



ACEBOTT

Robot Arm Tutorial

Perface

Our Company

ACEBOTT STEM Education Tech Co.,Ltd

Founded in China's Silicon Valley in 2013, ACEBOTT is a STEM education solution leader. We have a team of 150 individuals, including members from research and development, sales, and logistics. Our goal is to provide high-quality STEM education products and services to our customers. We are working together with STEM education experts and our business partners to produce successful STE products together. Our self-owned factory also provides OEM services for our clients, including logo customization on product packaging and PCB.

Our Tutorial

This course and robot arm learning kit are designed for children and teenagers aged 8 and above. It aims to provide a deeper understanding of the ESP32 controller board, robot arm knowledge, and electronic hardware. If you want to learn about robot arms, this kit offers knowledge and operational steps to help you build your own robot arm.

Through this kit, you can:

1. Learn how to effectively use the ESP32 controller board, including downloading code, understanding its features, and coding in the Arduino IDE.
2. Gain a solid programming foundation in the basics of the C languages, as ESP32 utilizes the simplified C/C++ programming languages to control circuits and sensors.
3. Explore the working principles of the servo module and understand the collaborative work of multiple servos in the robot arm project.
4. Enhance your maker skills by building your own robot arm using the ACEBOTT kit through step-by-step tutorials.
5. Implement basic functions such as joystick control, teach learning, web control, and app control in the robot arm project.

6. Gain a comprehensive understanding of robot arm concepts, preparing for more advanced learning in the future.

Overall, the ACEBOTT robot arm is a learning kit designed specifically for beginners and is based on the ESP32. Using this kit, users can gain a comprehensive understanding of the controller board and servos in a robot arm. By following the tutorials provided in the kit, students of different age groups can acquire valuable knowledge about robot arms and successfully build their own robot arm projects.

Customer service

ACEBOTT is a dynamic and fast-growing STEM education technology company that strives to offer excellent products and quality services that meet your expectations.

We value your feedback and encourage you to drop us a line at support@acebott.com with any comments or suggestions you may have.

Our experienced engineers are dedicated to promptly addressing any problems or questions you may have about our products. We guarantee a response within 24 hours during business days.

Follow Us

Scan the QR codes to Follow Us for troubleshooting & the latest news.

We have a very large community that is very helpful for troubleshooting and we also have a support team at the ready to answer any questions.



ACEBOTT FB Group QR Code



YouTube QR Code

Contents

Lesson 1 Hardware Understanding and Software Installation	1
I .Hardware Understanding	2
II .Software Installation	3
III.Understanding Servos	29
Lesson 2 Assembly of the robot arm	33
I .Parts List	33
II .List of Structural Components	34
III.Assembly Steps	35
Lesson 3 Robot Arm Joystick Control	59
I .Servos Control	59
II .Understanding the Joystick Module	60
III.Basic Movements of Robot Arm Controlled by Joystick	62
IV.Extending Tasks	65
Lesson 4 The Spatial Coordinates of The Robot Arm	66
I .The Cartesian Coordinate System	66
II .Joint Coordinate System	67
III.Forward and Inverse Kinematics	68
IV.Robot Arm Coordinate Diagram	69
V.Robot Arm Calibration Instructions	70
VI.Moving Spatial Coordinate Points	71
Lesson 5 Robot Arm Stacking	74
I .Robot Arm Palletizing Program	75
II .Extending Tasks	77
Lesson 6 Teaching and Learning of the Robot Arm	78
I .Teaching Program	78
II .Extending Tasks	80
Lesson 7 Web Control of the Robot Arm	81
I .Web Control Program	81
II .Login To the Webpage	82
III.Extending Tasks	83
Lesson 8 Robot Arm APP Control	85
I .APP Download	85
II .APP Control the Robot Arm	87

Lesson 1 Hardware Understanding and Software Installation

Robot arms are automated mechanical devices widely used in the field of robotics. The arm is a crucial component of the robot's execution mechanism, designed to transport the gripped workpiece to a given position. In the industrial sector, the application of robot arms can replace humans in monotonous and repetitive production tasks or in processing operations in dangerous and harsh environments.

A robot arm consists of three basic parts: the main body, the drive system, and the control system. The main body primarily includes the chassis, shoulder, Elbow, and end effector. Depending on different application scenarios, the main body of the robot arm can be divided into four-axis, five-axis, six-axis, or even multi-axis configurations. The drive system includes power devices and transmission mechanisms, with the core components being the reducer and servo motor, which drive the robot arm to perform corresponding actions. The control system issues command signals to the drive system according to the input program, controlling the movement of the robot arm.

The robot arm in this tutorial is a four-axis robot arm. The first axis is the chassis servo, the second axis is the shoulder servo, the third axis is the elbow servo, and the fourth axis is the claw servo.

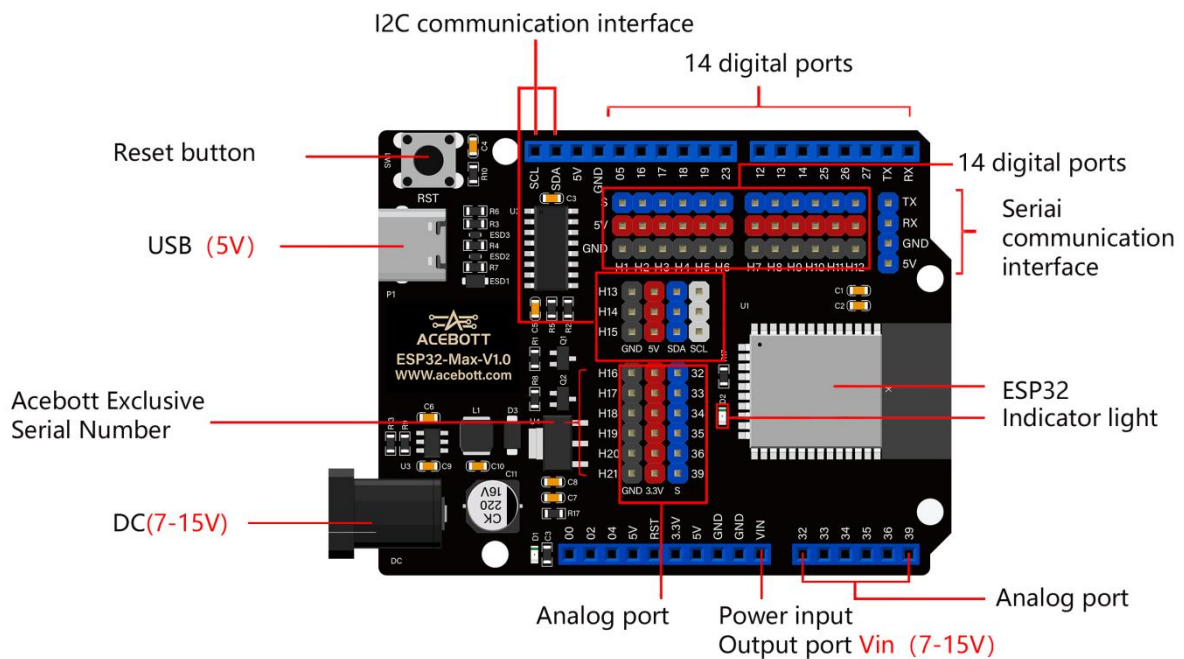
The robot arm in this tutorial uses a mainstream ESP32 controller board as its main controller, and programming in the Arduino IDE. Control methods include joystick control, web control and app control.

How is the robot arm specifically controlled? Let's continue with the tutorial and learn together.



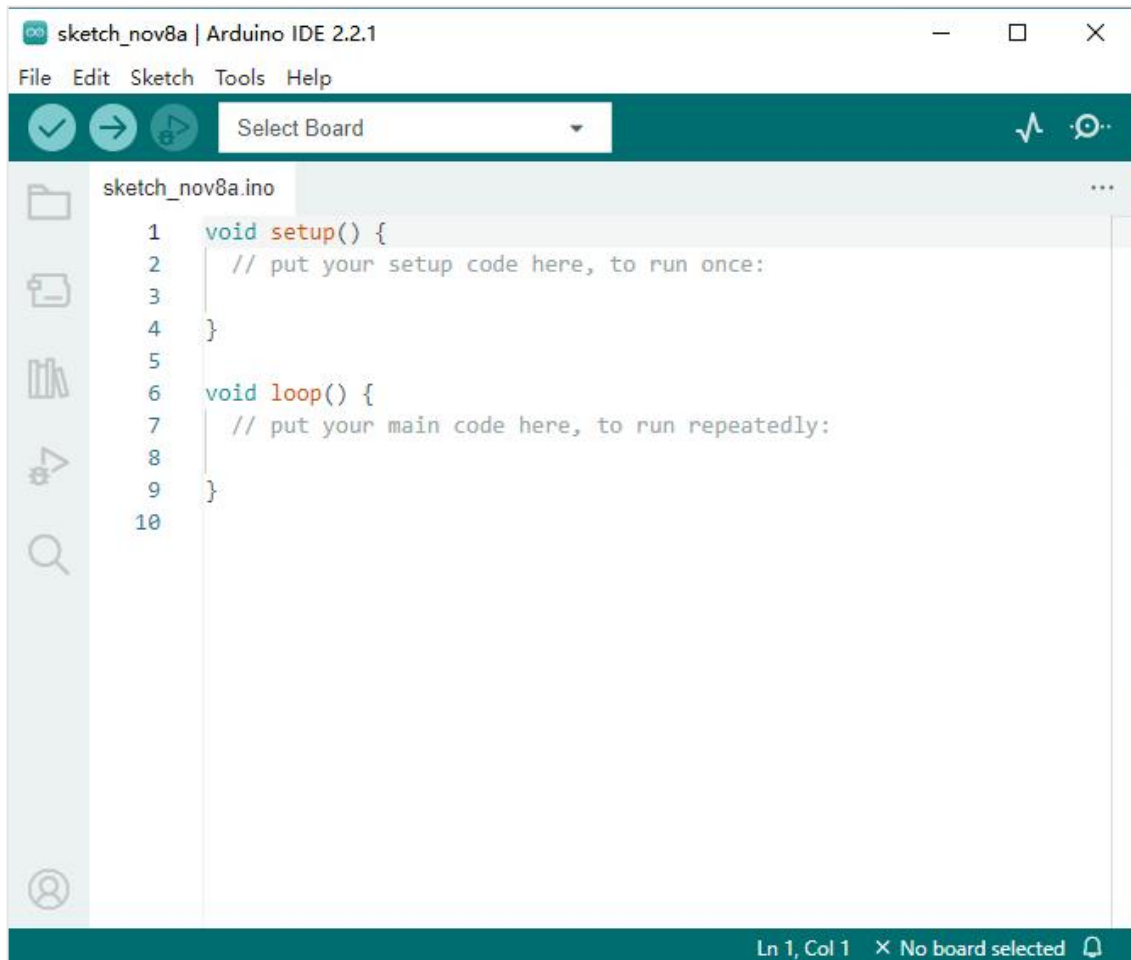
I .Hardware Understanding

The ESP32 controller board is a low-power, high-performance microcontroller ideal for IoT development. It features a dual-core processor running at 240MHz, 520KB of RAM, and 4MB of flash memory. It includes built-in WiFi and Bluetooth 4.2 modules for wireless communication. With 34 GPIO pins, it supports connectivity and control of various peripherals.



II .Software Installation

In the robot arm project, we will primarily use the Arduino IDE as the programming software. It is an open-source programming platform compatible with various boards such as Arduino UNO, ESP32, ESP8266, STM32, and more. Using the Arduino IDE, you simply write your program code within the IDE and then upload it to the controller board. The program will instruct the controller board on what actions to perform.



1.Installing Arduino IDE

First, open the official website where the Arduino IDE is downloaded:

<https://www.arduino.cc/en/Main/Software>

According to the user's computer system, choose the corresponding software version.

(1) Installation on Windows system

①Click the mouse at the position shown in the figure.

Attention: Due to potential compatibility issues with the new version, it is recommended to install version 2.2.1 of the Arduino IDE.



Arduino IDE 2.2.1

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

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

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

SOURCE CODE

The Arduino IDE 2.0 is open source and its source code is hosted on [GitHub](#).

DOWNLOAD OPTIONS

Windows Win 10 and newer, 64 bits

Windows MSI installer

Windows ZIP file

Linux AppImage 64 bits (X86-64)

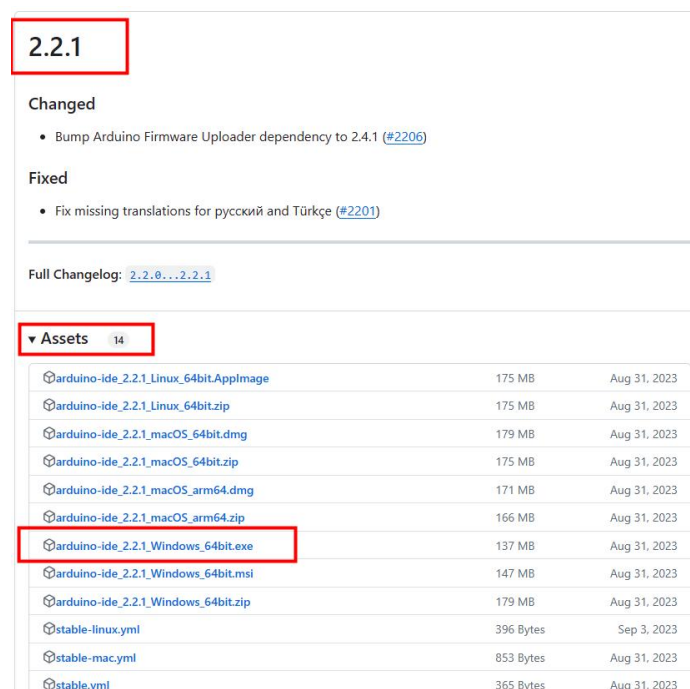
Linux ZIP file 64 bits (X86-64)

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

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

[Release Notes](#)

Link to version 2.2.1:<https://github.com/arduino/arduino-ide/releases>



2.2.1

Changed

- Bump Arduino Firmware Uploader dependency to 2.4.1 (#2206)

Fixed

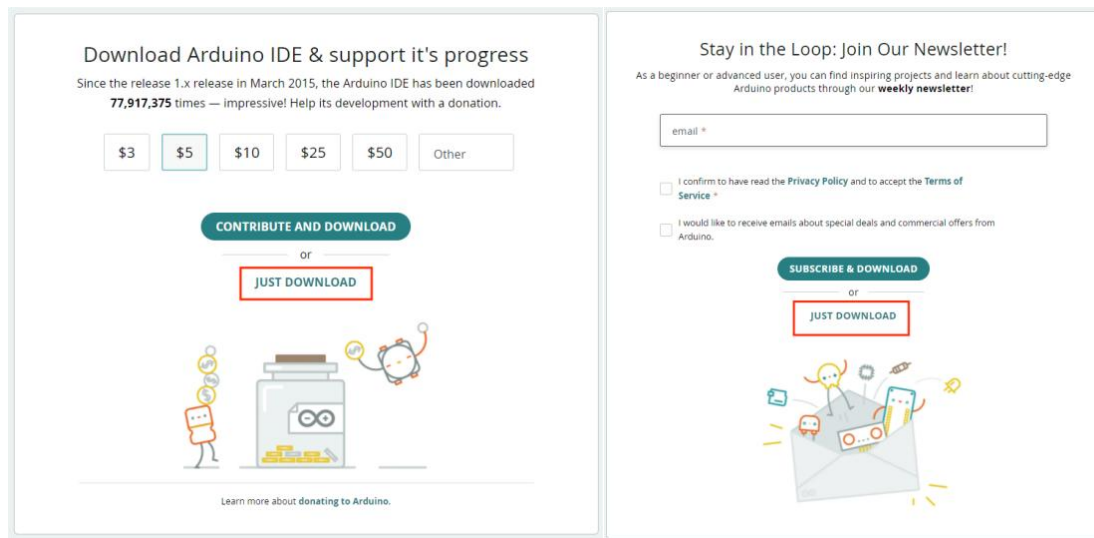
- Fix missing translations for русский and Türkçe (#2201)

Full Changelog: [2.2.0...2.2.1](#)

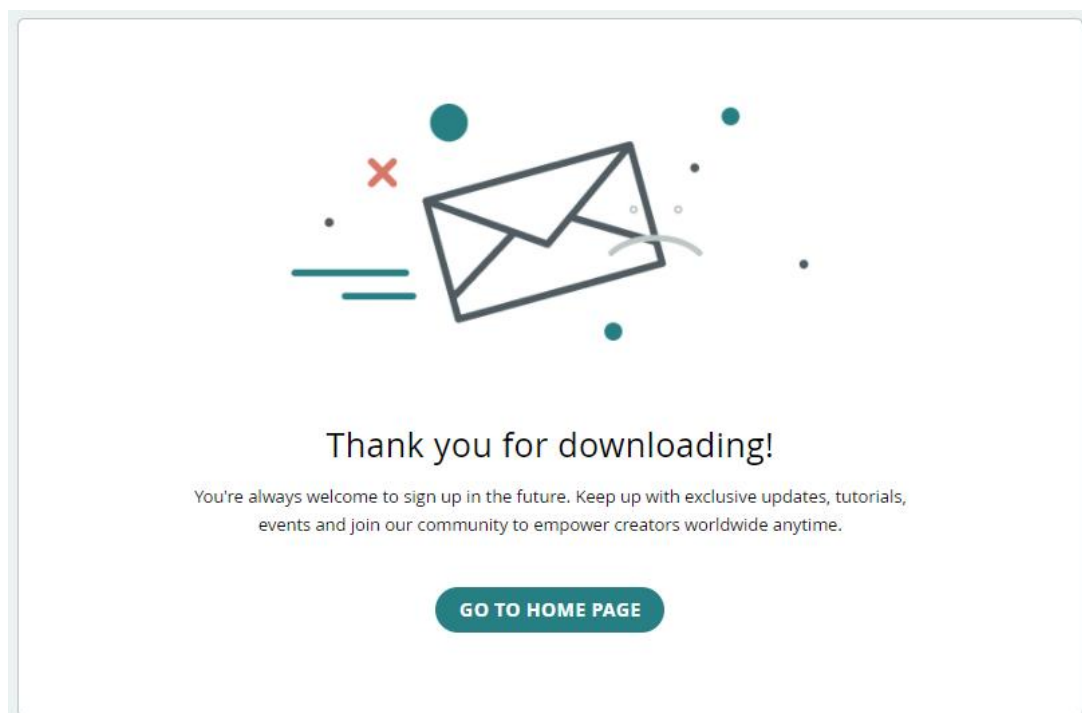
Assets 14

arduino-ide_2.2.1_Linux_64bit.AppImage	175 MB	Aug 31, 2023
arduino-ide_2.2.1_Linux_64bit.zip	175 MB	Aug 31, 2023
arduino-ide_2.2.1_macOS_64bit.dmg	179 MB	Aug 31, 2023
arduino-ide_2.2.1_macOS_64bit.zip	175 MB	Aug 31, 2023
arduino-ide_2.2.1_macOS_arm64.dmg	171 MB	Aug 31, 2023
arduino-ide_2.2.1_macOS_arm64.zip	166 MB	Aug 31, 2023
arduino-ide_2.2.1_Windows_64bit.exe	137 MB	Aug 31, 2023
arduino-ide_2.2.1_Windows_64bit.msi	147 MB	Aug 31, 2023
arduino-ide_2.2.1_Windows_64bit.zip	179 MB	Aug 31, 2023
stable-linux.yml	396 Bytes	Sep 3, 2023
stable-mac.yml	853 Bytes	Aug 31, 2023
stable.yml	365 Bytes	Aug 31, 2023

②Select JUSTDOWNLOAD.



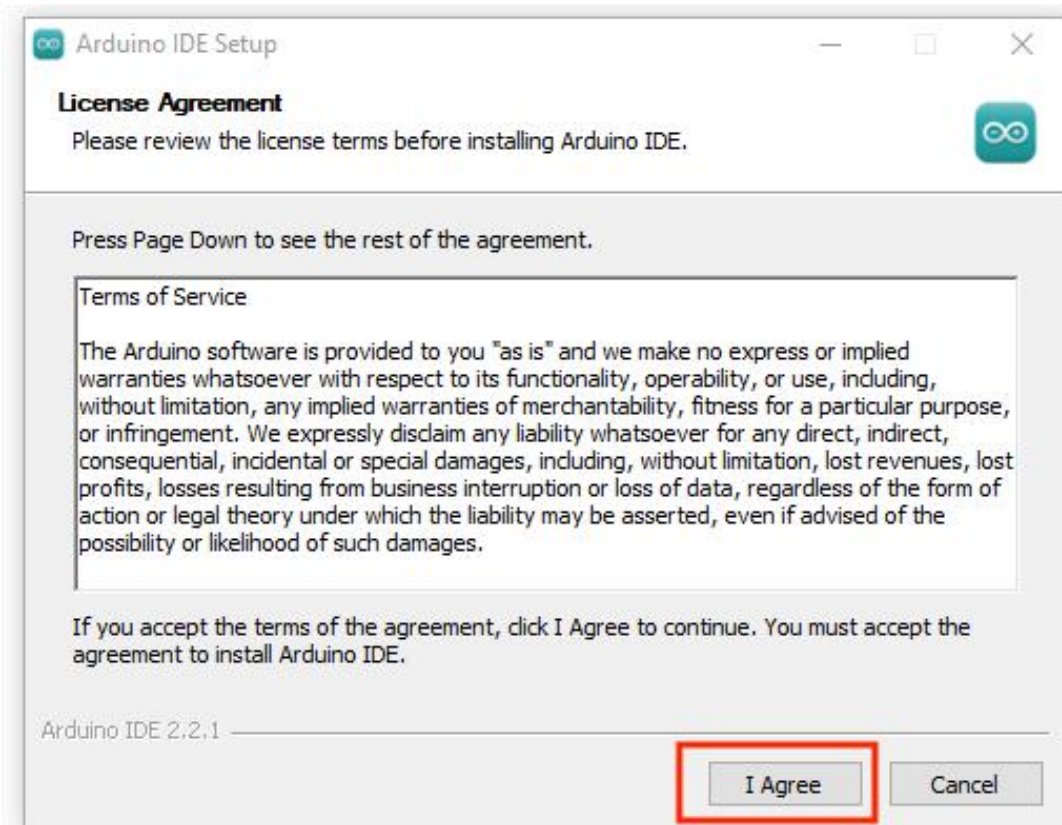
③When the following screen appears, it indicates that the Arduino IDE is downloading.



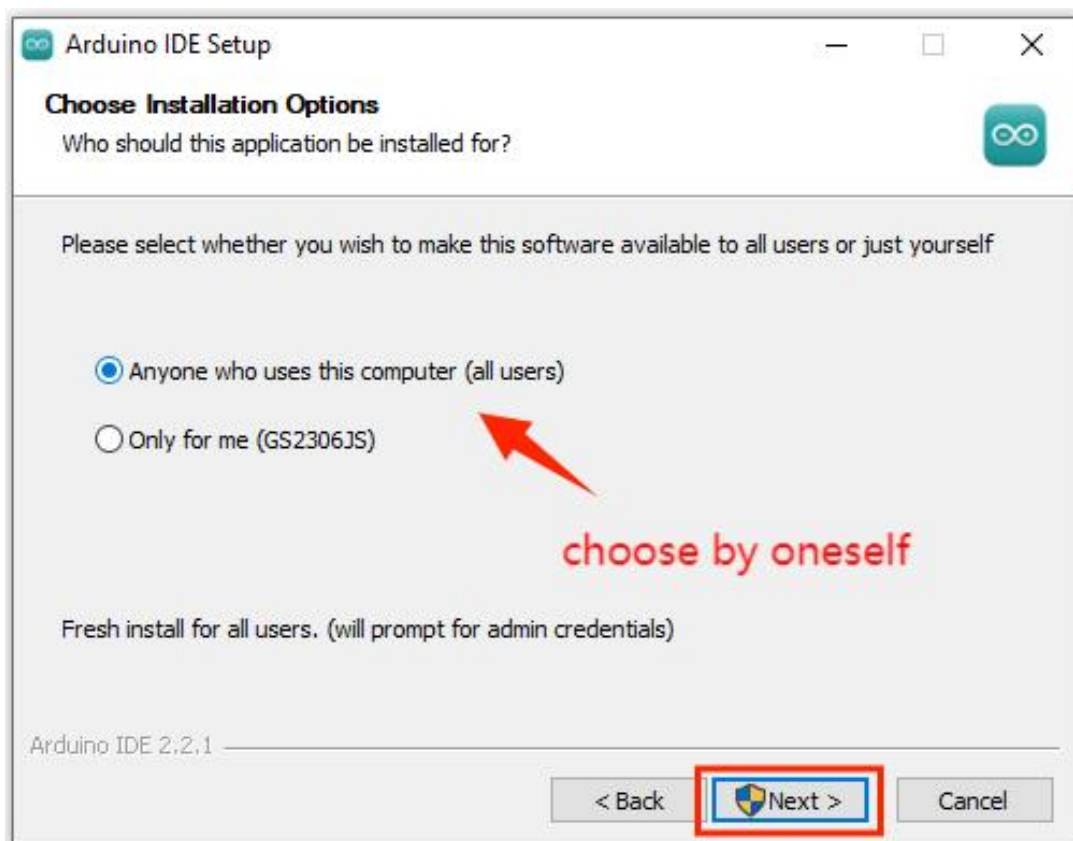
④When the download is complete, the icon file will appear. Click Install software.



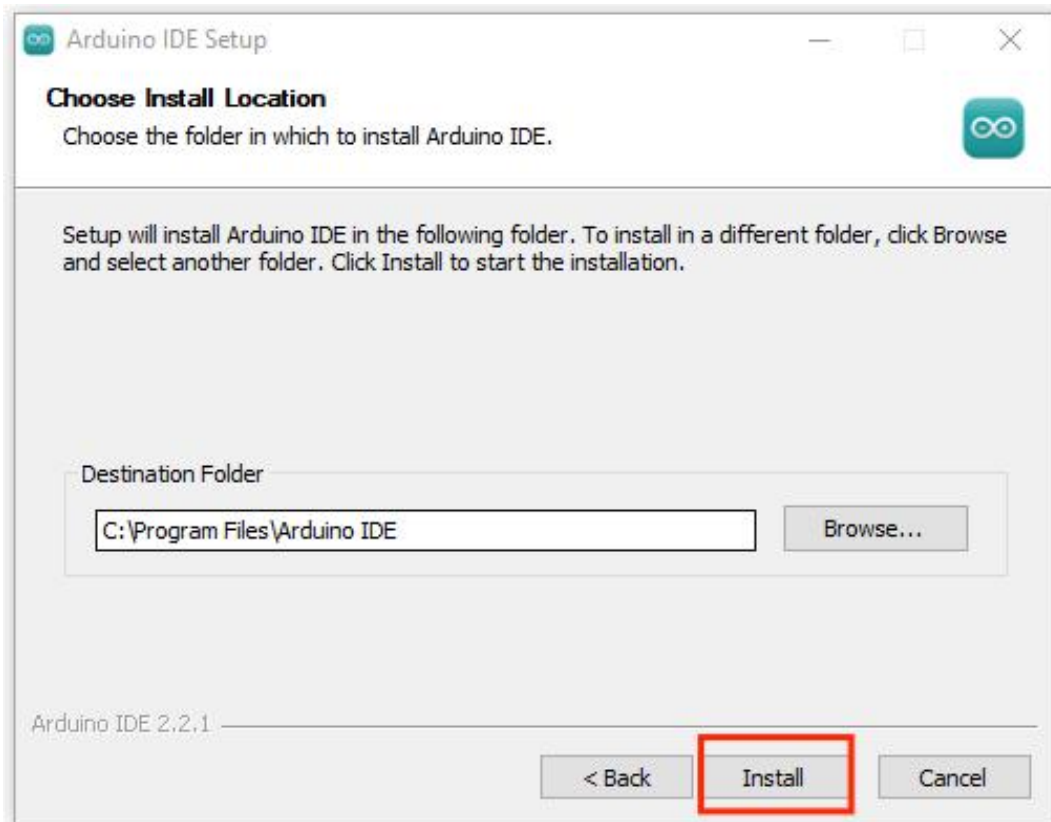
- ⑤ After installation, the following screen appears and select "I Agree".



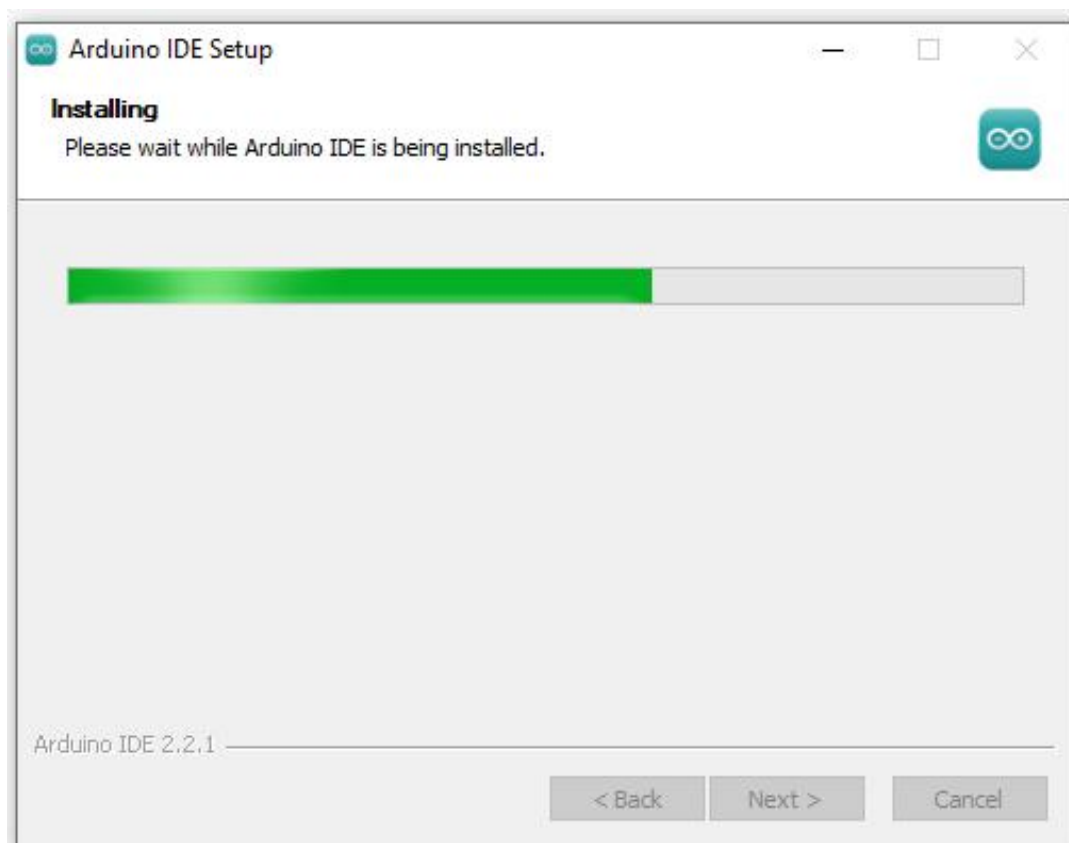
- ⑥ After selecting "I Agree", the following screen will appear and select "Next".



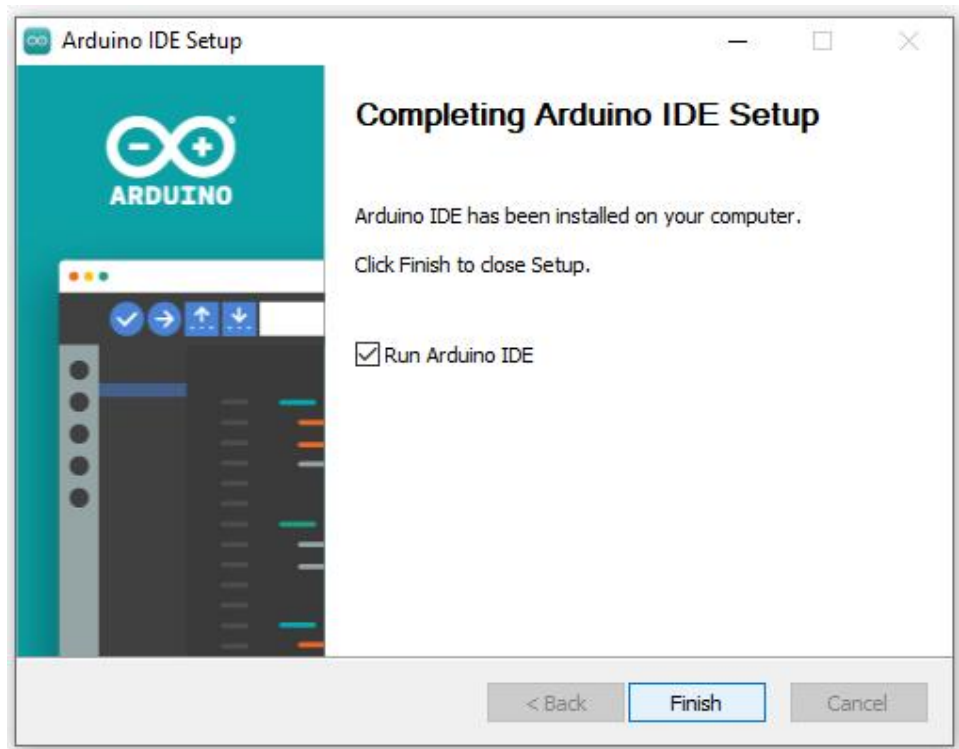
⑦Select "Next" and the following screen will appear. Select "Install".



⑧Arduino IDE software installed.



