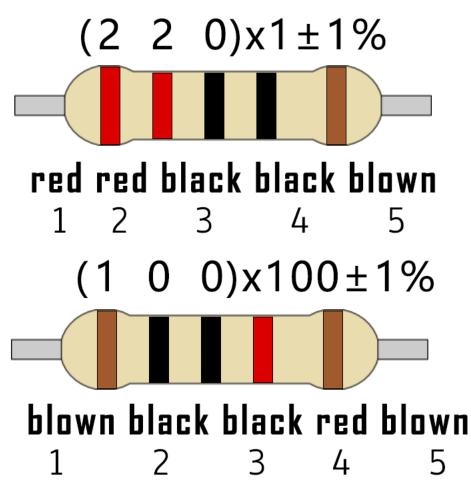


Note:

How to connect the LED



How to identify the 220 5-band resistor and 10K 5-band resistor



5. Test Code

```
/*
* Filename
          : Tilt And LED
* Description : Tilt switches and four leds to simulate an hourglass.
        : http//www.keyestudio.com
* Auther
*/
  #define SWITCH_PIN 15 // the tilt switch is connected to Pin15
  byte switch_state = 0;
  void setup()
  {
    for(int i=16;i<20;i++)</pre>
    {
      pinMode(i, OUTPUT);
    }
   pinMode(SWITCH_PIN, INPUT);
   for(int i=16;i<20;i++)
    {
   digitalWrite(i,0);
   }
    Serial.begin(9600);
  }
```

(continues on next page)

```
void loop()
{
switch_state = digitalRead(SWITCH_PIN);
Serial.println(switch_state);
if (switch_state == 0)
{
for(int i=16;i<20;i++)</pre>
 {
digitalWrite(i,1);
delay(500);
 }
 }
if (switch_state == 1)
{
for(int i=19;i>15;i--)
{
digitalWrite(i,0);
delay(500);
}
 }
}
```

# 6. Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that when you tilt the breadboard to an angle, the LEDs will light up one by one. When you turn the breadboard to the original angle, the LEDs will turn off one by one. Like the hourglass, the sand will leak out over time.

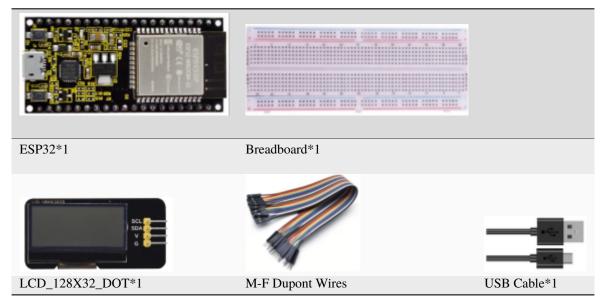
# 9.16 Project 16: I2C 128×32 LCD

#### 1. Introduction

In everyday life, we can do a host of experiments with the display module and also DIY a broad menu of small objects. For example, you can make a temperature meter with a temperature sensor and a display, or make a distance meter with an ultrasonic module and a display.

In this project, we will use the LCD\_128X32\_DOT module as the display and connect it to the ESP32, which will be used to control the LCD\_128X32\_DOT display to show various English words, common symbols and numbers.

#### 2. Components

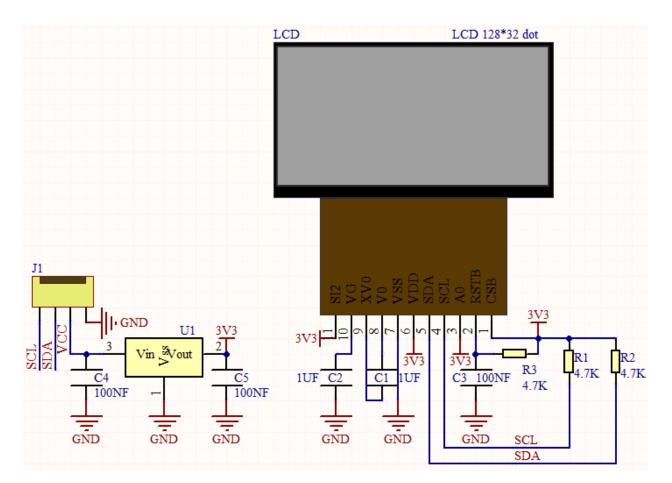


3. Component Knowledge

**LCD\_128X32\_DOT**: It is an LCD module with 128\*32 pixels and its driver chip is ST7567A. The module uses the IIC communication mode, while the code contains a library of all alphabets and common symbols that can be called directly.

When using, we can also set it in the code so that the English letters and symbols show different text sizes. To make it easy to set up the pattern display, we also provide a mold capture software that converts a specific pattern into control code and then copies it directly into the test code for use.

# Schematic diagram of LCD\_128X32\_DOT

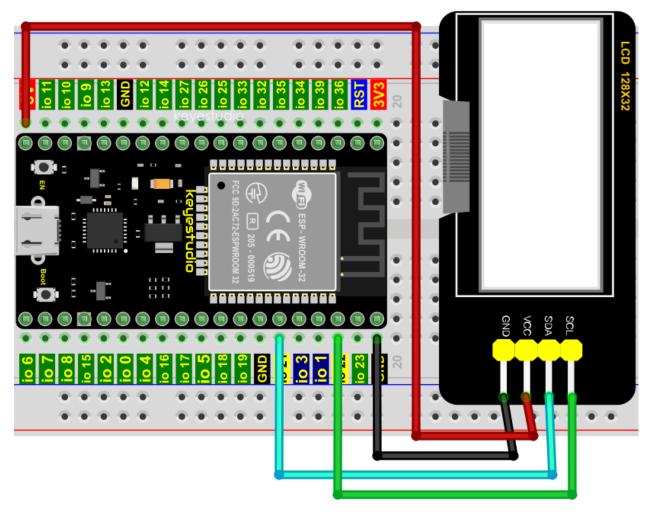


# Features:

Pixel: 128\*32 character Operating voltage(chip)4.5V to 5.5V Operating current100mA (5.0V)

Optimal operating voltage(module):5.0V

4. Wiring Diagram



5. Adding the lcd128\_32\_io library

This code uses a library named "**lcd128\_32\_io**", if you haven't installed it yet, please do so before learning. The steps to add third-party libraries are as follows:

Click on the link to download the library file:Arduino C "lcd128\_32\_io.h" Librarie

6. Test Code

(continues on next page)

# 6. Test Result

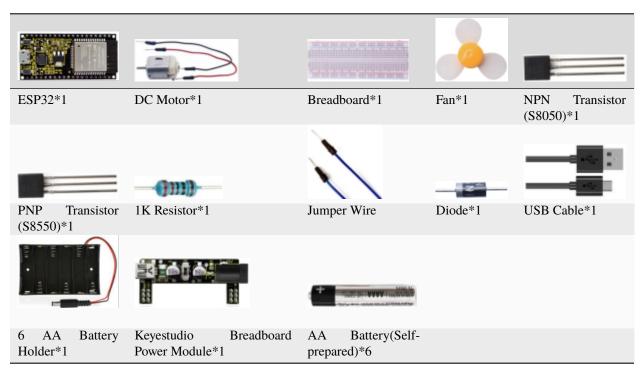
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the 128X32LCD module display will show "KEYESTUDIO" at the first line "ABCDEFGHIJKLMNOPQR" will be displayed at the second line "123456789 $\pm$ \*/<>=\$@" will be shown at the third line and "%^&(){}:;'?,.-\[]" will be displayed at the fourth line.

# 9.17 Project 17Small Fan

1. Introduction

In hot summer, we need electric fans to cool us down, so in this project, we will use a ESP32 to control a DC motor and small fan blades to make a **small electric fan.** 

2. Components



3. Component Knowledge:

Keyestudio Breadboard Power Supply Module



# Introduction:

This breadboard power supply module is compatible with 5V and 3.3V, which can be applied to MB102 breadboard. The module contains two channels of independent control, powered by the USB all the way.

The output voltage is constant for the DC5V, and another way is powered by DC6.5-12V, output controlled by the slide switch, respectively for DC 5V and DC 3.3V.

If the other power supply is DC 6.5-12v, when the slide switch is switched to +5V, the output voltages of the left and right lines of the module are DC 5V. When the slide switch is switched to +3V, the output voltage of the USB power supply terminal of the module is DC 5V, and the output voltage of the DC 6.5-12V power supply terminal of the other power supply is DC 3.3V.

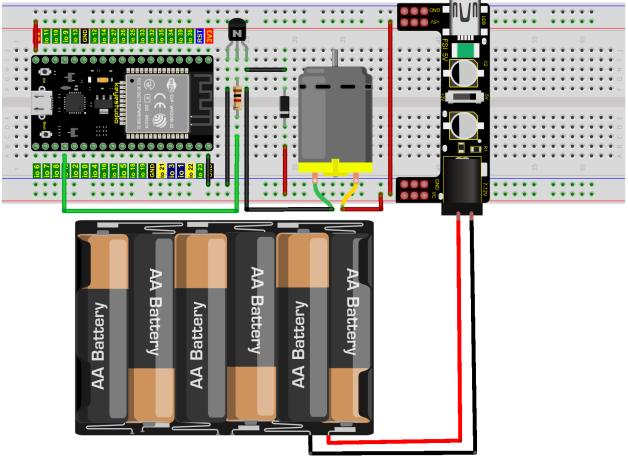
# Specification:

- Applied to MB102 breadboard;
- Input voltageDC 6.5-12V or powered by USB;
- Output voltage3.3V or 5V
- Max output current:<700ma
- Up and down two channels of independent control, one of which can be switched to 3.3V or 5V;

Comes with two sets of DC output pins, easy for external use.

