Preface

About SunFounder

SunFounder is a technology company focused on Raspberry Pi and Arduino open source community development. Committed to the promotion of open source culture, we strive to bring the fun of electronics making to people all around the world and enable everyone to be a maker.

Our products include learning kits, development boards, robots, sensor modules and development tools. In addition to high quality products, SunFounder also offers video tutorials to help you make your own project. If you have interest in open source or making something cool, welcome to join us!

About This Kit

This cute learning kit focuses on the popular open source platform Arduino. You can learn the knowledge of the Arduino servo and ultrasonic ranging module by applying this kit.

In this book, we will show you how to build the biped robot via description, illustrations of physical components, in both hardware and software respects. You may visit our website www.sunfounder.com to download the related code and view the user manual on LEARN -> Get Tutorials and watch related videos under VIDEO.

Free Support

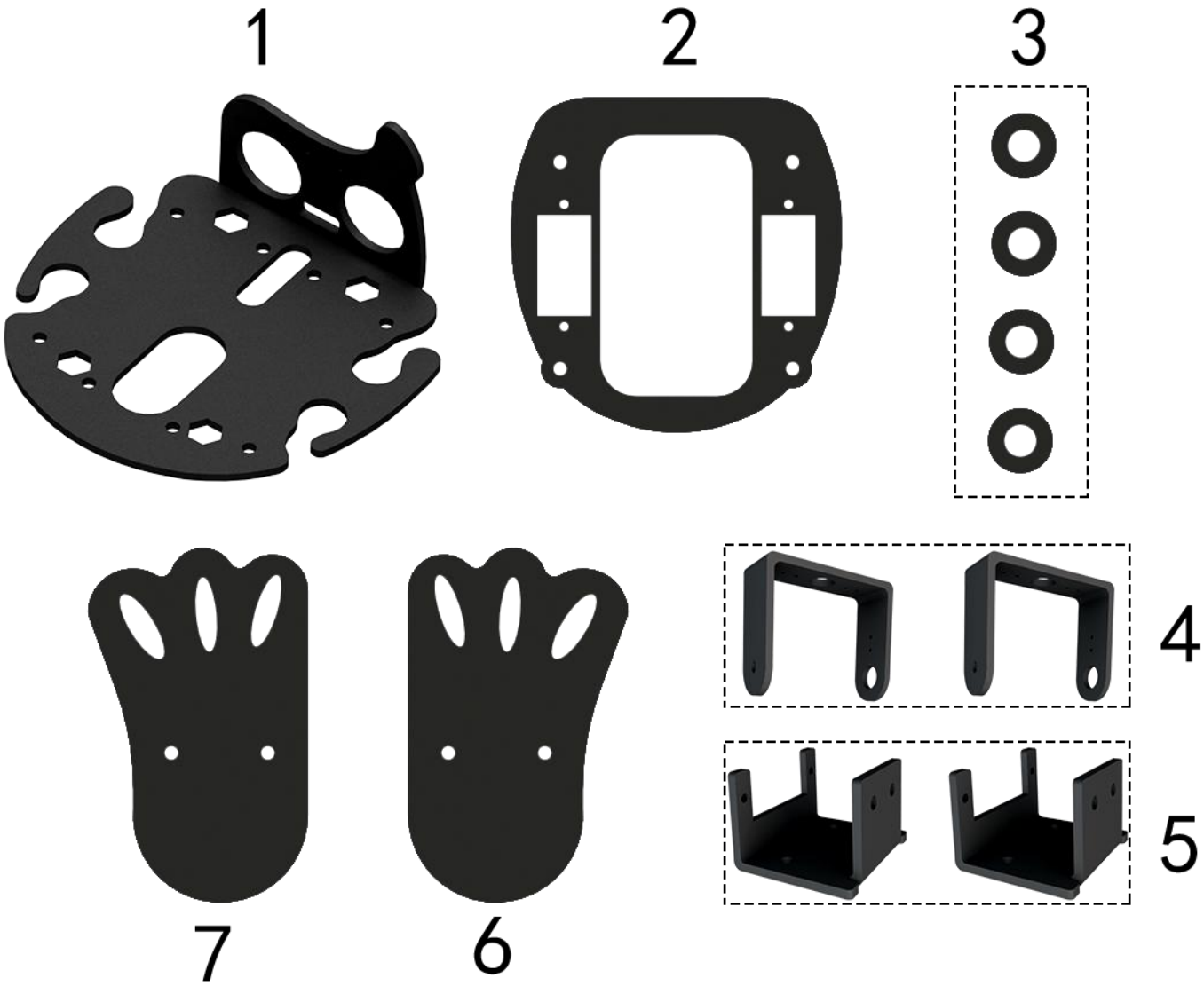
If you have any **TECHNICAL questions**, add a topic under **FORUM** section on our website and we'll reply as soon as possible.

For **NON-TECH questions** like order and shipment issues, please **send an email to service@sunfounder.com**. You're also welcomed to share your projects on FORUM.

Content

Components

Structural Plate



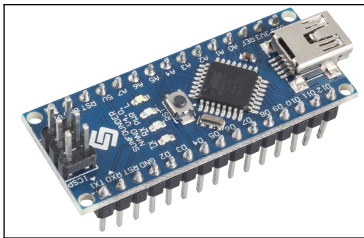
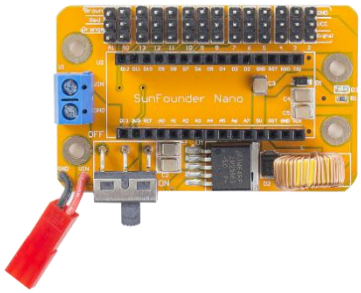
Note: Please check the structural plate and component list, if there misses any components, please take pictures of all the components you have received and inform us of the missing parts, and send E-mail to service@sunfounder.com.

Mechanical Fasteners

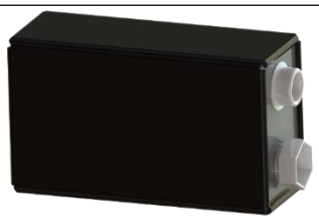
Parts	Name	Qty.
	M1.5*5 Self-tapping Screw	10
	M1.4*8 Screw	6
	M2*8 Screw	12
	M3*5 Screw	18
	M3*8 Countersunk Screw	8
	M1.4 Nut	6
	M2 Nut	12
	M3 Self-locking Nut	8
	M3*8 Bi-pass Copper Standoff	4
	M3*25 Bi-pass Copper Standoff	4

Electrical Components

Parts	Name	Qty.
	SF006C Servo	4
	Ultrasonic Module	1

	SunFounder Nano Board	1
	Expansion board	1
	Velcro Tape	1
	Mini USB Cable	1
	Battery buckle wire	1
	4-Pin Anti-reverse Cable	1
	Phillips Screw Driver	1
	Cross Socket Wrench	1

Battery (Not Included)

Parts	Name	Qty.
	9V battery	1

Introduction

This cute learning kit focuses on the popular open source platform Arduino. You can learn the knowledge of the Arduino servo and ultrasonic ranging module by applying this kit.

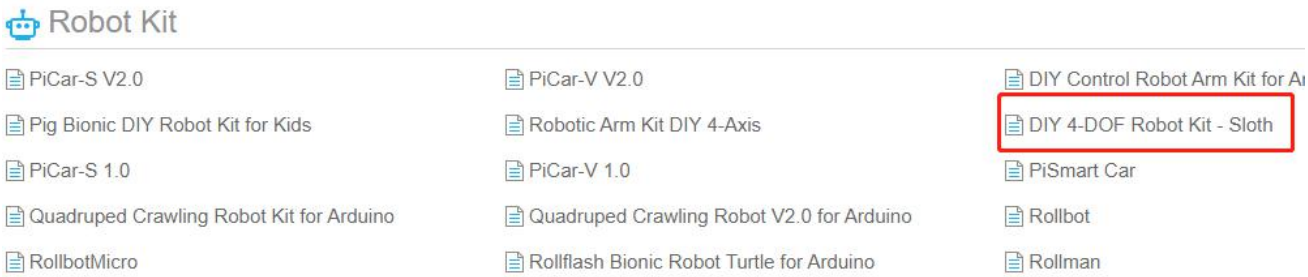
It is a new mobile robot called Sloth developed by SunFounder. Each leg has 2 joints driven by servo. One 9V chargeable lithium batteries are to supply the bot when the SunFounder Nano is used as the control board, compatible with the Arduino Nano. A servo control board connects with the batteries, servos, SunFounder Nano, and the HC-SR04 ultrasonic ranging module. Sloth can move forward and detect the range to make a turn when encountering an obstacle. In addition, when learning to program, you can also have the fun to build a pretty cool bio-robot by yourself.