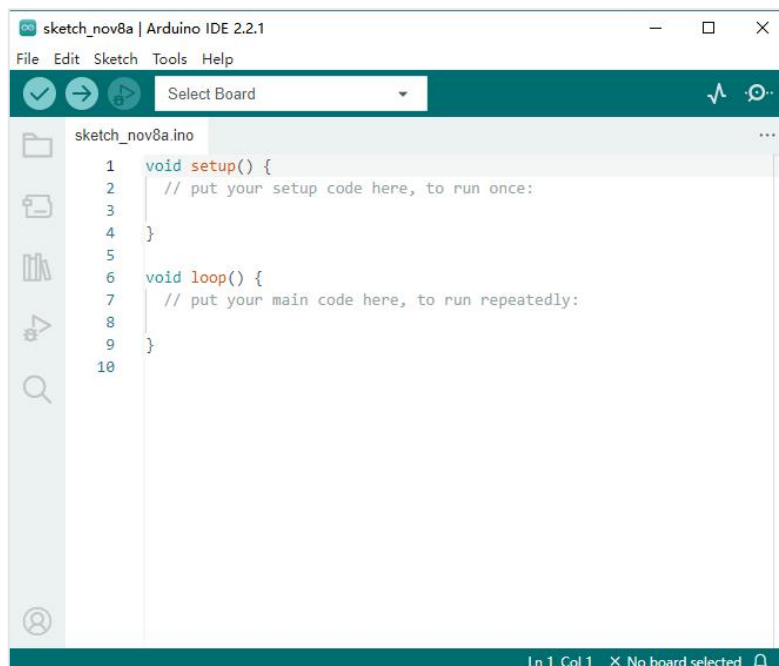
⑨ When the installation is complete, click Finish.



⑩ After installation, a shortcut icon of Arduino IDE will appear on your desktop.



⑪ After opening, the following software interface will appear.



(2) Installation on Mac OS

① Click the mouse at the position shown in the figure.

Attention: Due to potential compatibility issues with the new version, it is recommended to install version 2.2.1 of the Arduino IDE.



Arduino IDE 2.2.1

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

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

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

SOURCE CODE

The Arduino IDE 2.0 is open source and its source code is hosted on [GitHub](#).

DOWNLOAD OPTIONS

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

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

macOS Intel, 10.14: "Mojave" or newer, 64 bits
macOS Apple Silicon, 11: "Big Sur" or newer, 64 bits

[Release Notes](#)

Link to version 2.2.1: <https://github.com/arduino/arduino-ide/releases>

2.2.1

Changed

- Bump Arduino Firmware Uploader dependency to 2.4.1 (#2206)










Fixed

- Fix missing translations for русский and Türkçe (#2201)

Full Changelog: [2.2.0...2.2.1](#)

▼ Assets

 14

 arduino-ide_2.2.1_Linux_64bit.AppImage	175 MB	Aug 31, 2023
 arduino-ide_2.2.1_Linux_64bit.zip	175 MB	Aug 31, 2023
 arduino-ide_2.2.1_macOS_64bit.dmg	179 MB	Aug 31, 2023
 arduino-ide_2.2.1_macOS_64bit.zip	175 MB	Aug 31, 2023
 arduino-ide_2.2.1_macOS_arm64.dmg	171 MB	Aug 31, 2023
 arduino-ide_2.2.1_macOS_arm64.zip	166 MB	Aug 31, 2023
 arduino-ide_2.2.1_Windows_64bit.exe	137 MB	Aug 31, 2023
 arduino-ide_2.2.1_Windows_64bit.msi	147 MB	Aug 31, 2023
 arduino-ide_2.2.1_Windows_64bit.zip	179 MB	Aug 31, 2023

②Select JUSTDOWNLOAD.


Download Arduino IDE & support it's progress

Since the release 1.x release in March 2015, the Arduino IDE has been downloaded **77,917,375** times — impressive! Help its development with a donation.

CONTRIBUTE AND DOWNLOAD

or

JUST DOWNLOAD



[Learn more about donating to Arduino.](#)

Stay in the Loop: Join Our Newsletter!

As a beginner or advanced user, you can find inspiring projects and learn about cutting-edge Arduino products through our **weekly newsletter!**

email *


☐ I confirm to have read the [Privacy Policy](#) and to accept the [Terms of Service](#)

☐ I would like to receive emails about special deals and commercial offers from Arduino.

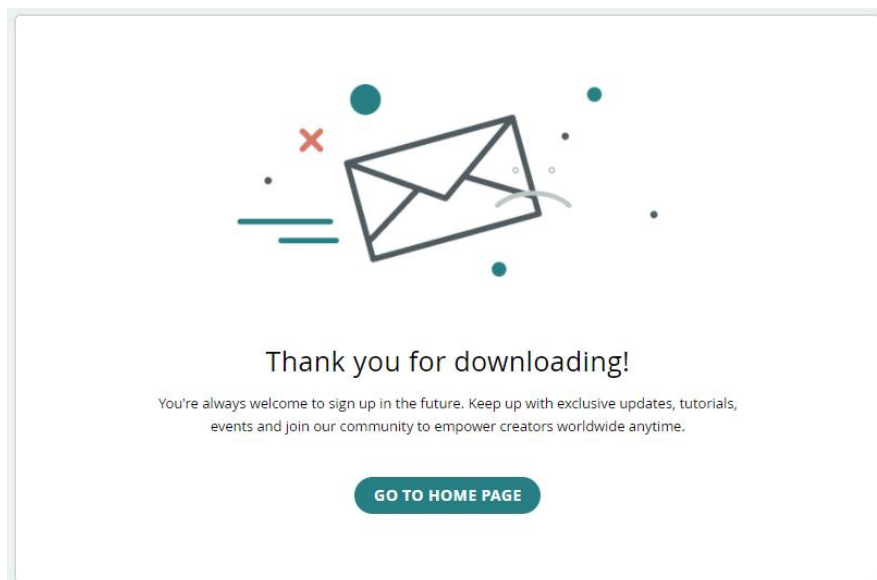
SUBSCRIBE & DOWNLOAD

or

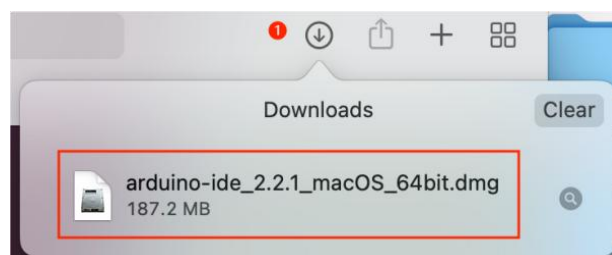
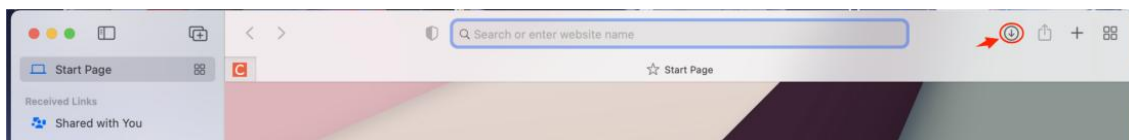
JUST DOWNLOAD



③When the following screen appears, the Arduino IDE is downloading.



④When the download is complete click the download icon in your browser and find the Arduino IDE installer.



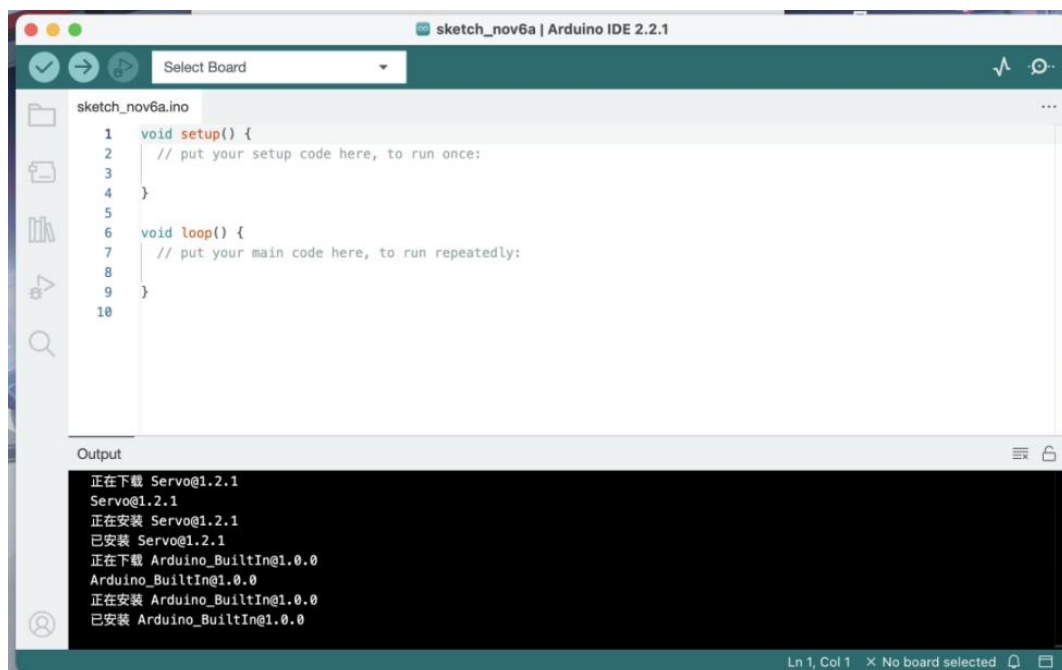
⑤Click on the install package to appear the installation screen, just select the Arduino IDE icon, move to the Applications to install the program.



⑥In the workbench, find the Arduino IDE and open it.



⑦After opening, you can see the following software interface.



2. Install the serial driver (skip it if installed)

Serial port is a computer communication interface, usually used for the computer and other devices (such as modems, sensors, printers, microcontrollers, etc.) for data transmission, USB serial port is our most commonly used communication interface.

Generally, after installing the Arduino IDE software for the first time, it is necessary to install the serial port driver to transmit the edited program to the controller board.

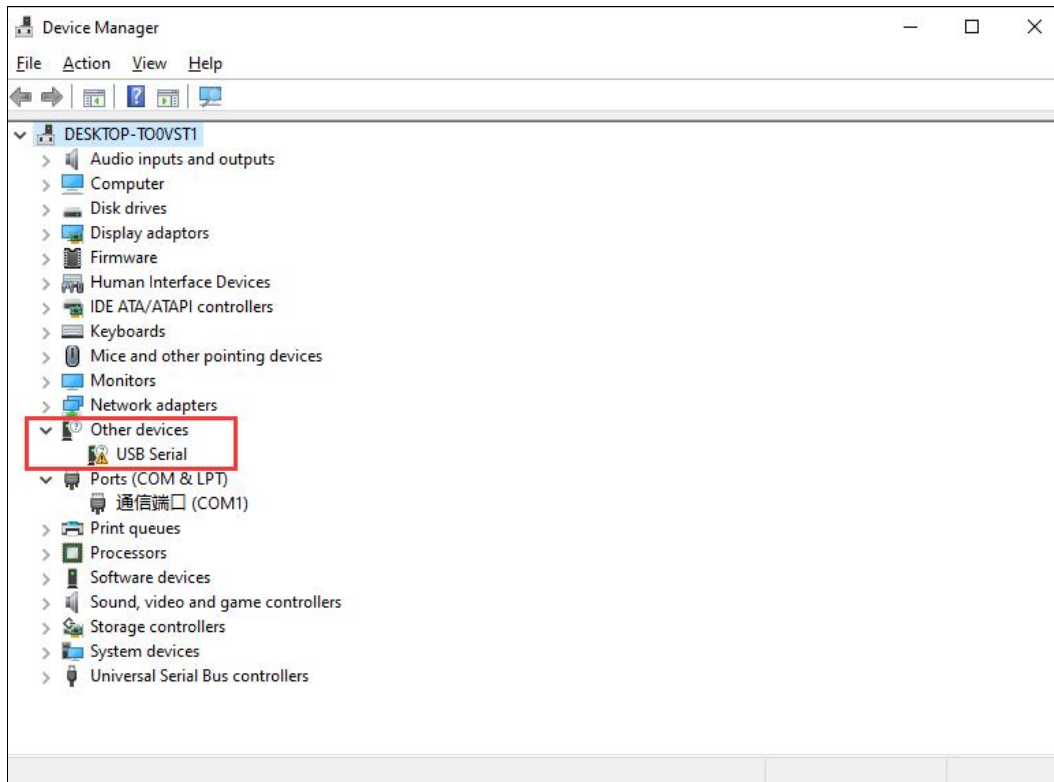
The USB-to-serial chip of the ESP32 controller board is CH340, so you need to install the driver for this chip. After connecting the controller board to the computer with a USB cable, the driver will usually install automatically on MacOS and Windows systems. If the driver fails to install automatically, you will need to install the driver manually.

The Driver can be downloaded from the Internet, or you can find the folder named "[CH340 Driver](#)" in the resource package provided by us. Choose the appropriate package according to your computer system type, and then follow the steps below to install the driver.

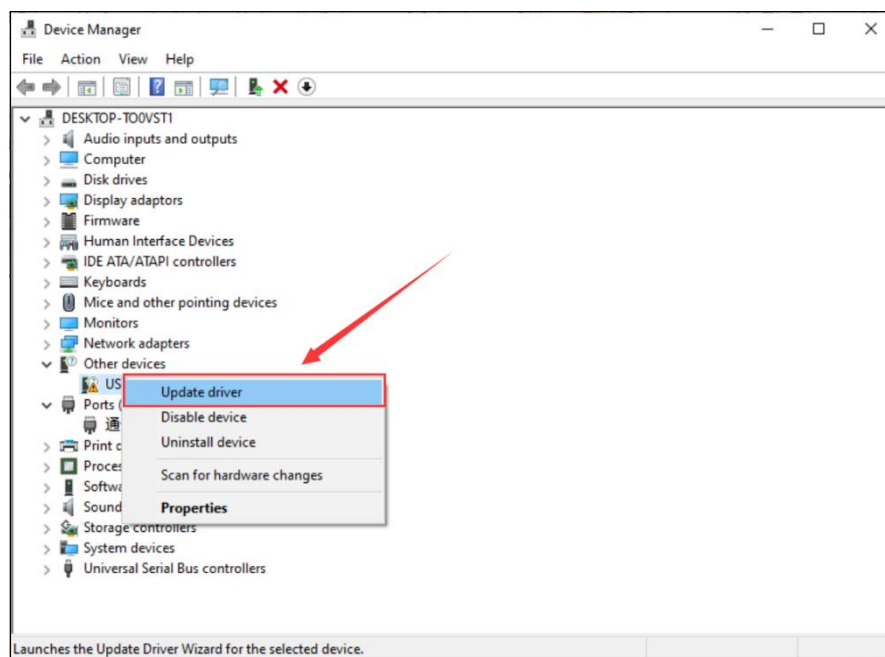
(1) Windows system driver installation mode

① Insert one end of the USB cable into the ESP32 controller board and the other end into a USB port on your computer.

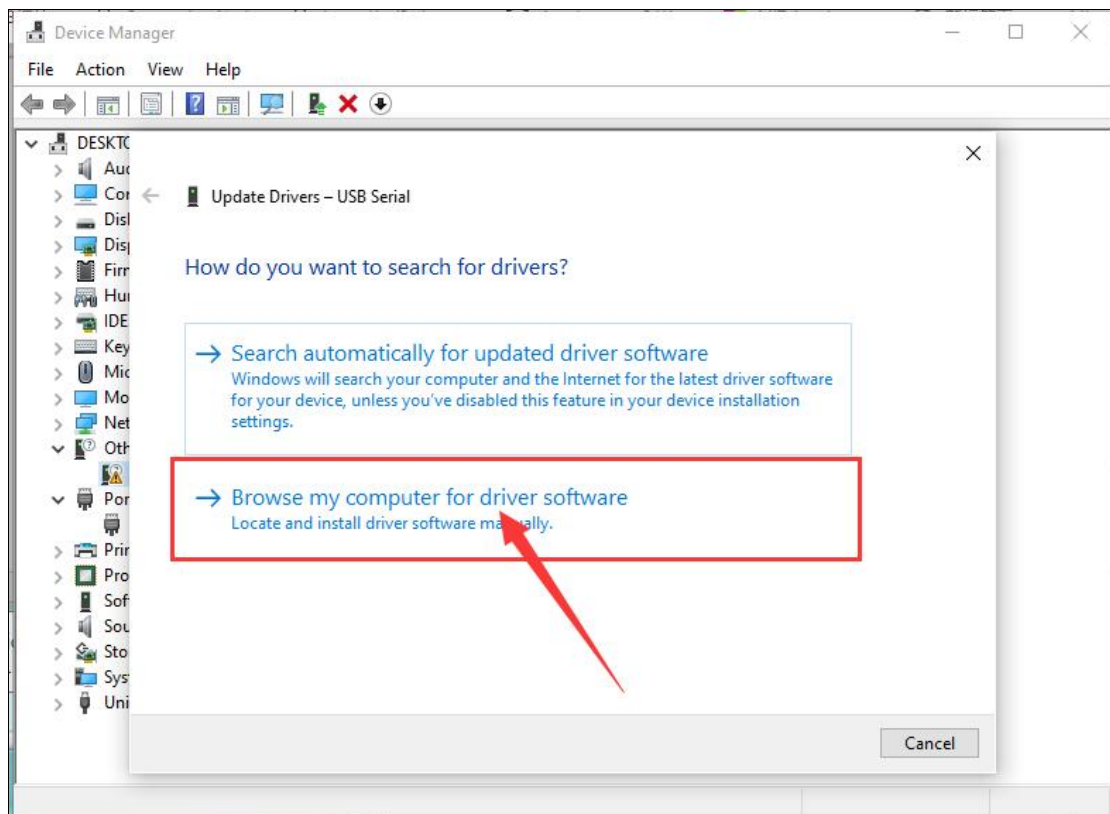
② When connecting the ESP32 controller board to the computer for the first time, right-click "My Computer" -> "Properties" -> click "Device Manager." Under "Other devices," you will see "USB-Serial" or "Unknown Device."



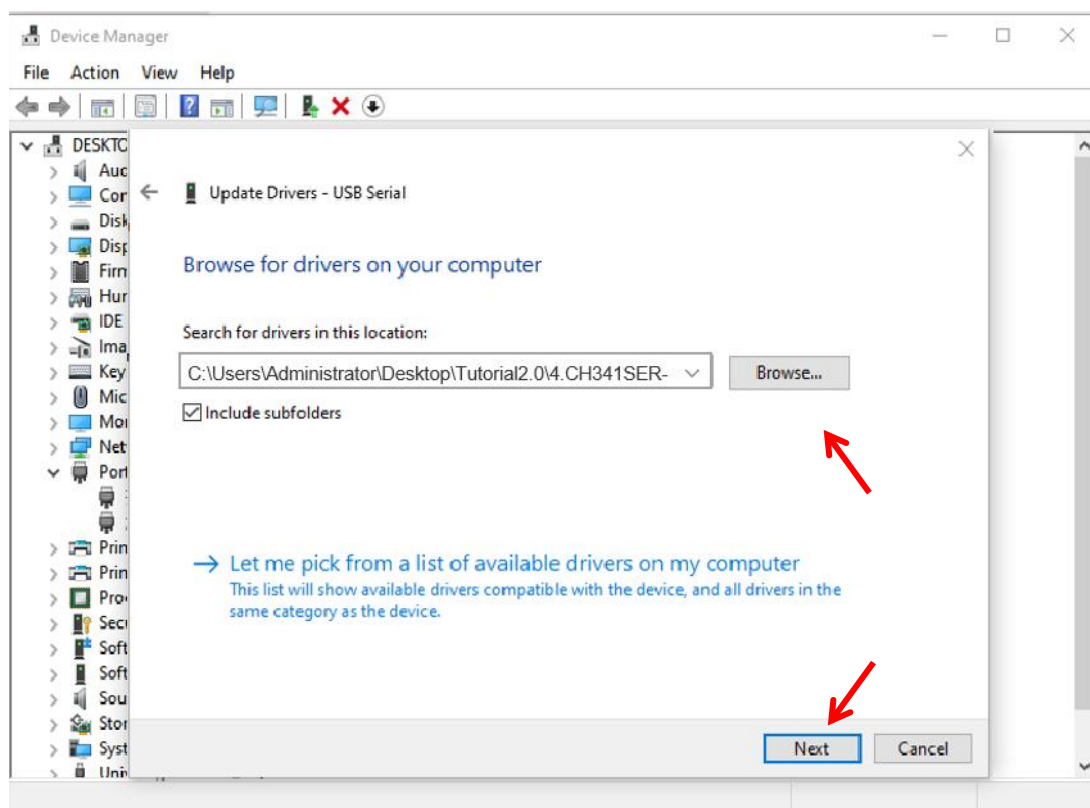
③Right-click on the device and select the top menu option ("Update Driver Software"), as shown in the image below.



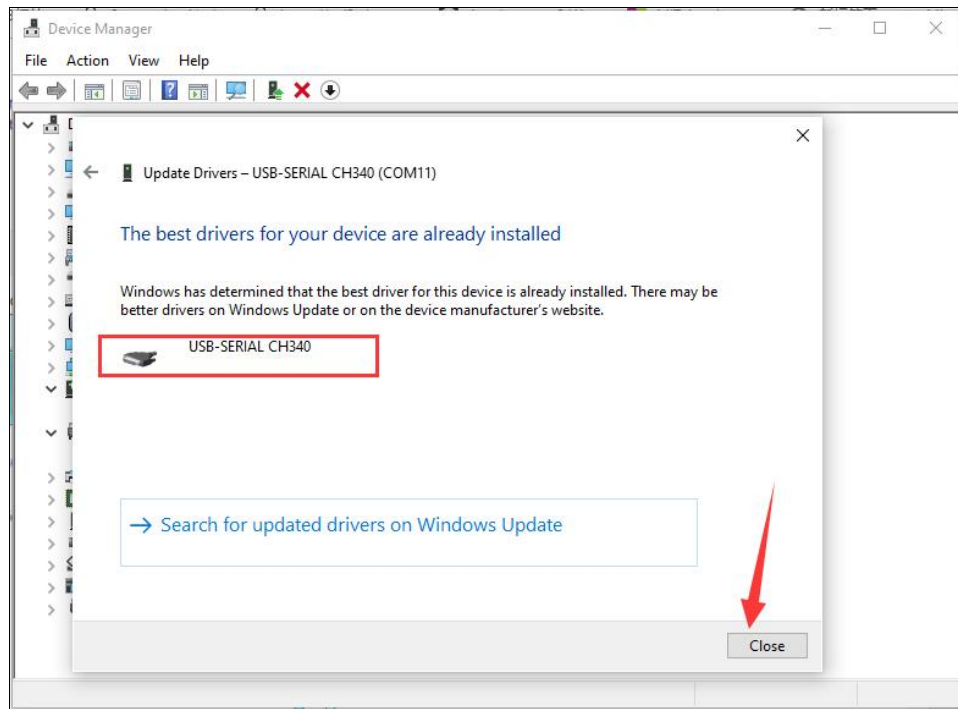
④You will then be prompted with "Search automatically for updated driver software" or "Browse my computer for driver software," as shown in the image below. On this page, select "Browse my computer for driver software."



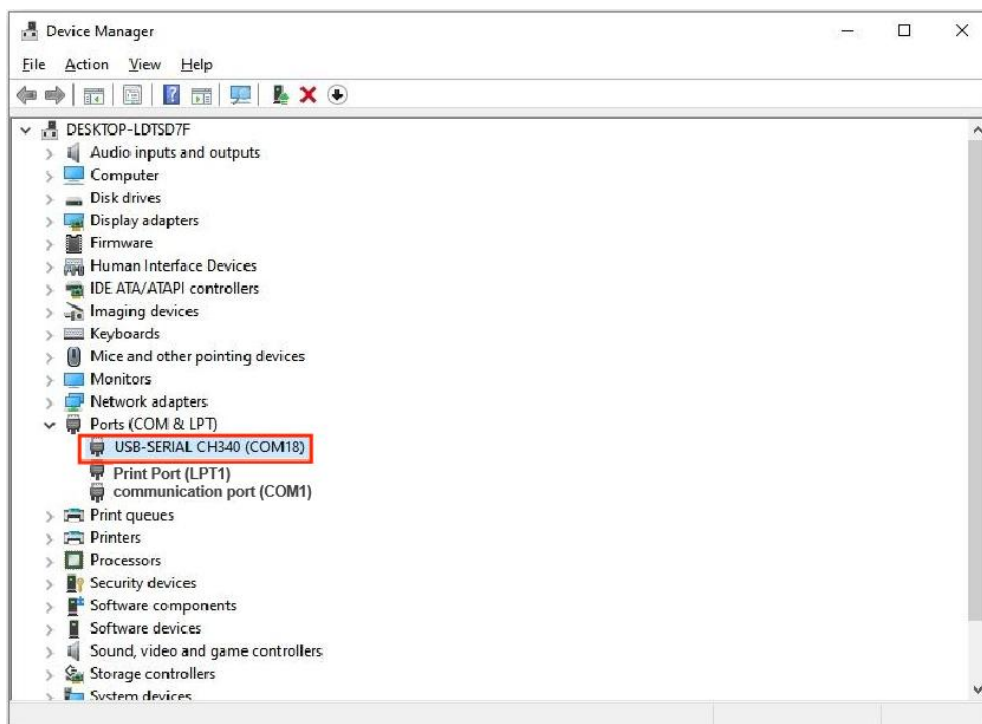
⑤ Then, add the path to the driver files.



⑥ After the software installation is complete, you will receive a confirmation message. Once the installation is finished, click "Close."



⑦ To confirm if the installation was successful, plug one end of the USB cable into the ESP32 controller board and the other end into a USB port on your computer. Right-click "My Computer" -> "Properties" -> click "Device Manager". Once the controller board is connected, the successful installation will be shown as indicated in the image below.

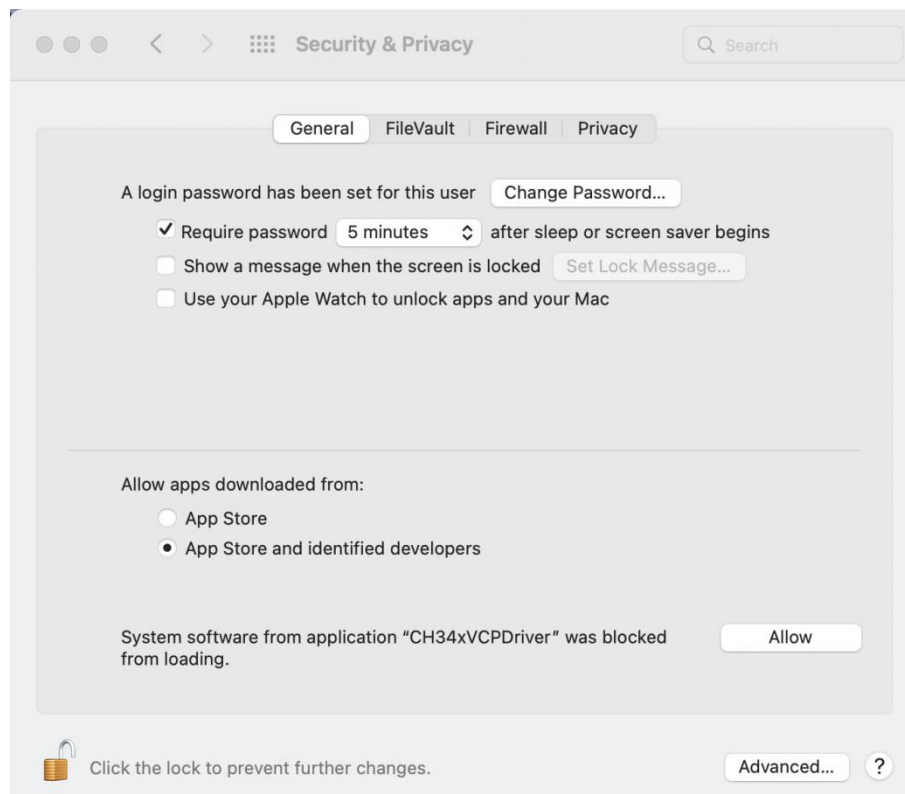


(2) macOS System driver installation mode

Open the "[CH340 Driver file-mac](#)" folder, you can see the driver installation File we provided. In general, the pkg format driver is installed by default, but if Rosetta is not supported in OS X 11.0 or later, install the dmg format driver.

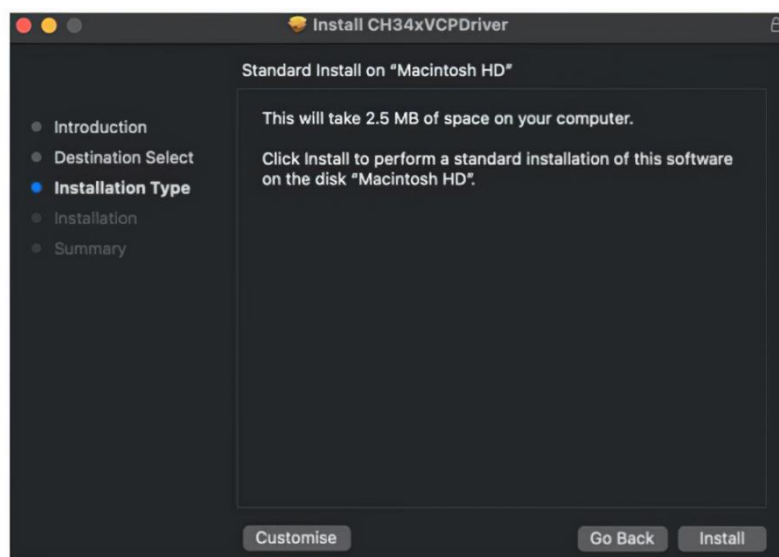
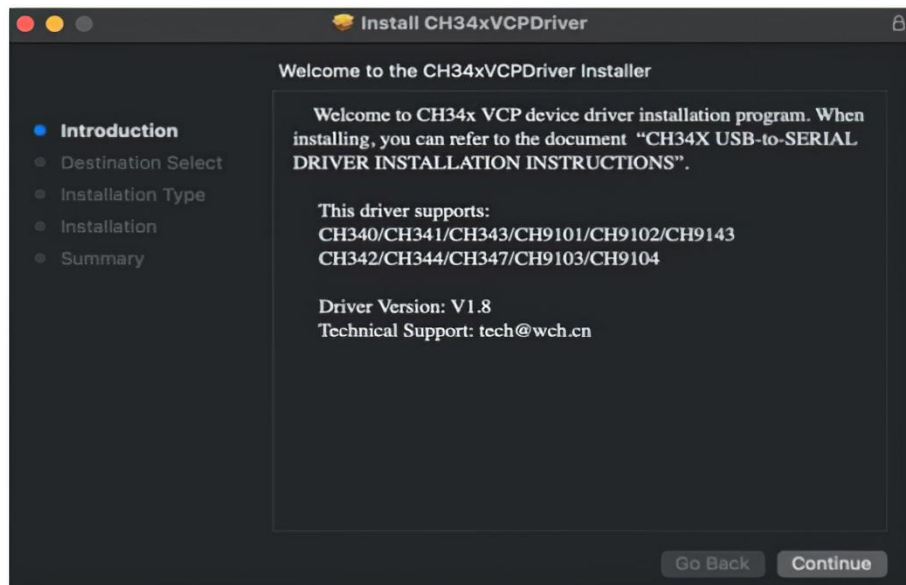


Before installing, go to "System Preferences "->" Security & Privacy "->" General" and select ->"Mac App Store and identified developers "under the heading" Allow applications to be downloaded from the following addresses "so that the driver will work correctly.



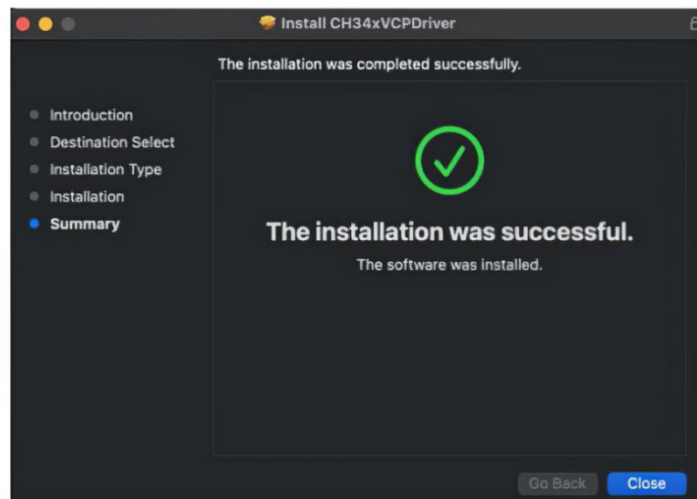
1).Install the driver in pkg format

①Click driver file -> Continue -> Install.

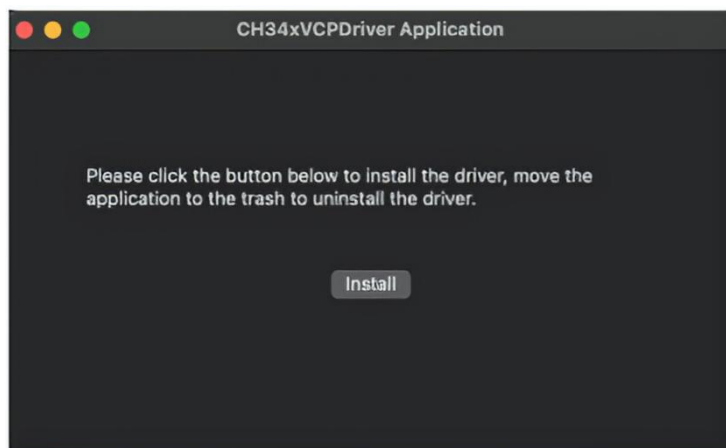


②The installation was successful.





③Install the pkg format driver on OS X 11.0 and later: open "LaunchPad"->"CH34xVCPDriver"-> Install.



④When using OS X 10.9 through OS X 10.15, click Restart to restart your computer, and perform the following steps after restarting.



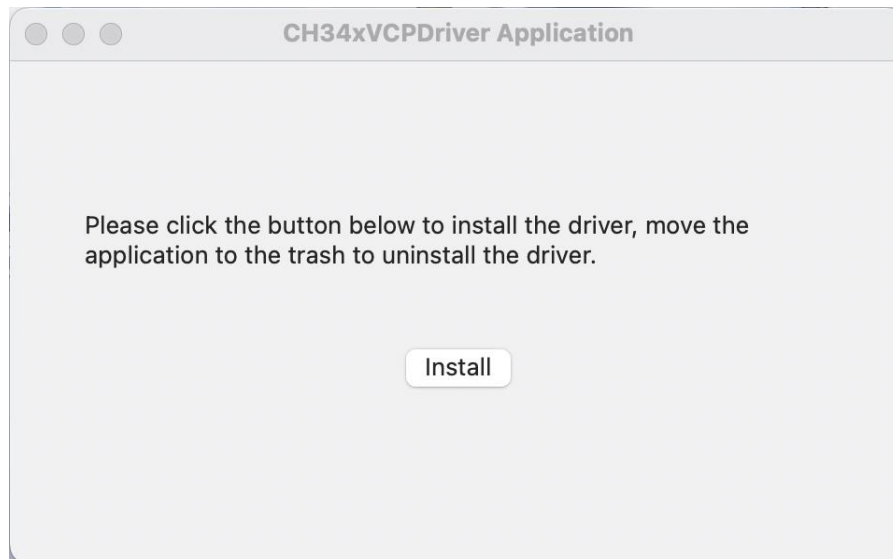
2)Install the dmg format driver (Attention:If the pkg file of Step A is successfully installed, skip Step B)

①Install the dmg driver, click the dmg file and drag "CH34xVCPDriver" into the application folder of the operating system.