# 4. Wiring Diagram 1

We use the S8050NPN transistor) to control the motor



fritzing

Wire up first, then connect a fan at the DC motor

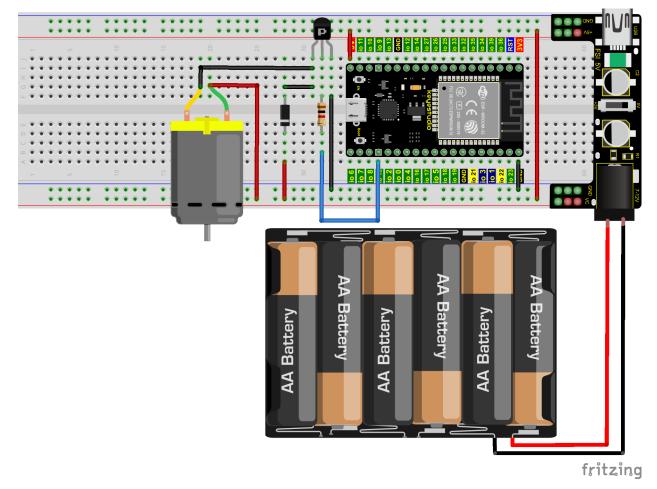
# 5. Test Code 1

6. Test Result 1

Upload the code to the ESP32 and power up. You will view the motor rotate for 4s, stop for 2s, in a loop way

7. Wiring Diagram 2

We use the S8050PNP transistor) to control the motor



Wire up first, then connect a fan at the DC motor

8. Test Code 2

```
//*
* Filename : Small_Fan
* Description : S8550 triode drives the motor working
* Auther : http//www.keyestudio.com
*/
```

9. Test Result 2

Upload the code to the ESP32 and power up. You will view the motor rotate for 4s, stop for 2s, in a loop way

# 9.18 Project 18: Dimming Light

#### 1. Introduction

A potentiometer is a three-terminal resistor with sliding or rotating contacts that forms an adjustable voltage divider. It works by changing the position of the sliding contacts across a uniform resistance. In the potentiometer, the entire input voltage is applied across the whole length of the resistor, and the output voltage is the voltage drop between the fixed and sliding contact.

In this project, we will learn how to use ESP32 to read the values of the potentiometer, and make a dimming lamp with LED.

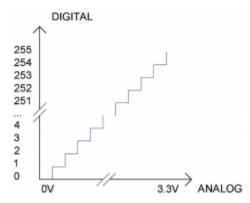
- ESP32\*1Breadboard\*1Potentiometer\*1Red LED\*1220Resistor\*1Jumper WiresUSB Cable\*1
- 2. Components

3. Component Knowledge



Adjustable potentiometer: It is a kind of resistor and an analog electronic component, which has two states of 0 and 1(high level and low level). The analog quantity is different, its data state presents a linear state such as  $1 \sim 1024$ 

**ADC :** An ADC is an electronic integrated circuit used to convert analog signals such as voltages to digital or binary form consisting of 1s and 0s. The range of our ADC on ESP32 is 12 bits, that means the resolution is 2^12=4096, and it represents a range (at 3.3V) will be divided equally to 4096 parts. The rage of analog values corresponds to ADC values. So the more bits the ADC has, the denser the partition of analog will be and the greater the precision of the resulting conversion.



Subsection 1: the analog in rang of 0V—3.3/4095 V corresponds to digital 0.

Subsection 2: the analog in rang of 3.3/4095 V—2\*3.3 /4095V corresponds to digital 1;

The following analog will be divided accordingly.

The conversion formula is as follows:

$$ADCValue = \frac{Ana \log Voltage}{3.3} * 4095$$

\*\*DAC\*\*The reversing of this process requires a DAC, Digital-to-Analog Converter. The digital I/O port can output high level and low level (0 or 1), but cannot output an intermediate voltage value.

This is where a DAC is useful. ESP32 has two DAC output pins with 8-bit accuracy, GPIO25 and GPIO26, which can divide VCC (here is 3.3V) into  $2^8=256$  parts. For example, when the digital quantity is 1, the output voltage value is 3.3/256 \*128=1.65V, the higher the accuracy of DAC, the higher the accuracy of output voltage value will be.

The conversion formula is as follows:

# $Ana\log Voltage = \frac{DACValue}{255} * 3.3(V)$

# ADC on ESP32

ESP32 has 16 pins can be used to measure analog signals. GPIO pin sequence number and analog pin definition are shown in the following table

ADC number in ESP32	ESP32 GPIO number
ADC0	GPIO 36
ADC3	GPIO 39
ADC4	GPIO 32
ADC5	GPIO33
ADC6	GPIO34
ADC7	GPIO 35
ADC10	GPIO 4
ADC11	GPIO0
ADC12	GPIO2
ADC13	GPIO15
ADC14	GPIO13
ADC15	GPIO 12
ADC16	GPIO 14
ADC17	GPIO27
ADC18	GPIO25
ADC19	GPIO26

# DAC on ESP32

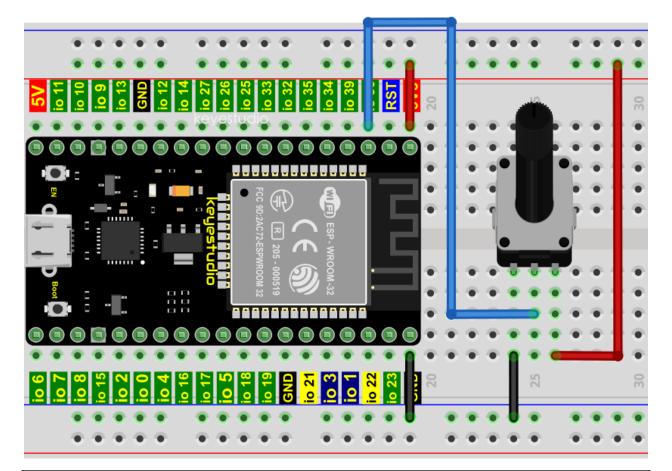
ESP32 has two 8-bit digital analog converters to be connected to GPIO25 and GPIO26 pins, respectively, and it is immutable. As shown in the following table

Simulate pin number	GPIO number
DAC1	GPIO25
DAC2	GPIO26

The DAC pin number is already defined in ESP32's code base; for example, you can replace GPIO25 with DAC1 in the code.

4. Read the values of the potentiometer

We connect the potentiometer to the analog IO port of ESP32 to read the ADC value, DAC value and voltage value of the potentiometer, please refer to the wiring diagram below



```
//******
                    /*
* Filename : Read Potentiometer Analog Value
* Description : Basic usage of ADCDAC and Voltage
* Auther
          : http//www.keyestudio.com
*/
#define PIN_ANALOG_IN 36 //the pin of the Potentiometer
void setup() {
 Serial.begin(115200);
}
//In loop() the analogRead() function is used to obtain the ADC value, and then the map().
\rightarrow function is used to convert the value into an 8-bit precision DAC value. The input and
-output voltage are calculated according to the previous formula, and the information
\rightarrow is finally printed out.
void loop() {
 int adcVal = analogRead(PIN_ANALOG_IN);
 int dacVal = map(adcVal, 0, 4095, 0, 255);
 double voltage = adcVal / 4095.0 * 3.3;
 Serial.printf("ADC Val: %d, \t DAC Val: %d, \t Voltage: %.2fV\n", adcVal, dacVal,
→voltage);
 delay(200);
}
```

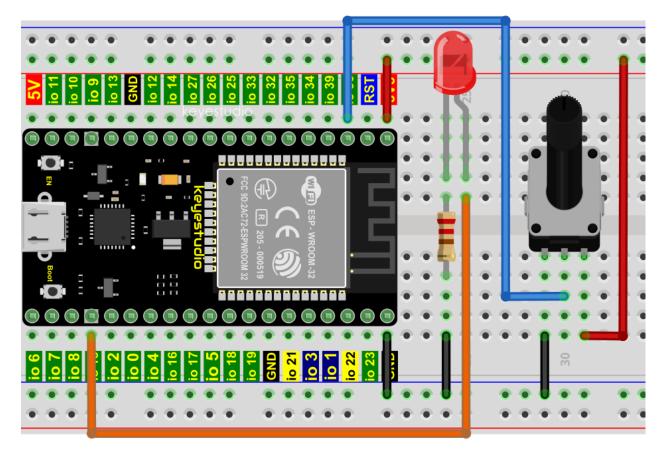
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable, open the serial monitor and set the baud rate to 115200 and press the reset button first.

You will see that the serial monitor window will print out the ADC value, DAC value and voltage value of the potentiometer. When turning the potentiometer handle, the ADC value, DAC value and voltage value will change. As shown below:

		/dev/tty	USB0		~ ^ X
					Send
ADC Val: 43,       D         ADC Val: 38,       D         ADC Val: 443,       D         ADC Val: 477,       D         ADC Val: 490,       D         ADC Val: 753,       D         ADC Val: 928,       D         ADC Val: 925,       D         ADC Val: 1359,       D         ADC Val: 1421,       D         ADC Val: 1463,       D         ADC Val: 1775,       D	AC Val: 4, Volta AC Val: 3, Volta DAC Val: 3, Volta DAC Val: 2, Volta DAC Val: 28, Volta DAC Val: 30, Volta DAC Val: 31, Volta DAC Val: 47, Volta DAC Val: 58, Volta DAC Val: 58, Volta DAC Val: 85, Volta DAC Val: 88, Volta DAC Val: 91, Volta DAC Val: 111, Volta DAC Val: 119, Volta	ge: 0.03V ge: 0.03V ge: 0.36V ge: 0.38V ge: 0.39V ge: 0.61V ge: 0.75V ge: 0.75V ge: 1.10V ge: 1.15V ge: 1.43V			
Autoscroll Show	w timestamp		Newline	▼ 115200 baud	<ul><li>✔</li><li>✔</li><li>✔</li><li>Clear output</li></ul>

5. Wiring diagram of the dimming lamp

In the previous step, we read the ADC value, DAC value and voltage value of the potentiometer. Now we need to convert the ADC value of the potentiometer into the brightness of the LED to make a lamp that can adjust the brightness. The wiring diagram is as follow:

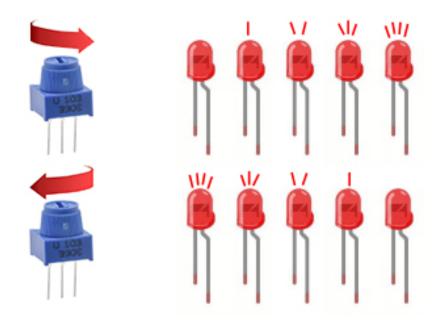


```
6. Test Code
```

//************************************
<pre>* Filename : Dimming Light * Description : Controlling the brightness of LED by potentiometer. * Auther : http//www.keyestudio.com */</pre>
#define PIN_ANALOG_IN 36 //the pin of the potentiometer
#define PIN_LED 15 // the pin of the LED
#define CHAN 0
<pre>void setup() {</pre>
ledcSetup(CHAN, 1000, 12);
<pre>ledcAttachPin(PIN_LED, CHAN);</pre>
}
<pre>void loop() {</pre>
<pre>int adcVal = analogRead(PIN_ANALOG_IN); //read adc</pre>
<pre>int pwmVal = adcVal; // adcVal re-map to pwmVal</pre>
<pre>ledcWrite(CHAN, pwmVal); // set the pulse width.</pre>
<pre>delay(10);</pre>
}
//***********************************

6. Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that turn the potentiometer handle and the brightness of the LED will change accordingly.