Download the Code

Step 1

Go to our official website https://www.sunfounder.com/ by visiting **LEARN** -> **Get Tutorials** -> **DIY 4-DOF Robot Kit - Sloth**.

Note: The location of the kit will change, but the name will not change.

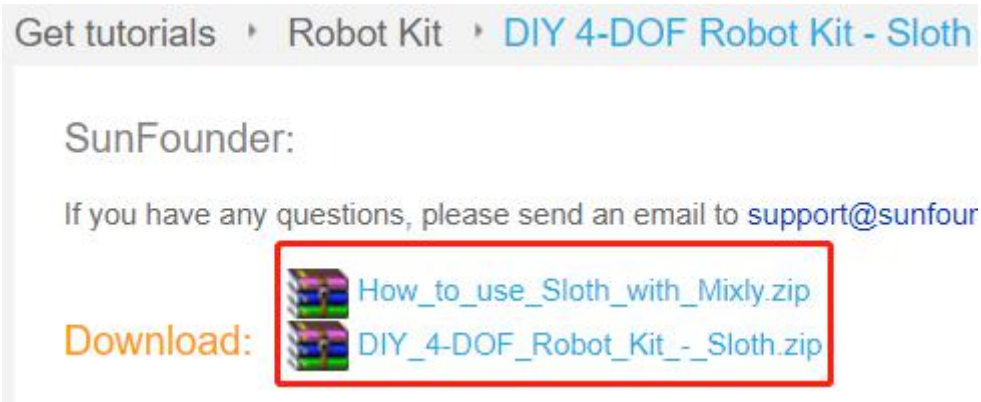


You can also directly download the file by visiting the **following link**:

https://www.sunfounder.com/learn/category/DIY-4-DOF-Robot-Kit-Sloth.html

Step 2

After entering the **DIY 4-DOF Robot Kit-Sloth** category, you can see 2 zip files. It is recommended to download **DIY_4-DOF_Robot_Kit_-_Sloth.zip** first, including the Arduino code, drivers, library, schematic and user manual.

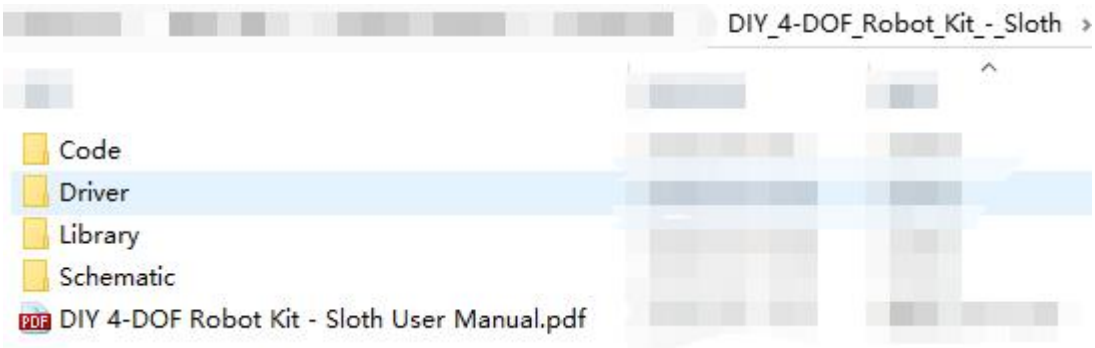


- **How_to_use_Sloth_with_Mixly.zip**: This is an extended use of Sloth via graphical programming software - Mixly to program. The manual contains only a basic introduction to Mixly and how to program in Mixly, but does not include hardware assembly part, you need to download DIY_4-DOF_Robot_Kit_-_Sloth.zip first, and after Sloth is completely assembled, you can use Mixly for programming.

Step 3

Unzip the download files.

- **DIY_4-DOF_Robot_Kit_-_Sloth.zip**



Code: The Arduino code to make Rollarm work.

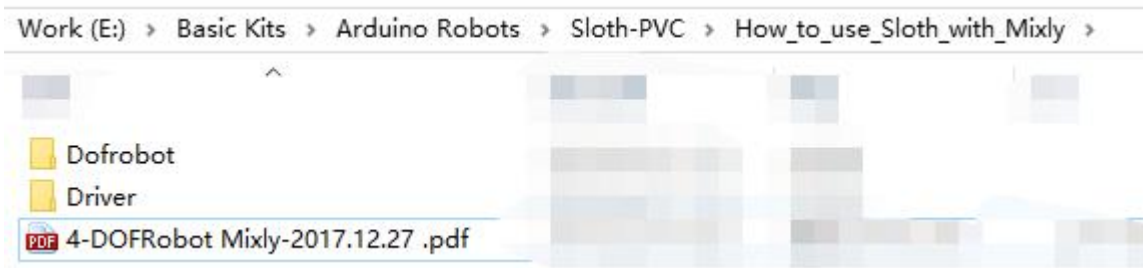
Driver: The Nano board driver software for Windows and Mac OS X system.

Library: The library of Ultrasonic sensor module.

Schematic: The schematic of the Servo Control board.

..User Manual.pdf: Sloth's complete manual, including Arduino installation and basic use, Sloth's hardware assembly and code use.

- **How_to_use_Sloth_with_Mixly.zip**



Dofrobot: The library that you need to add in the Mixly.

Driver: The Nano board driver software for Windows and Mac OS X system.

..User_Manual.pdf: The user manual of how to program in Mixly.

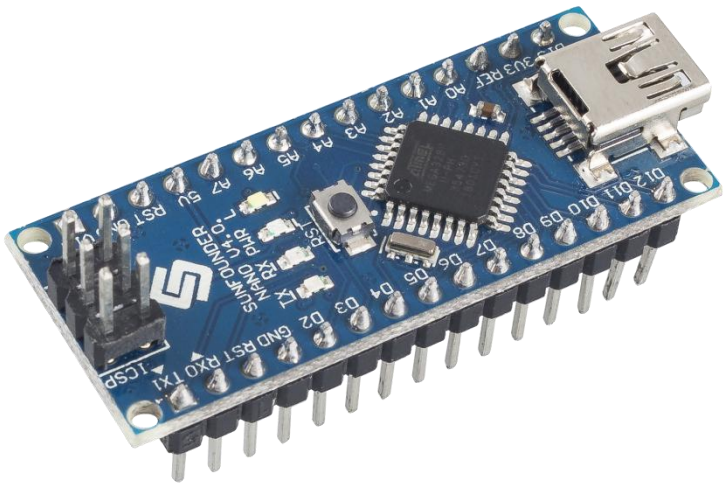
Getting Started

Arduino

Arduino is an open source platform that applies simple software and hardware. You can get it in a short even when you know little of it. It provides an integrated development environment (IDE) for code editing and compiling, compatible with multiple control boards. So you can just download the Arduino IDE, upload the sketches (i.e. the code files) to the board, and then you can see experimental phenomena. For more information, refer to http://www.arduino.cc.