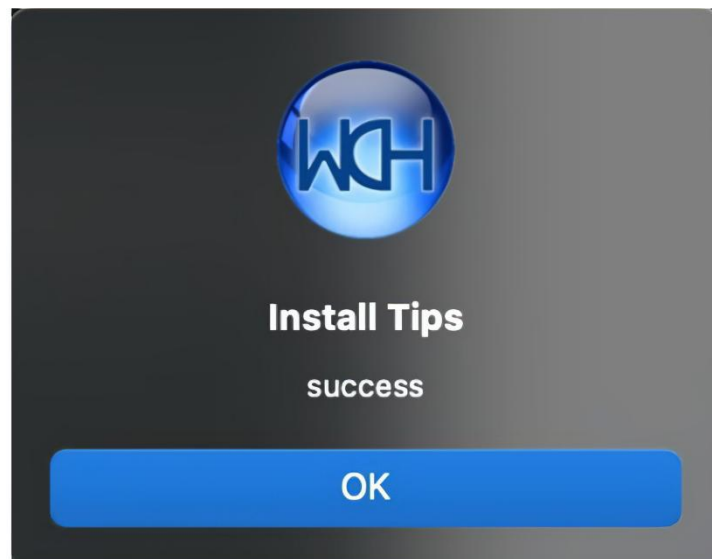


②Then open "LaunchPad"->"CH34xVCPDriver"-> Install.



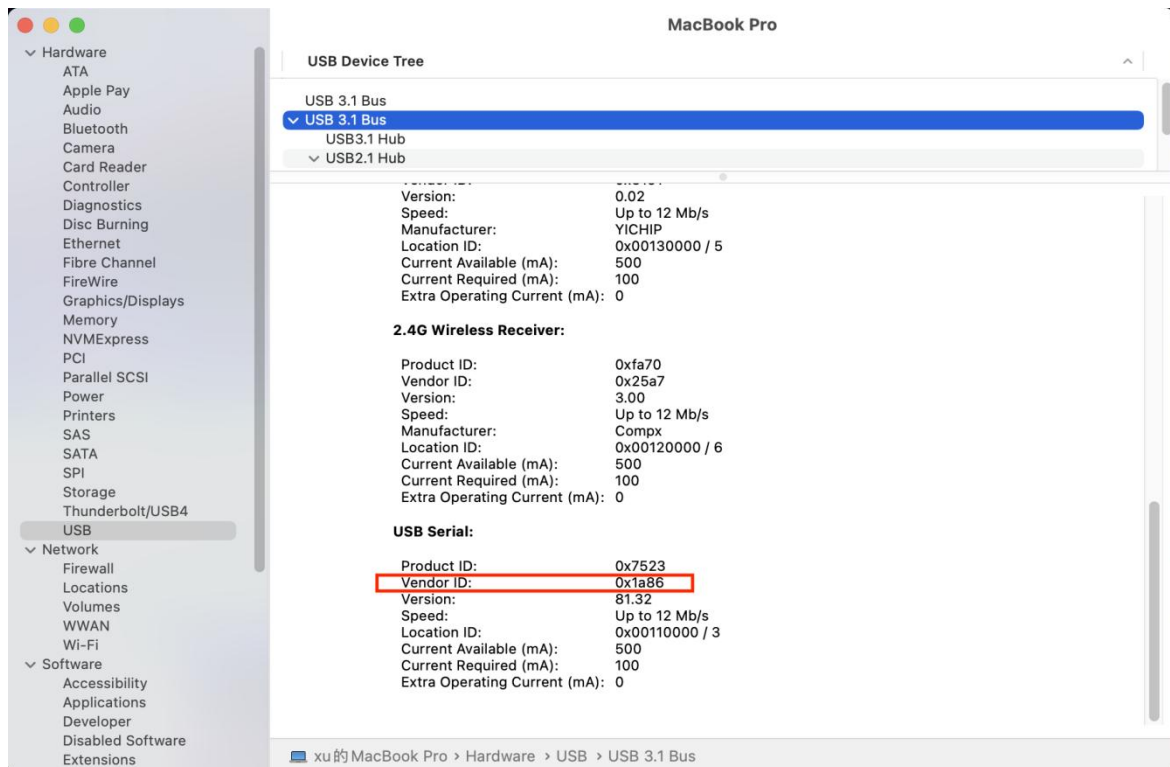


③The installation was successful.



3)Check whether the CH340 serial port driver is installed

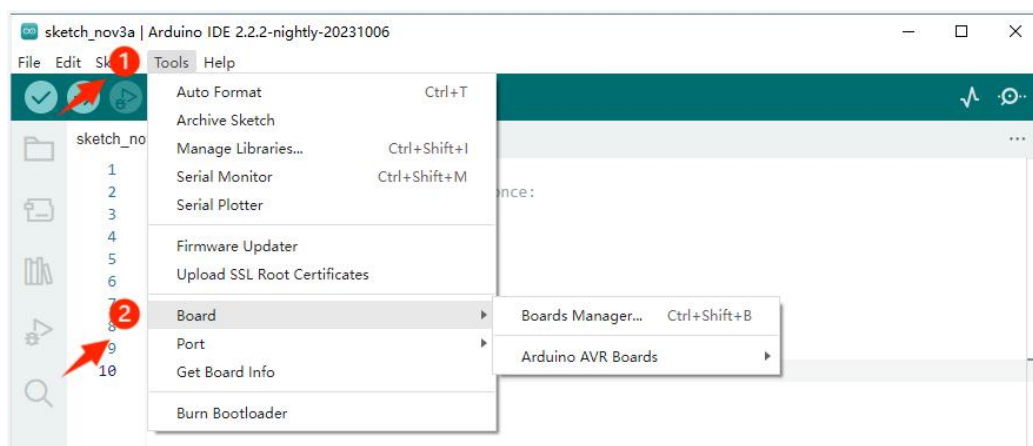
When the controller board is inserted into the USB port, open System Report -> Hardware ->USB. On the right is the USB device. If the USB device is working correctly, you can find a device with a "Vendor ID" of [0x1a86].



3.Install the ESP32 library

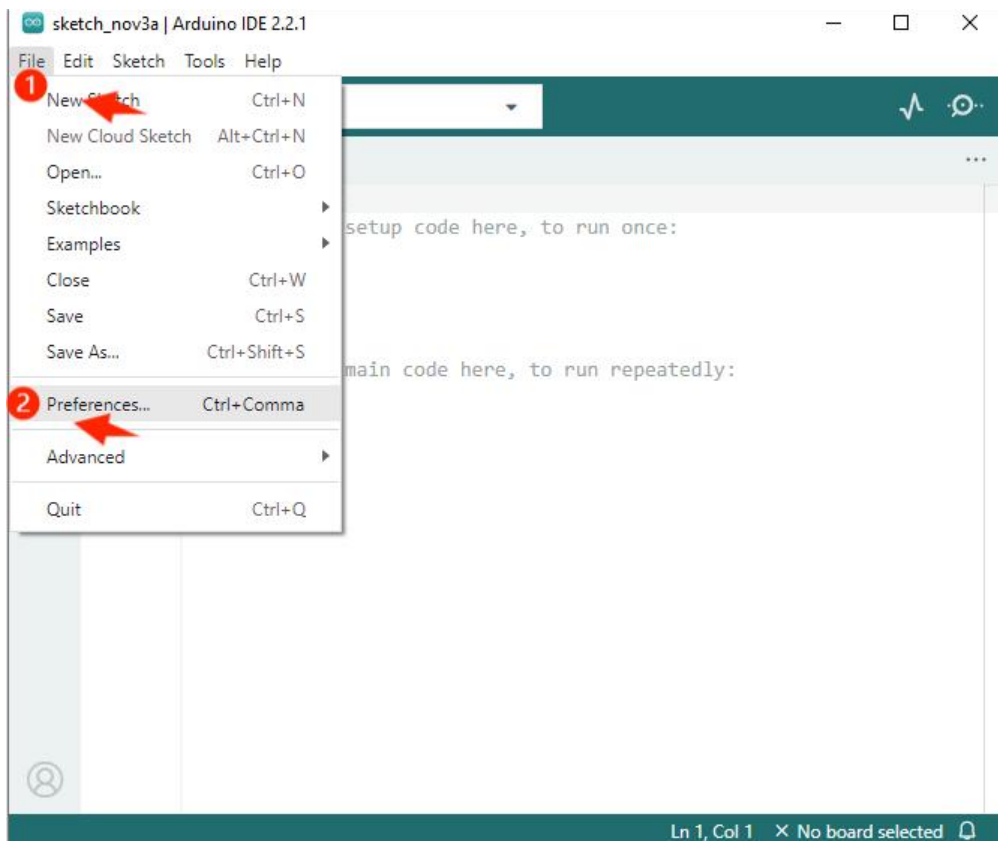
Because the controller board used by the robot arm is ESP32, it is also necessary to add the library of ESP32 in order to program the ESP32 board on the Arduino IDE.

When you open the Arduino IDE, select Tools>Board, you will find that the Arduino IDE only has Arduino AVR Boards and no ESP32.

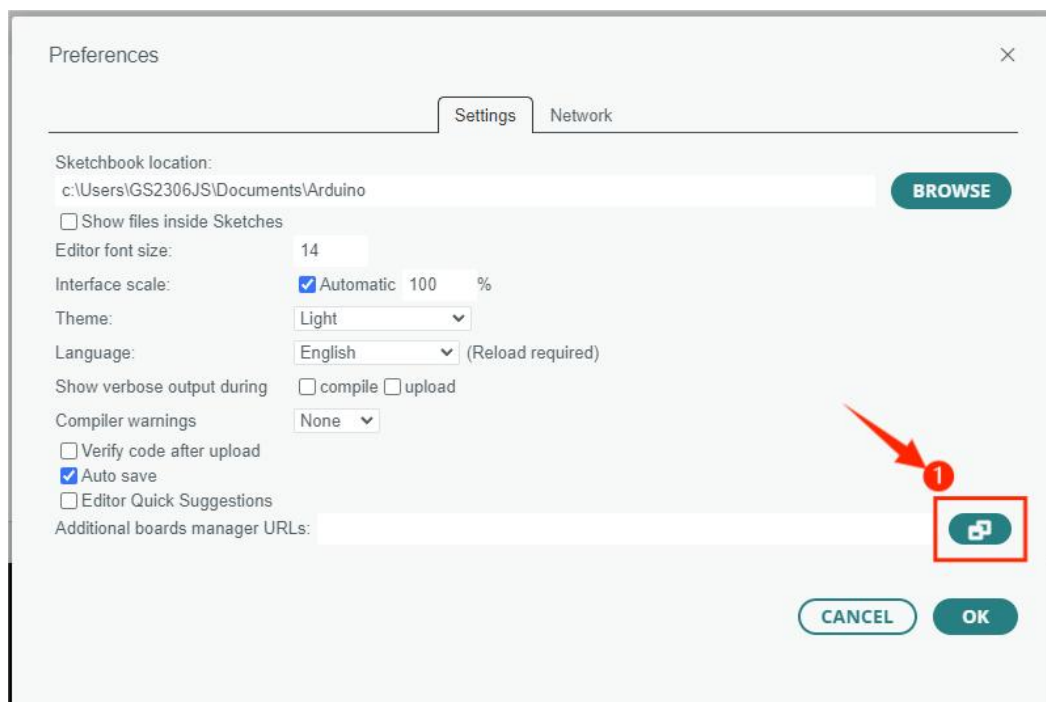


Next we need to install ESP32 library, follow these steps:

① Open File > Preferences

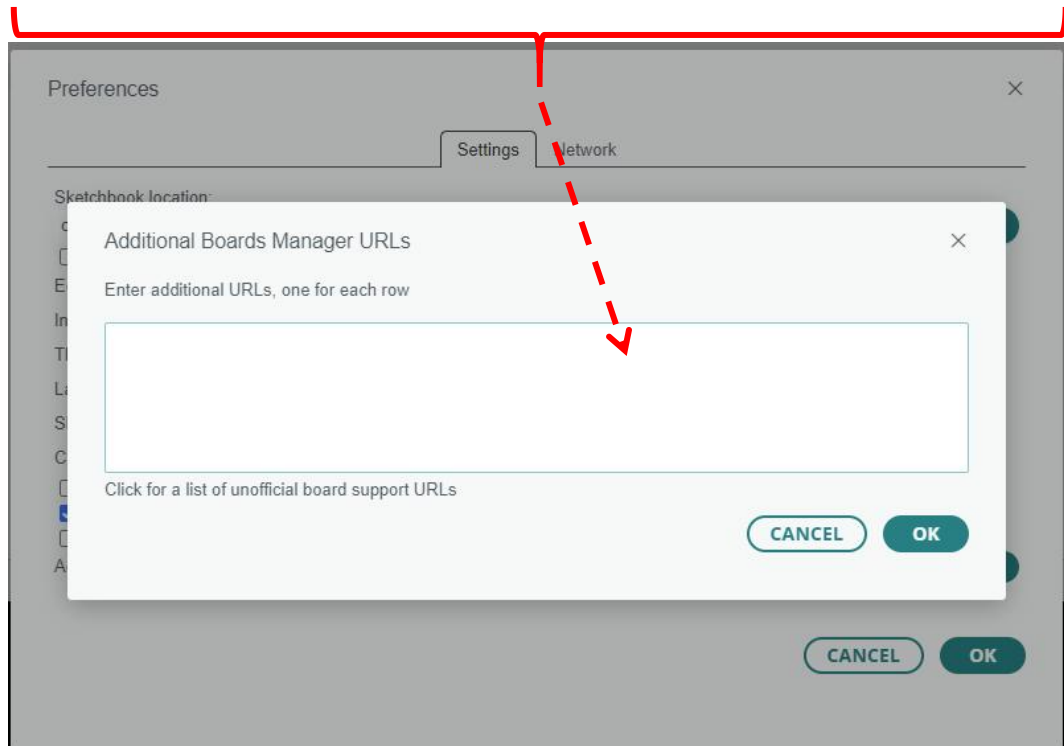


② Add the controller board management address URL.

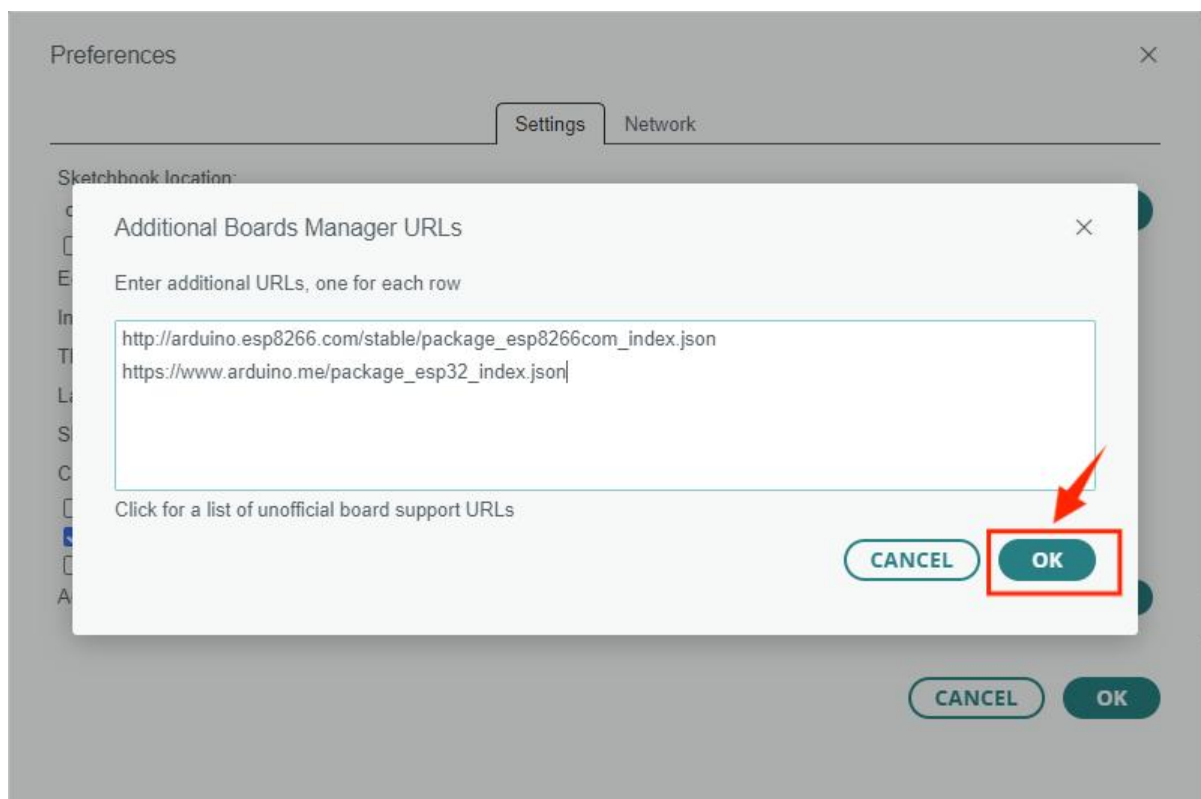


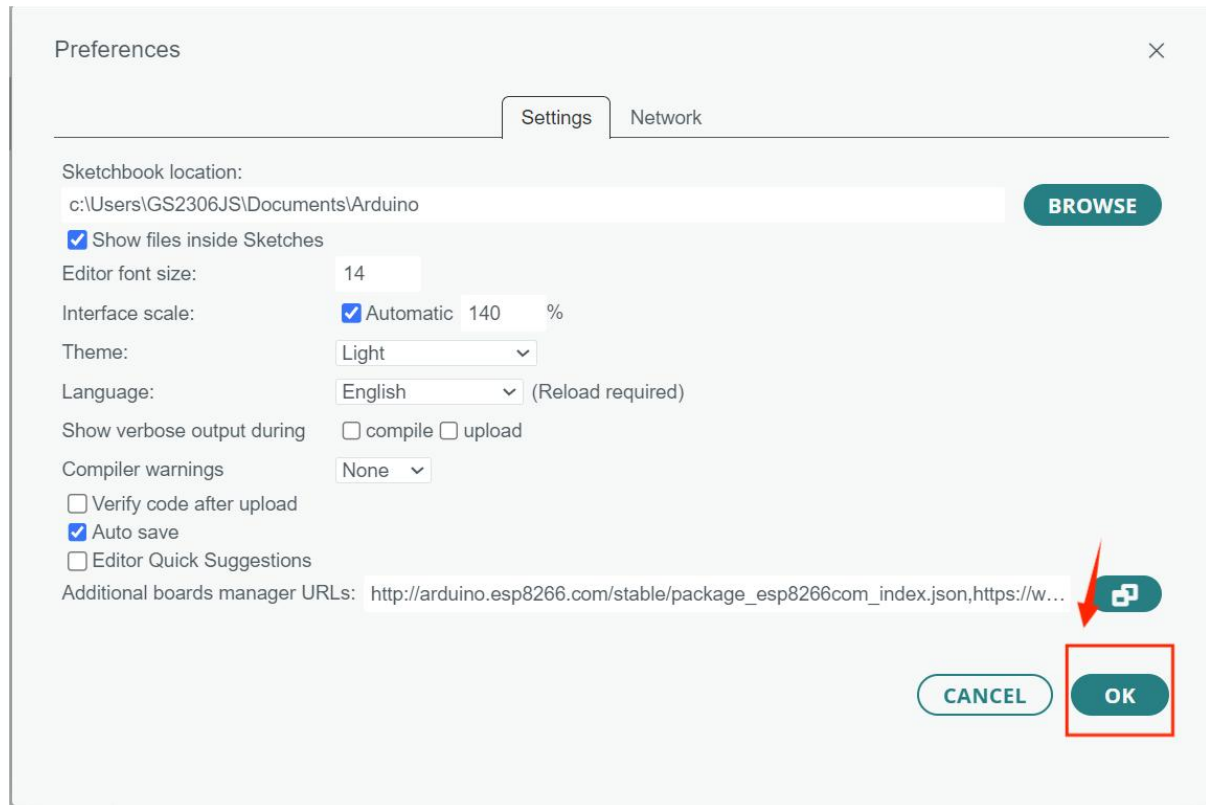
③ [Copy the URL in the lower text box and add it to "Additional Boards Manager URLs"](#)

http://arduino.esp8266.com/stable/package_esp8266com_index.json
https://www.arduino.me/package_esp32_index.json

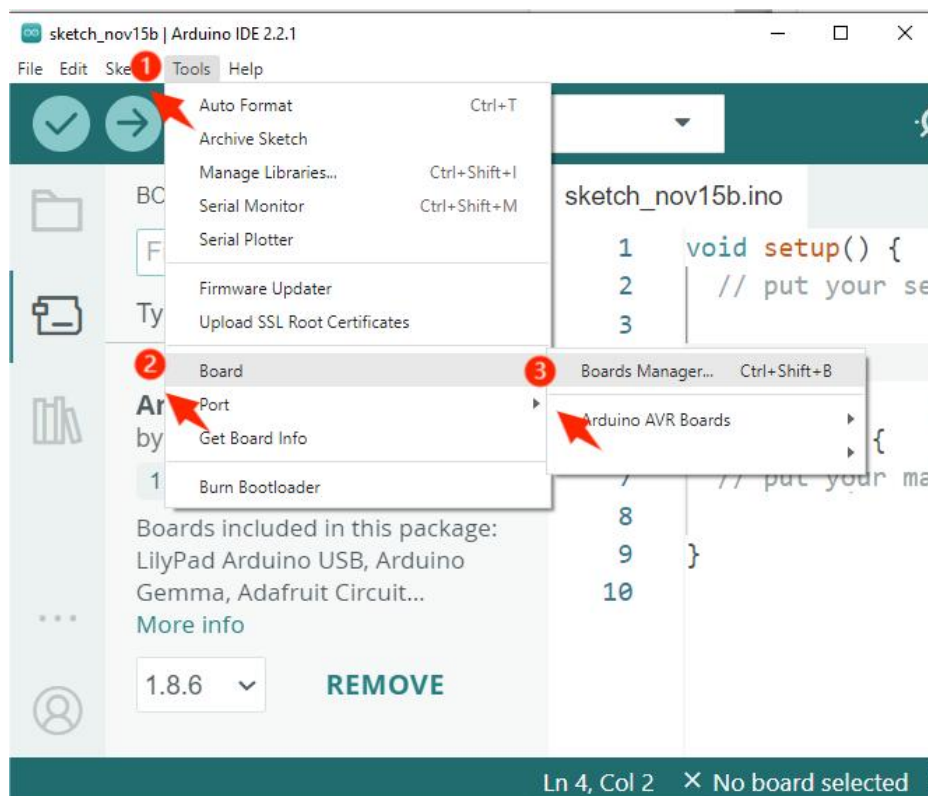


④After adding the URL, click "OK".



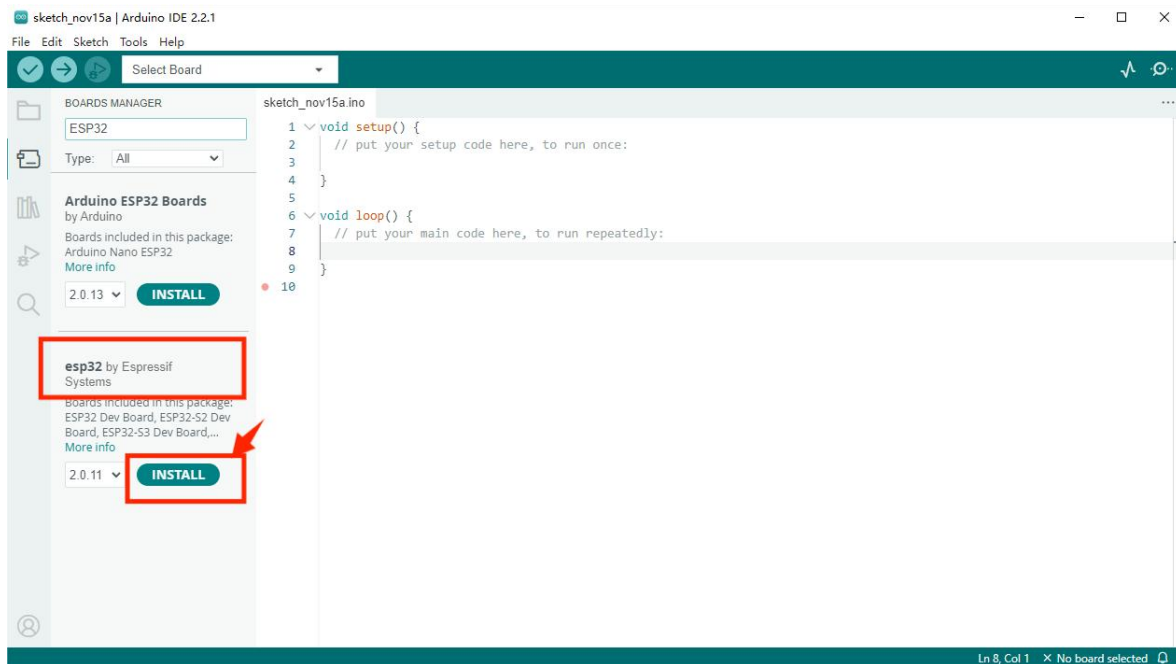


⑤ Click Tools > Board > Boards Manager.

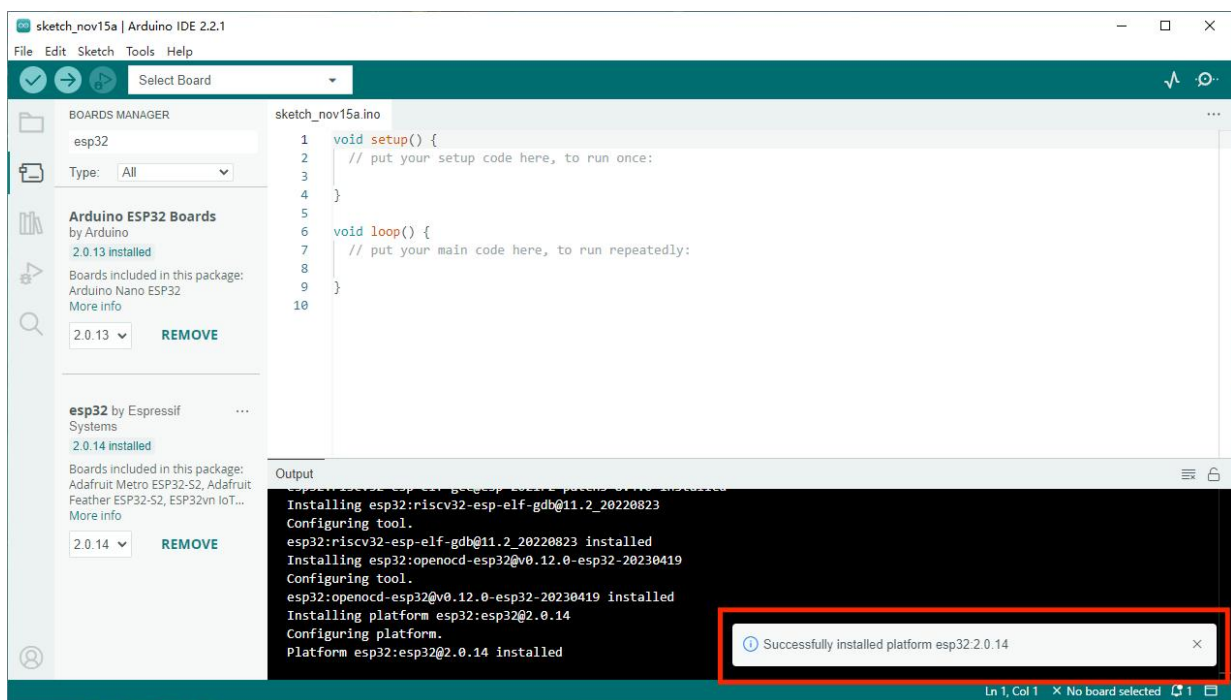


⑥ Search for "ESP32" in the BOARDS MANAGER's search bar and install it.

Attention: Please install version 2.0.12 of ESP32 because the newer versions are incompatible with the tutorial's libraries, which may cause errors in the program! If you have already installed version 3.0, please uninstall it and reinstall version 2.0 of ESP32.

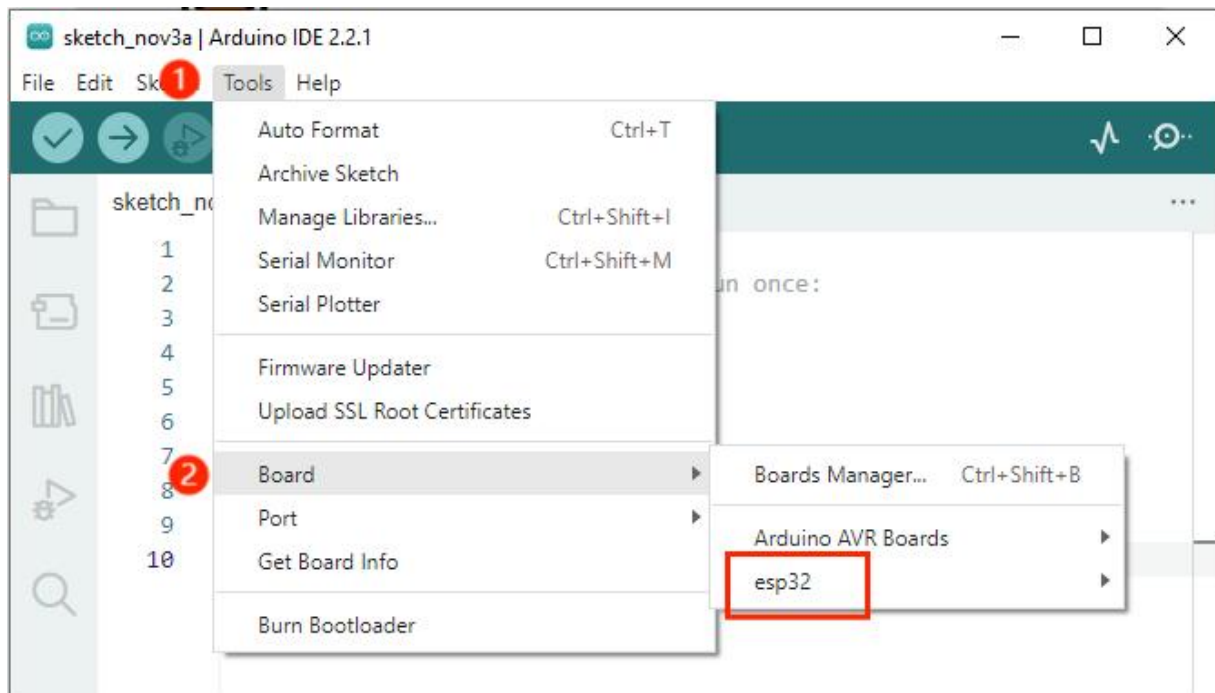


⑦ When the following interface appears, please wait until the installation is complete, then close Arduino IDE.



Attention: Since the installation package is hosted on GitHub, it may be affected by network speed. If the installation fails, try several times.

⑧Reopen Arduino IDE, select Tools > Board, and you will see the esp32 board appear.



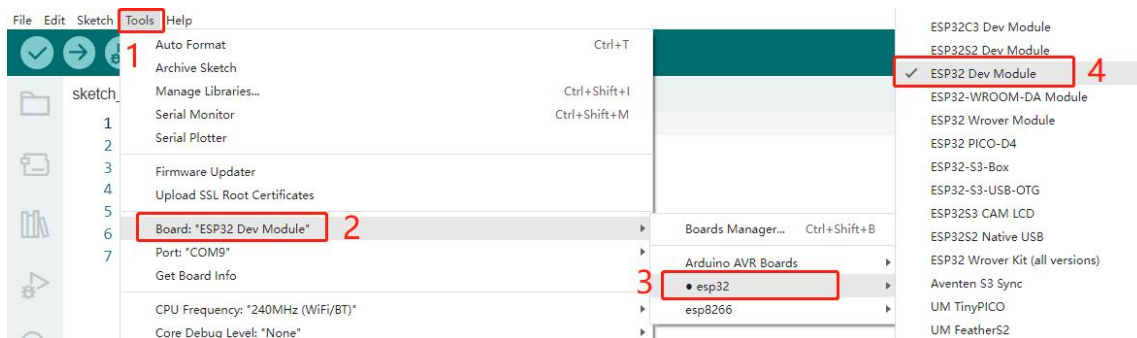
4.Add library files

Open the ["Add libraries"](#) file in English\Arduino\4.Add_libraries and follow the instructions to add them.