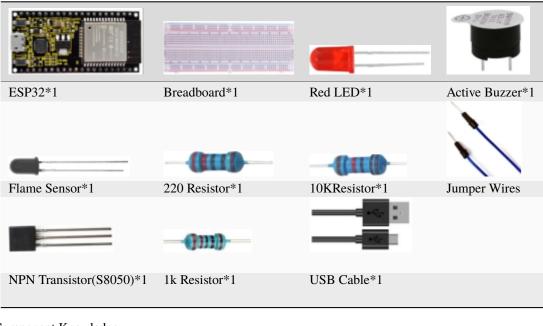


# 9.19 Project 19Flame Alarm

# 1. Introduction

Fire is a terrible disaster and fire alarm systems are very useful in housescommercial buildings and factories. In this project, we will use ESP32 to control a flame sensor, a buzzer and a LED to simulate fire alarm devices. This is a meaningful maker activity.

2. Component

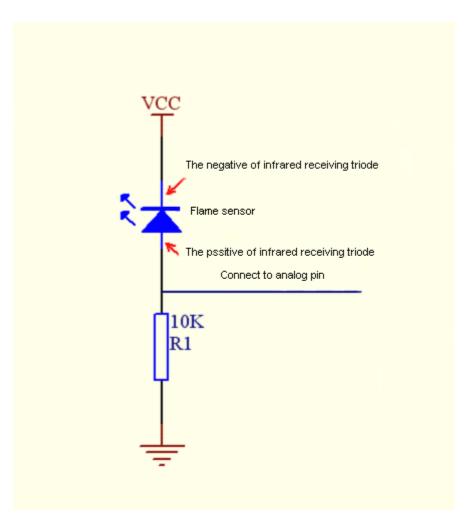


3. Component Knowledge



The flame emits a certain amount IR light that is invisible to the human eye, but our flame sensor can detect it and alert a microcontroller(such as ESP32) that a fire has been detected. It has a specially designed infrared receiver tube to detect the flame and then convert the flame brightness into a fluctuating level signal.

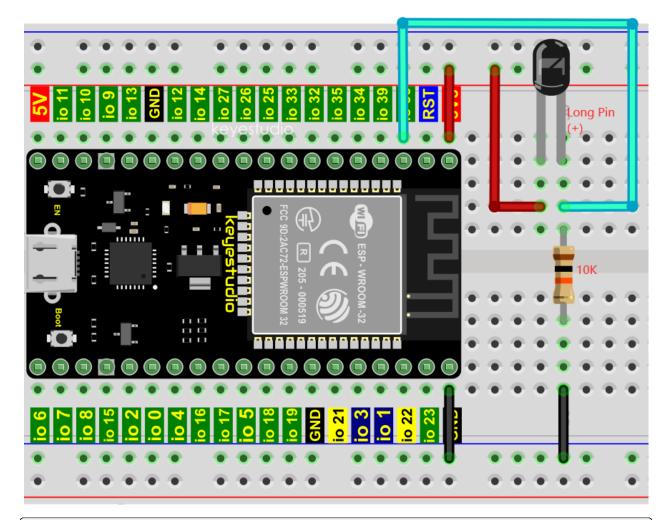
The short pin of the receiving triode is negative pole and the other long pin is positive pole. We should connect the short pin (negative) to 5V and the long pin (positive) to the analog pin, a resistor and GND. As shown in the figure below



**Note:** Since vulnerable to radio frequency radiation and temperature changes, the flame sensor should be kept away from heat sources like radiators, heaters and air conditioners, as well as direct irradiation of sunlight, headlights and incandescent light.

4 .Read the values of the flame sensor

We first use a simple code to read the ADC value, DAC value and voltage value of the flame sensor and print them out. Please refer to the wiring diagram below



```
/*
* Filename : Read Analog Value Of Flame Sensor
* Description : Basic usage of ADCDAC and Voltage
* Auther
           : http//www.keyestudio.com
*/
#define PIN_ANALOG_IN 36 //the pin of the Flame sensor
void setup() {
 Serial.begin(115200);
}
//In loop()the analogRead() function is used to obtain the ADC value, and then the map()_
\rightarrow function is used to convert the value into an 8-bit precision DAC value. The input and
\rightarrowoutput voltage are calculated according to the previous formula, and the information
\rightarrow is finally printed out.
void loop() {
 int adcVal = analogRead(PIN_ANALOG_IN);
 int dacVal = map(adcVal, 0, 4095, 0, 255);
 double voltage = adcVal / 4095.0 * 3.3;
 Serial.printf("ADC Val: %d, \t DAC Val: %d, \t Voltage: %.2fV\n", adcVal, dacVal, __
                                                                      (continues on next page)
```

د_	<pre>→voltage);</pre>
	delay(200);
}	
1	/*******************************

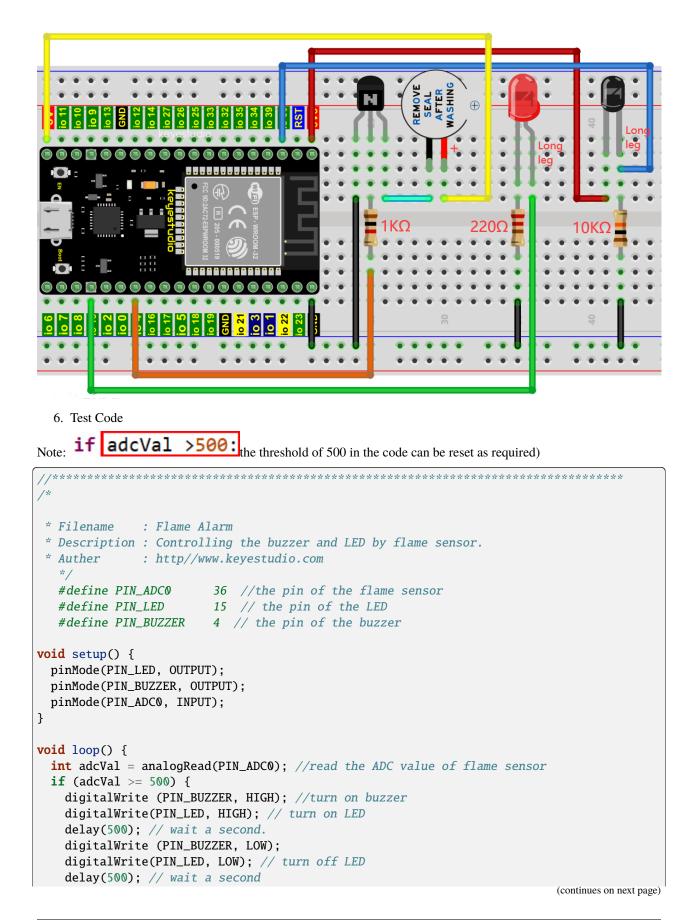
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable, open the serial monitor and set the baud rate to 115200 and press the reset button first.

You will see that the serial monitor window will print out the ADC value, DAC value and voltage value of the flame sensor. When the sensor is closed to fire, the ADC value,DAC value and voltage value will get greater. Conversely, the ADC value,DAC value and voltage value decrease.

		/dev/ttyUSB0	~ ^ X
			Send
ADC Val: 38, ADC Val: 443, ADC Val: 477, ADC Val: 490, ADC Val: 753, ADC Val: 928, ADC Val: 925, ADC Val: 1359, ADC Val: 1421, ADC Val: 1463, ADC Val: 1775,	DAC Val: 47, DAC Val: 58, DAC Val: 58, DAC Val: 85, DAC Val: 88,	Voltage: 0.03V Voltage: 0.03V Voltage: 0.36V Voltage: 0.38V Voltage: 0.39V Voltage: 0.61V Voltage: 0.75V Voltage: 0.75V Voltage: 1.10V Voltage: 1.15V Voltage: 1.18V Voltage: 1.43V	
<ul> <li>✓ Autoscroll □ S</li> </ul>	Show timestamp	Newline 🔹 11	5200 baud 💌 Clear output

#### 5. Wiring diagram of the flame alarm

Next, we will use a flame sensor, a buzzer, and a LED to make an interesting project, that is flame alarm. When flame is detected, the LED flashes and the buzzer alarms.



```
}
else
{
    digitalWrite(PIN_LED, LOW); //turn off LED
    digitalWrite (PIN_BUZZER, LOW); //turn off buzzer
    }
}
```

7. Test Result

Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that when the flame sensor detects the flame, the LED will flash and the buzzer will alarm, otherwise, the LED does not light up and the buzzer does not sound.