Arduino Board – SunFounder Nano Board

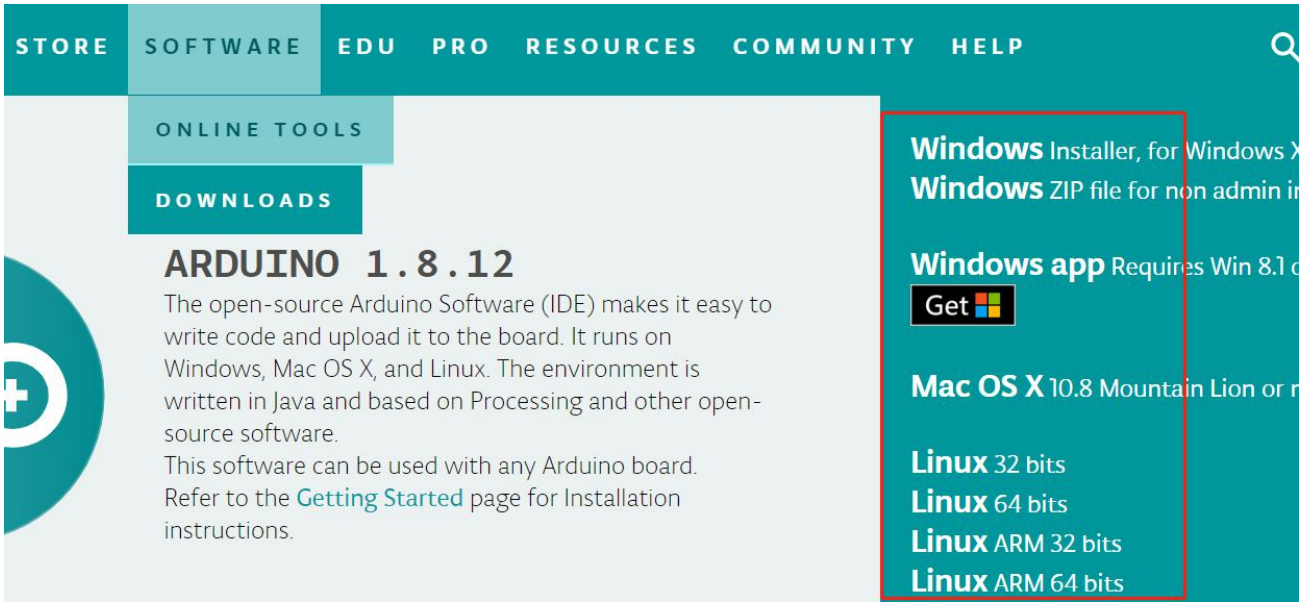
The SunFounder Nano is a small, complete, and breadboard-friendly board based on the ATmega328P (Arduino Nano 3.x). It is mostly the same as the Arduino Nano, except that the Sunfounder Nano uses the PL2303TA driver chip, while the Arduino Nano uses the Atmega16U2. So in the first use, some computers had to install the driver manually to allow the computer to recognize the SunFounder Nano board.



Install Arduino IDE

The code in this kit is written based on Arduino, so you need to install the IDE first. Skip it if you have done this.

Now go to the arduino.cc website and click **DOWNLOAD**. On the page, check the software list on the right side under **Download the Arduino Software**.



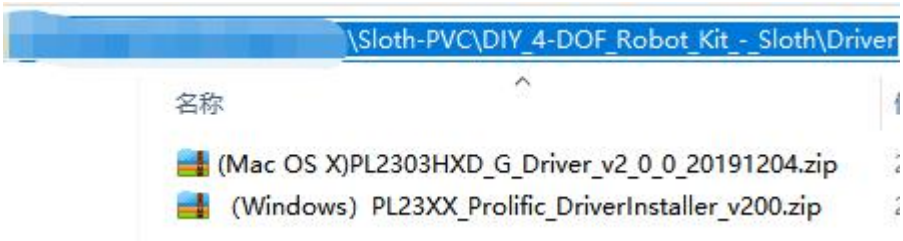
Find the one that suits your operation system and click to download. There are two versions of Arduino for Windows: **Installer** or **ZIP file**. You're recommended to download the former. Just download the package, and run the executable file to start installation. It will download the driver needed to run Arduino IDE. After downloading, follow the prompts to install. For the details of installing steps, you can refer to the guide on **Learning**->**Getting Started with Arduino**, scroll down and see **Install the Arduino Software**.

After installing, you will see Arduino icon on your desk and double click to open it.

Install the Driver

The USB driver chip of SunFounder Nano board is **PL2303TA**. After installing the Arduino IDE, use Mini USB Cable to connect the SunFounder Nano to your computer. Some Windows computers may install the driver automatically after a while, and the corresponding COMxx pops up. Congratulations, you can use it directly without having to install the driver manually.

For some users who cannot install the Driver automatically, we provide the Driver files for Windows and Mac OS X systems in the path **DIY_4-DOF_Robot_Kit_-_Sloth\Driver**.



Unzip and open the PDF inside and install the Driver according to the **Windows/Mac OS X Driver Installation** section.

Since the PL2303 driver installer is always being updated, the driver provided by us may not be compatible with your computer system. Please go to the link below to download the latest driver.

For Windows users:

http://www.prolific.com.tw/US/ShowProduct.aspx?p_id=225&pcid=41.

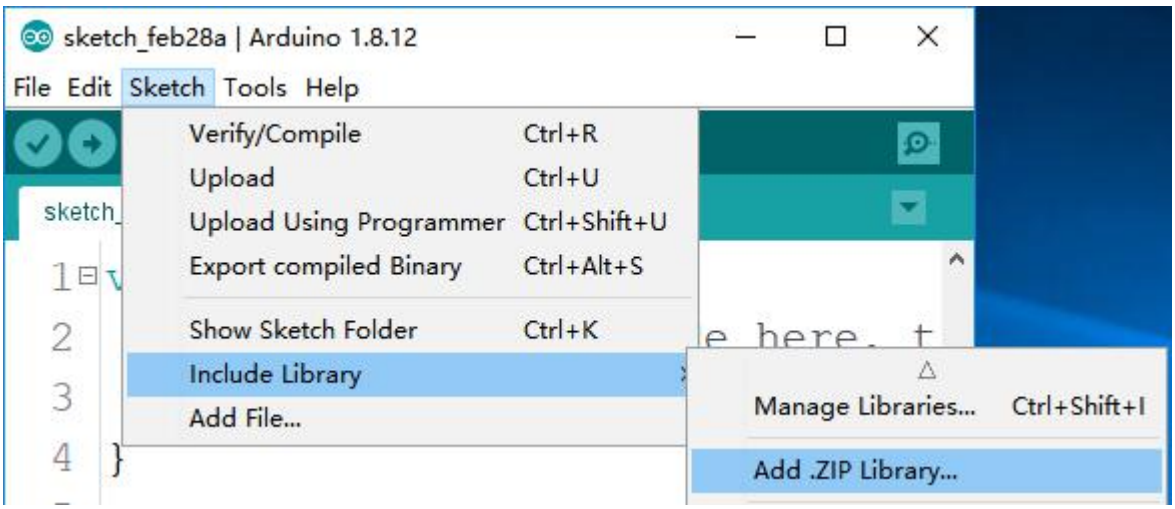
For Mac users:

http://www.prolific.com.tw/US/ShowProduct.aspx?p_id=229&pcid=41

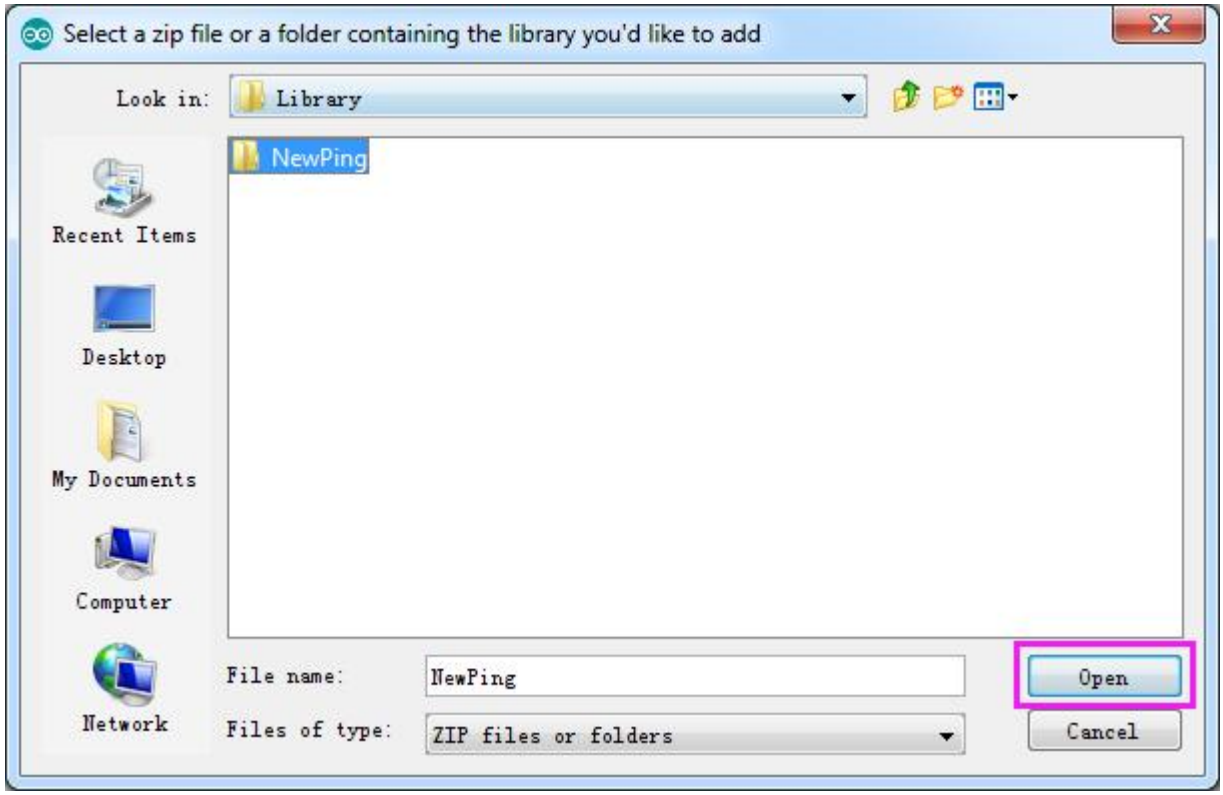
Add Libraries

In the following codes, we'll use an ultrasonic sensor to measure the distance, but its library doesn't default to the Arduino system library, and it'll come out when you run the code about ultrasonic sensor later: NewPing.h: no such file or directory. So we need to add it manually. The steps are as follows:

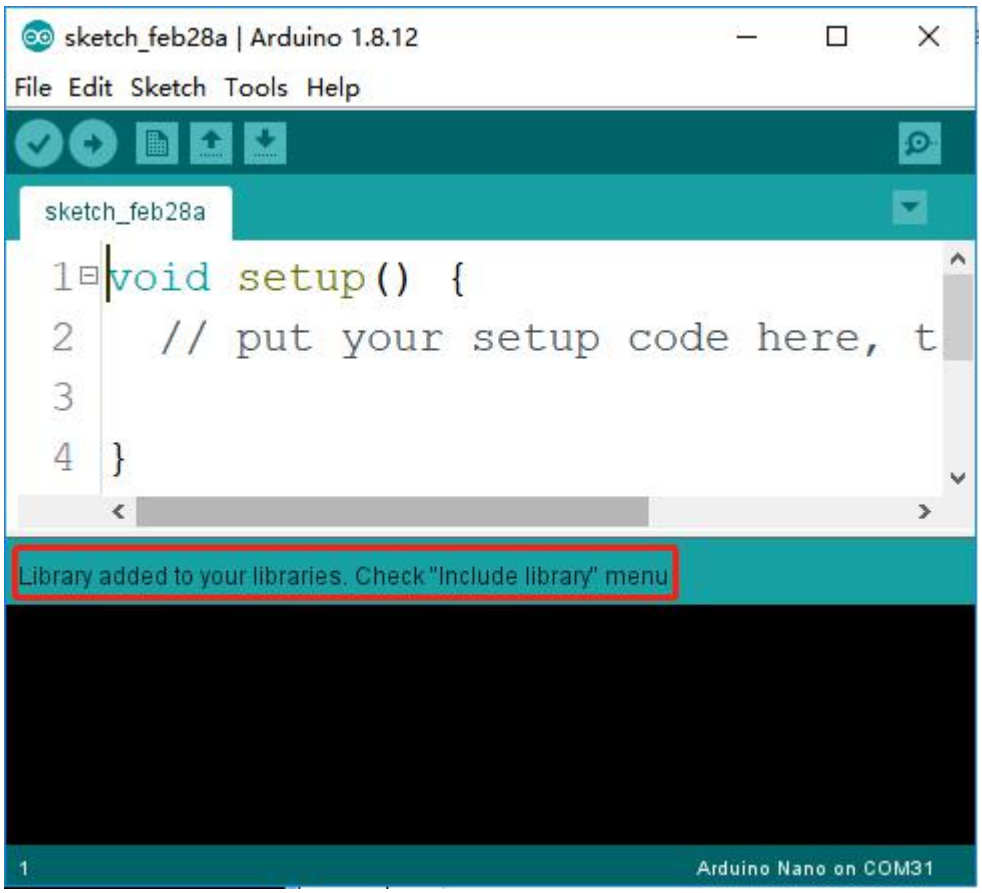
Open Arduino IDE, select Sketch -> Included Library -> Add .ZIP Library.



Enter the path ***DIY_4-DOF_Robot_Kit_-_Sloth\Library*** and select to open NewPing.



Then you will see "Library added to your libraries", indicating the library has been included successfully.



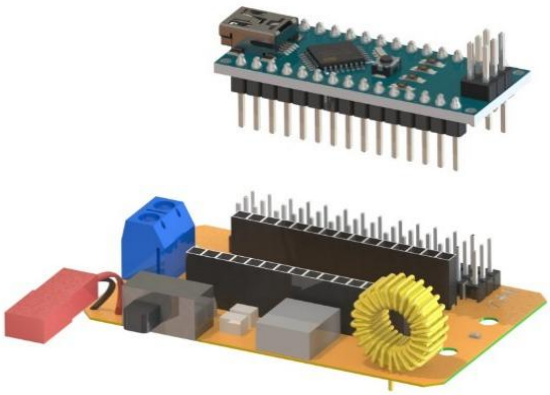
Test for Servos and Ultrasonic Module

Before assembling, you need to test the servos and the ultrasonic module according to the following steps.

Servo Test

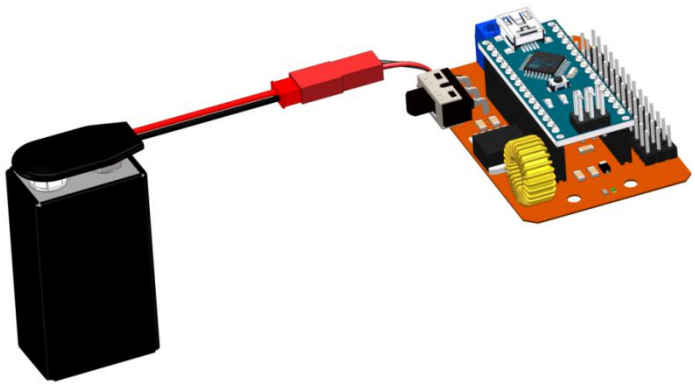
Step 1: Insert SunFounder Nano board into the Servo Control Board.

Note: The USB port should be at the same side with blue power supply terminal.



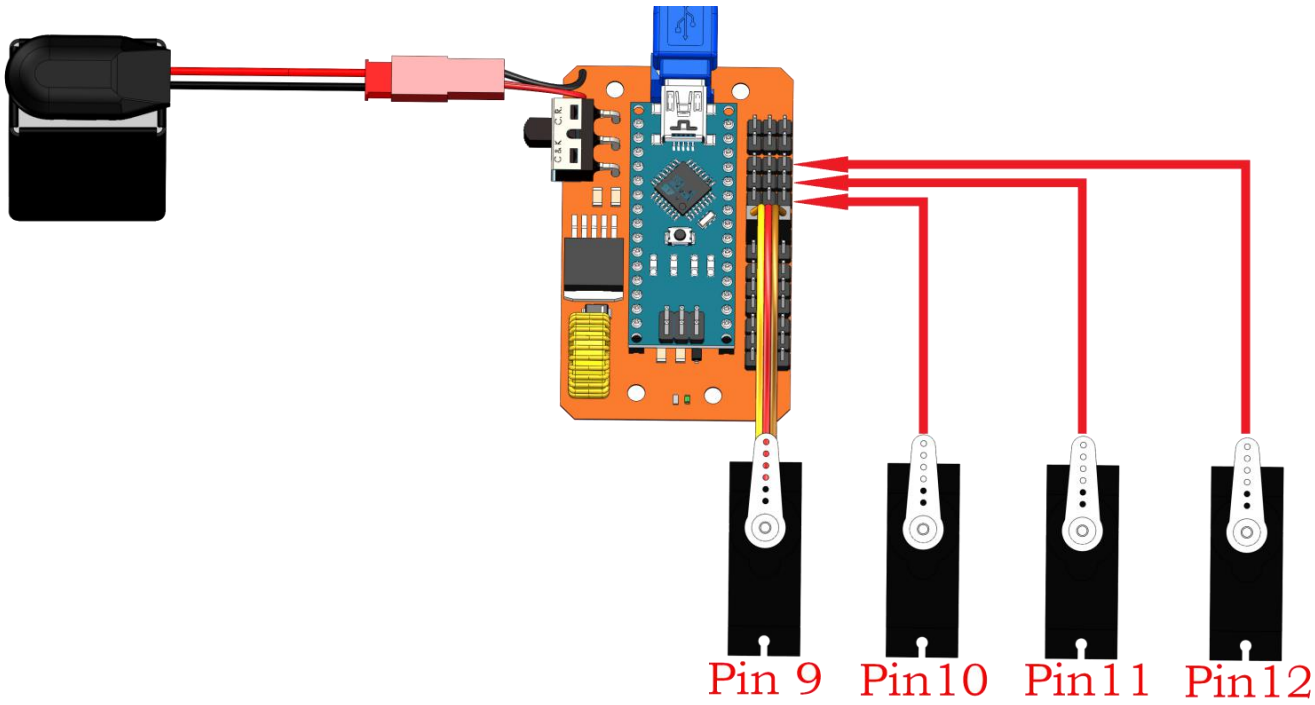
Step 2: Insert the battery to the battery cable.