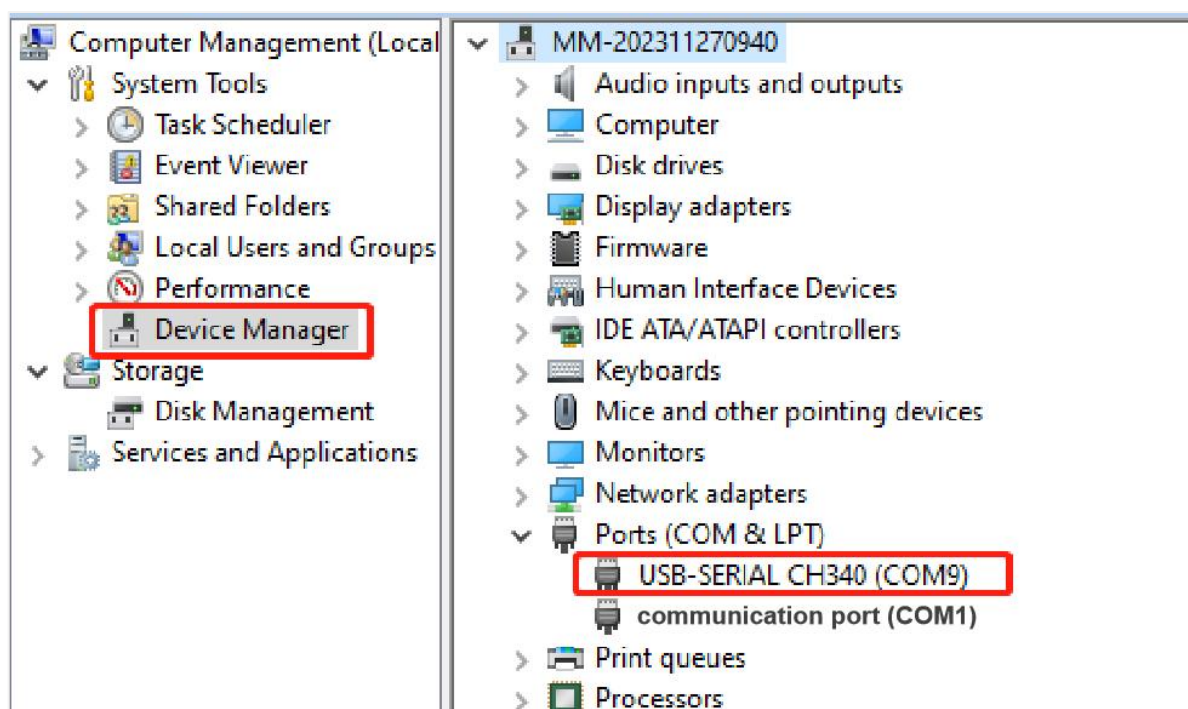
5.Test the development environment

After installing the Arduino IDE and the ESP32 board extension library, you can test if the development environment is set up successfully with a simple program. Please follow these steps:

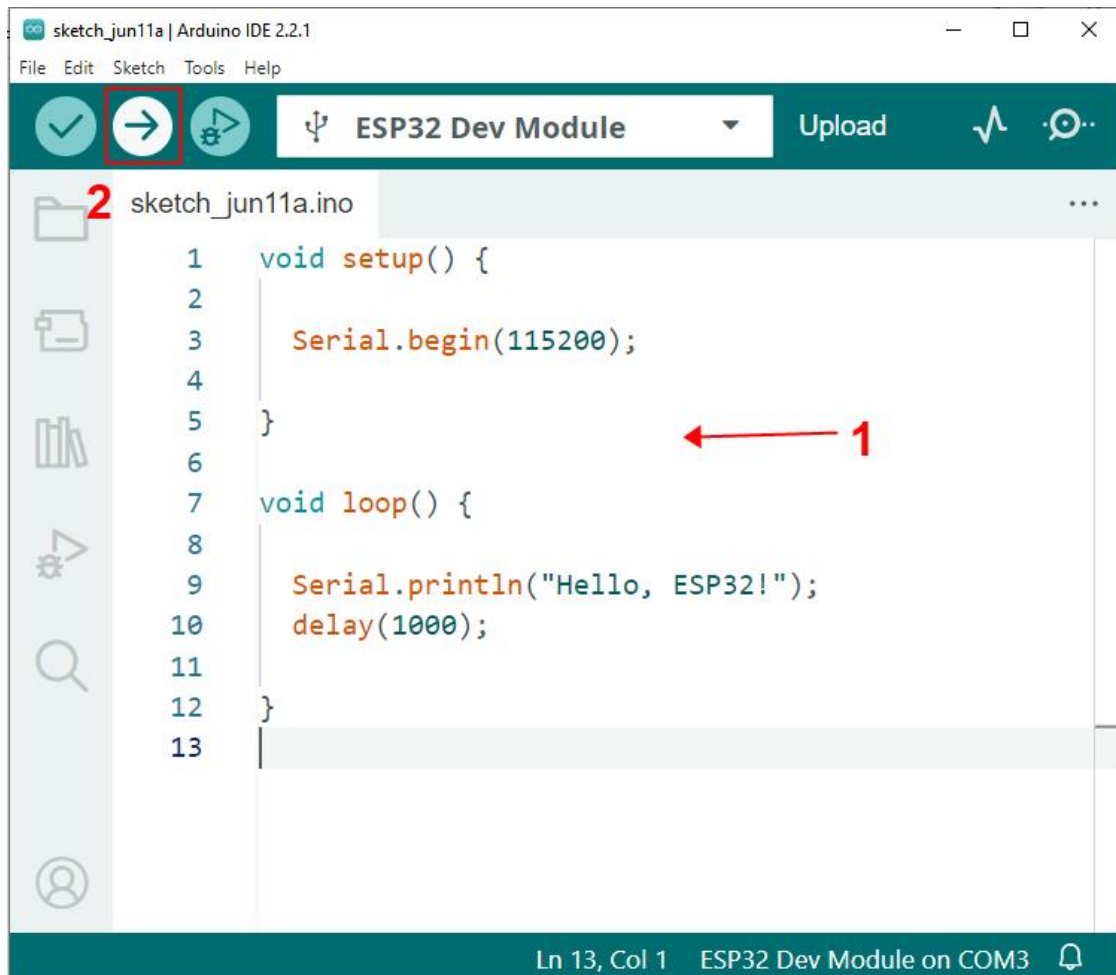
①Connect the controller board to your computer > Open Arduino IDE > Click on Tools > Select ESP32 > Choose (ESP32 Dev Module).

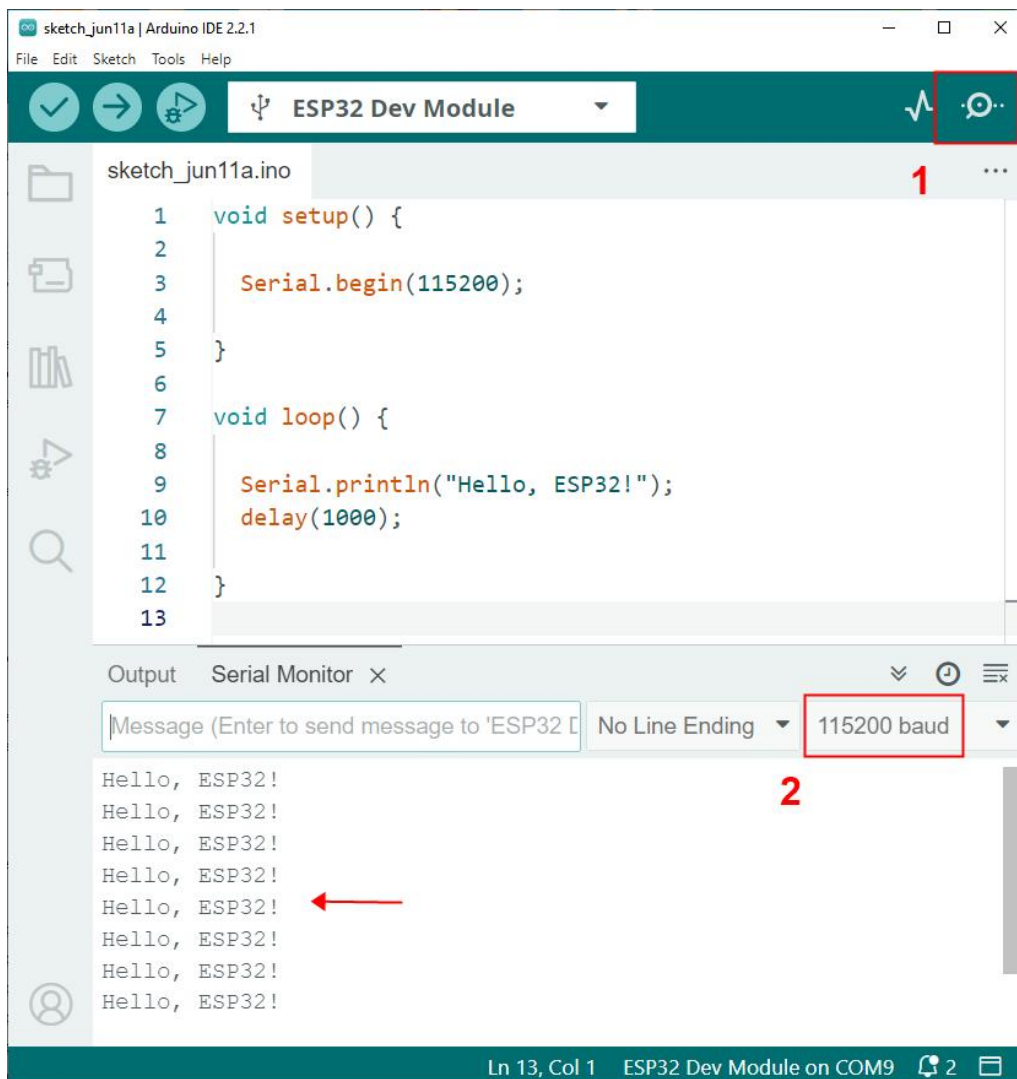


②Select the serial port (you can check the serial port number in the computer's Device Manager, then see if the corresponding port appears below. Since each board has a different COM number, choose the COM number displayed accordingly).



③Open ["Hello_esp32.ino"](#) in English\Arduino\2.Arduino program\Lesson 1\Hello_esp32, connect the ESP32 controller board to the computer with a USB cable, select the correct controller board and port, Upload the code to the ESP32 controller board, select the baud rate of 115200, you can see the serial port monitor constantly output "Hello, ESP32!" .





III.Understanding Servos

1.Introduction to servos

The main structure of the servo is shown in the following figure, which mainly has several parts: shell, variable speed gear set, motor, adjustable potentiometer, control circuit board, and steering wheel.

Its working principle is to control the circuit board to receive control signals from the signal source and drive the motor to rotate. The gear set reduces the speed of the motor many times and amplifies the output torque of the motor accordingly. The

potentiometer and the final stage of the gear set rotate together to measure the actual angle of rotation of the servo shaft. The control circuit board receives feedback from the potentiometer to determine the actual angle of the motor and compares it with the target angle. If there is an error, it controls the servo to rotate to the target angle position.

The workflow is as follows: Control Signal → Electronic Control Board → Motor Rotation → Gear Set Reduction → Servo Rotation → Feedback of Actual Motor Angle → Control Circuit Board adjusts the motor position based on feedback to the target angle.

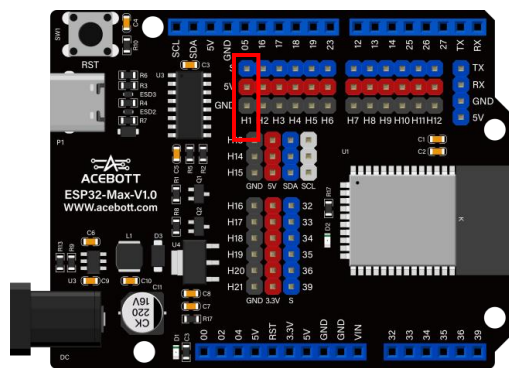
2.The pin definition of the steering gear

①Usually, a servo has three control wires: power wire, ground wire, and signal wire.



②Servo pin definitions: Brown wire - GND, Red wire - 5V, Orange wire - signal.

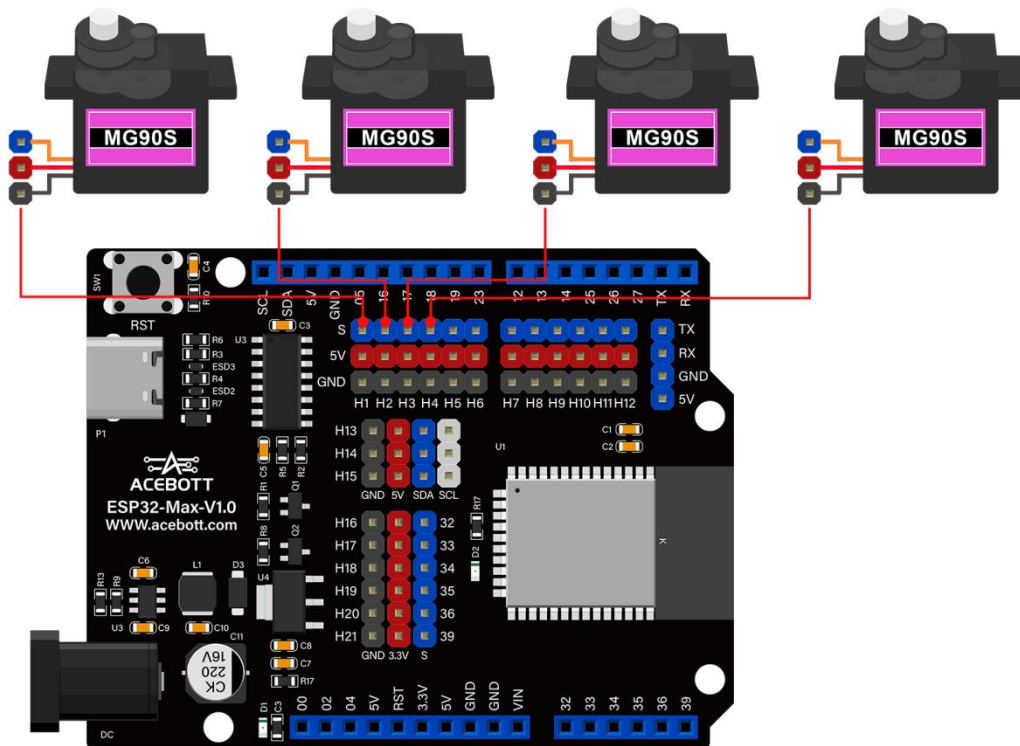
③Connect the servo to the ESP32 controller board according to the following diagram.

Servo	ESP32 Controller Board	Diagram
Brown wire	GND	
Red wire	5V	
Orange wire	GPIO5	

Attention: Please make sure to strictly follow the wiring instructions when connecting the module to the ESP32 controller board. Incorrect wiring may cause a short circuit and damage the ESP32 controller board.

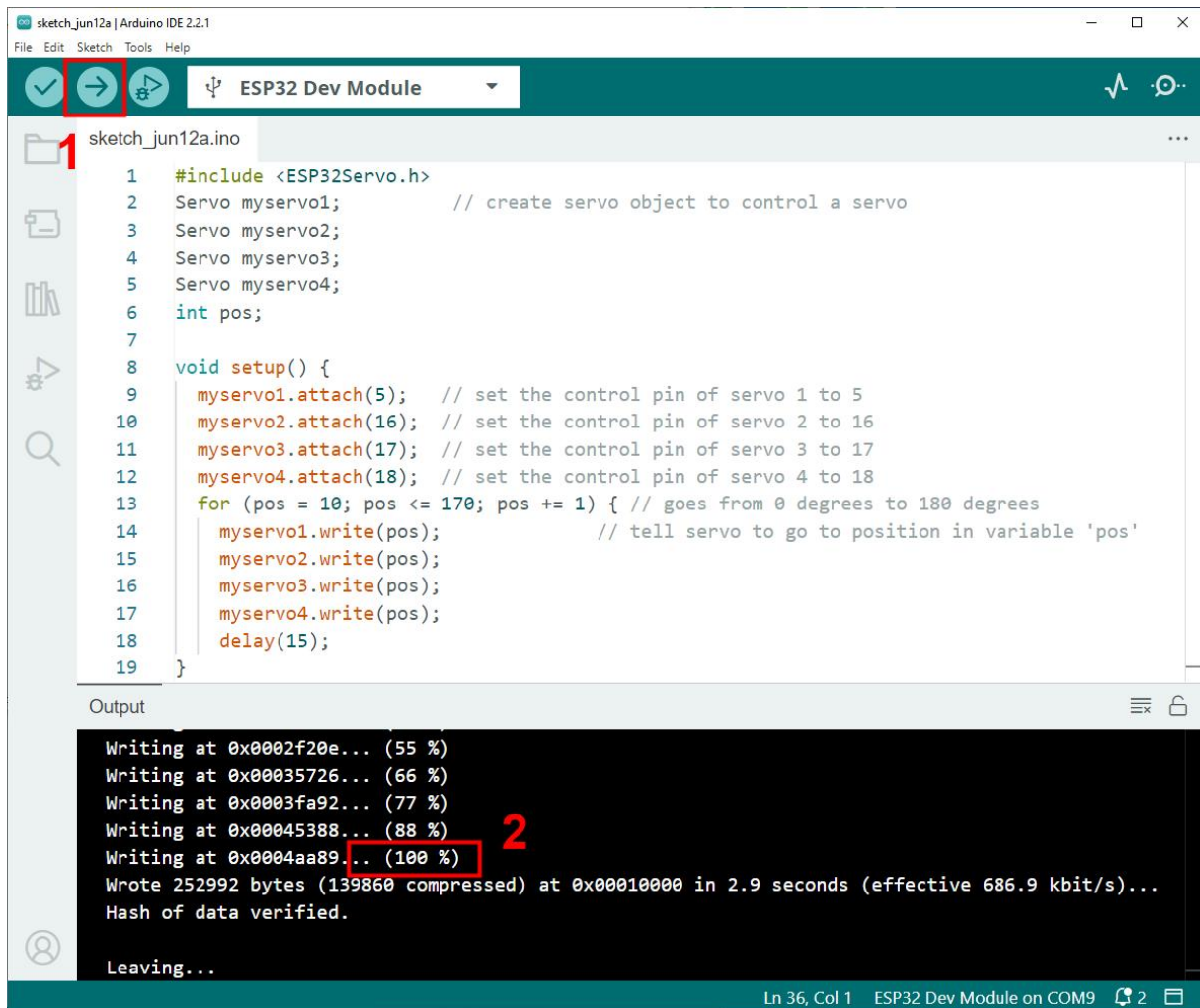
3.Servo testing

①Connect the four servos as shown in the diagram.



②Open "[Servo_test.ino](#)" in English\Arduino\2.Arduino program\Lesson 1\Servo_test, connect the ESP32 controller board to the computer with a USB cable, select the correct controller board, processor, and port, Upload the code to the ESP32 controller board.

Attention: To maintain a stable power output, you need to install 18650 batteries in the battery holder, connect it to the controller board's power port, and switch it to the "on" position before operating.



The screenshot shows the Arduino IDE interface. At the top, the board is set to 'ESP32 Dev Module'. The sketch file 'sketch_jun12a.ino' is open, displaying the following code:

```

1  #include <ESP32Servo.h>
2  Servo myservo1;           // create servo object to control a servo
3  Servo myservo2;
4  Servo myservo3;
5  Servo myservo4;
6  int pos;
7
8  void setup() {
9      myservo1.attach(5);    // set the control pin of servo 1 to 5
10     myservo2.attach(16);   // set the control pin of servo 2 to 16
11     myservo3.attach(17);   // set the control pin of servo 3 to 17
12     myservo4.attach(18);   // set the control pin of servo 4 to 18
13     for (pos = 10; pos <= 170; pos += 1) { // goes from 0 degrees to 180 degrees
14         myservo1.write(pos); // tell servo to go to position in variable 'pos'
15         myservo2.write(pos);
16         myservo3.write(pos);
17         myservo4.write(pos);
18         delay(15);
19     }

```

The 'Output' window at the bottom shows the upload progress:

```

Writing at 0x0002f20e... (55 %)
Writing at 0x00035726... (66 %)
Writing at 0x0003fa92... (77 %)
Writing at 0x00045388... (88 %)
Writing at 0x0004aa89... (100 %)
Wrote 252992 bytes (139860 compressed) at 0x00010000 in 2.9 seconds (effective 686.9 kbit/s)...
Hash of data verified.
Leaving...

```

Red annotations in the image include a '1' pointing to the sketch file name and a '2' pointing to the 100% progress bar in the output window.

③If the servo is functioning properly, it will rotate from 0 degrees to 180 degrees, then from 180 degrees back to 0 degrees, and finally settle at the 90-degree position.

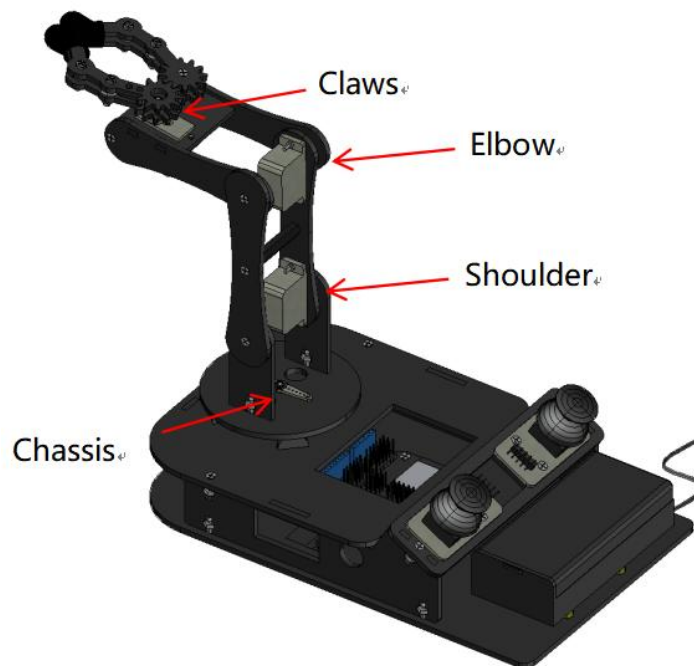
4.Servo Motor Zeroing Program

Before installing the robot arm, it's important to pre-flash the servo motor zeroing program to facilitate smooth assembly of its structure.




























Open ["servo_90.ino"](#) in English\Arduino\2.Arduino program\Lesson 1\Servo_90,connect the ESP32 controller board to the computer with a USB cable, select the correct controller board, processor, and port, Upload the code to the ESP32 controller board.

Lesson 2 Assembly of the robot arm

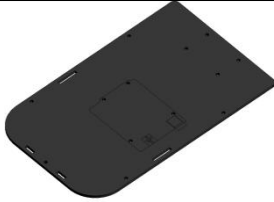
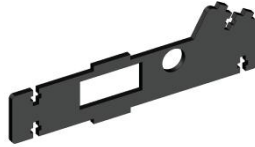


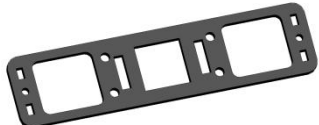
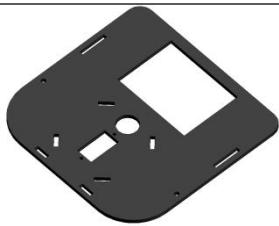

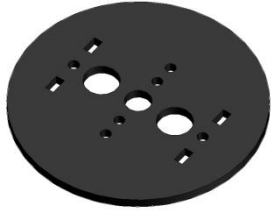

Before assembly, to facilitate the process and distinguish the servos at different positions, we will name the servos from top to bottom as Claws, Elbow, Shoulder, and Chassis.




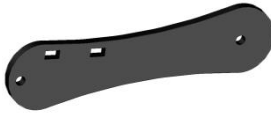
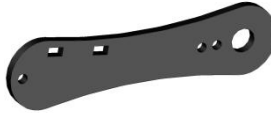
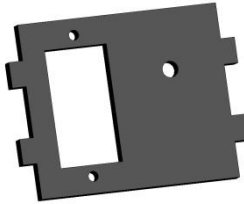




I .Parts List

 ESP32 Max V1.0 Controller Board 1PC	 Acrylic Board 1Set	 Servo MG90 9G 4PCS	 F-F 1P Dupont Wire 12PCS	 Joystick Module 2PCS	 USB Cable 1M 1PC
 18650 Battery Holder 1PC	 Nylon Cable Ties 2PCS	 Screwdriver 1PC	 L-Angled Socket Spanner 1PC	 Non-Slip Mat 6PCS	 Non-Slip Sleeve 4PCS
 M3*12MM Dual-pass Copper Pillar 9PCS	 M3*8MM Flat Head Screws 17PCS	 M3*10MM Flat Head Screws 24PCS	 M3*14MM Round Head Screws 5PCS	 M3 Nickel-Plated Nuts 24PCS	 M3 Nickel-Plated Lock Nuts 4PCS
 M2*10MM Round Head Screws 10PCS	 M2 Nickel-Plated Nuts 10PCS	 M1.7*6 Large Round Flat Head Tapping Screws 10PCS	 M3*22MM Flat Head Screws 2PCS	 M3*35 Nylon Column 1PC	 M3*40 Nylon Column 1PC
 M3*3 Nylon Gasket 6PCS	 M3*6 Nylon Gasket 2PCS	 Block 4PCS			

II .List of Structural Components

Acrylic Structure	Quantity	Image
Base plate	1	
Base plate bracket	1	
	1	
	1	
Joystick mounting plate	1	
Chassis servo mounting plate	1	
Pin	4	
Chassis disk	1	
Disk bracket 1	1	

Disk bracket 2	1	
Shoulder bracket 1	1	
Shoulder bracket 1	1	
Elbow bracket 1	1	
Elbow bracket 2	1	
Claw servo mounting plate	1	
Claw component 1	2	
Claw component 2	2	

III.Assembly Steps

Attention: If you want to watch the assembly video, please click the link below.

<https://www.youtube.com/watch?v=RtOe7knGhkl>

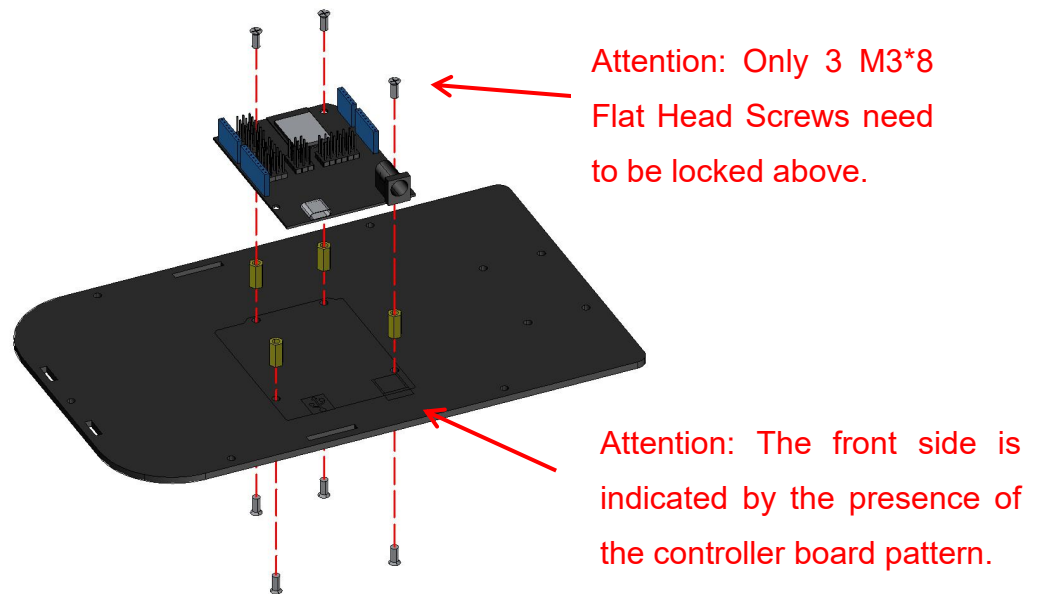
Or scan the QR code below.



1.Remove the protective paper attached to the acrylic structure

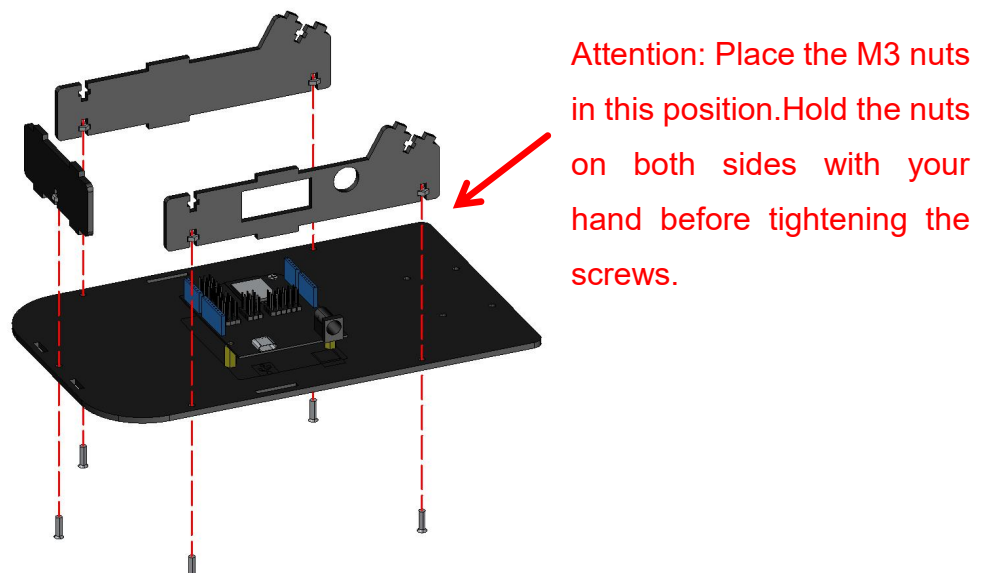
2.Installing the ESP32 controller board on the base plate

Part List	
Name	Quantity
esp32 Controller Board	1
Base Plate	1
M3*8 Flat Head Screw	7
M3*12 Double-pass Copper Pillar	4



3.Installing the base plate

Part List	
Name	Quantity
Base Plate Bracket	3
M3*10 Flat Head Screw	5
M3 Nut	5

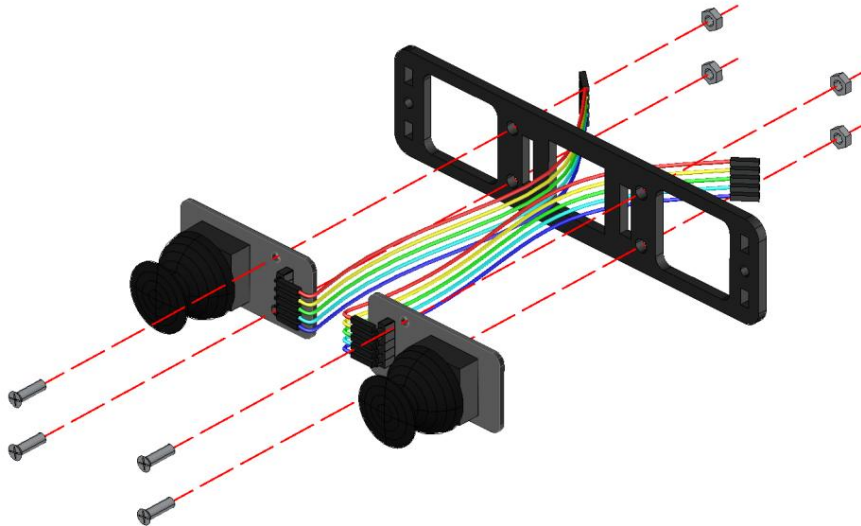


4.Install the joystick module

Part List	
Name	Quantity
Joystick Module	2

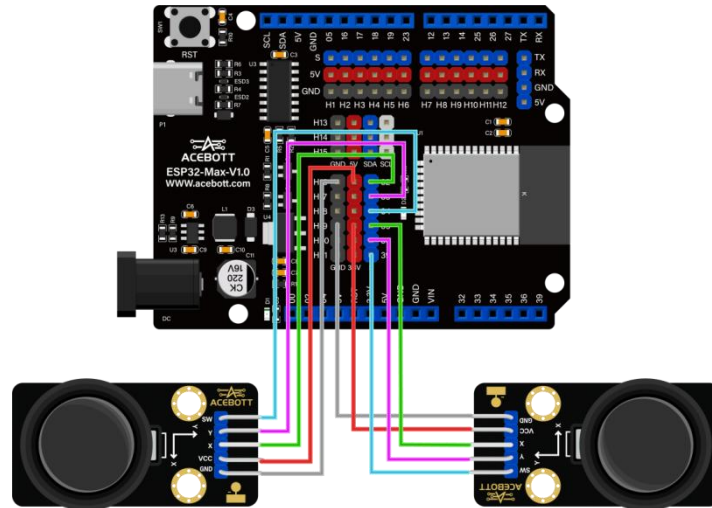
Joystick Mounting Plate	1
M3*10 Flat Head Screws	4
M3 Nuts	4

Attention: When the joystick module is connected to the dupont wire, the actual Dupont wire color may be different from the figure, so you only need to follow the pin identification on the joystick module to connect.



This step requires connecting the wires from the two joystick modules to the mainboard first. For the left joystick module, connect SW to pin 34, X to pin 32, Y to pin 33, and connect VCC and GND to the VCC and GND pins in the same row as pin 32. For the right joystick module, connect SW to pin 39, X to pin 35, Y to pin 36, and connect VCC and GND to the VCC and GND pins in the same row as pin 35.

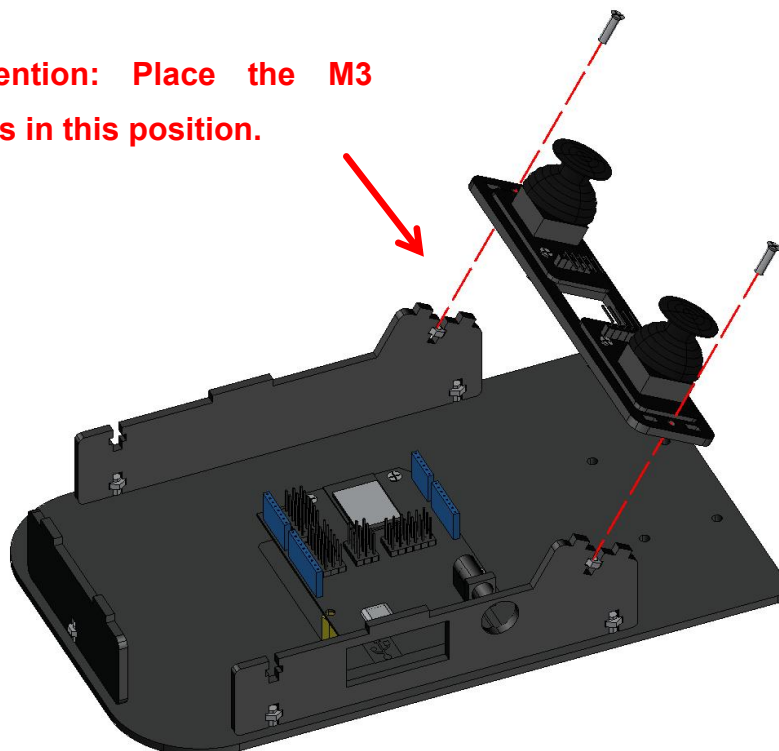
Attention: Please make sure to strictly follow the wiring instructions when connecting the module to the ESP32 controller board. Incorrect wiring may cause a short circuit and damage the ESP32 controller board.