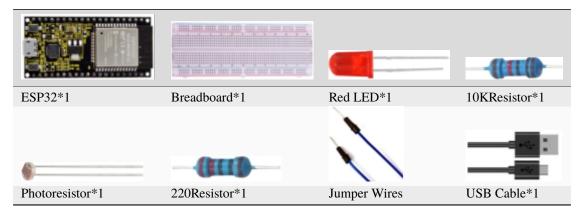
# 9.20 Project 20: Night Lamp

#### 1. Introduction

Sensors or components are ubiquitous in our daily life. For example, some public street lamps will automatically turn on at night and turn off during the day. In fact, this make use of a photosensitive element that senses the intensity of external ambient light. When the outdoor brightness decreases at night, the street lights will turn on automatically. In the daytime, the street lights will automatically turn off.

In this project, we use a ESP32 to control a LED to achieve the effect of the street light.

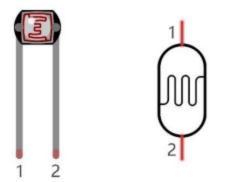
2. Components



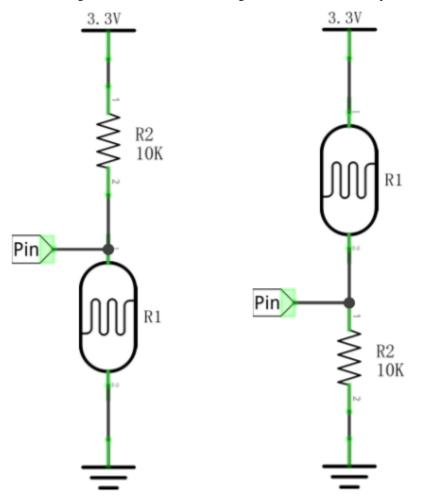
3. Component Knowledge



**Photoresistor :** It is a kind of photosensitive resistor, its principle is that the photoresistor surface receives brightness (light) to reduce the resistance, the resistance value will change with the detected intensity of the ambient light. With this characteristic, we can use the photoresistor to detect the light intensity. Photoresistor and its electronic symbol are as follows



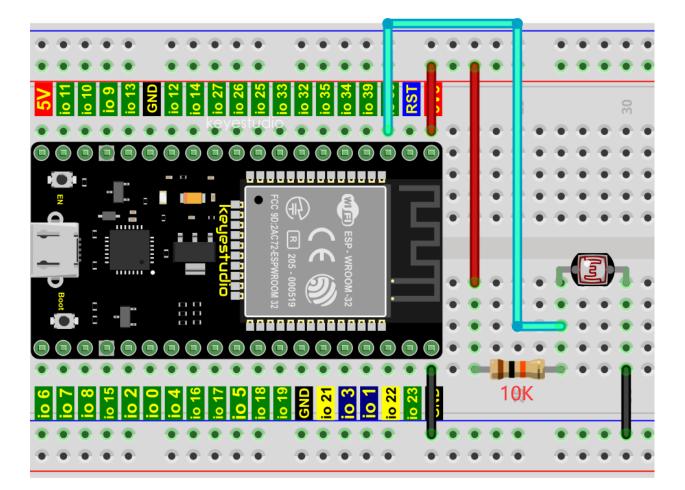
The following circuit is used to detect changes in resistance values of photoresistors



In the circuit above, when the resistance of the photoresistor changes due to the change of light intensity, the voltage between the photoresistor and resistor R2 will also change. Thus, the intensity of light can be obtained by measuring this voltage.

4. Read the values of the photoresistor

We first use a simple code to read the ADC value, DAC value and voltage value of the photoresistor and print them out. Please refer to the following wiring diagram



```
/*
* Filename : Read Photosensitive Analog Value
* Description : Basic usage of ADC
* Auther
            : http//www.keyestudio.com
*/
#define PIN_ANALOG_IN 36 //the pin of the photosensitive sensor
void setup() {
 Serial.begin(115200);
}
//In loop() the analogRead() function is used to obtain the ADC value, and then the map().
\rightarrow function is used to convert the value into an 8-bit precision DAC value. The input and
-output voltage are calculated according to the previous formula, and the information
\rightarrow is finally printed out.
void loop() {
 int adcVal = analogRead(PIN_ANALOG_IN);
 int dacVal = map(adcVal, 0, 4095, 0, 255);
 double voltage = adcVal / 4095.0 * 3.3;
 Serial.printf("ADC Val: %d, \t DAC Val: %d, \t Voltage: %.2fV\n", adcVal, dacVal, __
→voltage);
 delay(200);
```

(continues on next page)

```
}
```

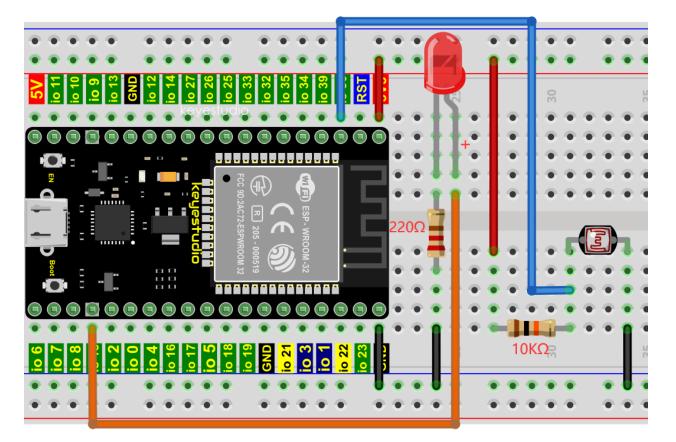
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable, open the serial monitor and set the baud rate to 115200 and press the reset button first.

You will see that the serial monitor window will print out the ADC valueDAC value and voltage value of the photoresistor. When the light intensity around the photoresistor is gradually reduced, the ADC value, DAC value and voltage value will gradually increase. On the contrary, the ADC value, DAC value and voltage value decrease gradually.

					/dev/tty	USB0			~ ^ X
									Send
ADC ADC ADC ADC ADC ADC ADC ADC ADC ADC	Val: 4 Val: 4 Val: 4 Val: 9 Val: 9 Val: 9 Val: 9 Val: 1 Val: 1 Val: 1 Val: 1	43, 38, 443, 477, 490, 753, 928, 925, 1359, 1421, 1463, 1775,	DAC Va DAC Va DAC Va DAC Va DAC Va DAC Va DAC Va DAC Va DAC Va DAC Va	l: 3, l: 2, l: 28, l: 30, l: 31, l: 31, l: 58, l: 58, l: 58, l: 88, l: 88, l: 91, l: 111	Voltage: 0.05V Voltage: 0.03V Voltage: 0.03V Voltage: 0.36V Voltage: 0.38V Voltage: 0.39V Voltage: 0.61V Voltage: 0.75V Voltage: 0.75V Voltage: 1.10V Voltage: 1.15V Voltage: 1.43V Voltage: 1.54V				Î
< <b>■</b>	Autoscr	oll 🗌 Sh	ow times	tamp		Newline	<ul> <li>▼ 11520</li> </ul>	00 baud 🔻	Clear output

# 5. Wiring diagram of the light-controlled lamp

Now we will make a light controlled lamp. The principle is the same as the small dimming lamp, that is, the ESP32 takes the analog values of the sensor, and then adjusts the brightness of the LED.



```
6. Test Code
```

```
/*
* Filename : Night Lamp
* Description : Controlling the brightness of LED by photosensitive sensor.
* Auther
         : http//www.keyestudio.com
  */
  #define PIN_ANALOG_IN 36 // the pin of the photosensitive sensor
  #define PIN_LED 15 // the pin of the LED
  #define CHAN
                     0
  #define LIGHT_MIN
                     372
  #define LIGHT_MAX
                     2048
  void setup() {
   ledcSetup(CHAN, 1000, 12);
   ledcAttachPin(PIN_LED, CHAN);
  }
void loop() {
 int adcVal = analogRead(PIN_ANALOG_IN); //read adc
 int pwmVal = map(constrain(adcVal, LIGHT_MIN, LIGHT_MAX), LIGHT_MIN, LIGHT_MAX, 0,...
→4095); // adcVal re-map to pwmVal
 ledcWrite(CHAN, pwmVal); // set the pulse width.
 delay(10);
}
```

7. Test Result

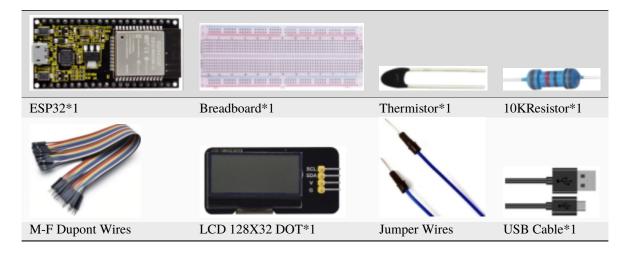
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that when the intensity of light around the photoresistor is reduced, the LED will be bright, on the contrary, the LED will be dim.

# 9.21 Project 21Temperature Instrument

# 1.Introduction

Thermistor is a kind of resistor whose resistance depends on temperature changes, which is widely used in gardening, home alarm systems and other devices. Therefore, we can use the features to make a temperature instrument.

2.Components



# 3.Component Knowledge

**Thermistor:** It is a temperature sensitive resistor. When it senses a change in temperature, the resistance of the thermistor will change. We can take advantage of this characteristic to detect temperature intensity. The thermistor and its electronic symbol are shown below:



The relationship between resistance and temperature of the thermistor is

# $Rt = R * EXP[B * \left(\frac{1}{T2} - \frac{1}{T1}\right)]$

Rt is the thermistor resistance under T2 temperature;

**R** is the nominal resistance of thermistor under T1 temperature;

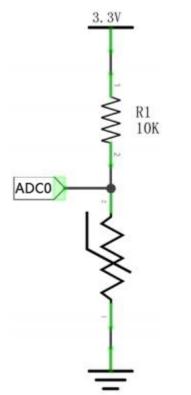
**EXP**[**n**] is nth power of e;

**B** is temperature index;

**T1, T2** is Kelvin temperature (absolute temperature). Kelvin temperature=273.15 + Celsius temperature.

Parameters : B=3950, R=10k, T1=25.