And connect the battery cable to the expansion board.



Step 3: Connect four servos to pin 9 to pin 12 of the expansion board.

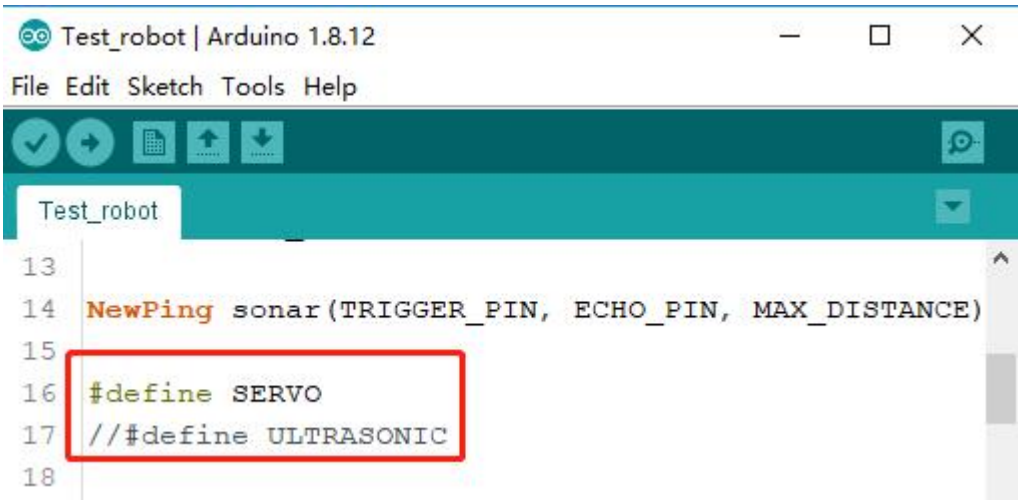
Note: The **yellow**, **red**, and **brown** wires connect to **Signal**, **VCC**, and **GND** on the expansion board, respectively.



Step 4: Open the **Test_robot.ino** under this path of DIY_4-DOF_Robot_Kit_-_Sloth\Code\Test_robot.

Go to line 16 and delete the comment sign "//" (if any) before #define SERVO to start the corresponding servo test code; add the comment sign "//" before #define ULTRASONIC to disable the corresponding ULTRASONIC test code.

Note: In this sketch, starting both test functions at the same time affects test results.



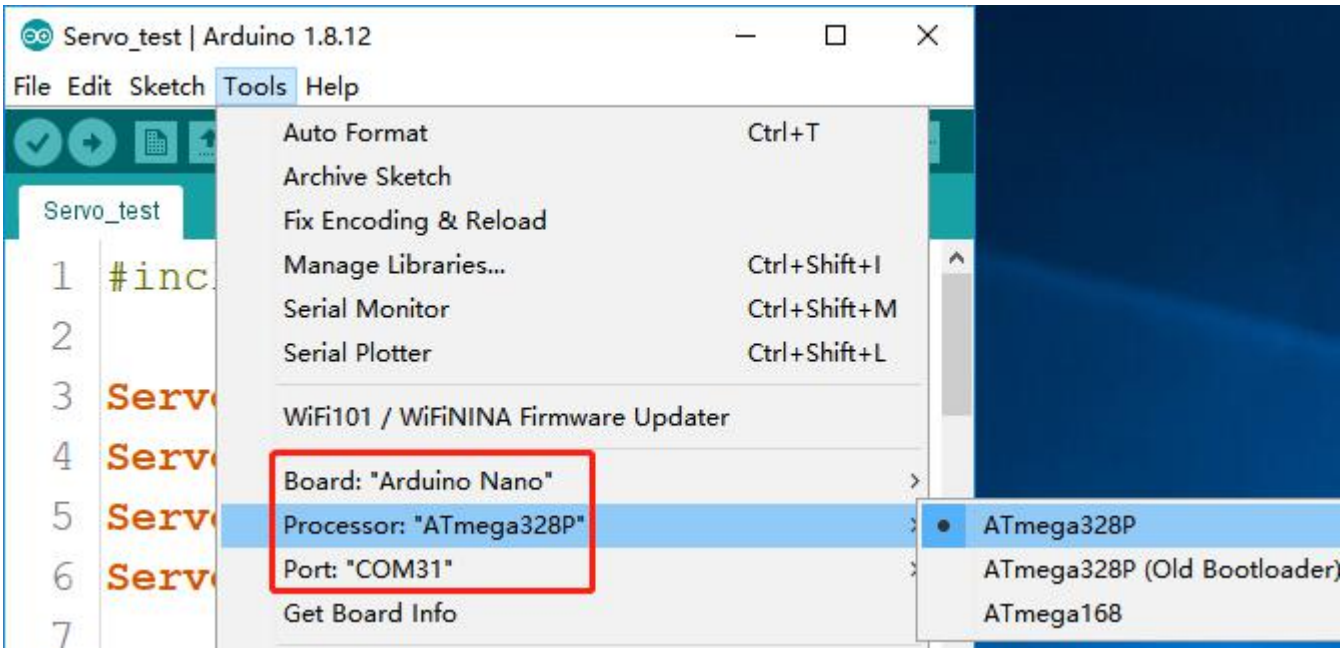
Step 5: Select the corresponding Board, Processor and Port.

- **Board:** Arduino Nano.

- **Processor:** ATmega328P.

Note: If the code cannot be uploaded successfully for a long time, it needs to be changed to ATmega328P (Old Bootloader).

- **Port:** Random allocation. The corresponding option can be determined by pulling out the USB cable and reconnecting the nano. Usually a combination of "COM + Numbers".

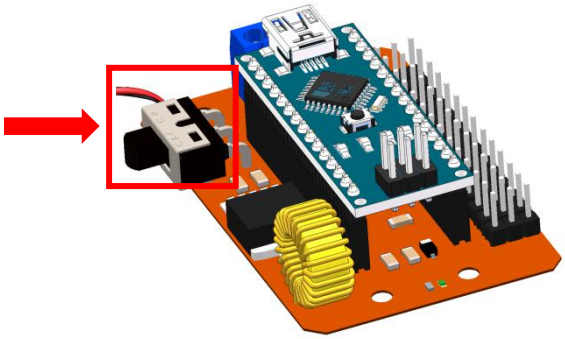


Step 6: Upload codes to SunFounder Nano board.

After waiting for a few seconds, the download process is successful. The following window will prompt “Done uploading”.