5.Install the joystick mounting plate

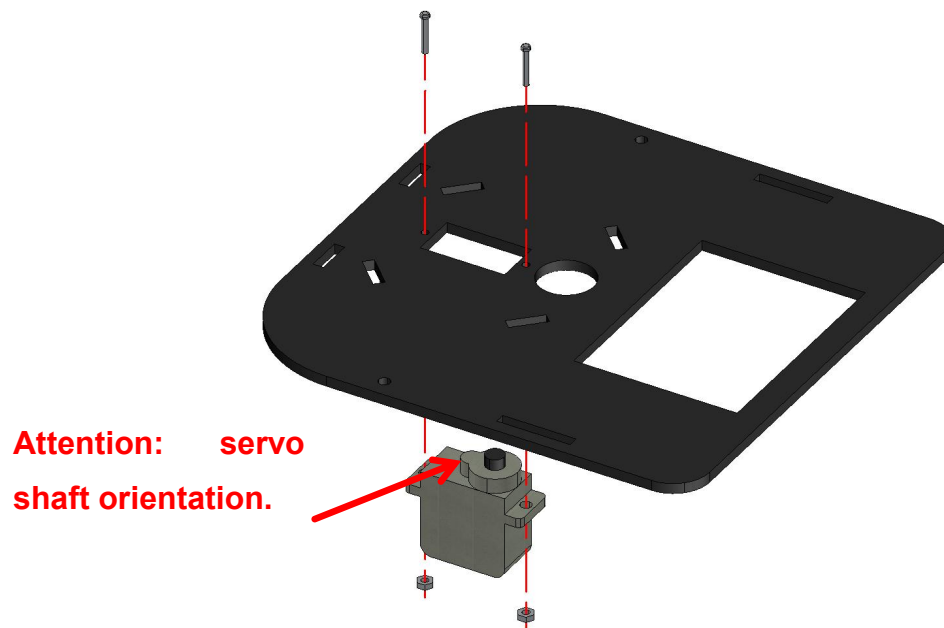
Part List	
Name	Quantity
M3*10 Flat Head Screw	2
M3 Nut	2

Attention: Place the M3 nuts in this position.



6.Install chassis servo

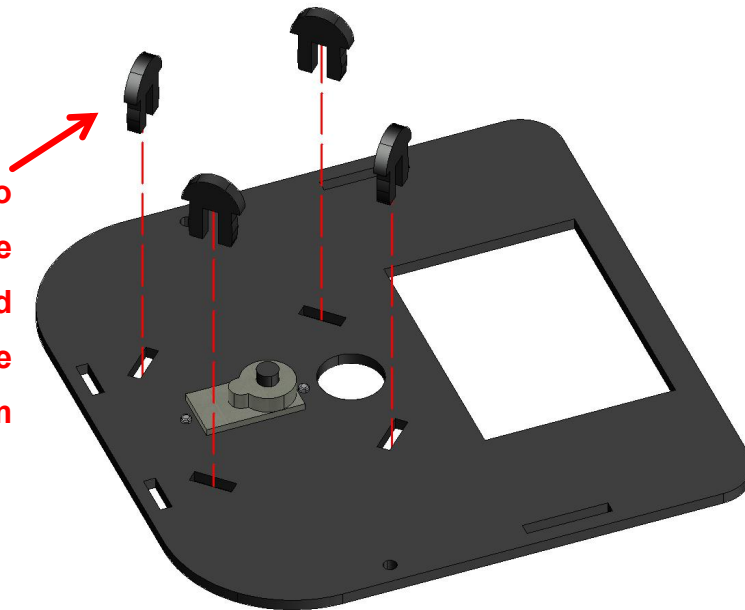
Part List	
Name	Quantity
Servo	1
Chassis Servo Mounting Plate	1
M2*10 Round Head Screw	2
M2 Nut	2



7.Install pin

Part List	
Name	Quantity
Pin	4

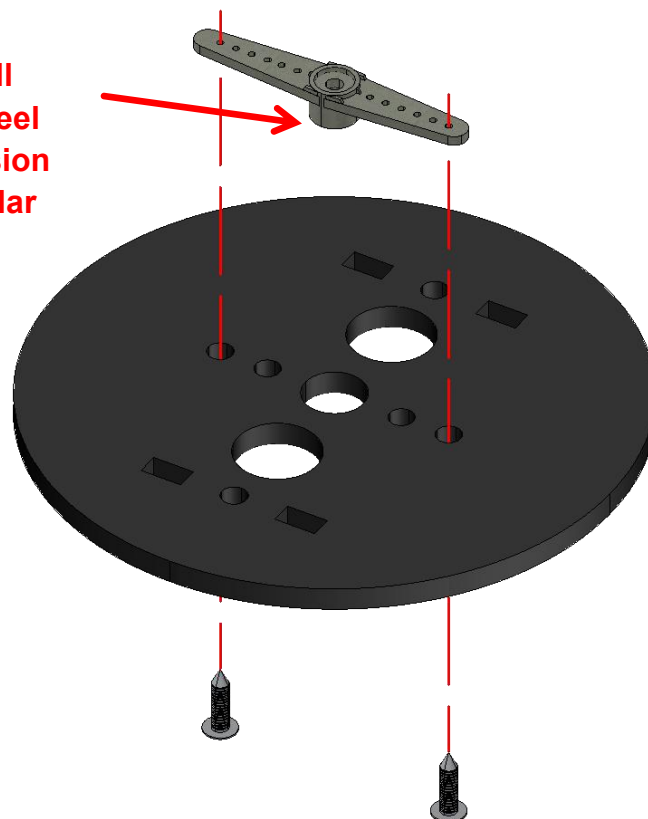
Attention: Do not press the pin too hard to prevent the pin from breaking.



8.Install the steering wheel of the chassis servo

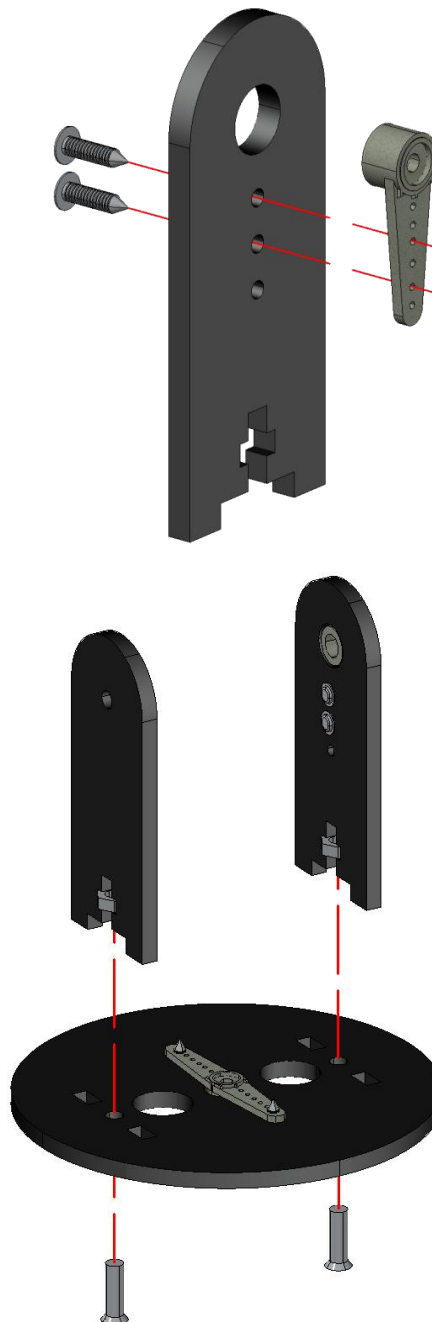
Part List	
Name	Quantity
Straight Steering Wheel	1
Chassis Disk	1
M1.7*6 Large Round Flat Head Tapping Screw	2

Attention: Install the steering wheel with the protrusion facing the circular hole.



9. Install the disk bracket

Part List	
Name	Quantity
Half Straight Steering Wheel	1
Disk bracket 1	1
Disk bracket 2	1
M1.7*6 Large Round Flat Head Tapping Screws	2
M3*10 Flat Head Screws	2
M3 Nuts	2

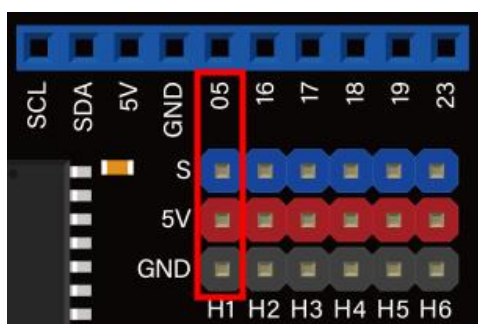


10.Fixed robot arm chassis structure

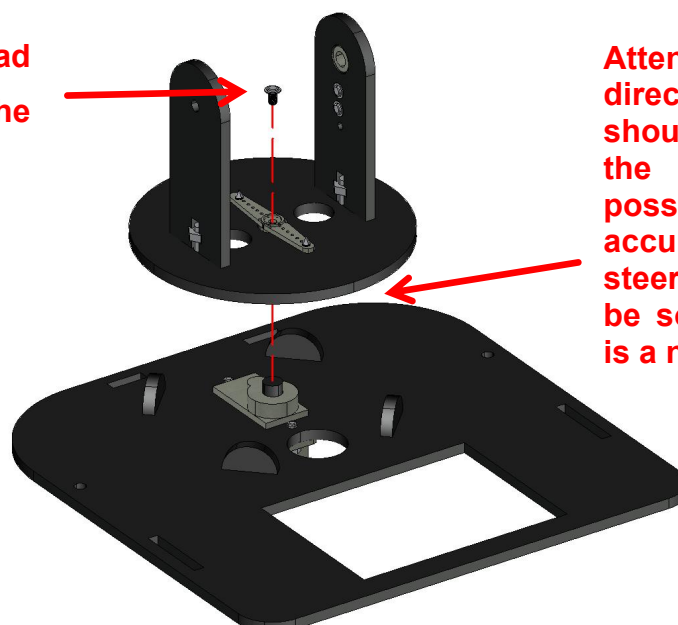
Part List	
Name	Quantity
M2.5*4 Round Head Screw	1

Attention: ①Before this step of installation, first connect the servo to the GPIO5 pin of the controller board, and then turn on the power of the controller board to keep the servo in the 90° position.

②In order to avoid damage to the steering gear during installation, please do not rotate the shaft of the steering gear.



M2.5*4 round head screws are in the bag of servo.

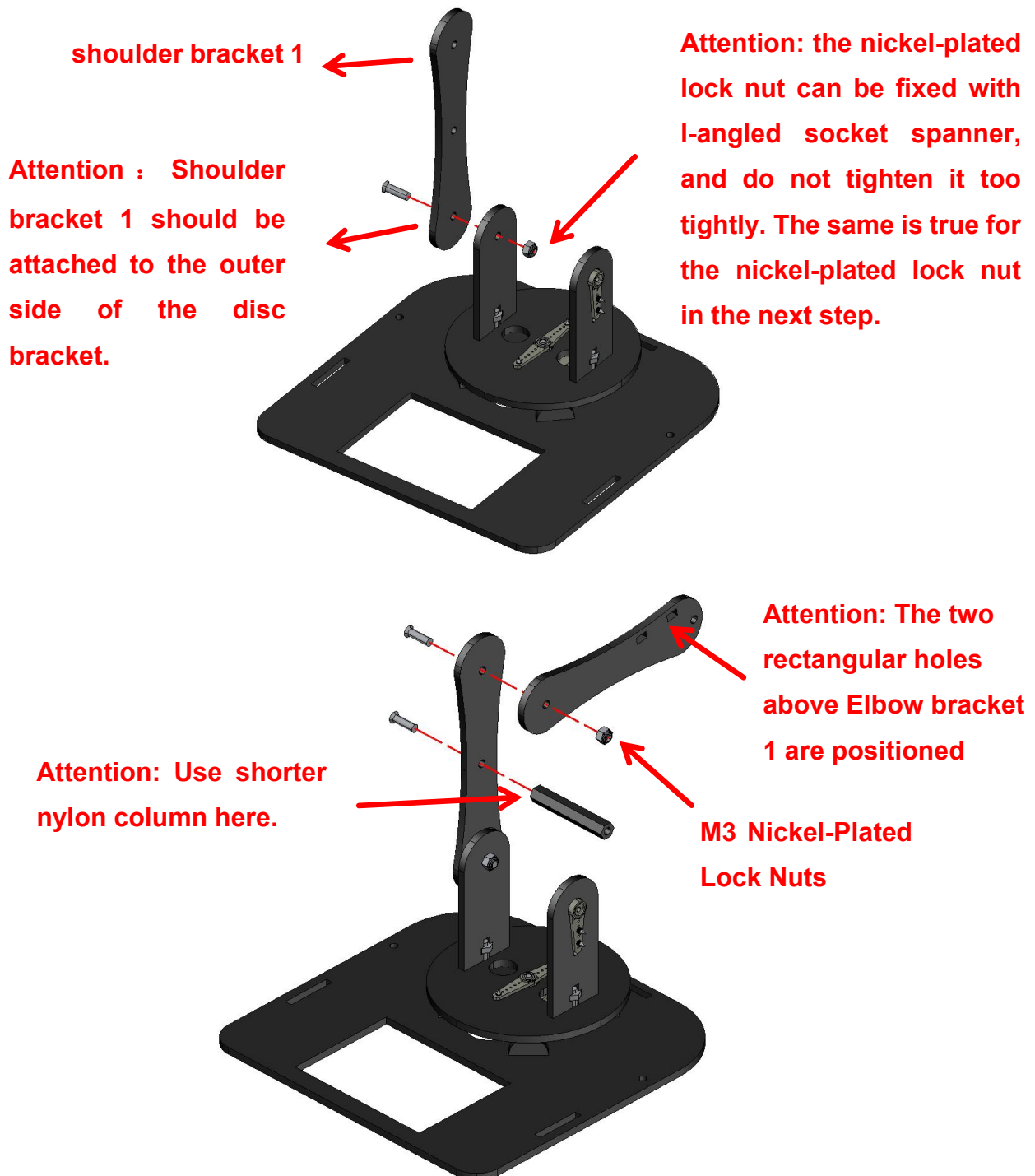


Attention: The installation direction of this structure should be consistent with the figure as far as possible. Due to the accuracy problem of the steering gear, there may be some deviation, which is a normal situation.

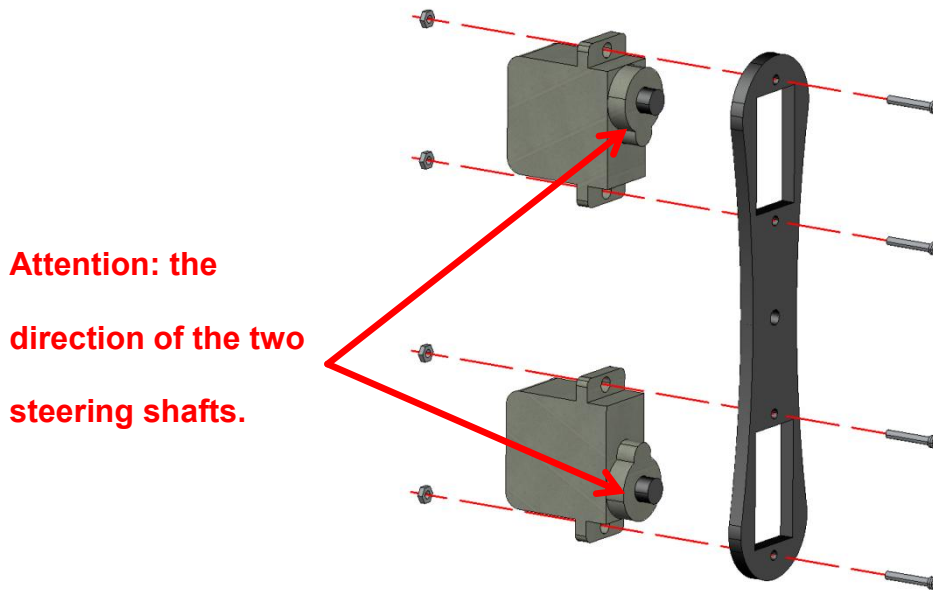
11.Install the shoulder bracket

Part List	
Name	Quantity

Shoulder Bracket 1	1
Elbow Bracket 1	1
M3*35 Nylon Column	1
M3*10 Flat Head Screw	3
M3 Nickel-Plated Lock Nut	2



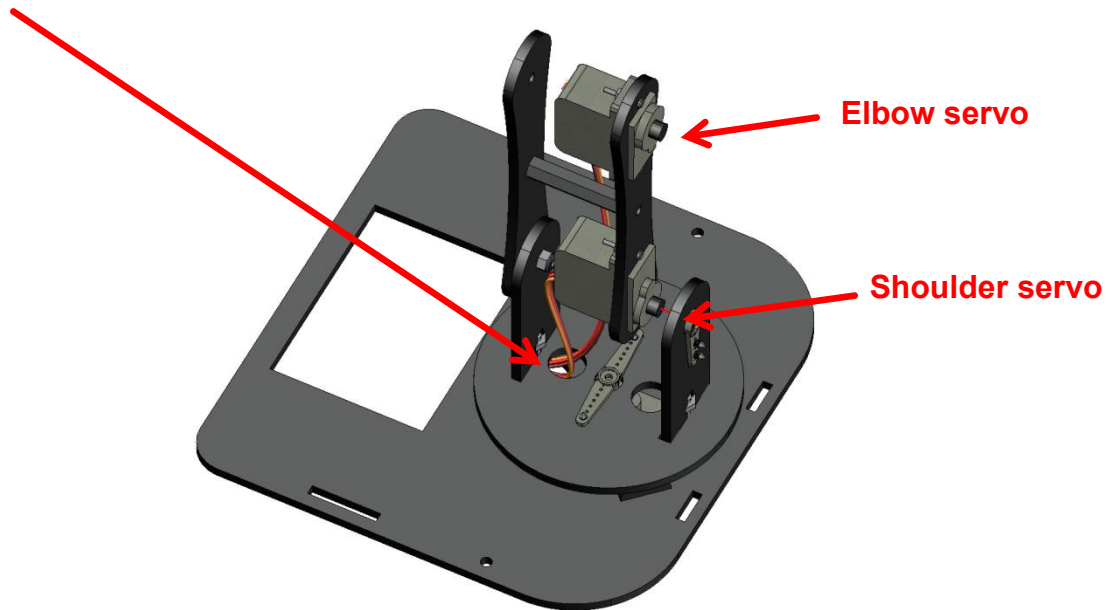
Part List	
Name	Quantity
Servo	2
Shoulder Bracket 2	1
M2*10 Round head screw	4
M2 Nut	4



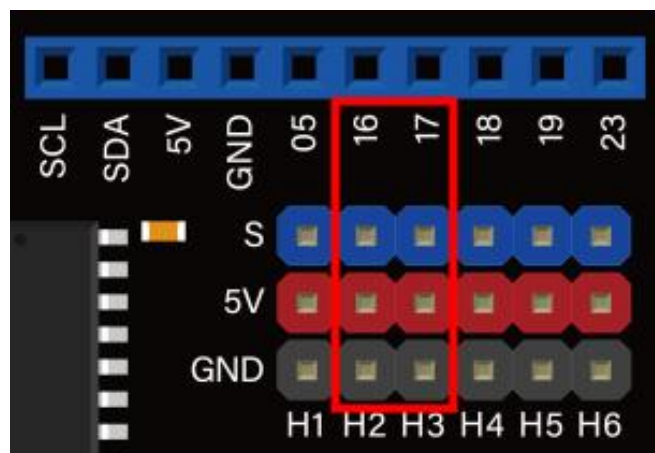
Connecting the shoulder servo and elbow servo to the ESP32.

Attention: ①First, unplug the USB cable, disconnect the power supply of the controller board, and make the robot arm servo in a state of no power;

②Then, the threaded hole on the chassis disc is gently rotated to align with the threaded hole on the bottom plate, and then the dupont lines of the shoulder servo and elbow servo of the robot arm are passed through the two threaded holes.

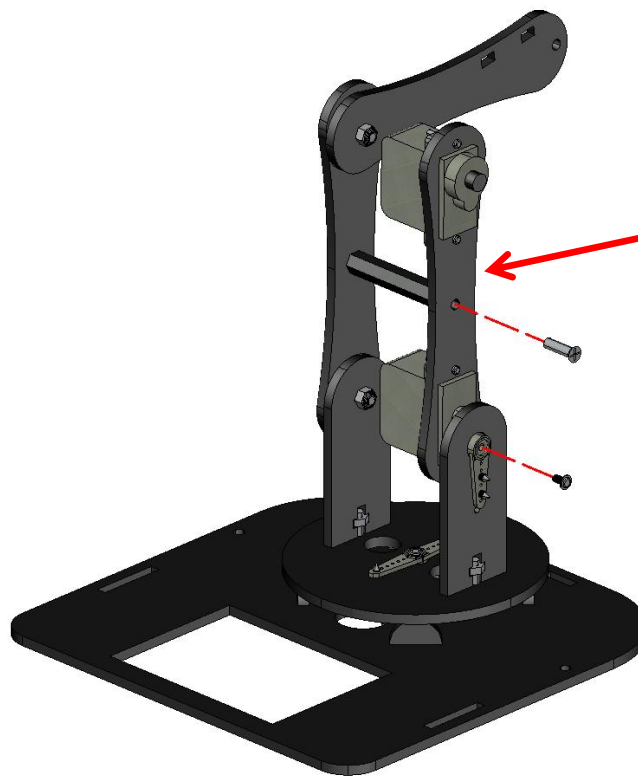


③Connect the shoulder servo wire to GPIO16 pin, and the elbow servo wire to GPIO17 pin.



④Then turned on the power again, and keep the chassis, shoulder and elbow servos at 90° position.

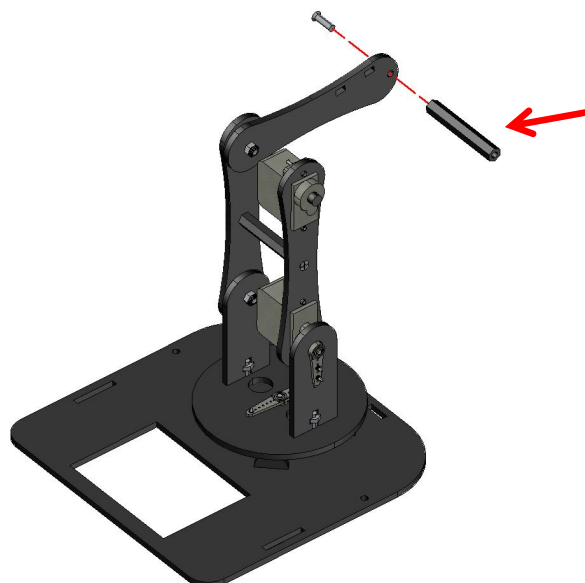
Part List	
Name	Quantity
M2.54 Round Head Screw	1
M3*10 Flat Head Screw	1



Attention:The shoulder bracket should be installed vertically to the ground before tightening the screws of the servo motor and nylon column.

12.Fixed M3*40 nylon column

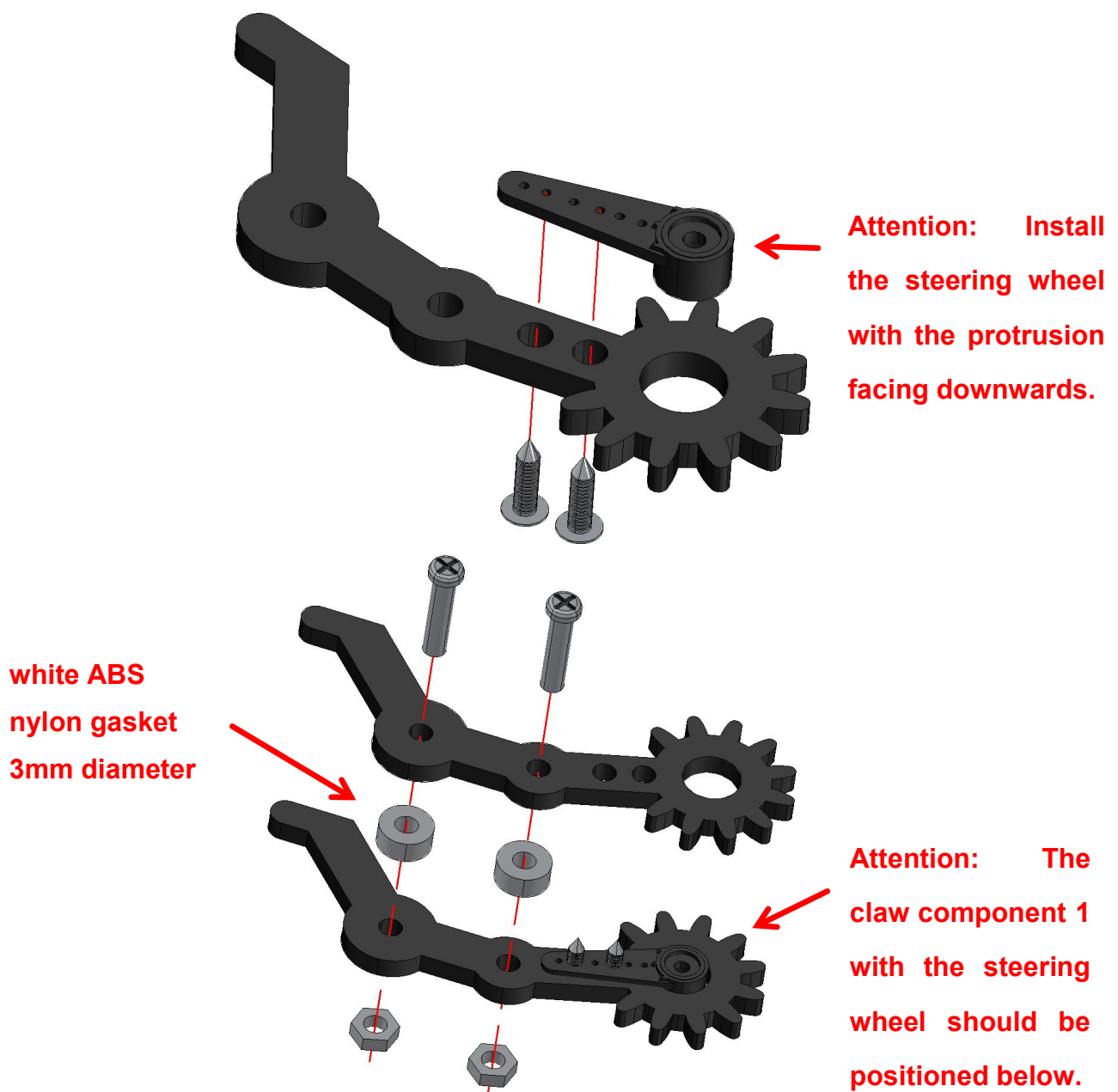
Part List	
Name	Quantity
M3*40 Nylon Column	1
M3*10 Flat Head Screw	1

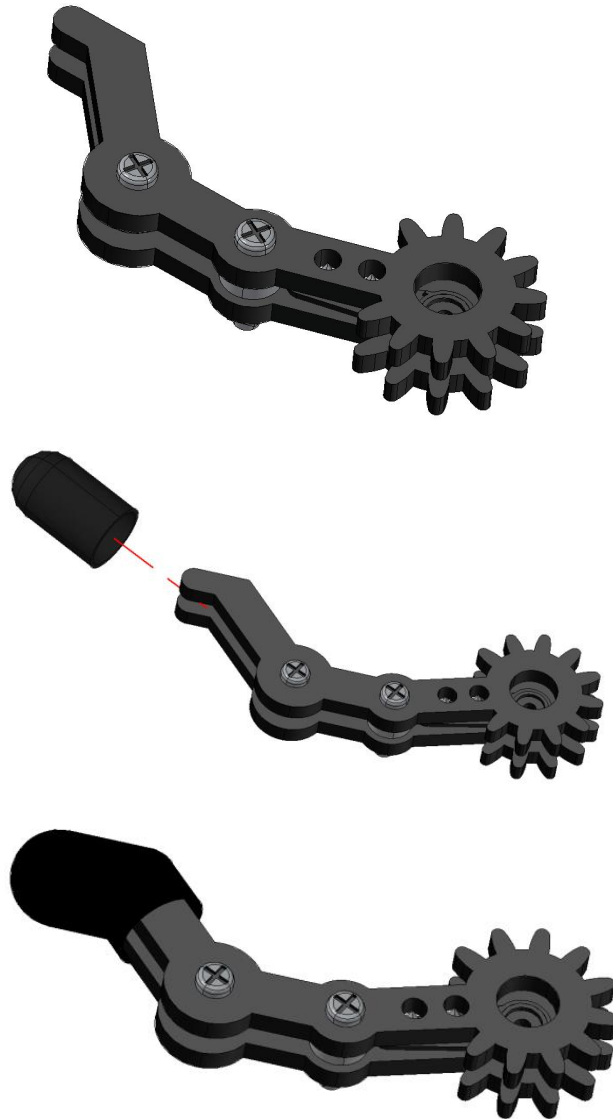


Attention: Use longer nylon column here.

13.Install the left paw structure of the robot arm

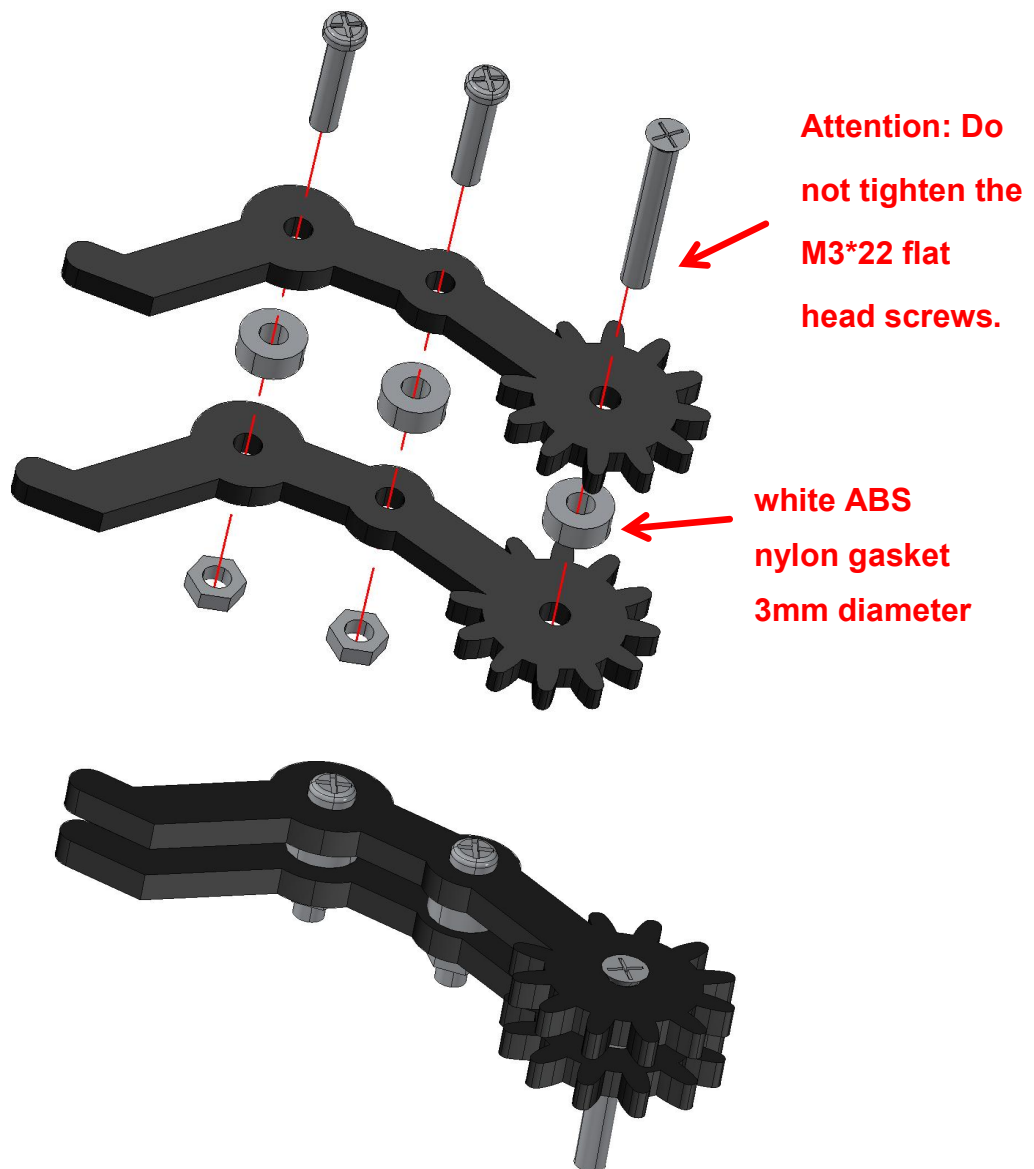
Part List	
Name	Quantity
Half Straight Steering Wheel	1
Claw Component 1	2
M1.7*6 Large Round Flat Head Tapping Screws	2
M3*14 Round Head Screws	2
3MM White ABS Nylon Gaske	2
M3 Nuts	2
Non-Slip Sleeve	1



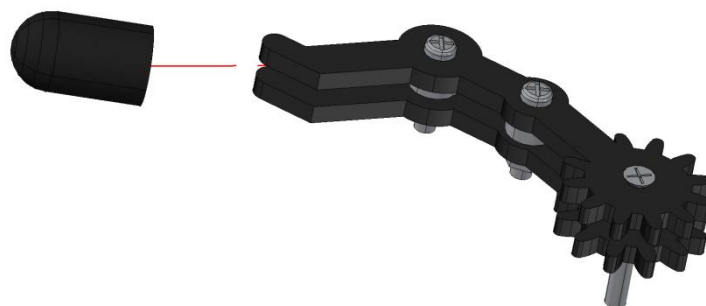


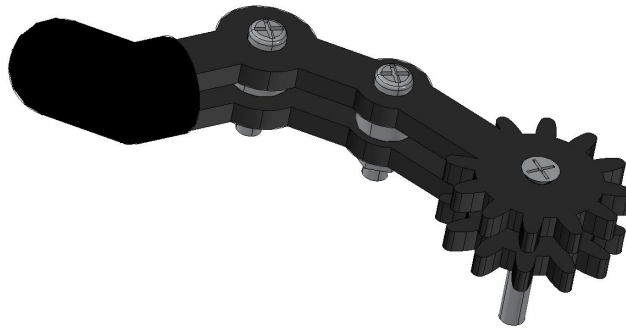
14.Install the right paw structure of the robot arm

Part List	
Name	Quantity
Claw Component 2	2
M3*22 Flat Head Screws	1
M3*14 Round Head Screws	2
3MM White ABS Nylon Gaske	3
M3 Nuts	2
Non-Slip Sleeve	1



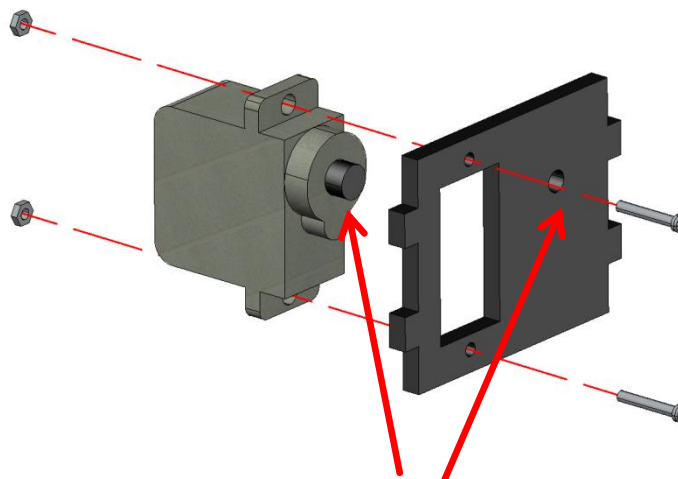
Install the non-slip sleeve.