The circuit connection method of the thermistor is similar to the photoresistor, as shown below



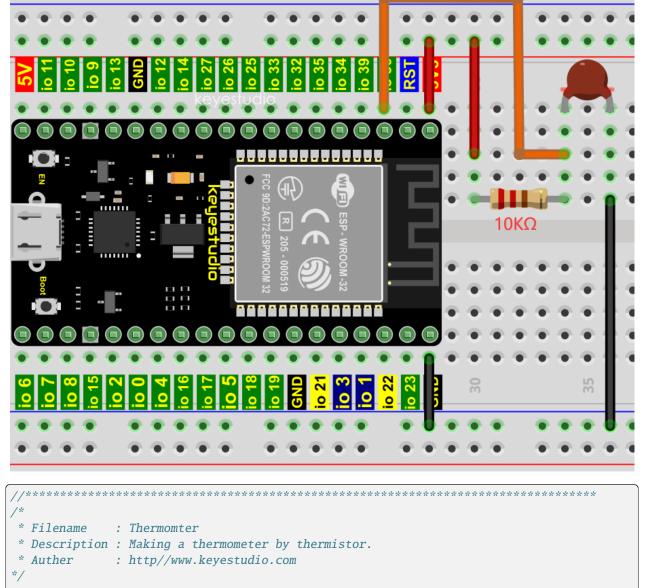
We can use the value measured by the ADC converter to obtain the resistance of thermistor, and then we can use the formula to obtain the temperature value.

Therefore, the temperature formula can be derived as:

 $\frac{Rt}{R}$  $T2 = 1 / \left( \frac{1}{T1} + \right)$ 

4.Read the value of the thermistor

First we will learn the thermistor reading the current ADC value, voltage value and temperature value and print them out. Please connect the wirings according to the wiring diagram below



(continues on next page)

```
#define PIN_ANALOG_IN
                    36
void setup() {
 Serial.begin(115200);
}
void loop() {
 int adcValue = analogRead(PIN_ANALOG_IN);
                                                         //read ADC pin
 double voltage = (float)adcValue / 4095.0 * 3.3;
                                                         // calculate voltage
 double Rt = 10 * voltage / (3.3 - voltage);
                                                         //calculate resistance...
→ value of thermistor
 double tempK = 1 / (1 / (273.15 + 25) + log(Rt / 10) / 3950.0); //calculate_
→temperature (Kelvin)
 double tempC = tempK - 273.15;
                                                          //calculate
→temperature (Celsius)
 Serial.printf("ADC value : %d,\tVoltage : %.2fV, \tTemperature : %.2fC\n", adcValue,
→voltage, tempC);
 delay(1000);
}
```

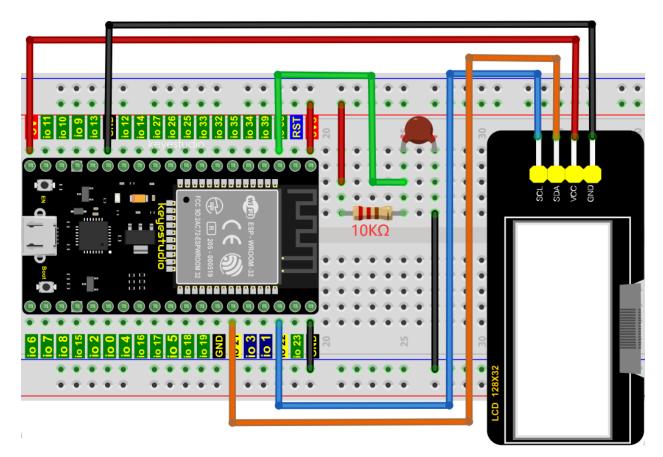
Upload the code to the ESP32, power up with a USB cable, open serial monitor and set baud rate to 115200. Press the rest button of the ESP32 board, then you will view ADC value, voltage value and temperature value of the thermistor displayed.

Pinch the thermistor a while, the temperature value will increase.

		/dev	ı/ttyUSB0	~ ^ X
				Send
	value : 1872,	Voltage : 1.51V,	Temperature : 28.92C	<b>^</b>
	,	Voltage : 1.67V,	Temperature : 24.35C	
	,	Voltage : 1.48V,		
		Voltage : 1.48V,	-	
ADC	value : 1861,	Voltage : 1.50V,	Temperature : 29.17C	
ADC	value : 1835,	Voltage : 1.48V,	Temperature : 29.76C	
ADC	value : 1802,	Voltage : 1.45V,	Temperature : 30.52C	
ADC	value : 1776,	Voltage : 1.43V,	Temperature : 31.13C	
ADC	value : 1834,	Voltage : 1.48V,	Temperature : 29.79C	
ADC	value : 1745,	Voltage : 1.41V,	Temperature : 31.85C	
ADC	value : 1808,	Voltage : 1.46V,	Temperature : 30.38C	
ADC	value : 1843,	Voltage : 1.49V,	Temperature : 29.58C	
ADC	value : 1870,	Voltage : 1.51V,	Temperature : 28.96C	
ADC	value : 2160,	Voltage : 1.74V,	Temperature : 22.54C	
		Voltage : 1.50V,	Temperature : 29.21C	
			•	•
•	Autoscroll 🗌 Show	timestamp	Newline 🔹 115200 baud	▼ Clear output

5. Wiring diagram of the temperature instrument

(continued from previous page)



6.Adding the lcd128\_32\_io library

Please add the lcd128\_32\_io library first

📮 🗧 2. Raspberry Pi System

Home Share View

👻 🛧 📙 « 3. Arduino Tutorial 🔸 2. Raspberry Pi System		~
Name	Date modified	Туре
1. Development Environment Configuration	7/8/2022 3:03 PM	File folder
2. Projects	7/7/2022 12:05 PM	File folder
3. Arduino Libraries	7/7/2022 1:14 PM	File folder

7.Test Code

```
//*
* Filename : Temperature Instrument
* Description : LCD displays the temperature of thermistor.
* Auther : http//www.keyestudio.com
*/
#include "lcd128_32_io.h"
#define PIN_ANALOG_IN 36
(continues on next page)
```

```
lcd lcd(21, 22); //Create lCD128 *32 pinsda->21 scl->22
void setup() {
 lcd.Init(); //initialize
 lcd.Clear(); //clear
}
char string[10];
void loop() {
 int adcValue = analogRead(PIN_ANALOG_IN);
                                                             //read ADC pin
 double voltage = (float)adcValue / 4095.0 * 3.3;
                                                             // calculate voltage
 double Rt = 10 * voltage / (3.3 - voltage);
                                                             //calculate resistance_
\rightarrow value of thermistor
 double tempK = 1 / (1 / (273.15 + 25) + log(Rt / 10) / 3950.0); //calculate_
→temperature (Kelvin)
 double tempC = tempK - 273.15;
                                                             //calculate
→temperature (Celsius)
 lcd.Cursor(0,0); //Set display position
 lcd.Display("Voltage:"); //Setting the display
 lcd.Cursor(0,8);
 lcd.DisplayNum(voltage);
 lcd.Cursor(0,11);
 lcd.Display("V");
 lcd.Cursor(2, ◊);
 lcd.Display("tempC:");
 lcd.Cursor(2,8);
 lcd.DisplayNum(tempC);
 lcd.Cursor(2,11);
 lcd.Display("C");
 delay(200);
}
```

# 8.Test Result

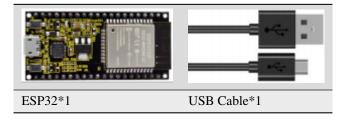
Compile and upload the code to ESP32, after the code is uploaded successfully, power up with a USB cable and you will see that the LCD 128X32 DOT displays the voltage value of the thermistor and the temperature value in the current environment.

# 9.22 Project 22Bluetooth

This chapter mainly introduces how to make simple data transmission through Bluetooth of ESP32 and mobile phones. Project 22.1 is classic Bluetooth while project 22.2 is Bluetooth control LED.

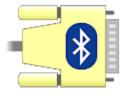
# 9.22.1 Project 22.1Classic Bluetooth

1.Components



In this tutorial we need to use a Bluetooth APP called serial Bluetooth terminal to assist in the experiment.

Download link: https://www.appsapk.com/serial-bluetooth-terminal/ .



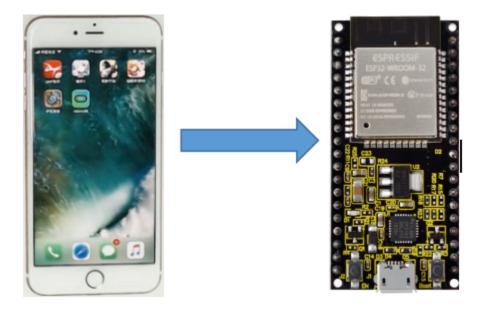
2.Component Knowledge

Bluetooth is a short-distance communication system that can be divided into two types, namely low power bluetooth (BLE) and classic bluetooth. There are two modes for simple data transfer: master mode and slave mode.

**Master Mode**: In this mode, work is done on the master device and can be connected to the slave device. When the device initiates a connection request in the main mode, information such as the address and pairing password of other bluetooth devices are required. Once paired, you can connect directly to them.

**Slave Mode**: A bluetooth module in the slave mode can only accept connection requests from the host, but cannot initiate connection requests. After being connected to a host device, it can send and receive data through the host device.

Bluetooth devices can interact with each other, when they interact, the bluetooth device in the main mode searches for nearby devices. While a connection is established, they can exchange data. For example, when a mobile phone exchanges data with ESP32, the mobile phone is usually in master mode and the ESP32 is in slave mode.