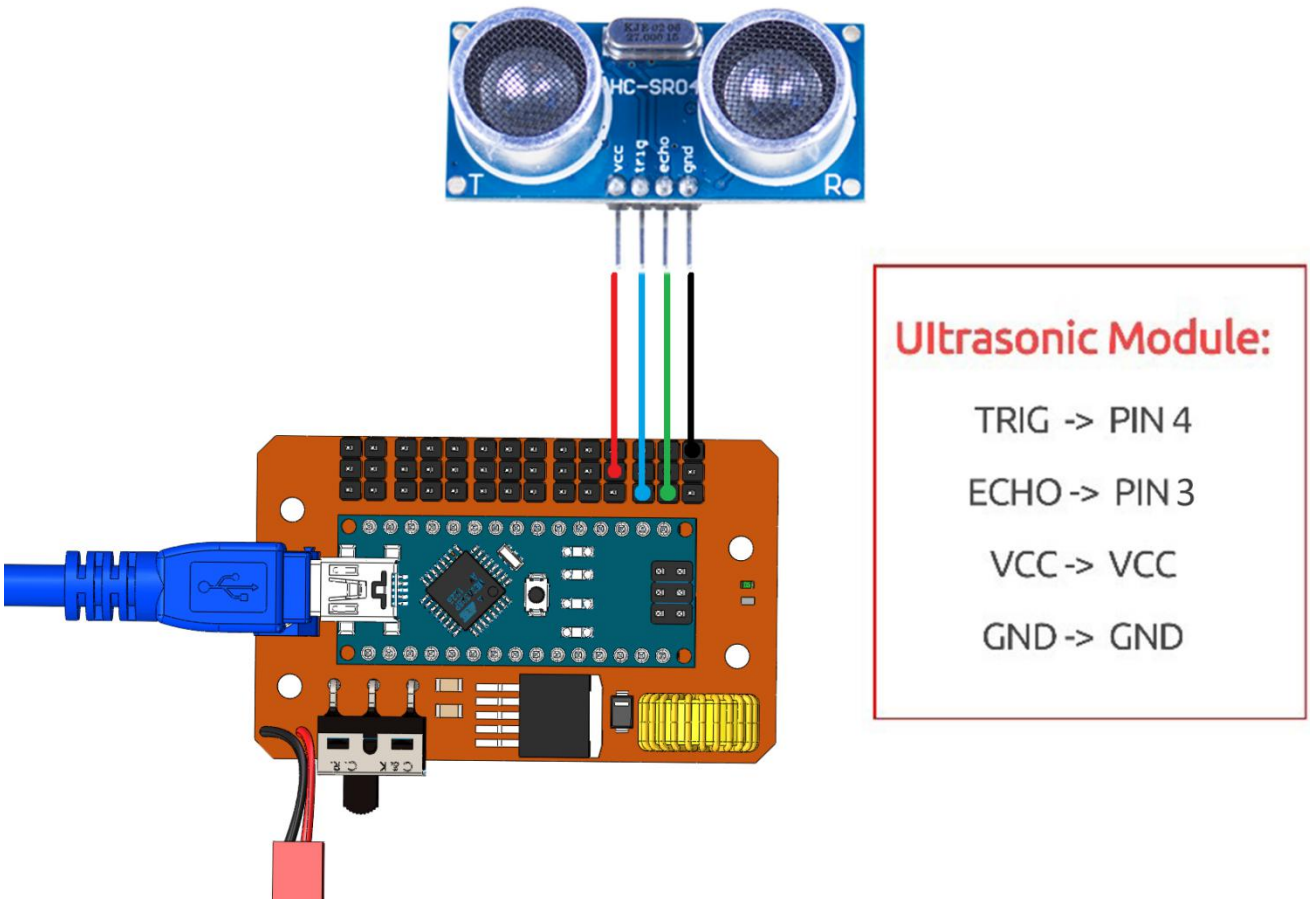
Note: If the code cannot be uploaded successfully for a long time, Processor needs to be changed to ATmega328P (Old Bootloader).

Step 7: **Slide the power switch on the servo control board to ON**. You will see the rocker arm rotates within 0-180 degrees, indicating the servo can work.



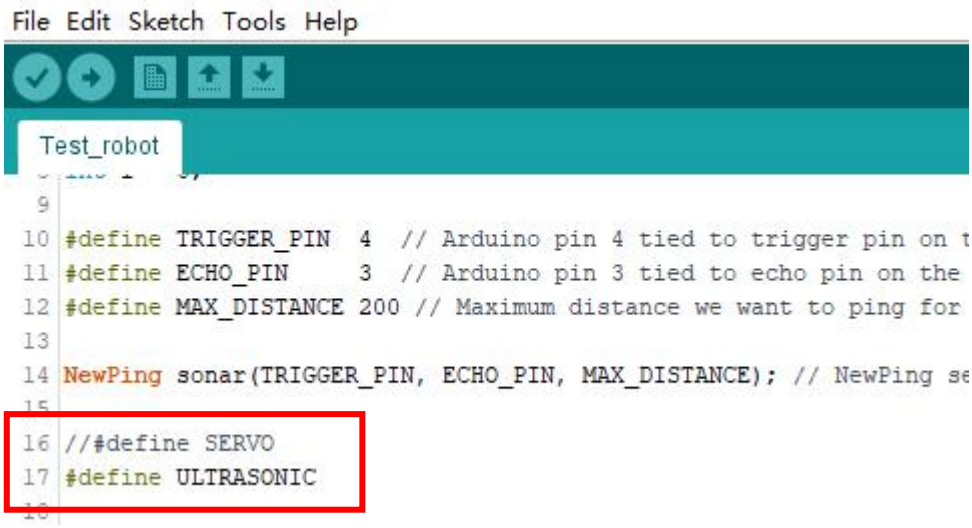
Ultrasonic Test

Step 1: Connect Ultrasonic module to Servo Control Board via 4-Pin Anti-reverse Cable.



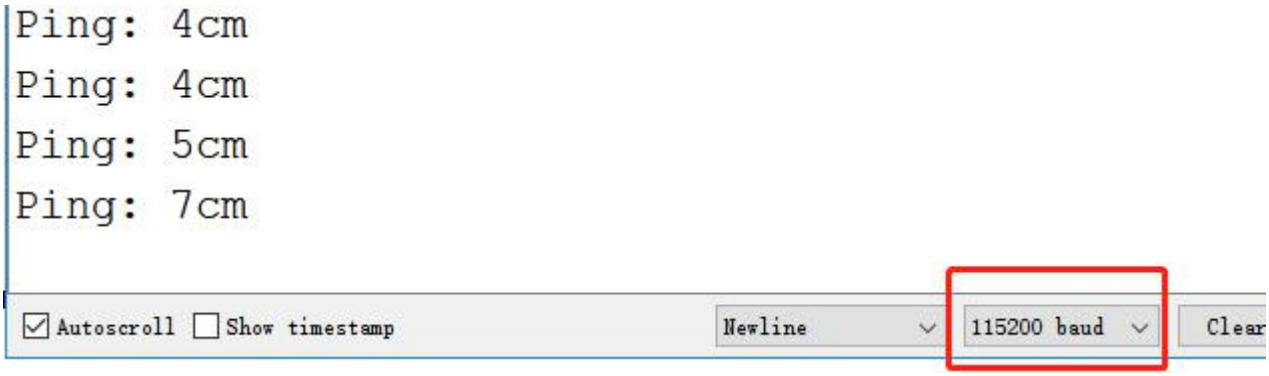
Step 2: Open the **Test_robot.ino** and select Board, Processor and Port.

Step 3: Go to line 16 and add the comment sign "//" before #define SERVO; delete the comment sign "//" before #define ULTRASONIC.



Step 4: Upload the code first. After it's uploaded successfully, click the Serial Monitor icon on the right to enter it.

Step 5: Set the baud rate to 115200 (started by line 25 serial.bein (115200)).



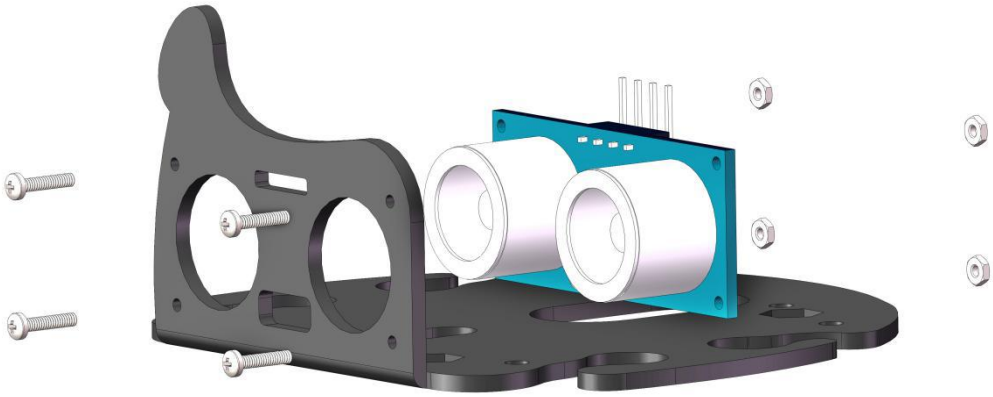
Step 6: Similar to the last step of **i. Servo Test, slide the power switch on the servo control board to ON, now you can see the detected distance.**

Note: The detected data of 0cm is likely to be invalid data, because the trigger probe of ultrasonic module cannot emit ultrasonic wave, or after the ultrasonic wave is emitted, the echo probe cannot receive the echo. When using, be careful not to cover individual probes.