15.Fixed the robot arm's claw servo mounting plate

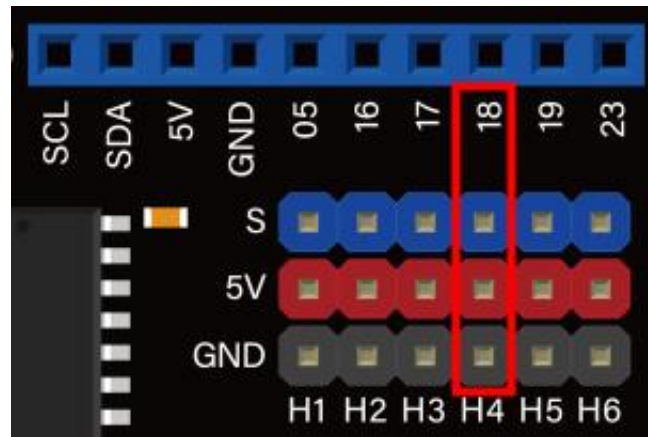
Part List	
Name	Quantity
Servo	1
Claws Servo Mounting Plate	1
M2*10 Round Head Screw	2
M2 Nut	2



Attention: Ensure the steering shaft and the circular hole are both positioned upwards.

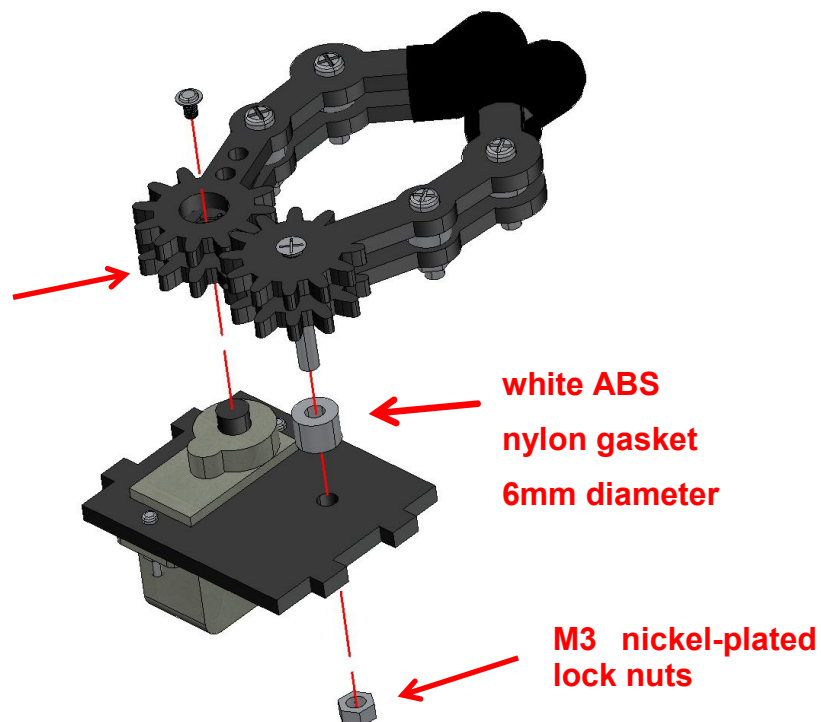
Connect the wire of the claws servo to the GPIO18 pin.

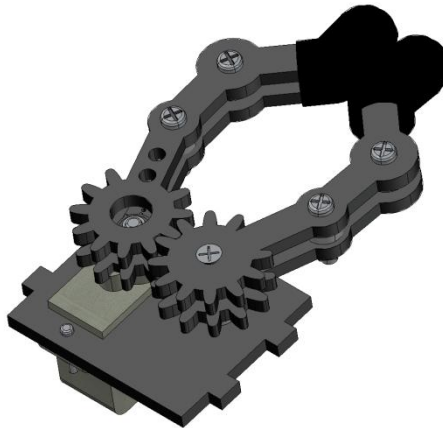
Attention: For this step, please power on the controller board and ensure the claws servo remains at the 90° position.



Part List	
Name	Quantity
M2.5*4 Round Head Screw	1
6MM White ABS Nylon Gaske	1
M3 Nickel-Plated Lock Nut	1

Attention: The initial form of the claw is closed.

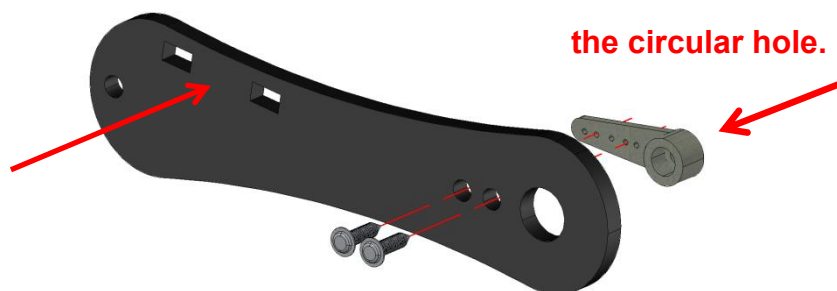




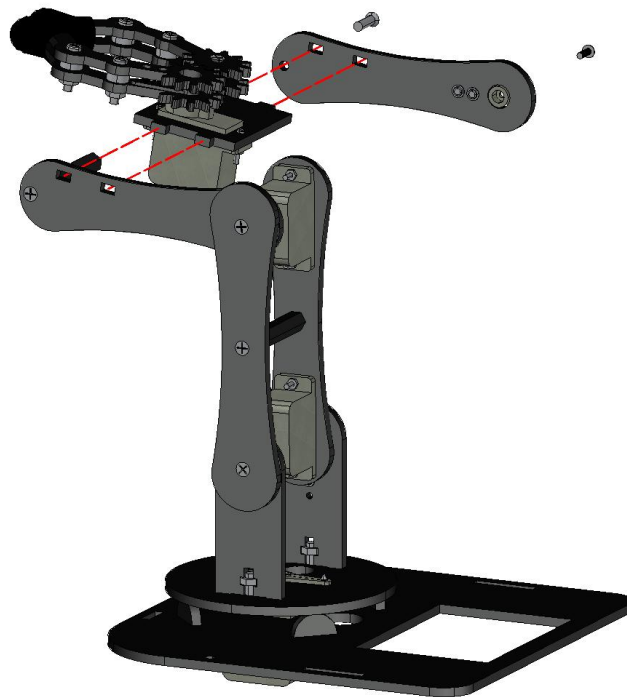
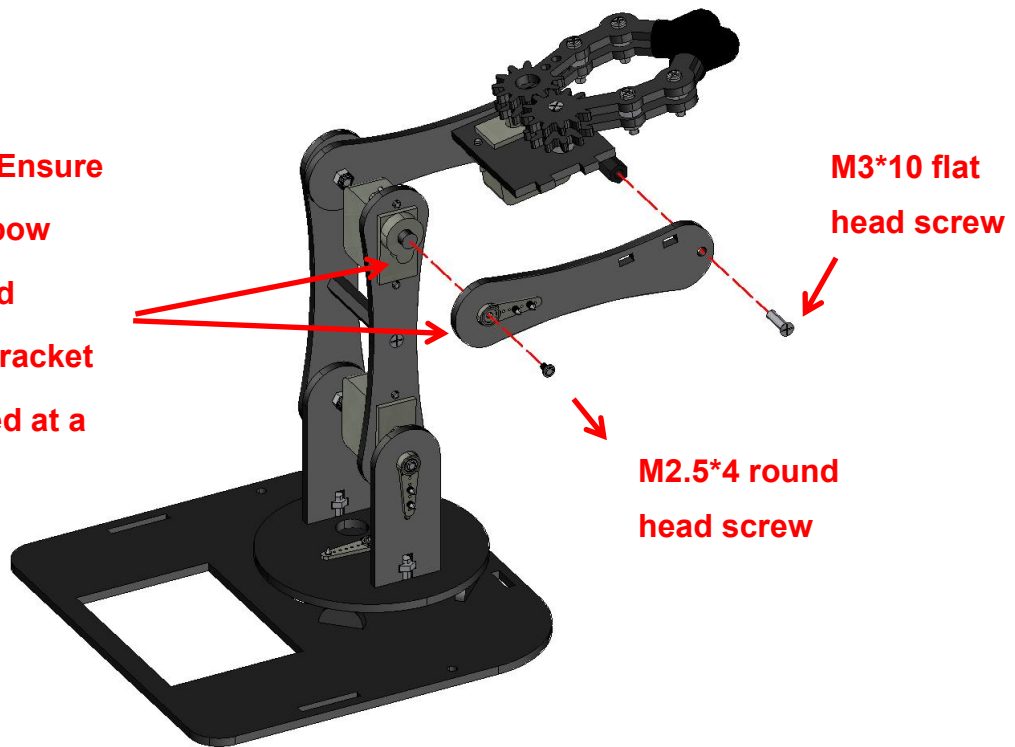
Part List	
Name	Quantity
Half Straight Steering Wheel	1
Elbow Bracket 2	1
M1.7*6 Large Round Flat Head Tapping Screws	2
M3*10 Flat Head Screw	1
M2.5*4 Round Head Screw	1

Attention: Install the steering wheel with the protrusion facing towards the circular hole.

Attention: Ensure the rectangular hole is facing upwards.



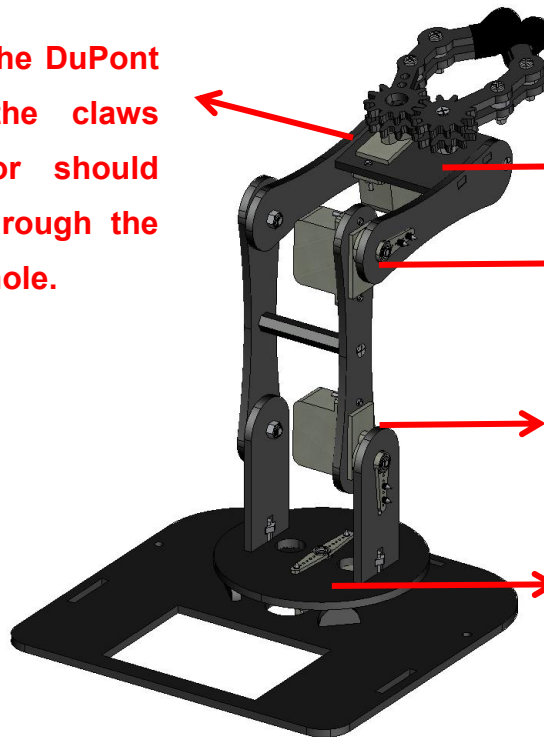
Attention: Ensure that the elbow bracket and shoulder bracket are installed at a 90° angle.



16. Organize the servo wires

Please ensure that all four servo motors' DuPont wires are correctly connected to the controller board.

Attention: The DuPont wires of the claws servo motor should not pass through the concentric hole.



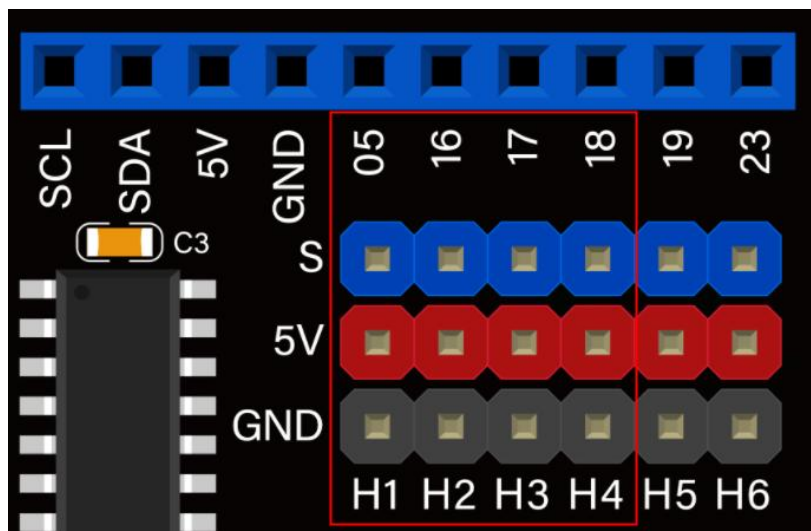
The claws servo motor is connected to pin 18.

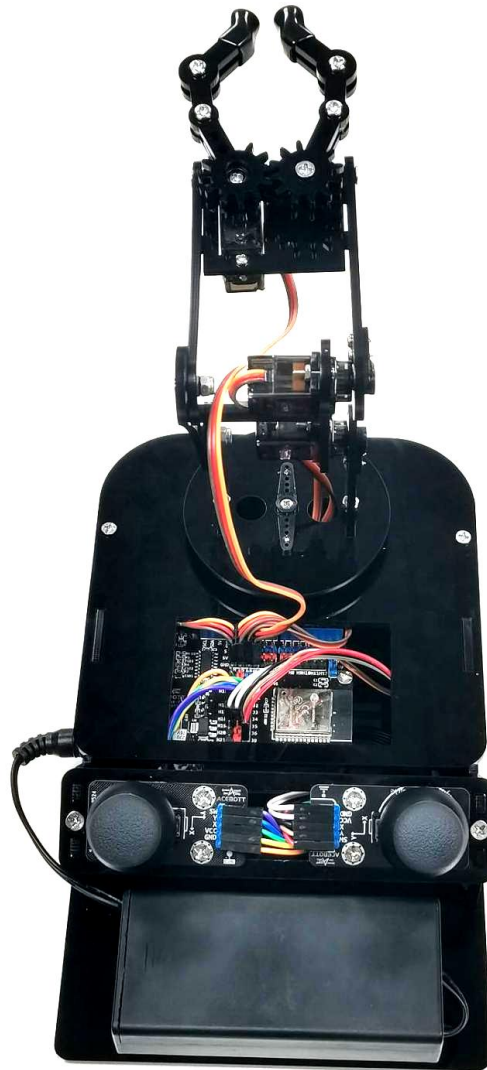
The elbow servo motor is connected to pin 17.

The shoulder servo motor is connected to pin 16.

The chassis servo motor is connected to pin 5.

Attention: Please make sure to strictly follow the wiring instructions when connecting the module to the ESP32 controller board. Incorrect wiring may cause a short circuit and damage the ESP32 controller board.

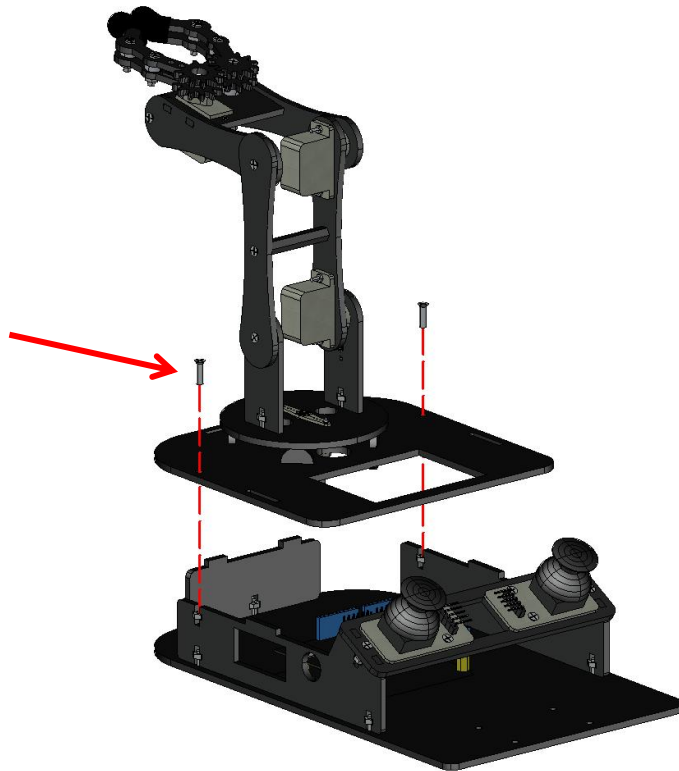




17.Fixed the base of the robot arm

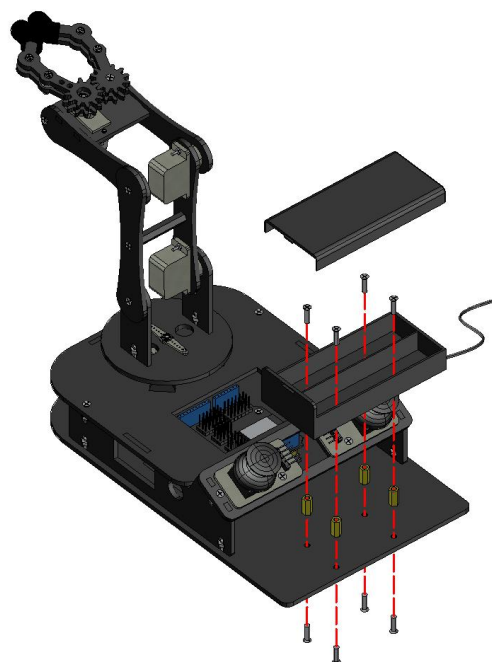
Part List	
Name	Quantity
M3*10 Flat Head Screws	2
M3 Nuts	2

Attention: Ensure all four servo motor wires are connected to the controller board before tightening the screws.



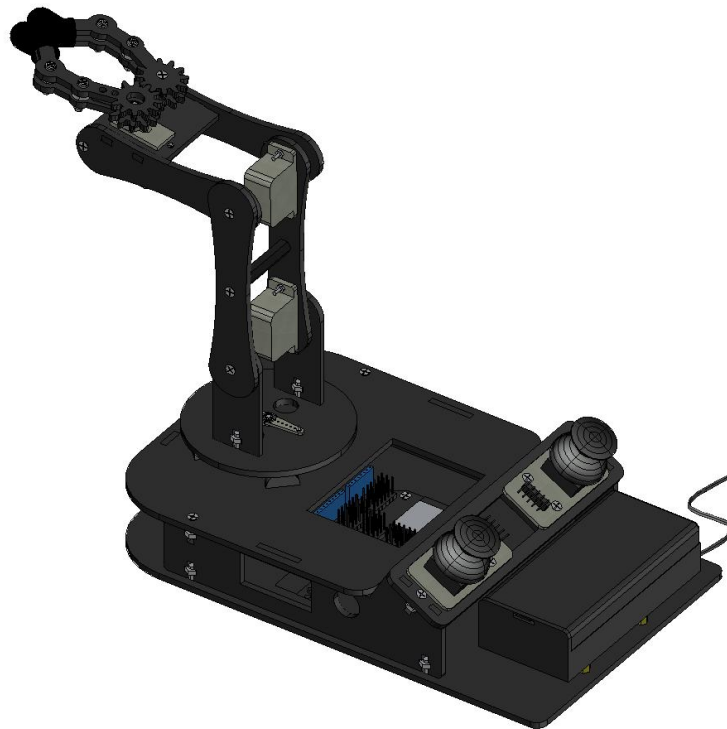
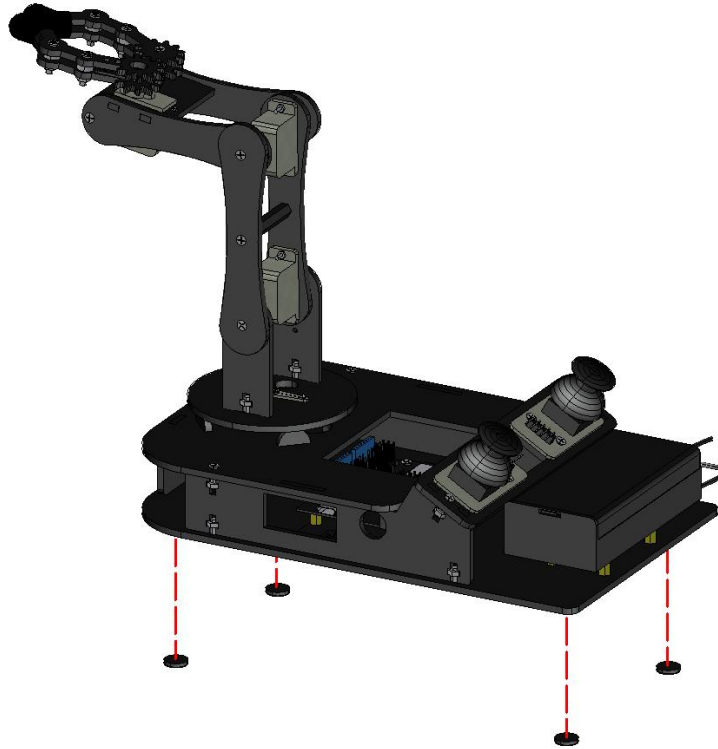
18.Fixed battery holder

Part List	
Name	Quantity
18650 Battery Holder	1
M3*8 Flat Head Screws	8
M3*12 Double-pass Copper Pillar	4



19. Install the non-slip mat

Peel off the adhesive film from the non-slip mat and stick the non-slip mat on the four corners underneath the robot arm base.



The robot arm is now fully assembled!

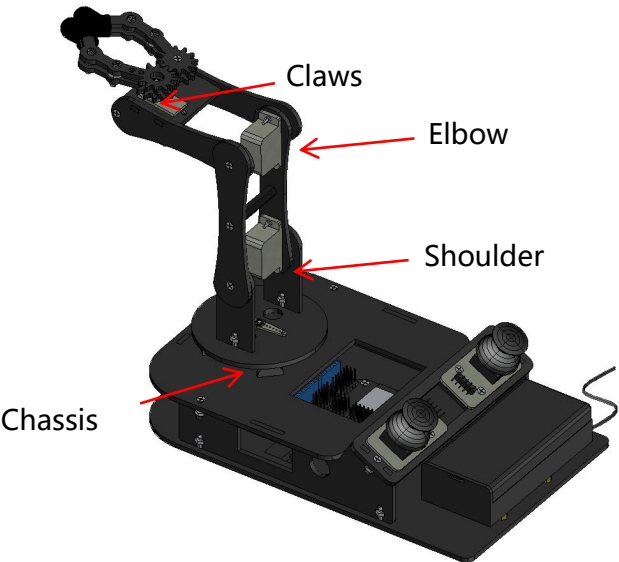
Lesson 3 Robot Arm Joystick Control

I .Servos Control

1.Pin description of robot arm servos

After installing the robot arm, each servo assumes different functions. To control each servo's angle programmatically and achieve various functionalities, it's essential to know the corresponding pin numbers for each servo.

The robot arm uses a total of 4 servos, with their corresponding pin numbers as follows:

NO.	Pin Number	Servo Position	Figure
1	GPIO18	Claws	
2	GPIO17	Elbow	
3	GPIO16	Shoulder	
4	GPIO5	Chassis	

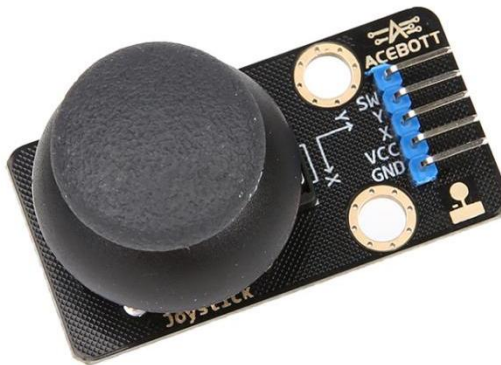
2.Motion principles of each joint in the robot arm

NO.	Pin Number	Servo Position	Movement Patterns
1	GPIO18	Claws	The larger the servo angle, the wider the claw opens
2	GPIO17	Elbow	The larger the servo angle, the higher the elbow of the robot arm moves up
3	GPIO16	Shoulder	The larger the servo angle, the lower the shoulder of the robot arm moves down
4	GPIO5	Chassis	The larger the servo angle, the more the base of the robot arm turns left

II .Understanding the Joystick Module

1.Introduction to the Joystick Module

The joystick module consists of two sliding potentiometers and one button. When you move the joystick, the resistance values of the potentiometers change, resulting in corresponding X/Y voltage values. Pressing down on the joystick triggers the button, causing the SW signal to go low. This module is commonly used in applications such as RC models, gaming controllers, remote-controlled vehicles, and camera gimbals.



2.Testing the Serial Output of the Joystick Module

When you move the joystick, you can monitor the corresponding X/Y values in real-time by printing them to the Serial Monitor in the Arduino IDE.

Open ["Joystick_test.ino"](#) in English\Arduino\2.Arduino program\Lesson 3\Joystick_test, connect the ESP32 controller board to the computer with a USB cable, select the correct controller board, processor, and port, Upload the code to the ESP32 controller board.

Sample Code:

```
const int left_X = 32; //Define the right X pin to 32
const int left_Y = 33; //Define the right Y pin to 33
const int left_key = 34;
int X1 = 0, Y1 = 0, Z1 = 0;

void setup() {
  pinMode(left_key, INPUT);
  Serial.begin(115200);
}

void loop() {
  X1 = analogRead(left_X); //Read the right X value
  Serial.print(X1);
  Y1 = analogRead(left_Y); //Read the right Y value
  Serial.print(" ");
  Serial.print(Y1);
  Z1 = digitalRead(left_key); //Read the right Z value
  Serial.print(" ");
  Serial.println(Z1);
  delay(1000);
}
```



III. Basic Movements of Robot Arm Controlled by Joystick

The basic control actions of a robot arm mainly include left-right rotation, up-down elbow movement, and opening-closing of the claw. Once you master these basic movements, you can combine and expand other actions based on these fundamentals.

Attention: After the robot arm is powered on, it is forbidden to rotate the servo directly by hand to prevent damage to the servo.

1. Joystick control program

Open [“JoyStick_Controlled_Robot_Arm.ino”](#) in English\Arduino\2.Arduino program\Lesson 3\JoyStick_Controlled_Robot_Arm, connect the ESP32 controller board to the computer with a USB cable, select the correct controller board, processor, and port, Upload the code to the ESP32 controller board.

Sample Code:

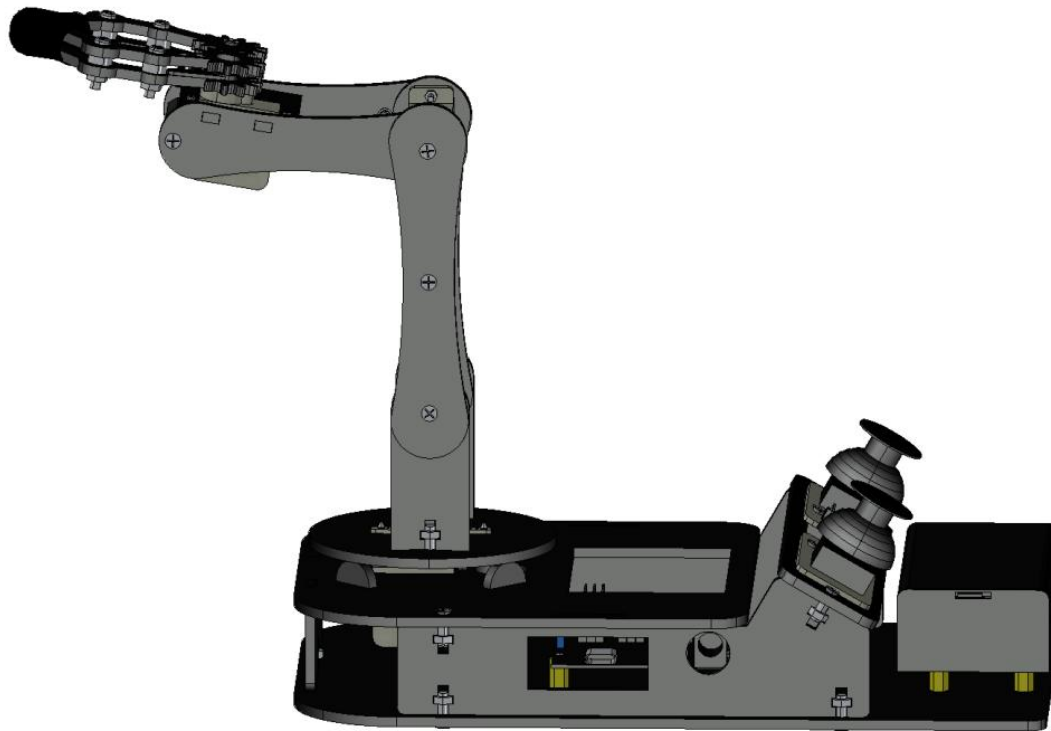
```
#include <ACB_ARM.h> //Add the Robot arm libraries
ACB_ARM ARM;

void setup() {
  ARM.ARM_init(5,16,17,18); //The parameters are four Servo pins
  ARM.JoyStick_init(32,33,34,35,36,39); //Joystick initialization
}

void loop() {
  ARM.get_JoyStick();
  if (ARM.JoyY1 < 50) { //chassis left
    ARM.chassis_angle = ARM.chassis_angle + 1;
    ARM.JoyChassisCmd(ARM.chassis_angle);
  }
  if (ARM.JoyY1 > 3500) { //chassis right
    ARM.chassis_angle = ARM.chassis_angle - 1;
    ARM.JoyChassisCmd(ARM.chassis_angle);
  }
  if (ARM.JoyX1 < 50) { //Shoulder down
    ARM.shoulder_angle = ARM.shoulder_angle + 1;
    ARM.JoyShoulderCmd(ARM.shoulder_angle);
  }
  if (ARM.JoyX1 > 4000) { //Shoulder up
    ARM.shoulder_angle = ARM.shoulder_angle - 1;
    ARM.JoyShoulderCmd(ARM.shoulder_angle);
  }
  if (ARM.JoyX2 < 50) { //Elbow up
    ARM.elbow_angle = ARM.elbow_angle + 1;
    ARM.JoyElbowCmd(ARM.elbow_angle);
  }
  if (ARM.JoyX2 > 4000) { //Elbow down
    ARM.elbow_angle = ARM.elbow_angle - 1;
    ARM.JoyElbowCmd(ARM.elbow_angle);
  }
  if (ARM.JoyY2 > 4000) { // Claws open
    ARM.claws_angle = ARM.claws_angle + 1;
    ARM.JoyClawsCmd(ARM.claws_angle);
  }
  if (ARM.JoyY2 < 50) { // Claws close
    ARM.claws_angle = ARM.claws_angle - 1;
    ARM.JoyClawsCmd(ARM.claws_angle);
  }
}
```

2.Joystick control patterns

After uploading the program, you will notice that the four arms of the robot arm are arranged in a "7" shape, which is the initial posture. You can then use the joystick to control the robot arm.



The following is a joystick function introduction:

Joystick	Direction	Joint Position	Law of Motion
Left joystick	Left	Chassis	Turn to the left
	Right	Chassis	Turn to the right
	Up	Shoulder	Upward motion
	Down	Shoulder	Downward motion
Right joystick	Up	Elbow	Upward motion
	Down	Elbow	Downward motion
	Left	Claws	Claws open
	Right	Claws	Claws close

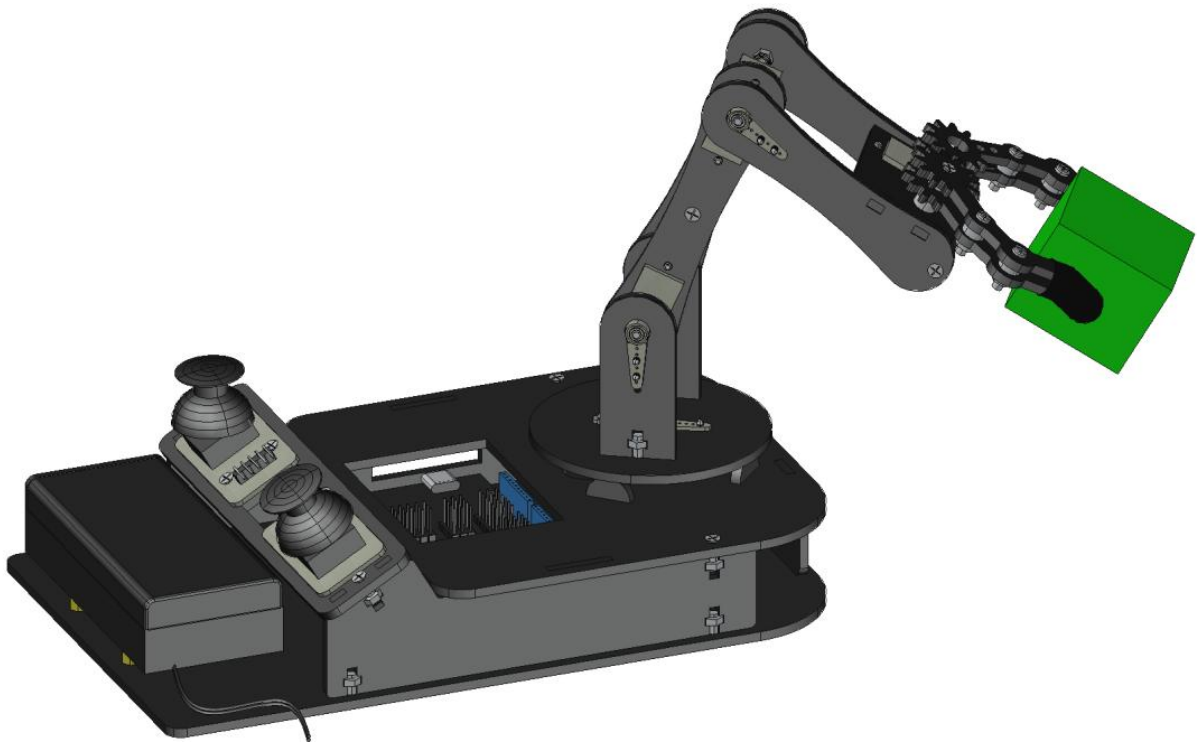
IV.Extending Tasks

Based on the basic operational patterns of the joystick on the robot arm, next we can achieve the functionality of using the joystick to control the robot arm for object handling tasks.