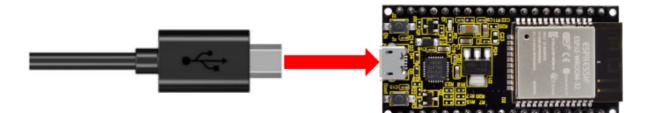
Master

Slave

# Master Slave

3.Wiring Diagram

We can use a USB cable to connect ESP32 mainboard to the USB port on the Raspberry P.



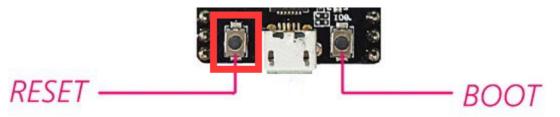
4.Test Code

Project_22.1_Classic_Bluetooth   Arduino 1.8.19 🗸 🗸 🗙
<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp
Project_22.1_Classic_Bluetooth
/**************************************
<pre>/* Filename : Classic BluetoothSerialToSerialBT  * Description : ESP32 communicates with the phone by bluetooth and print phone's da  * Auther : http//www.keyestudio.com  */</pre>
<pre>#include "BluetoothSerial.h"</pre>
<pre>BluetoothSerial SerialBT; String buffer; void setup() { Serial.begin(115200); SerialBT.begin("ESP32test"); //Bluetooth device name Serial.println("\nThe device started, now you can pair it with bluetooth!"); }</pre>
<pre>void loop() {</pre>
Invalid library found in /home/pi/Downloads/arduino-1.8.19/libraries/examples: no head SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0
//************************************
<pre>* Filename : Classic BluetoothSerialToSerialBT * Description : ESP32 communicates with the phone by bluetooth and print phone's data</pre>
*/ #include "BluetoothSerial.h"
<pre>BluetoothSerial SerialBT; String buffer; void setup() { Serial.begin(115200); SerialBT.begin("ESP32test"); //Bluetooth device name Serial.println("\nThe device started, now you can pair it with bluetooth!"); }</pre>
<pre>void loop() {     if (Serial.available()) {         SerialBT.write(Serial.read());     } }</pre>
<pre>if (SerialBT.available()) {</pre>

<pre>Serial.write(SerialBT.read());</pre>	
}	
<pre>delay(20);</pre>	
}	
//*************************************	

# 5.Test Result

Compile and upload the code to the ESP32. After uploading successfullywe will use a USB cable to power on. Open the serial monitor and set the baud rate to **115200 then press the reset button first**. When you see the serial monitor prints out the character string as below, it indicates that the Bluetooth of ESP32 is ready and waiting for connection with a phone. (If open the serial monitor and set the baud rate to 115200, the information is not displayed, please press the RESET button of the ESP32)

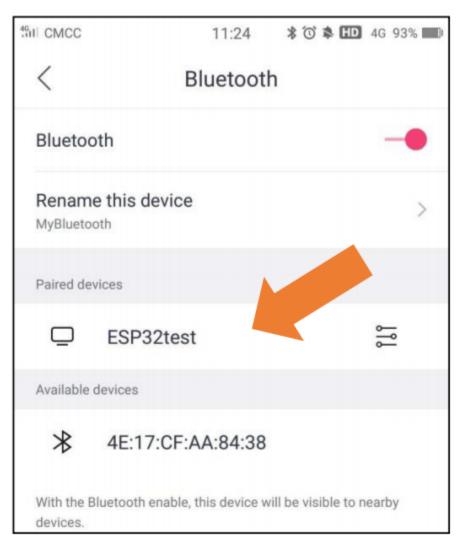


/dev/ttyUSB0			``	/ ^ X
				Send
<pre>rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT) configsip: 0, SPIWP:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x mode:DI0, clock div:1 load:0x3fff0018,len:4 load:0x3fff001c,len:1216 ho 0 tail 12 room 4 load:0x40078000,len:10944 load:0x40080400,len:6388 entry 0x400806b4 E (137) psram: PSRAM ID read error: 0xffffffff</pre>	(00,wp_dr	v:0x00		
The device started, now you can pair it with bluetooth!				Į
Autoscroll Show timestamp Newline	•	115200 baud	▼ Clea	r output

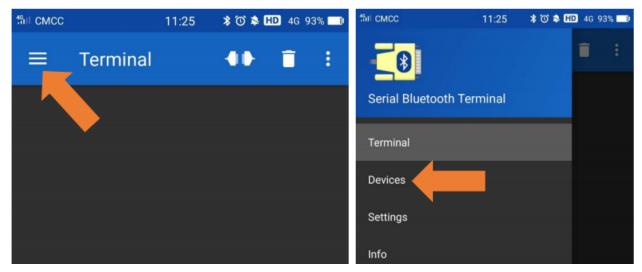
Make sure that the Bluetooth of your phone has been turned on and "Serial Bluetooth Terminal" has been installed.



Click"Search"search for the nearby Bluetooth and select to connect the"ESP32 test".



Turn on software APP, click the left of the terminal. Select "Devices" .



Select ESP32test in classic Bluetooth mode, and a successful connecting prompt will appear as shown below.

← Devices	٠		
BLUETOOTH CLASSIC	BLUETOOTH LE	11:25:41.094 Connecting to ESP32test 11:25:41.195 Connected	
ESP32test C4:4F:33:22:B6:3B		M1 M2 M3 M4 M5	ľ

Data can be transferred between your phone and the Raspberry Pi via ESP32 now.

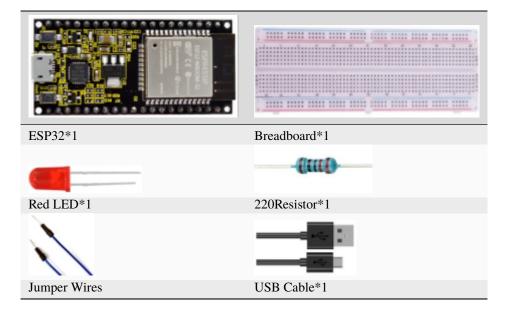
Send"Hello!", When the Raspberry Pi receives it, which will reply with "Hi!".

/dev/ttyUSB0			×
Fil		Sen	nd
configsip: 0, SPIWP:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00			•
mode:DIO, clock div:1 load:0x3fff0018,len:4 load:0x3fff001c,len:1216 ho 0 tail 12 room 4 load:0x40078000,len:10944 load:0x40080400,len:6388 entry 0x400806b4 E (137) psram: PSRAM ID read error: 0xfffffff			
The device started, now you can pair it with bluetooth! Hello!			Į
✓ Autoscroll  Show timestamp  Newline  115200 baud  Clear	ar c	outp	ut

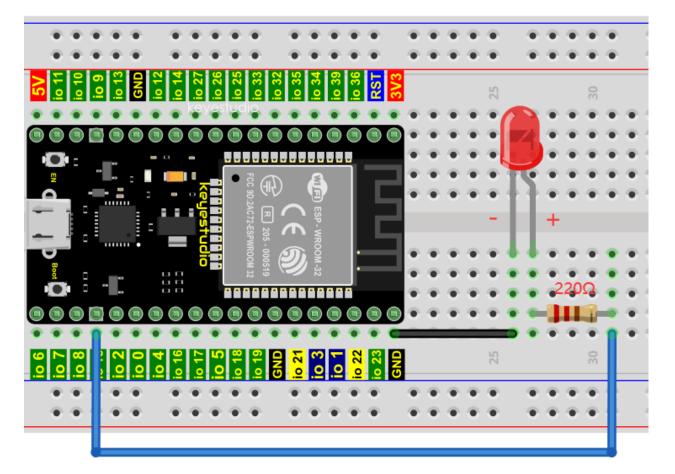
4Gul CMCC		11	:26 🖇	(i) 🛸 🖪	D 4G 92%	0
=	Termi	inal	-	<b>(1)</b> -	<b>i</b> :	
11:25:41	.094 Conn .195 Conn .913 Hello .759 Hi!	ected	ESP32tes	st		
M1	M2	М3	M4	M5	5 M6	
					>	

# 9.22.2 Project 22.2Bluetooth Control LED

1.Components



2.Wiring Diagram



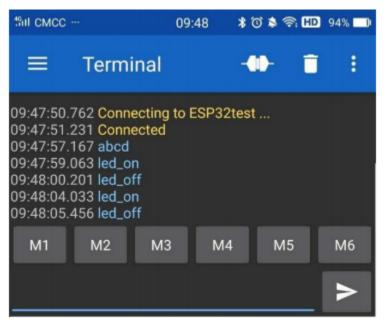
3.Test Code

Project_22.2_Bluetooth_Control_LED   Arduino 1.8.19 ~ ^ ×
Eile Edit Sketch Tools Help
Project_22.2_Bluetooth_Control_LED
/**************************************
<pre>* Filename : Bluetooth Control LED * Description : The phone controls esp32's led via bluetooth. When the phone sends "LED_on," ESP32's LED lights turn on. When the phone sends "LED_off," ESP32's LED lights turn off. * Auther : http//www.keyestudio.com */</pre>
<pre>#include "BluetoothSerial.h" #include "string.h" #define LED 15</pre>
<pre>BluetoothSerial SerialBT; char buffer[20]; static int count = 0;</pre>
<pre>void setup() {     pinMode(LED, OUTPUT);     SerialBT.begin("ESP32test"); //Bluetooth device name </pre>
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0
//************************************
<ul> <li>* Filename : Bluetooth Control LED</li> <li>* Description : The phone controls esp32's led via bluetooth.</li> <li>When the phone sends "LED_on," ESP32's LED lights turn on.</li> </ul>
<pre>When the phone sends "LED_off," ESP32's LED lights turn off. * Auther : http//www.keyestudio.com</pre>
<pre>*/ #include "BluetoothSerial.h" #include "string.h"</pre>
<pre>#define LED 15 BluetoothSerial SerialBT;</pre>
<pre>char buffer[20];</pre>
<pre>static int count = 0; void setup() {</pre>
<pre>pinMode(LED, OUTPUT); SerialBT.begin("ESP32test"); //Bluetooth device name Serial.begin(115200);</pre>
<pre>Serial.println("\nThe device started, now you can pair it with bluetooth!"); }</pre>
<pre>void loop() {</pre>
(continues on next page)