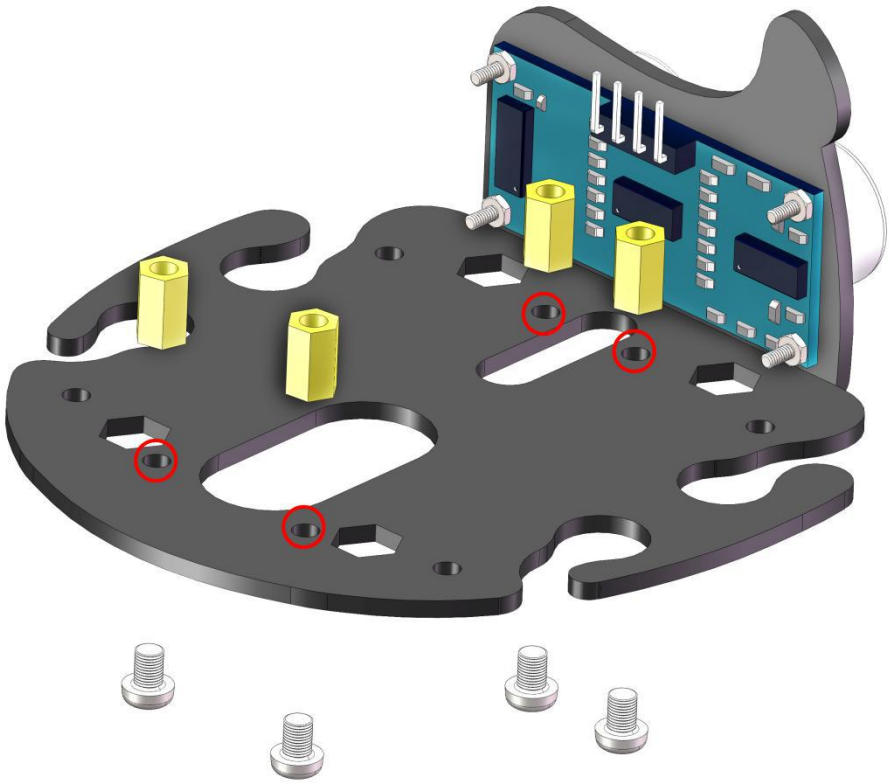
Assembly

Head Assembly

Insert the ultrasonic module into No. 1 board and secure it with **M1.4*8 screws** and **M1.4 nuts**.

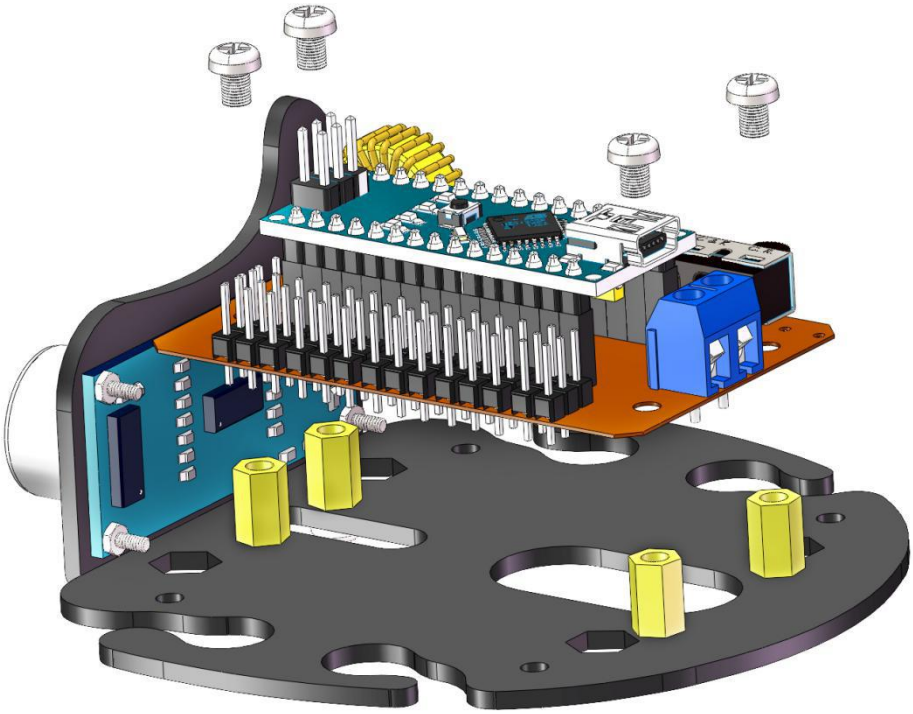


Use a **M3*5 screw** to secure the **M3*8 Bi-pass Copper Standoff** post on No. 1 board.

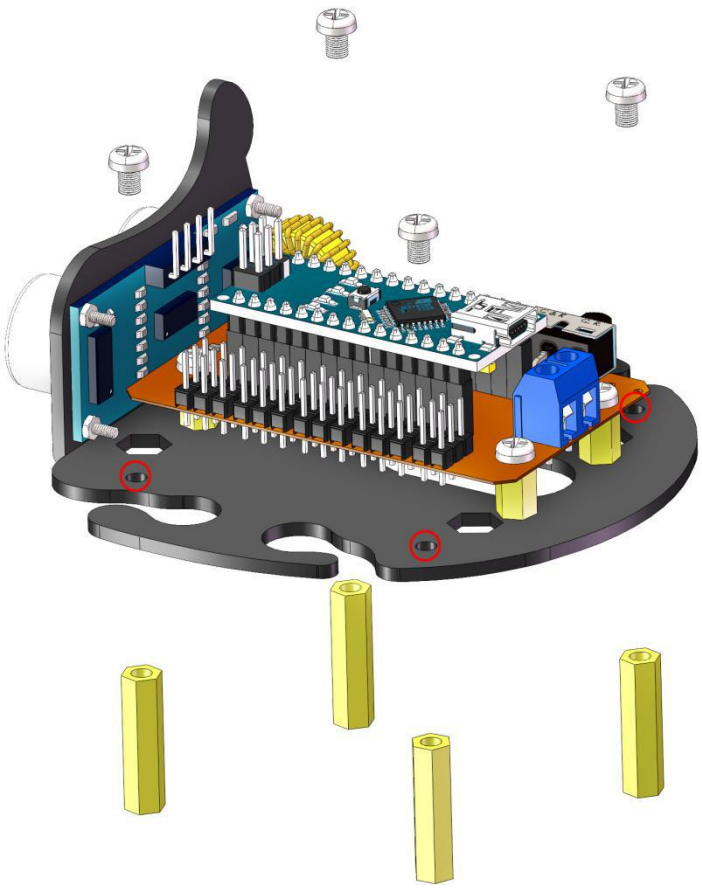


Electrical Module Assembly

Use a **M3*5mm screw** to mount the previously installed circuit board on No. 1 board.