Task description:

Using the joystick module control method, maneuver the robot arm to pick up an object from point A and transport it to point B for placement.

Attention: Define the positions of points A and B as desired.

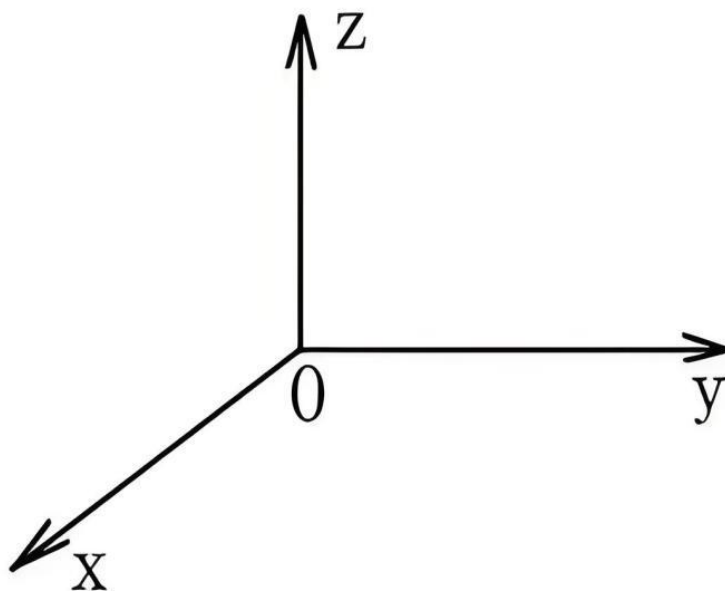


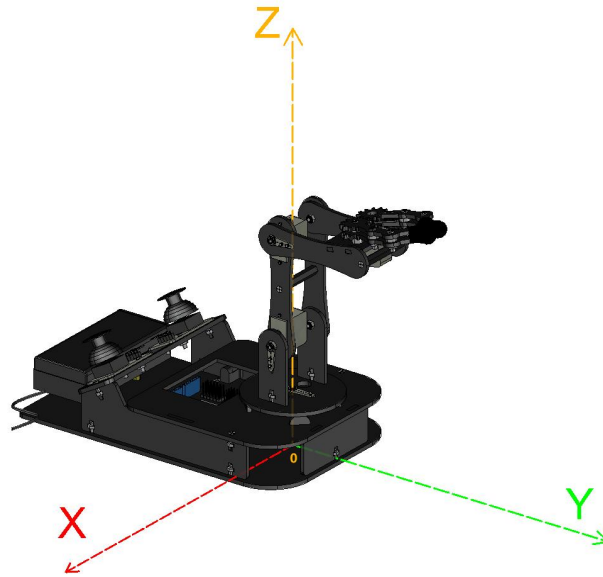
Lesson 4 The Spatial Coordinates of The Robot Arm

The spatial coordinates of the robot arm play a crucial role in its control and programming. Through precise spatial coordinates, the robot arm can achieve accurate positioning, optimize motion planning, effectively avoid obstacles, perform precise operations, and thereby enhance levels of automation and intelligence.

I .The Cartesian Coordinate System

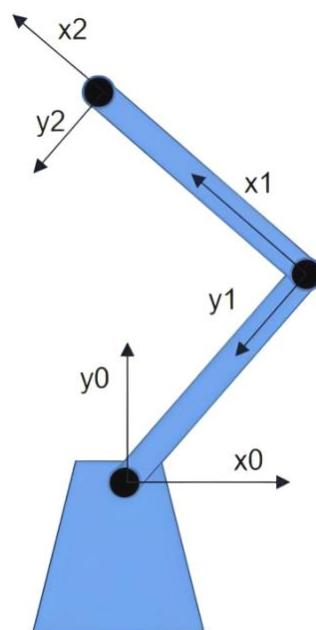
In robot arm spatial motion control, the Cartesian coordinate system is one of the most commonly used coordinate systems. It is a mathematical system for describing the position of points in space. In three-dimensional space, the Cartesian coordinate system consists of three mutually perpendicular axes (x, y, z). The intersection of these three axes is the origin (O) of the coordinate system. In this tutorial, the origin of the Cartesian coordinate system is located at the center of the robot arm Chassis servo disc.





II .Joint Coordinate System

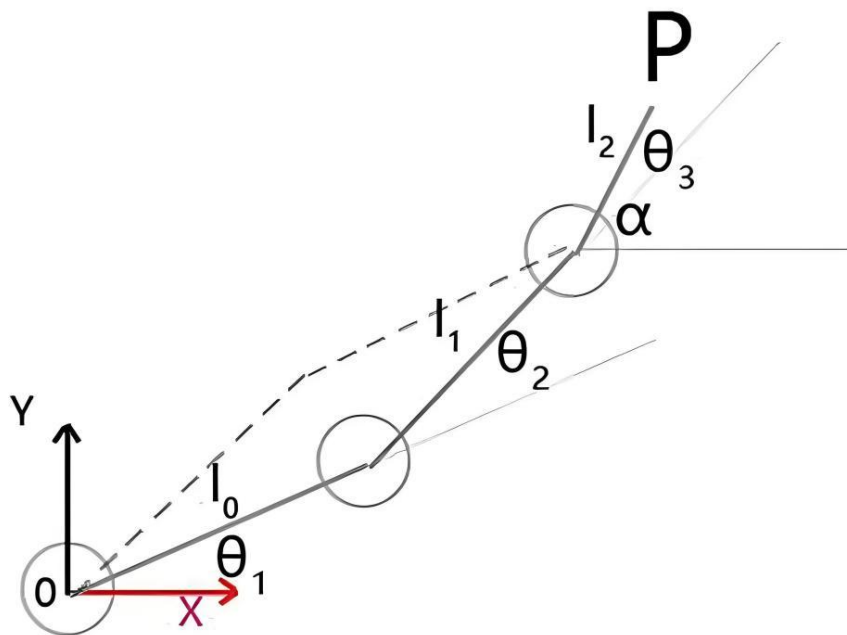
Apart from the Cartesian coordinate system, each joint of the robot arm has its own coordinate system known as the joint coordinate system. Its origin typically lies at the joint connection point, and its axes are defined along the rotation axis of the joint. Each joint coordinate system is associated with a joint coordinate that describes the rotation angle or extension of that joint. Because each joint of the robot arm can rotate or extend, the joint coordinate systems can transform based on the current posture of the robot arm.



III. Forward and Inverse Kinematics

Forward kinematics refers to calculating the position and orientation of the robot arm in Cartesian coordinates based on the joint angles. Inverse kinematics, on the other hand, involves calculating the joint angles based on the Cartesian coordinates to achieve the desired movement of the robot arm.

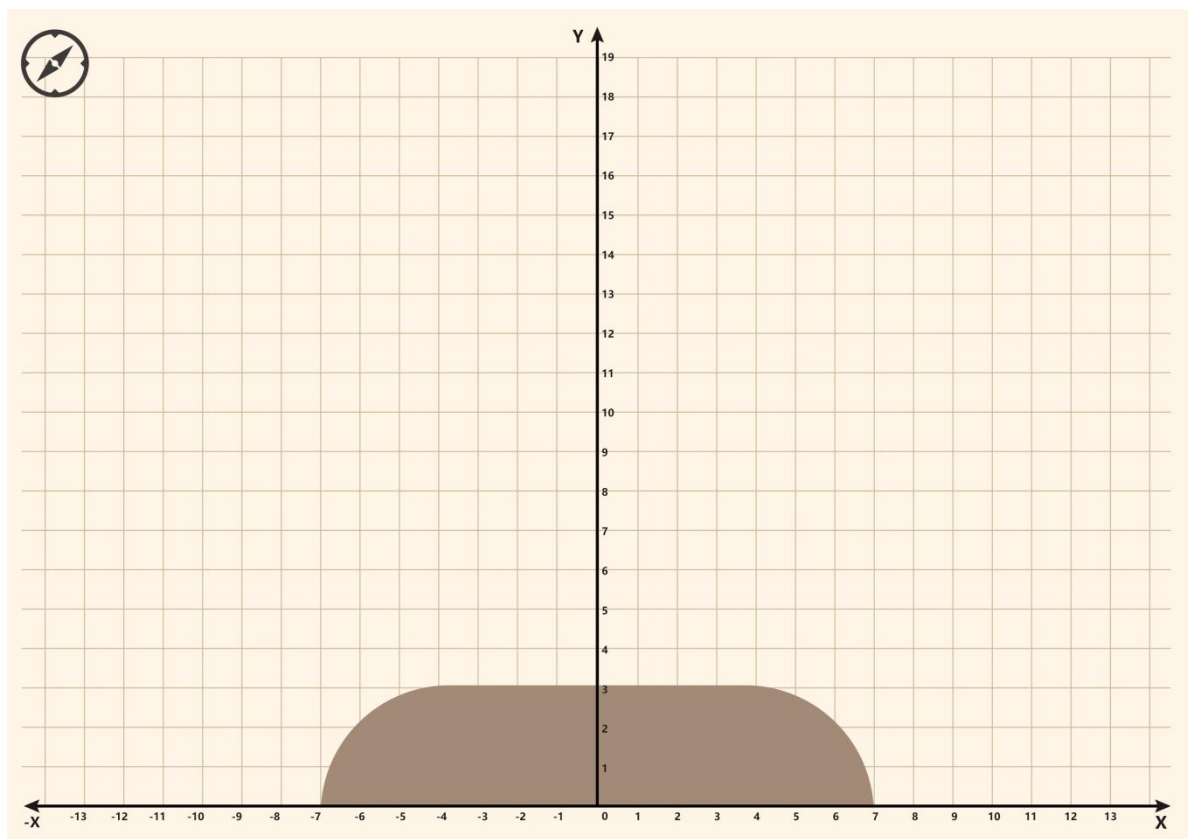
The forward kinematics problem can be solved through matrix transformations, that is, by comparing the transformation matrix of each joint coordinate system with the coordinate transformation formulas of the robot arm to determine the angles of each joint. The inverse kinematics problem is usually more complex, requiring the solution of a set of nonlinear equations and often having multiple solutions.



In this tutorial, we will mainly build a Cartesian coordinate system with the center of the Chassis steering wheel as the origin point. When the specified position coordinates (X,Y,Z) are entered, the library file in the tutorial will automatically use the inverse kinematics algorithm to calculate the coordinates of each joint of the mechanical arm, and then convert them into the angles of each joint steering gear. Thus, the end of the manipulator is controlled to reach the position of the target point on the space coordinate.

IV.Robot Arm Coordinate Diagram

There will inevitably be some errors in the assembly process of the robot arm. In order to better calibrate the robot arm, we need to use the coordinate diagram. The coordinate diagram of the robot arm is composed of X and Y coordinates, with the intersection point of X and Y being the origin of the map. The coordinate range of X is $[-13, 13]$, and the coordinate range of Y is $[0, 19]$. The chamfered rectangle in the shaded area is the reference position for the robot arm. Place the upper edge of the robot arm's base against this shape.



[【 Click to get the robot arm coordinate diagram PDF file 】](#)

Attention: Please print the diagram of the robot arm on A4 paper according to the PDF file.

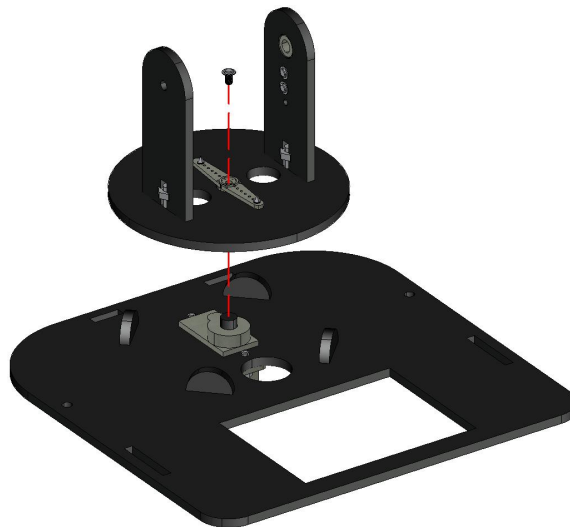
V.Robot Arm Calibration Instructions

The robot arm is composed of multiple servos, and in practical work, there may be certain errors. To reduce these errors, we have added instructions in the robot arm control program that allow adjustment of the arm's errors.

```
ARM.Chassis_angle_adjust(8); // Default 0  
ARM.Slight_adjust(0,2); // Default 0,0
```

First, the command 'ARM.Chassis_angle_adjust(8)' is used to adjust the central angle deviation of the Chassis servo. The default center point value is 90 degrees, with angles ranging from 90 to 180 degrees on the left side of the -X axis, and angles ranging from 0 to 90 degrees on the right side of the X axis.

However, due to the gear precision during installation, it's not guaranteed that the Chassis servo and steering gear will be exactly at 90 degrees. They might slightly tilt left or right. Therefore, at this point, we need to calibrate it in the program. For example, if it tilts 8 degrees to the right, we need to increase the deviation value. So, in the parentheses of the 'ARM.Chassis_angle_adjust()' command, you would enter 8. If the shift is 8 degrees to the left, then you need to subtract the offset value, "ARM.Chassis_angle_adjust()" parentheses with -8.



Next is the 'ARM.Slight_adjust(0,2)' command. This command has two parameters, both defaulting to 0. Suppose there is a slight offset error when the robot arm end effector reaches a specified spatial point. In that case, this command is used to make fine adjustments.

The first parameter corresponds to the robot arm's X positive semiaxis. If the end effector tilts 1 degree to the right, you would write 1; if it tilts 1 degree to the left, you would write -1. If there is no tilt, you would write 0.

The second parameter corresponds to the robot arm's X negative semiaxis. If the end effector tilts 1 degree to the right, you would write 1; if it tilts 1 degree to the left, you would write -1. If there is no tilt, you would write 0.

Next please borrow the coordinate diagram, according to the space coordinates below the program and your actual situation, adjust the calibration parameters.

VI. Moving Spatial Coordinate Points

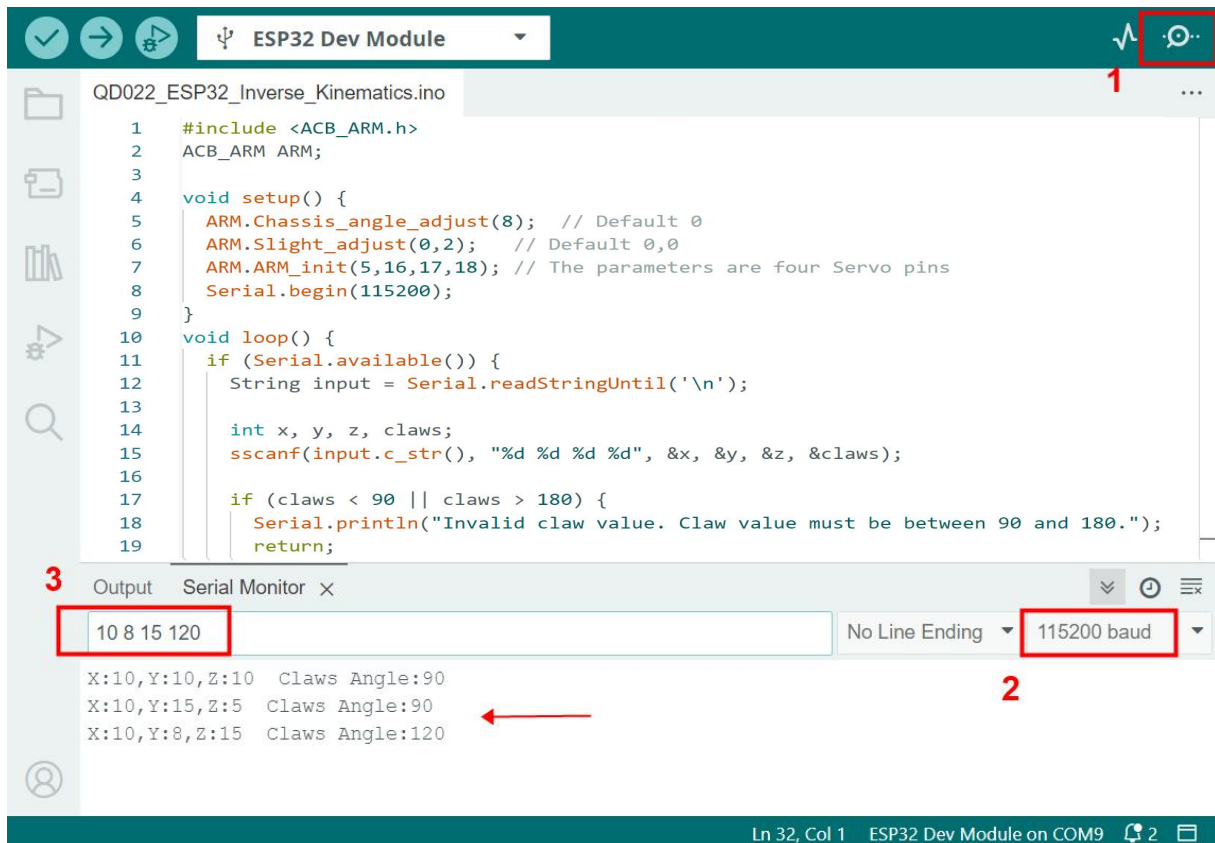
Open ["Inverse Kinematics.ino"](#) in English\Arduino\2.Arduino program\Lesson 4\Inverse_Kinematics, connect the ESP32 controller board to the computer with a USB cable, select the correct controller board, processor, and port, Upload the code to the ESP32 controller board.

Sample Code:

```
#include <ACB_ARM.h> //Add the Robot arm libraries
ACB_ARM ARM;

void setup() {
  ARM.Chassis_angle_adjust(8);
  ARM.Slight_adjust(0,2);
  ARM.ARM_init(5,16,17,18); //The parameters are four Servo pins
  Serial.begin(115200);
}

void loop() {
  if (Serial.available()) {
    String input = Serial.readStringUntil('\n');
    int x, y, z, claws; //Define variables for coordinates x, y, z, and claws
    sscanf(input.c_str(), "%d %d %d %d", &x, &y, &z, &claws);
    if (claws < 90 || claws > 180) {
      Serial.println("Invalid claw value. Claw value must be between 90 and 180.");
      return;
    }
    Serial.print("X:");
    Serial.print(x);
    Serial.print(",Y:");
    Serial.print(y);
    Serial.print(",Z:");
    Serial.print(z);
    Serial.print(" Claws Angle:");
    Serial.print(claws);
    Serial.println(" ");
    ARM.ClawsCmd(claws);
    delay(1000);
    ARM.PtpCmd(x, y, z);
  }
}
```



After uploading the program, enter four values in the serial monitor: the X coordinate, the Y coordinate, the Z coordinate, and the claw opening angle (ranging from 90 degrees to 180 degrees). These four values need to be separated by spaces. After entering the values, press Enter.

ARM.PtpCmd(x, y, z) is a robot arm coordinate control command. Through this command, the robot arm's end effector moves to the specified position in the spatial coordinates, where x is the x-axis coordinate, y is the y-axis coordinate, and z is the z-axis coordinate.

ARM.JoyClawsCmd(claws) is a robot arm claw control command. This command controls the opening and closing of the end effector's claw, where claws represents the angle parameter for the claw servo, with an input range of 90 to 180 degrees.

If the coordinates are correct, the robot arm will move to the specified point in space. If the serial monitor displays "Out of range!" after entering the coordinates, it means the input values exceed the arm's reachable range, as the movement range is within a spherical area. In this case, you need to adjust the values and re-enter them.

Lesson 5 Robot Arm Stacking

In today's era of rapid technological development, robot arms have become an indispensable part of modern industry, commercial services, and daily life. With their remarkable flexibility, precision, and efficiency, they have completely transformed traditional modes of operation.

Especially, robot arm palletizing technology is widely used in various fields, particularly in scenarios requiring high-efficiency item handling, where its application value is especially important. For example, in logistics and warehousing, robot arm palletizing technology can significantly improve the efficiency and accuracy of goods handling, reduce labor costs, and minimize human errors. Compared to manual palletizing, robot arm palletizing technology offers advantages such as high efficiency, good stability, and ease of operation.

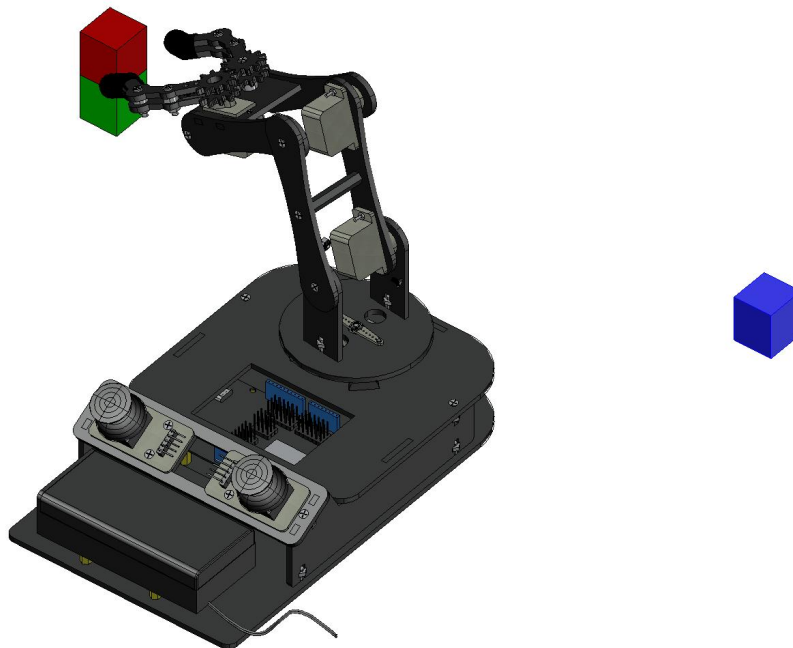


With the advancement of technology and market demand, robot arm palletizing technology will see broader applications in the future. Therefore, in this lesson, we will learn to implement basic palletizing functions using a robot arm.

I .Robot Arm Palletizing Program

Open ["Robot_Arm_Stacking.ino"](#) in English\Arduino\2.Arduino program\Lesson 5\Robot_Arm_Stacking, connect the ESP32 controller board to the computer with a USB cable, select the correct controller board, processor, and port, Upload the code to the ESP32 controller board.

First, stack two blocks vertically and place them at the (-7,13) coordinates on the robot arm map, ensuring that the center point of the blocks aligns with the coordinate point.



Sample Code:

```
#include <ACB_ARM.h> //Add the Robot arm libraries
ACB_ARM ARM;

bool RunningState = true;

int startx = 7, starty = 13, startz = 2; //Initial coordinates
int midz = 15; //Height of middle point
int endx = -7, endy = 13, endz = 0; //End point coordinates
int openAngle = 130, closeAngle = 90;
int count = 2; //Number of blocks
int i = 0;

void setup() {
  ARM.Chassis_angle_adjust(8); //Chassis error calibration
  ARM.Slight_adjust(0,2); //Small calibration error
  ARM.ARM_init(5,16,17,18); //The parameters are four Servo pins
  Serial.begin(115200);
}

void loop() {
  while(i < count) {
    ARM.ClawsCmd(openAngle); //open claws
    delay(1000);
    ARM.PtpCmd(startx, starty, startz-i*2); //The target point is the upper object
    delay(1000);
    ARM.ClawsCmd(closeAngle); //close claws
    delay(1000);
    ARM.PtpCmd(startx, starty, midz); //Lift the object after picking it up
    delay(1000);
    ARM.PtpCmd(endx, endy, midz); //Rotate above the target point
    delay(1000);
    ARM.PtpCmd(endx, endy, endz+i*2); //Placing Objects
    delay(1000);
    ARM.ClawsCmd(openAngle);
    delay(1000);
    ARM.PtpCmd(endx, endy, midz); //Raise arms
    delay(1000);
    i = i + 1;
  }
  ARM.Zero(); //Servo initialization
}
```

After uploading the program, we can observe that the end of the robot arm will reach the initial coordinates of the blocks, then grab the upper block, transport it to the destination coordinates, and place it down. Next, it will return to the initial coordinates of the blocks, grab the lower block, transport it to the destination coordinates, and finally stack it on top of the first block.

II .Extending Tasks

Based on the robot arm's coordinate map, we already know the palletizing pattern of the robot arm. As long as we know the position coordinates on the map, we can make the robot arm complete the palletizing function. Next, we will use the map to complete the robot arm's palletizing function.

Task description:

Based on the palletizing program example, try modifying the initial position coordinates and the endpoint coordinates of the objects in the program to enable the robot arm to achieve palletizing at different positions.

Attention: The two blocks need to be placed stacked together. Ensure the coordinates entered do not exceed the map range. Additionally, during the robot arm's grabbing process, there may be some errors due to placement or precision issues.

Lesson 6 Teaching and Learning of the Robot Arm

Teaching a robot arm involves the operator setting a fixed motion path for the robot arm to follow, allowing it to work according to preset steps.

Teaching a robot arm can be divided into three steps. The first step is the teaching action, where the operator sets a fixed motion path for the robot arm. The second step is the storage action, where the robot arm's control system records the taught actions. The third step is the reproduction of the teaching, where the robot arm repeats the actions recorded during the teaching process.

In this lesson, we will use the joystick module to teach and learn the robot arm's movements.

I .Teaching Program

Open ["Memory Controlled Robot Arm.ino"](#) in English\Arduino\2.Arduino program\Lesson 6\Memory_Controlled_Robot_Arm, connect the ESP32 controller board to the computer with a USB cable, select the correct controller board, processor, and port, Upload the code to the ESP32 controller board.

Sample Code:

```
#include <ACB_ARM.h> //Add the Robot arm libraries
ACB_ARM ARM;

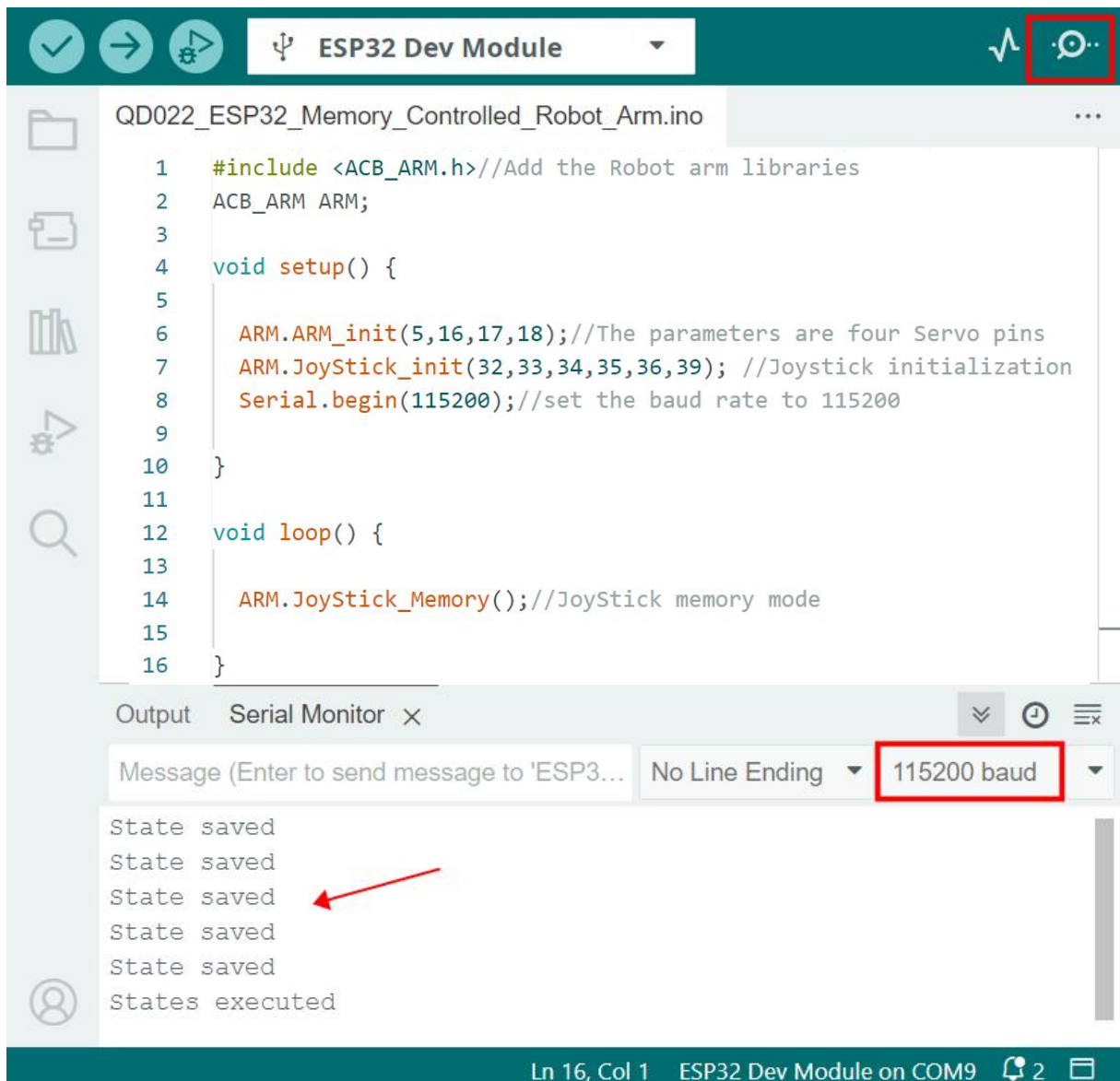
void setup() {
  ARM.ARM_init(5,16,17,18); //The parameters are four Servo pins
  ARM.Joystick_init(32,33,34,35,36,39); //Joystick initialization
  Serial.begin(115200); //set the baud rate to 115200
}

void loop() {
  ARM.Joystick_Memory(); //Joystick memory mode
}
```

- Attention:** ①When determining the first and last position points of the robot arm, it is necessary to save them promptly;
- ②The memory function can save up to 20 action groups at a time.

Instruction of teaching operation:

Joystick	Joystick Operation	Actions
Left joystick	Short press	Save action
	Long press	Clear action
Right joystick	Short press	Run action



The screenshot shows the Arduino IDE interface. The top toolbar includes a red box around the upload button. The file name is `QD022_ESP32_Memory_Controlled_Robot_Arm.ino`. The code in the editor is as follows:

```

1  #include <ACB_ARM.h> //Add the Robot arm libraries
2  ACB_ARM ARM;
3
4  void setup() {
5
6      ARM.ARM_init(5,16,17,18); //The parameters are four Servo pins
7      ARM.Joystick_init(32,33,34,35,36,39); //Joystick initialization
8      Serial.begin(115200); //set the baud rate to 115200
9
10 }
11
12 void loop() {
13
14     ARM.Joystick_Memory(); //Joystick memory mode
15
16 }

```

The Serial Monitor is open, showing the output:

```

State saved
State saved
State saved
State saved
State saved
States executed

```

A red arrow points to the third "State saved" line. The Serial Monitor settings show "No Line Ending" and "115200 baud", which is also highlighted with a red box.

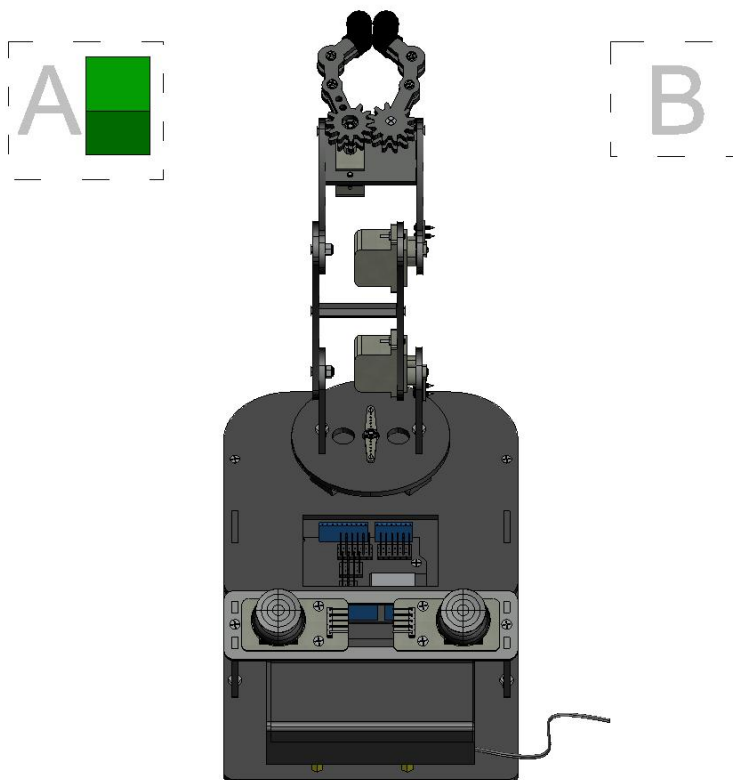
II .Extending Tasks

Based on the basic operational rules of teaching and learning for the robot arm, next we can use the joystick module to achieve teaching and learning functions for transporting objects with the robot arm.