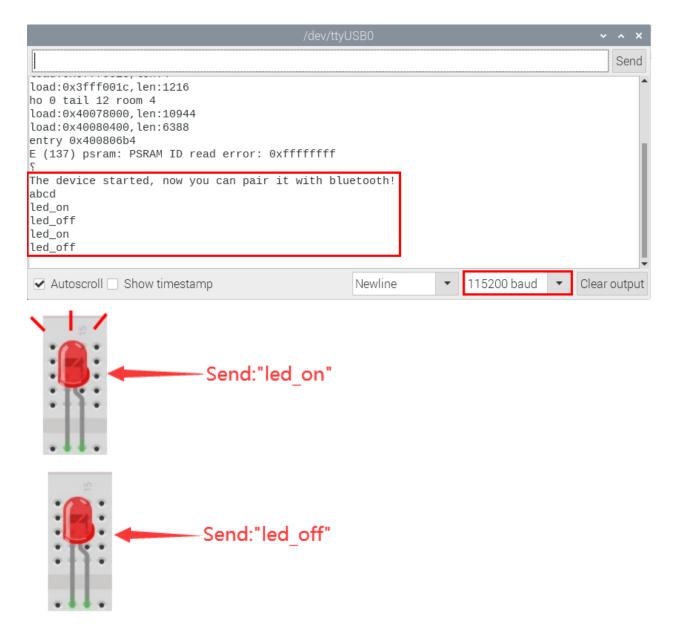
```
while(SerialBT.available())
 {
   buffer[count] = SerialBT.read();
   count++;
 }
 if(count>0){
   Serial.print(buffer);
   if(strncmp(buffer, "led_on", 6)==0){
     digitalWrite(LED,HIGH);
   }
   if(strncmp(buffer,"led_off",7)==0){
     digitalWrite(LED,LOW);
   }
   count=0;
   memset(buffer,0,20);
 }
}
              /****
```

### 4.Test Result

Compile and upload the code to the ESP32. The APP operation is the same as the project 22.1. To make the external LED on and off, simply change the sending content to "led\_on" and "led\_off". Moving the APP to send data:



The serial monitor will display as follows:



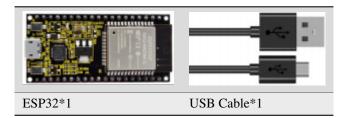
**Note:** If the "led-on 'or" led-off " is not sent, the status of the LED will not change. If the LED is on, it remains on when irrelevant content is received; Conversely, if the LED is off, it continues to be off when irrelevant content is received.

# 9.23 Project 23WiFi Station Mode

#### 1.Introduction

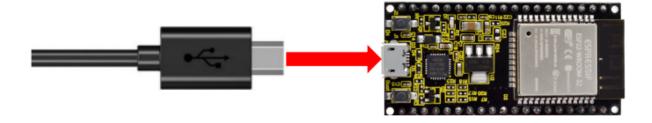
ESP32 has three different WiFi operating modes : Station modeAP mode and AP+Station mode. All WiFi programming projects must be configured with WiFi operating mode before using, otherwise WiFi cannot be used. In this project, we are going to learn the WiFi Station mode of the ESP32.

2.Components



### 3.Wiring Diagram

Plug the ESP32 to the USB port of the Raspberry Pi



#### 4.Component Knowledge

#### Station mode

When setting Station mode, the ESP32 is taken as a WiFi client. It can connect to the router network and communicate with other devices on the router via a WiFi connection. As shown in the figure below, the PC and the router have been connected. If the ESP32 wants to communicate with the PC, the PC and the router need to be connected.

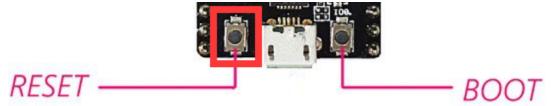


5.Test Code

Project_23_WiFi_Station_Mode   Arduino 1.8.19	~ ^ X
<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp	
	<mark>,⊘</mark> -
Project_23_WiFi_Station_Mode	
<pre>//***********************************</pre>	*******
<pre>//Enter correct router name and password. const char *ssid_Router = "ChinaNet-2.4G-0DF0"; //Enter the router name const char *password_Router = "ChinaNet@233"; //Enter the router password</pre>	- 1
<pre>void setup(){    Serial.begin(115200);    delay(2000);    Serial.println("Setup start");    WiFi.begin(ssid Router, password Router);//Set ESP32 in Station mode and content </pre>	onnect_i∙
	ttyUSB0

Since WiFi names and passwords vary from place to place, thereby users need to enter the correct WiFi names and passwords in the box shown below before running the program code.

Project_23_WiFi_Station_Mode   Arduino 1.8.19 🔹 🔺 🗙
<u>File Edit Sketch Tools H</u> elp
Project_23_WiFi_Station_Mode
//*************************************
* Filename : WiFi Station
* Description : Connect to your router using ESP32 * Auther : http// <u>www.keyestudio.com</u> Enter the correct Router
*/ #include <wifi.h> //Include the WiFi Library header file of ESP32.</wifi.h>
<pre>//Enter correct router name and password const char *ssid_Router = "ChinaNet-2.4G-0DF0"; //Enter the router name const char *password_Router = "ChinaNet@233"; //Enter the router password</pre>
<pre>void setup(){    Serial.begin(115200);    lu (2022)</pre>
<pre>delay(2000); Serial.println("Setup start");</pre>
WiFi.begin(ssid Router, password Router);//Set ESP32 in Station mode and connect iv
Invalid library found in /home/pi/Downloads/arduino-1.8.19/libraries/examples: no hea
SP32 Wrover Module Default 4MB with spiffs (1 2MB APP/1 5MB SPIFES) OID 80MHz 921600 None on /dev/ttyl ISB0
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0 ///**********************************
//************************************
//************************************
<pre>//***********************************</pre>
<pre>//* /* /* /* /* /* /* /* /* /* /* /* /*</pre>



### 6.Test Result

After making sure the router name and password are entered correctly, compile and upload the code to ESP32, open serial monitor and set baud rate to 115200 then press the reset button first. When ESP32 successfully connects to "ssid\_Router", serial monitor will print out the IP address, then monitor will display as follows: (If open the serial monitor and set the baud rate to 115200 and the information is not displayed, please press the RESET button of the ESP32).

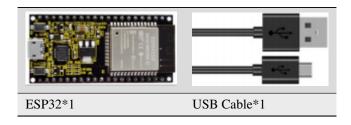
	/dev/ttyl	JSB0						×
							Ser	nd
load:0x3fff0018,len:4 load:0x3fff001c,len:1216 ho 0 tail 12 room 4 load:0x40078000,len:10944								•
load:0x40080400,len:6388 entry 0x400806b4 E (52) psram: PSRAM ID read error Setup start	0xffffffff							
Connecting to ChinaNet-2.4G-0DF0								I
Connected, IP address: 192.168.0.154 Setup End								ļ
Autoscroll 🗌 Show timestamp		Newline	•	115200 baud	•	Clear	outp	ut

# 9.24 Project 24WiFi AP Mode

# 1.Introduction

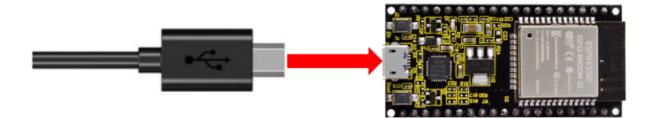
In this project, we are going to learn the WiFi AP mode of the ESP32.

## 2.Components



## 3.Wiring Diagram

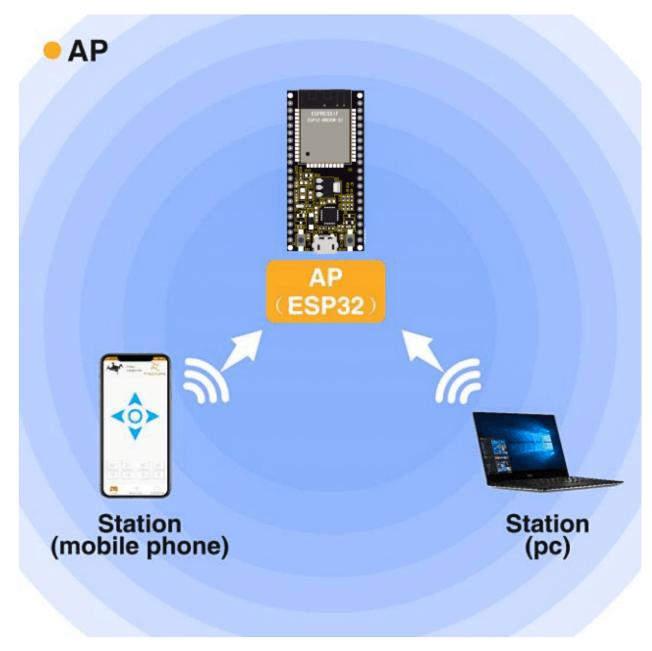
Plug the ESP32 mainboard to the USB port of the Raspberry Pi



# 4.Component Knowledge

# **AP Mode:**

When setting AP mode, a hotspot network will be created, waiting for other WiFi devices to connect. As shown below;



5.Test Code

Project_24_WiFi_AP_Mode   Arduino 1.8.19	~ ^ X
<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp	
	<mark>.⊘</mark> .
Project_24_WiFi_AP_Mode	
//************************************	*******
<pre>#include <wifi.h> //Include the WiFi Library header file of ESP32.</wifi.h></pre>	
<pre>const char *ssid_AP = "ESP32_WiFi"; //Enter the router name const char *password_AP = "12345678"; //Enter the router password</pre>	
<pre>IPAddress local_IP(192,168,1,108);//Set the IP address of ESP32 itself IPAddress gateway(192,168,1,1); //Set the gateway of ESP32 itself IPAddress subnet(255,255,255,0); //Set the subnet mask for ESP32 itself</pre>	
<pre>void setup(){     Serial.begin(115200); </pre>	,*
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/	ttyUSB0

Before running the code , you can make any changes to the ESP32 AP name and password in the box as shown below, but in a default circumstance, it doesn't need to modify.

```
File Edit Sketch Tools Help
   Project 24 WiFi AP Mode
                                               * * * * * * * * * * * * * * * * * * * *
  * Filename : WiFi AP
 * Auther : http://www.keyestudio.com password for ESP32 AP.
 */
#include <WiFi.h> //Include the WiFi Lib ary header file of ESP32.
const char *ssid_AP
                       = "ESP32_WiFi"; //Enter the router name
const char *password_AP = "12345678"; //Enter the router password
IPAddress local_IP(192, 168, 1, 108);//Set the IP address of ESP32 itself
IPAddress gateway(192,168,1,1); //Set the gateway of ESP32 itself
IPAddress subnet(255,255,255,0); //Set the subnet mask for ESP32 itself
void setup(){
  Serial.begin(115200);
 4.1
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0
/*
* Filename : WiFi AP
* Description : Set ESP32 to open an access point
* Auther : http//www.keyestudio.com
*/
#include <WiFi.h> //Include the WiFi Library header file of ESP32.
                     = "ESP32_WiFi"; //Enter the router name
const char *ssid_AP
const char *password_AP = "12345678"; //Enter the router password
IPAddress local_IP(192,168,1,108);//Set the IP address of ESP32 itself
IPAddress gateway(192,168,1,1); //Set the gateway of ESP32 itself
IPAddress subnet(255,255,255,0); //Set the subnet mask for ESP32 itself
void setup(){
 Serial.begin(115200);
 delay(2000);
 Serial.println("Setting soft-AP configuration ... ");
 WiFi.disconnect();
 WiFi.mode(WIFI_AP);
 Serial.println(WiFi.softAPConfig(local_IP, gateway, subnet) ? "Ready" : "Failed!");
```

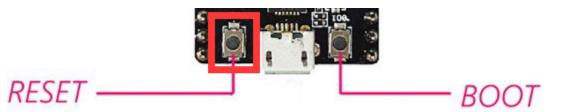
```
(continues on next page)
```

```
Serial.println("Setting soft-AP ... ");
boolean result = WiFi.softAP(ssid_AP, password_AP);
if(result){
   Serial.println("Ready");
   Serial.println(String("Soft-AP IP address = ") + WiFi.softAPIP().toString());
   Serial.println(String("MAC address = ") + WiFi.softAPmacAddress().c_str());
}else{
   Serial.println("Failed!");
}
Serial.println("Setup End");
}
void loop() {
}
```

6.Test Result

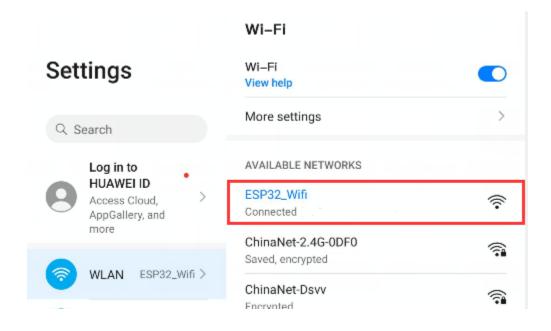
Enter the ESP32 AP name and password correctly, compile and upload the code to ESP32, open the serial monitor and set the baud rate to **115200 and press the reset button first**, then monitor will display as follows:

(If open the serial monitor and set the baud rate to 115200, the information is not displayed, please press the RESET button of the ESP32)



	/dev/ttyUSB0		×
		Se	end
F (53) psram: PSRAM ID read error: Setting soft-AP configuration Setting soft-AP Ready Soft-AP IP address = 192.168.4.1 MAC address = 58:BF:25:8A:2F:E1	0xfffffff		
Setting Station configuration Connecting to ChinaNet-2.4G-0DF0  Connected, IP address: 192.168.0.154 Setup End			
Autoscroll 🗌 Show timestamp	Newline   I15200 baud  Clea	r out	• put

When observing the printed information of the serial monitor, turn on the WiFi scanning function of the mobile phone, you can see the ssid\_AP on ESP32, which is dubbed "ESP32\_Wifi" in this code. You can connect to it either by typing the password "12345678" or by modifying the code to change its AP name and password.

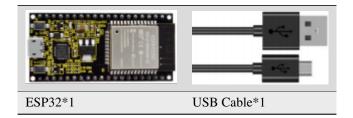


# 9.25 Project 25WiFi Station+AP Mode

#### 1.Introduction

In this project, we are going to learn the AP+Station mode of the ESP32.

2.Components



#### 3.Wiring Diagram



Plug the ESP32 mainboard to the USB port of the Raspberry Pi

4.Component Knowledge

#### **AP+Station mode:**

In addition to the AP mode and the Station mode, **AP+Station mode** can be used at the same time. Turn on the Station mode of the ESP32, connect it to the router network, and it can communicate with the Internet through the router. Then turn on the AP mode to create a hotspot network. Other WiFi devices can be connected to the router network or the hotspot network to communicate with the ESP32.

5.Test Code

Project_25_WiFi_Station_AP_Mode   Arduino 1.8.19 🗸 🗸	~ X
<u>File Edit Sketch Tools H</u> elp	
	<b>1</b> 20
Project_25_WiFi_Station_AP_Mode	
//*************************************	* * * *
<pre>* Filename : WiFi AP+Station * Description : ESP32 connects to the user's router, turning on an access point * Auther : http//<u>www.keyestudio.com</u></pre>	
*/ #include <wifi.h></wifi.h>	- 1
<pre>const char *ssid_Router = "ChinaNet-2.4G-0DF0"; //Enter the router name const char *password_Router = "ChinaNet@233"; //Enter the router password const char *ssid_AP = "ESP32_WiFi"; //Enter the router name const char *password_AP = "12345678"; //Enter the router password</pre>	
<pre>void setup(){    Serial.begin(115200);    Serial.println("Setting soft-AP configuration ");    WiFi.disconnect();</pre>	
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUS	5B0

Before running the code, you need to modify the ssid\_Router, password\_Routerssid\_AP and password\_AP, as shown in the box below:

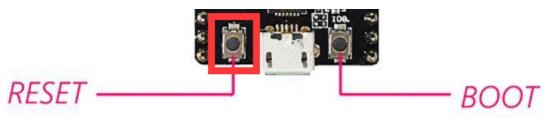
Project_25_WiFi_Station_AP_Mode   Arduino 1.8.19 🔹 🔺 🗙
<u>F</u> ile <u>E</u> dit <u>S</u> ketch <u>T</u> ools <u>H</u> elp
Project_25_WiFi_Station_AP_Mode
//*************************************
* Filename : WiFi AP+Station
* Description : ESP32 connects to the user's router, turning on an access point * Auther : http://www.keyestudio.com Please enter the correct names and
*/ nasswords of Pouter and AP
#include <wifi.h></wifi.h>
<pre>const char *ssid_Router = "ChinaNet-2.4G-0DF0"; //Enter the router name const char *password_Router = "ChinaNet@233"; //Enter the router password const char *ssid_AP = "ESP32_WiFi"; //Enter the router name const char *password_AP = "12345678"; //Enter the router password</pre>
<pre>void setup(){    Serial.begin(115200);    Serial.println("Setting soft-AP configuration ");    WiFi.disconnect(); </pre>
····· · · · · · · · · · · · · · · · ·
SP32 Wrover Module, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921600, None on /dev/ttyUSB0
SES2 Wrover Module, Derault 4MB with spins (1.2MB APP)1.5MB SPin 37, QIO, 50Min2, 921000, None on Jaevitty0580
//************************************
<pre>/* * Filename : WiFi AP+Station</pre>
<pre>* Description : ESP32 connects to the user's router, turning on an access point * Auther : http//www.keyestudio.com</pre>
*/
<pre>#include <wifi.h></wifi.h></pre>
<pre>const char *ssid_Router = "ChinaNet-2.4G-0DF0"; //Enter the router name</pre>
<pre>const char *password_Router = "ChinaNet@233"; //Enter the router password const char *ssid_AP = "ESP32_WiFi"; //Enter the router name</pre>
const char *password_AP = "12345678"; //Enter the router password
<pre>void setup(){    Serial.begin(115200);</pre>
Serial.println("Setting soft-AP configuration ");
WiFi.disconnect(); WiFi.mode(WIFI_AP);
<pre>Serial.println("Setting soft-AP ");</pre>
<pre>boolean result = WiFi.softAP(ssid_AP, password_AP); if(result){</pre>
Serial.println("Ready");

(continues on next page)



#### 6.Test Result

Ensure that the code in the program has been modified correctly, compile and upload the code to the ESP32. After uploading successfullywe will use a USB cable to power on. Open the serial monitor and set the baud rate to 115200 and press the reset button, then monitor will display as follows: (If open the serial monitor and set the baud rate to 115200 and the information is not displayed, please press the RESET button of the ESP32)



	/dev/ttyUSB0	~ ^ X
		Send
F (53) psram: PSRAM ID read error: Setting soft-AP configuration Setting soft-AP Ready Soft-AP IP address = 192.168.4.1 MAC address = 58:BF:25:8A:2F:E1 Setting Station configuration Connecting to ChinaNet-2.4G-0DF0  Connected, IP address: 192.168.0.154 Setup End	0xfffffff	
Autoscroll 🗌 Show timestamp	Newline   I15200 baud  (	Clear output

Open the WiFi scanning function of the mobile phone, you can see the ssid\_AP.

	Wi–Fi	
Settings	Wi-Fi View help	
Q Search	More settings	>
Log in to HUAWEI ID Access Cloud, AppGallery, and	AVAILABLE NETWORKS	
	ESP32_Wifi Connected	Ŷ
more	ChinaNet-2.4G-0DF0 Saved, encrypted	
WLAN ESP32_Wifi >	ChinaNet-Dsvv	1
	Encrynted	•