Task description:

Use the joystick module to control the robot arm to pick up a block from point A on the coordinate map, transport it to point B, and then return to the initial position. This process involves teaching and learning.

Attention: Define the positions of points A and B as you wish.



Lesson 7 Web Control of the Robot Arm

With the continuous development of wireless communication technology and Internet of Things (IoT) technology, remote control technology is widely used in many fields. It allows users to achieve precise remote control of terminal devices over long distances. There are many types of wireless communication technologies, and this tutorial mainly focuses on how to use WiFi communication technology to remotely control a robot arm.

WiFi communication technology is a type of Wireless Local Area Network (WLAN) technology that allows electronic devices such as smartphones, tablets, and laptops to connect wirelessly to the Internet or a local area network (LAN). WiFi communication technology connects devices to the same network using wireless routers or access points (APs), enabling them to transmit and receive data between each other.

Web-based device control is one of the primary applications of WiFi communication technology, widely used in areas such as smart homes and smart industries. Web-based device control connects devices and control terminals over the Internet. Interaction between devices and controllers can be achieved through simple HTTP protocols. When a device connects to a controller, the controller provides a straightforward web interface that users can access through a web page to control the device.

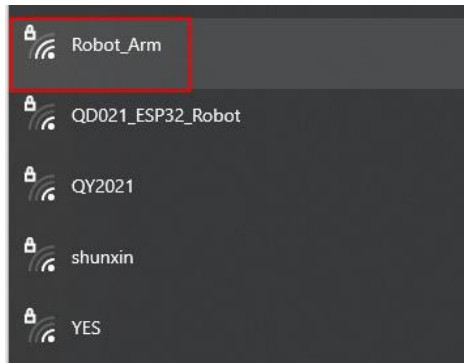
Next, we will use a web page to remotely control the operation of the robot arm.

I .Web Control Program

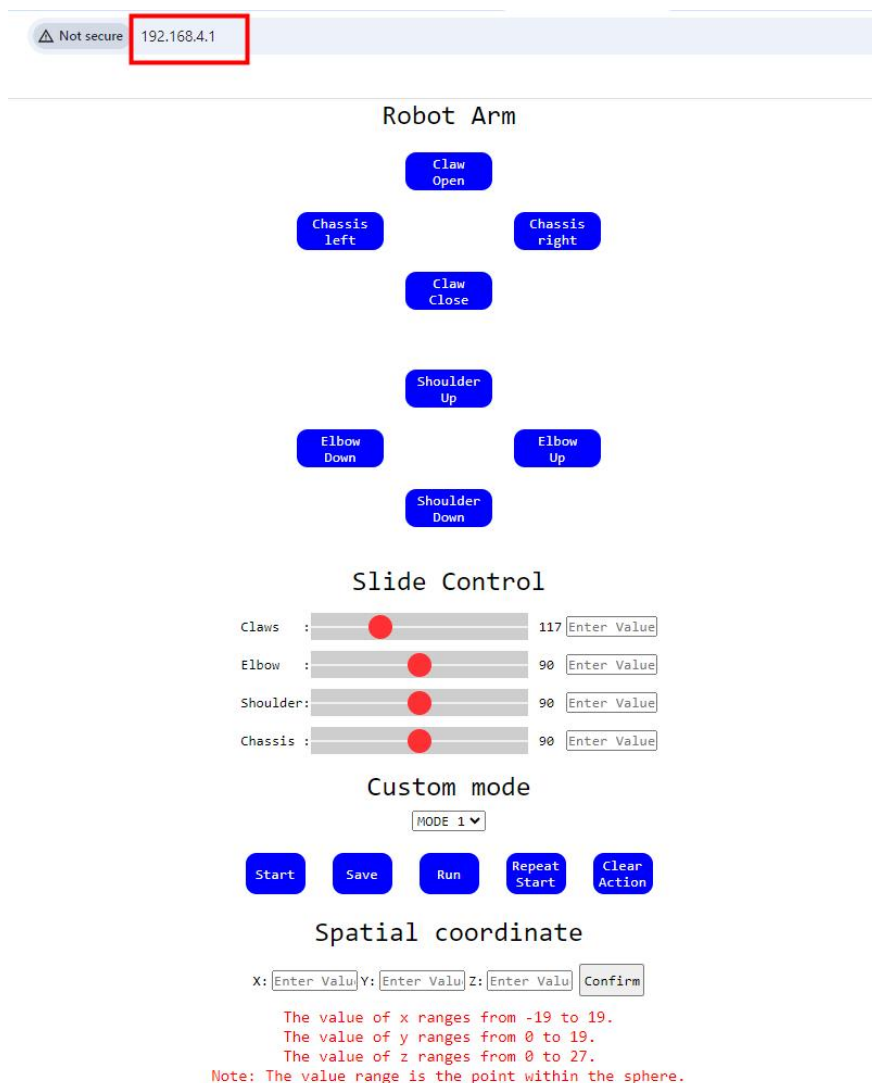
Open ["Web_Controlled_Robot_Arm.ino"](#) in English\Arduino\2.Arduino program\Lesson 7\Web_Controlled_Robot_Arm, connect the ESP32 controller board to the computer with a USB cable, select the correct controller board, processor, and port, Upload the code to the ESP32 controller board.

II .Login To the Webpage

After successful upload, next, use your computer or mobile phone to scan for WiFi networks. Connect to the WiFi hotspot named 'Robot_Arm' with the password 12345678, as shown in the following image.



After successfully connecting, enter "192.168.4.1" into the address bar of your browser. The web page interface will appear as shown in the following image.

A screenshot of a web browser interface for controlling a robot arm. The address bar shows '192.168.4.1' with a red box around it. The page title is 'Robot Arm'. It features several blue buttons for controlling the robot arm: 'Claw Open', 'Chassis left', 'Chassis right', 'Claw Close', 'Shoulder Up', 'Elbow Down', 'Elbow Up', and 'Shoulder Down'. Below these is a 'Slide Control' section with four sliders for 'Claws', 'Elbow', 'Shoulder', and 'Chassis', each with a red slider knob and a numerical value (117, 90, 90, 90 respectively) and an 'Enter Value' button. Underneath is a 'Custom mode' section with a 'MODE 1' dropdown menu and five buttons: 'Start', 'Save', 'Run', 'Repeat Start', and 'Clear Action'. The 'Spatial coordinate' section has input fields for 'X', 'Y', and 'Z' coordinates, each with an 'Enter Value' button, followed by a 'Confirm' button. At the bottom, there is a note: 'The value of x ranges from -19 to 19. The value of y ranges from 0 to 19. The value of z ranges from 0 to 27. Note: The value range is the point within the sphere.'

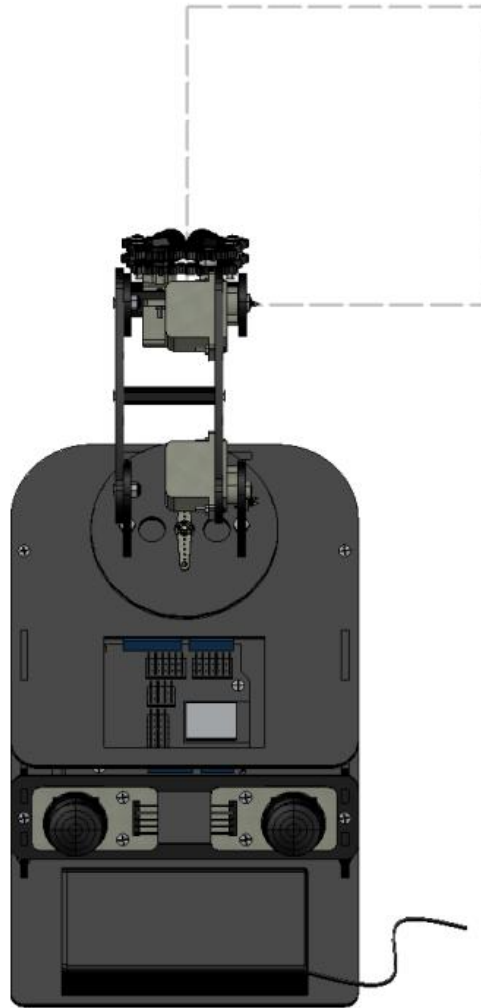
No.	Web Page Functionality	Function Description
1	Button control	Control the movement of the robot arm through buttons on the web page.
2	Slider control	Control the movement of the robot arm by either moving sliders or entering an angle into the input box on the webpage. Attention: Move the slider slowly; the faster you slide it, the faster the robot arm moves.
3	Save action	You can save a total of 6 sets of robot arm actions (Mode 1 to 6), with each set capable of storing up to 20 different actions, the specific operation process is as follows: ①Clicking "Start" changes the button to "End," then proceed to "Save" the action. Based on your action path, click "Save" step by step. Note to click "Save" for both the starting position and the ending position. ②Click "End" to complete the action save; ③Click "Run" to perform an act of memory; ④Click Repeat Start to repeat the memory action; ⑤Click "Reset" to reset the action group.
4	Spatial positioning	Enter the spatial coordinates x, y, z, then click "Confirm." The robot arm will move to the specified spatial coordinates. Attention: Below the x, y, z input boxes, there are corresponding value range descriptions. If values are outside the specified range, please enter them again.

III.Extending Tasks

According to the way of robot arm web page control, next we use three control methods to realize the function of action saving for robot arm in the web page.

Task description:

- (1)Use buttons to control the claws of the robot arm to draw square-like actions on the map and save them in mode1;
- (2)Use the slider to control the claws of the robot arm to draw a square-like action on the map and save it in mode2;
- (3)Spatial positioning is used to control the robot arm paw to draw a square-like action on the map and save it in mode3.



Lesson 8 Robot Arm APP Control

In the previous tutorial, we have learned to control the robot arm with a joy-stick and a web page. In order to control the robot arm more conveniently, we choose to use the mobile APP as the user end to control the robot arm through the mobile APP. Next, we will see how to control the work of the robot arm through a mobile APP.

I .APP Download

(1)If you are using an iOS device, search for the keyword "ACEBOTT" in the App Store and download it. If you are using an Android device, search for the keyword "ACEBOTT" in the Google Play Store and download it. The icon is shown as below.



Attention:

- ①This tutorial is applicable to ACEBOTT APP version 2.0 and above. You can click the settings button in the upper left corner of the APP to view the software version number. Please make sure that the software version you are using meets the requirements;
- ②If you need to update the ACEBOTT software version, you can refer to the method prompted in this tutorial to download the latest APP version.

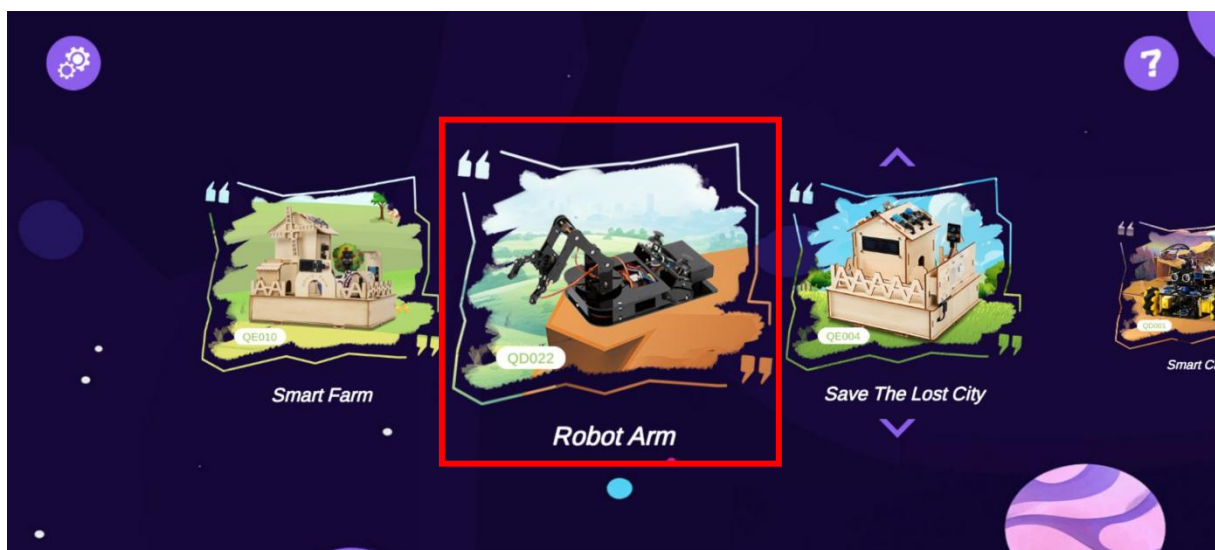
(2)Open the app to enter the splash screen.



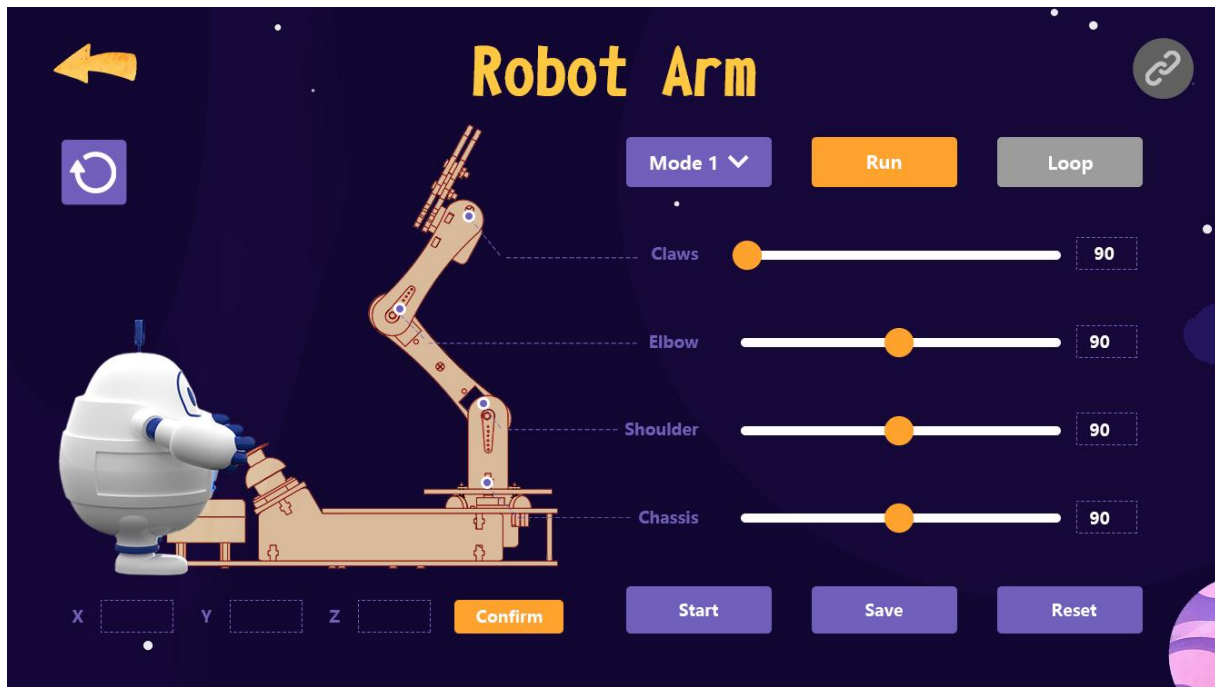
(3) Enter the selection screen and choose the robot arm.

Attention: If you need to watch the APP operation video, please click the link below.

<https://youtu.be/0JtV29RbKQs>



(4) Enter the robot arm control interface (it cannot be controlled directly yet, as the program needs to be uploaded).



II .APP Control the Robot Arm

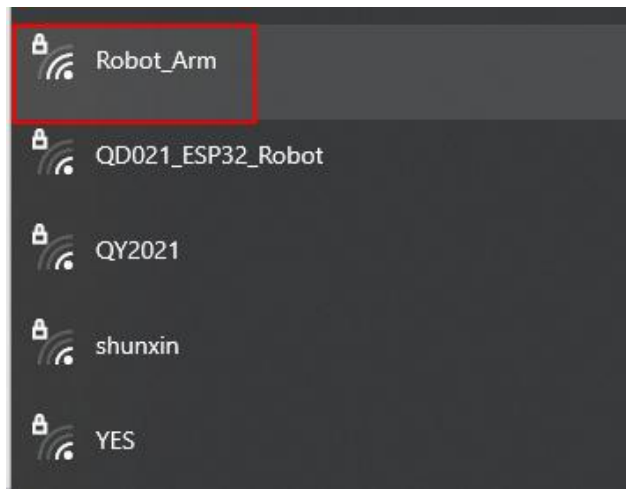
1.Upload the Arduino program for APP control of the robot arm.

Before using the APP to control the robot arm, you need to upload the Arduino program that enables communication between the robot arm and the APP to the robot arm.

Open ["APP Controlled Robot Arm.ino"](#) in English\Arduino\2.Arduino program\Lesson 8\APP_Controlled_Robot_Arm, connect the ESP32 controller board to the computer with a USB cable, select the correct controller board, processor, and port, Upload the code to the ESP32 controller board.

2.Connect to the Robot Arm's WiFi

Scan for WiFi networks on your computer or mobile phone and connect to the WiFi hotspot named "Robot_Arm" with the password 12345678, as shown in the image below.

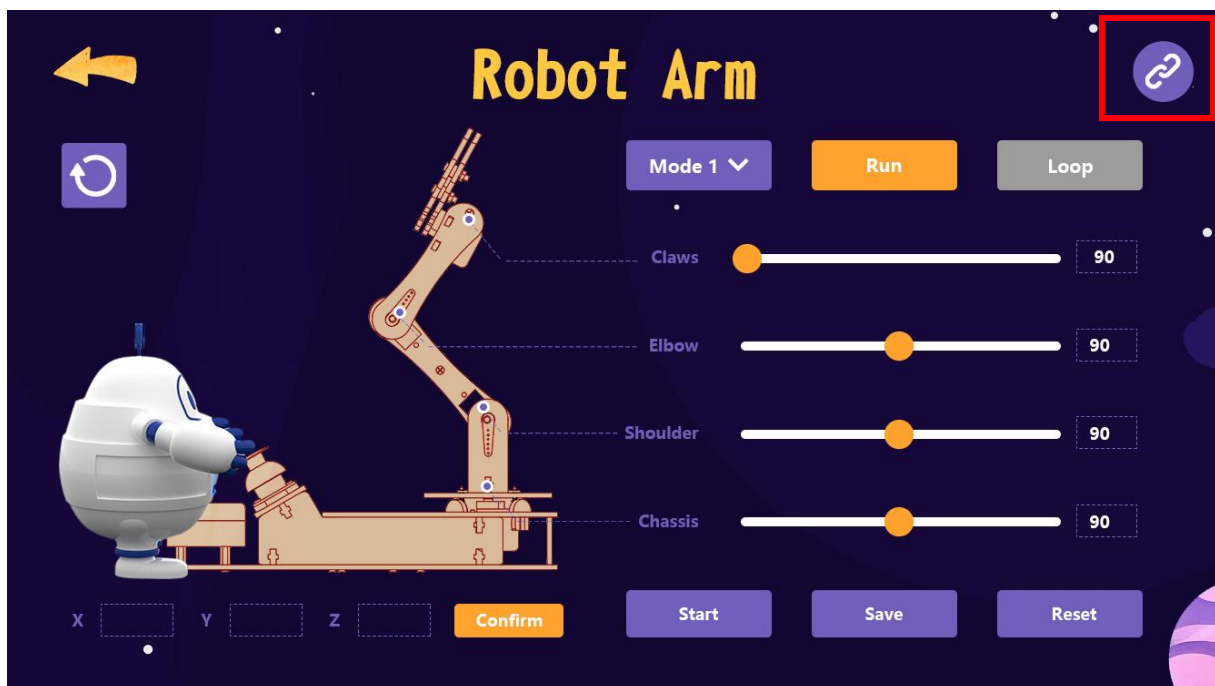


Attention: The hotspot name and password are predefined in the program, but users can customize them. When we have multiple robot arms, we can distinguish each one by using different WiFi names.

```
const char* ssid = "Robot_Arm";// Wifi name  
const char* password = "12345678";// WiFi password
```

3.Using APP controls

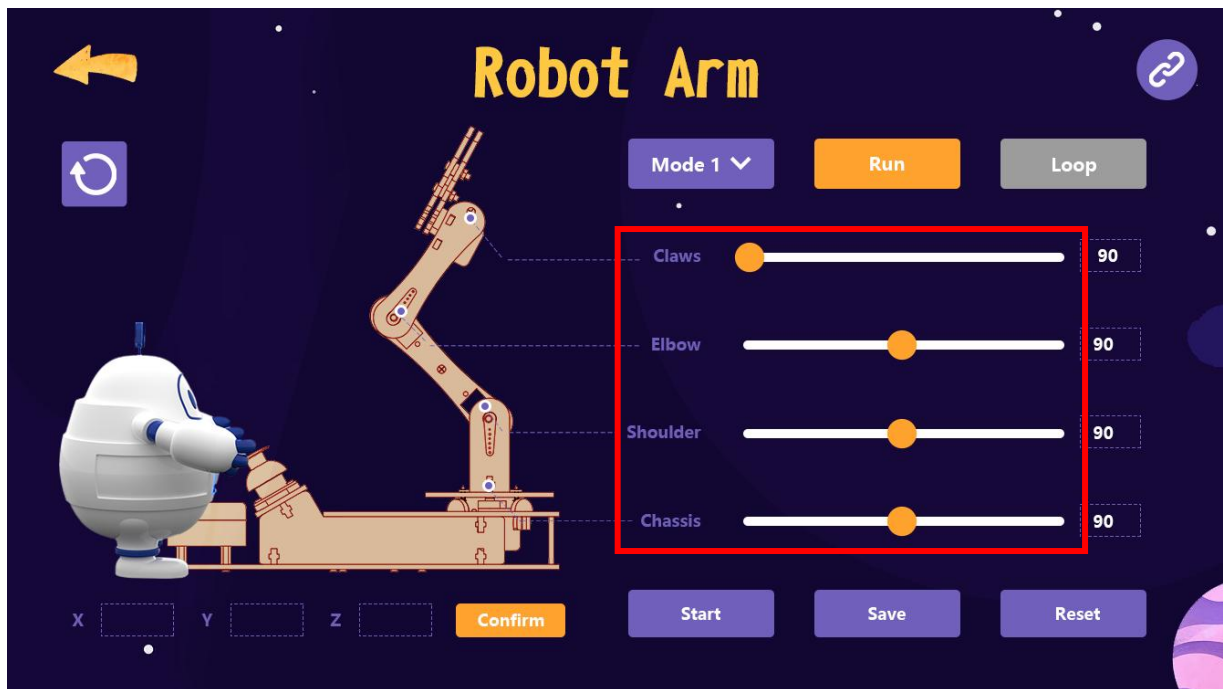
After connecting the WiFi, click the connection icon in the upper right corner of the APP to complete the connection.



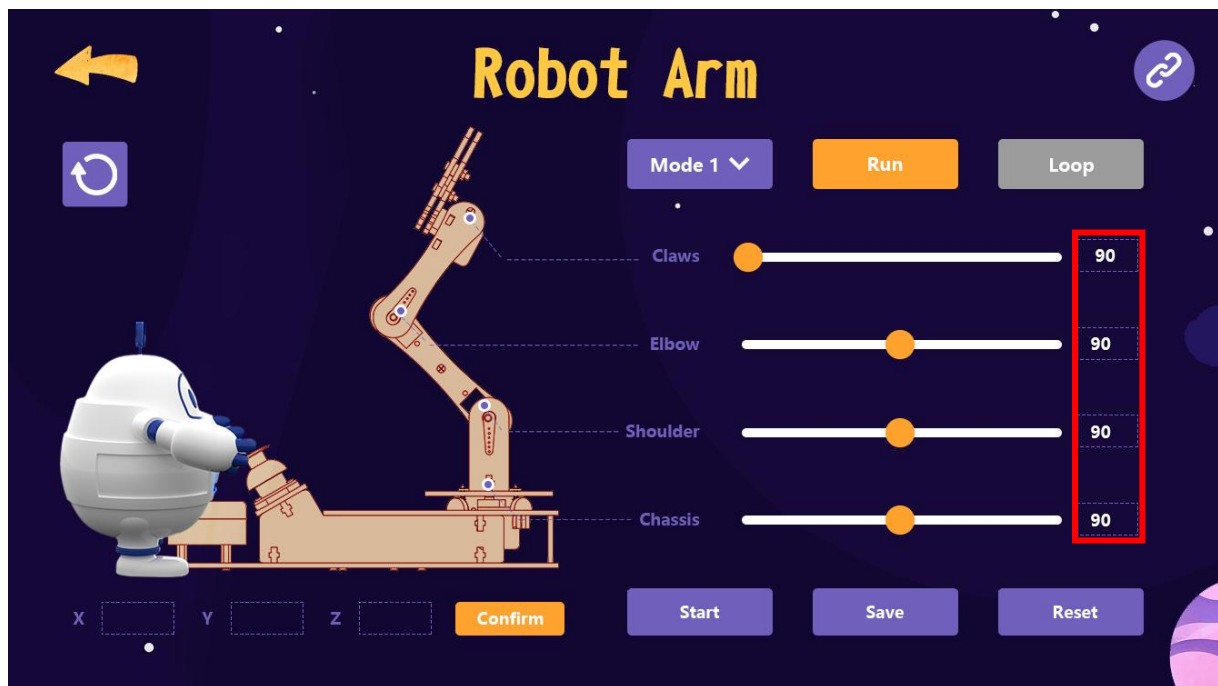
After completing the above operations, return to the interface shown below again, and then the control of the robot arm can be realized. The main control actions are: slider control, input box control, custom mode (start, end, save, run, reset), space positioning function and position recovery function.

Introduction of robot arm APP function:

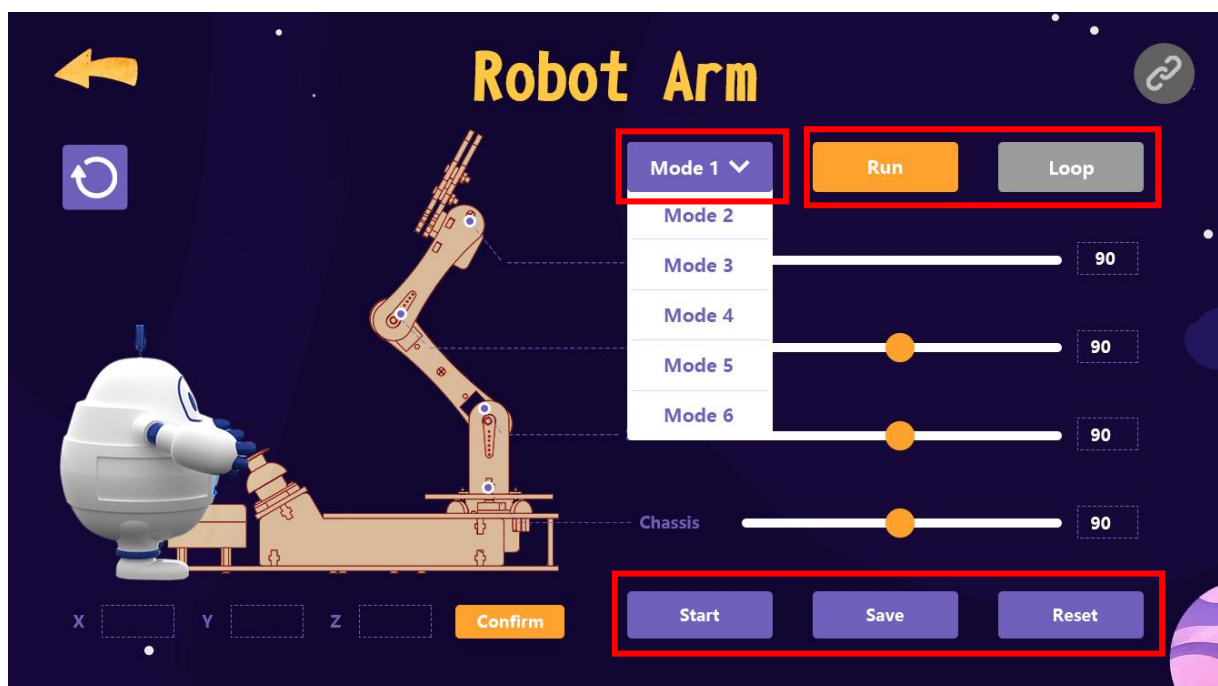
(1)**Slider control:** Move the sliders corresponding to different servo motors of the robot arm to adjust its posture and orientation.



(2)**Input box control:** Next to each slider, there is an input box where you can enter the desired servo angle to control the posture and orientation of the robot arm.

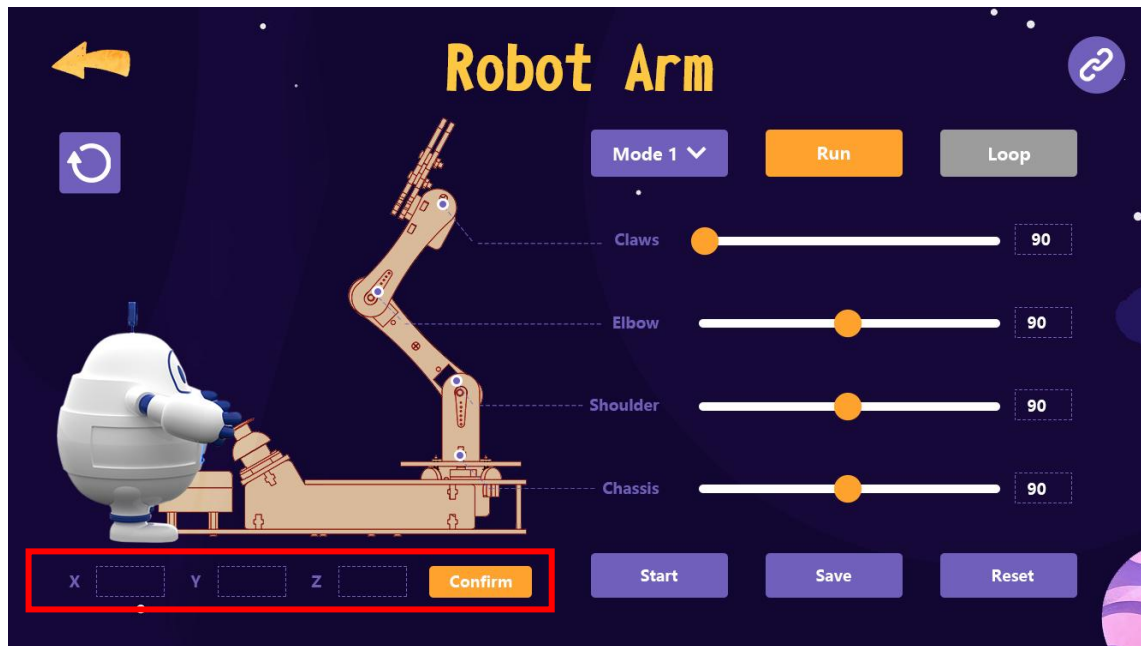


(3)**Action saving:** Click "mode1", there will be 6 modes to choose from (Mode 1~6), and 20 different action groups can be saved in each mode. Click "Run" to perform a memorized action, click "Loop" to repeat the memorized action, and click "Reset" to clear the memorized action. The specific operation process is the same as web page control.

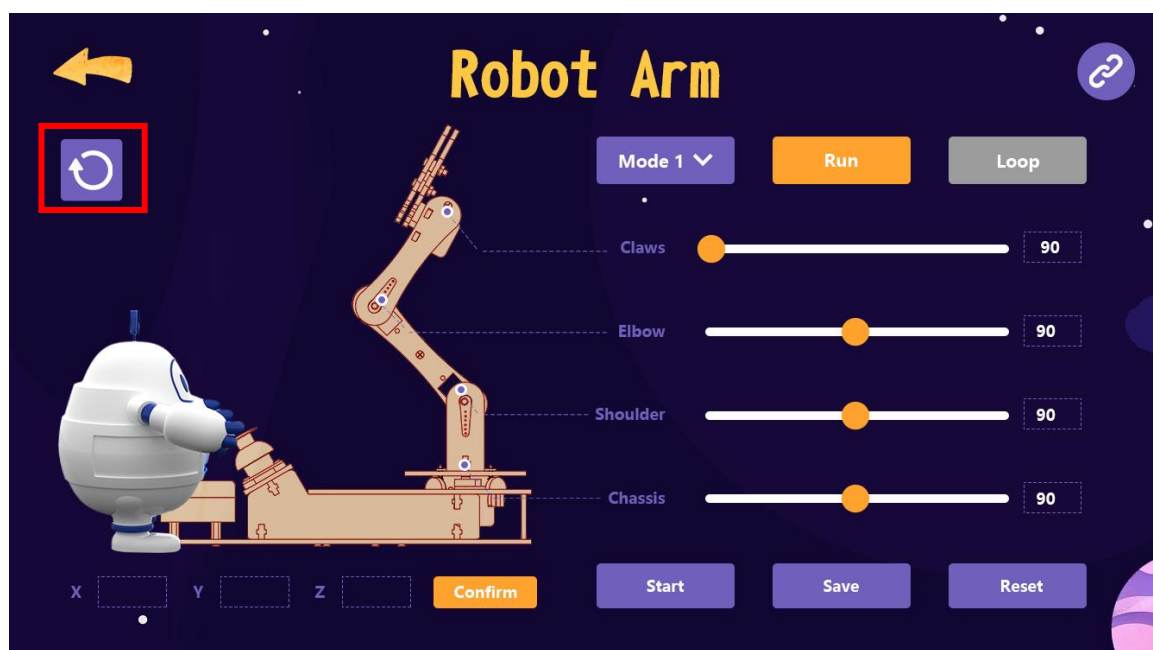


(4)**Spatial positioning:** Enter the x, y, z spatial coordinates, and the robot arm will move to the specified spatial coordinate point.

Attention: Due to the structure of the robot arm, there are limitations on its range of motion. When entering coordinates, if the values exceed the robot arm's range of motion, you can enter them again.



(5)**Position initialization:** Click on the refresh icon at the top left corner, and the robot arm will return to its initial position.



Follow Us

Scan the QR codes to Follow Us for troubleshooting & the latest news.

We have a very large community that is very helpful for troubleshooting and we also have a support team at the ready to answer any questions.



ACEBOTT FB Group QR Code



YouTube QR Code