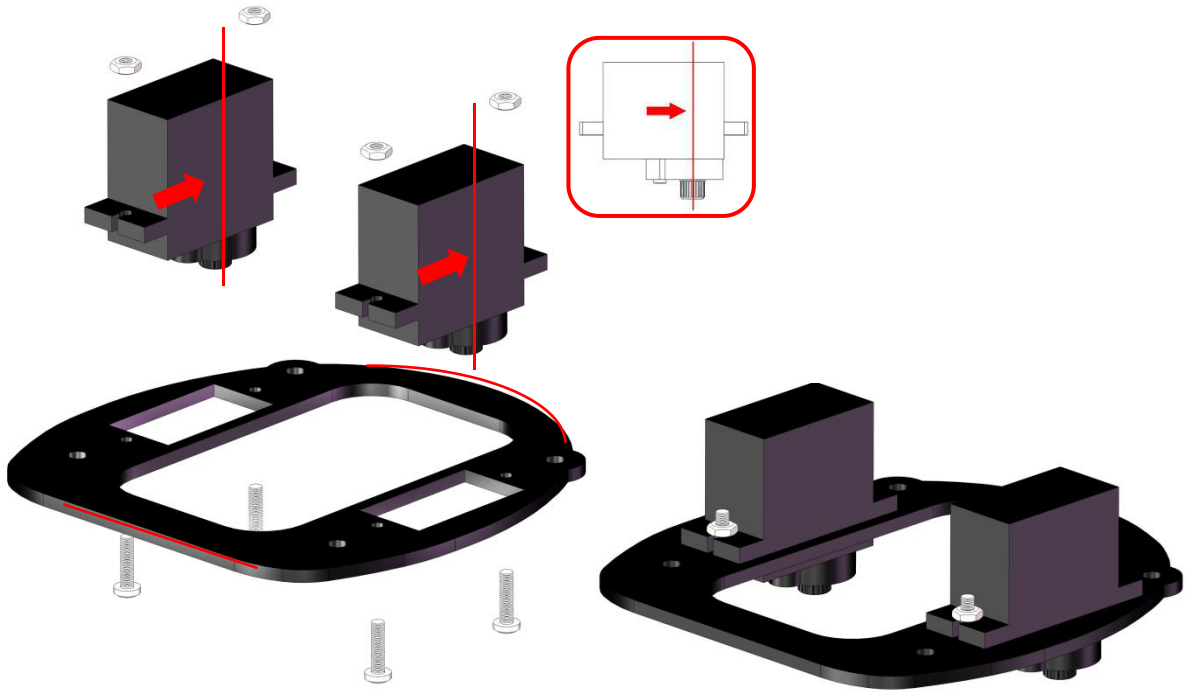


Use **M3*5 screws** to fix **M3*25 Bi-pass Copper Standoff** under the No. 1 board.

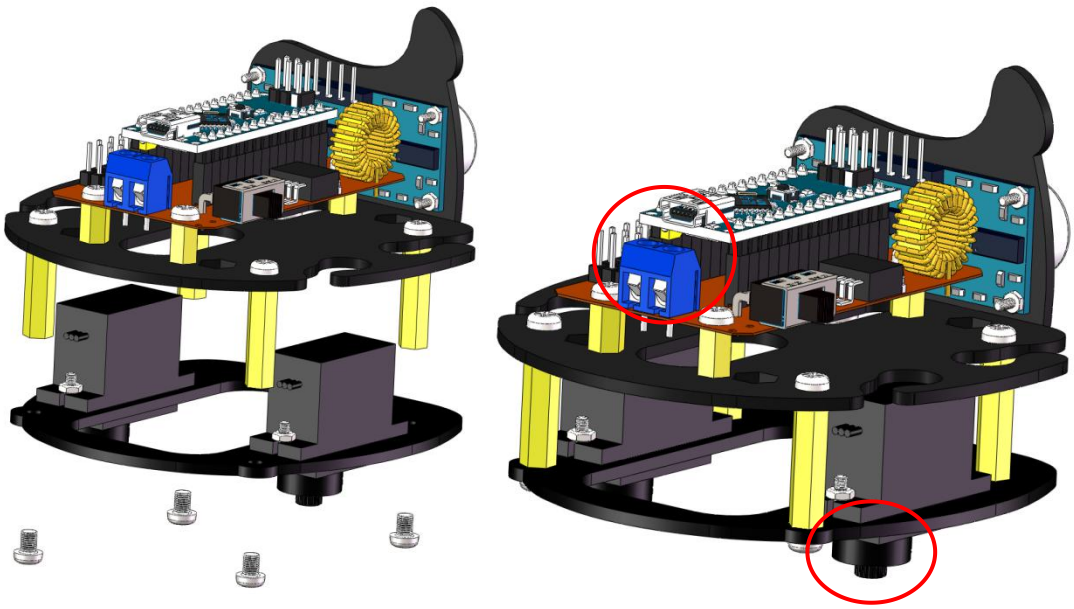


Servo Assembly

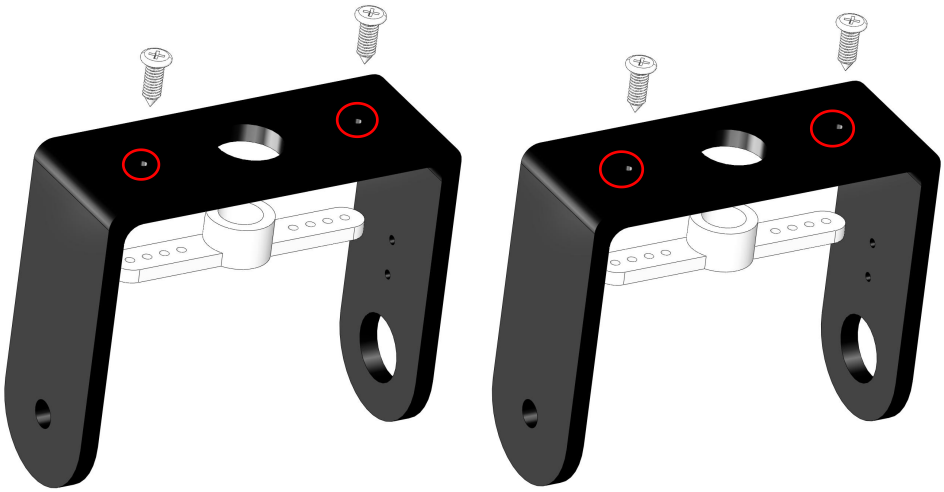
Use **M2*8 screws** and **M2 nuts** to mount the servo on the corresponding position on the No. 2 board. (Note the direction of the servo installation)



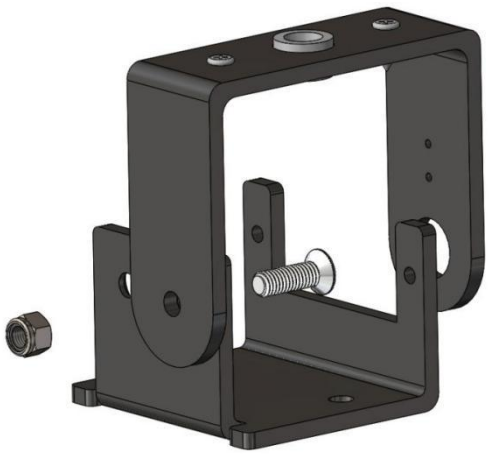
Secure the No. 1 and No. 2 boards with **M3*5 screws**. Note that the side of the servo shaft should be mounted on the side of the USB port.



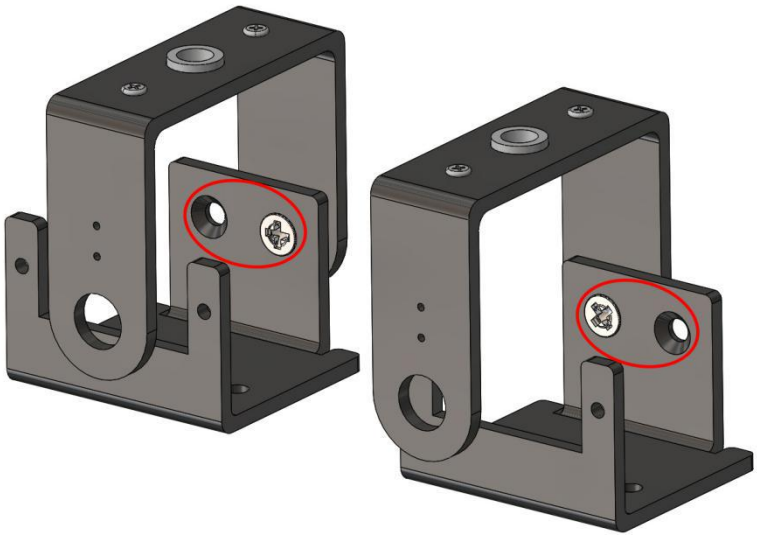
Use two **M1.5*5 self-tapping screws** to fix the 2-arm rocker arm to the No. 4 board and use the same method to install another No. 4 board.



Secure one of the round holes on the 4th and 5th boards with **M3*8 Countersunk screws** and **M3 self-locking nuts**.



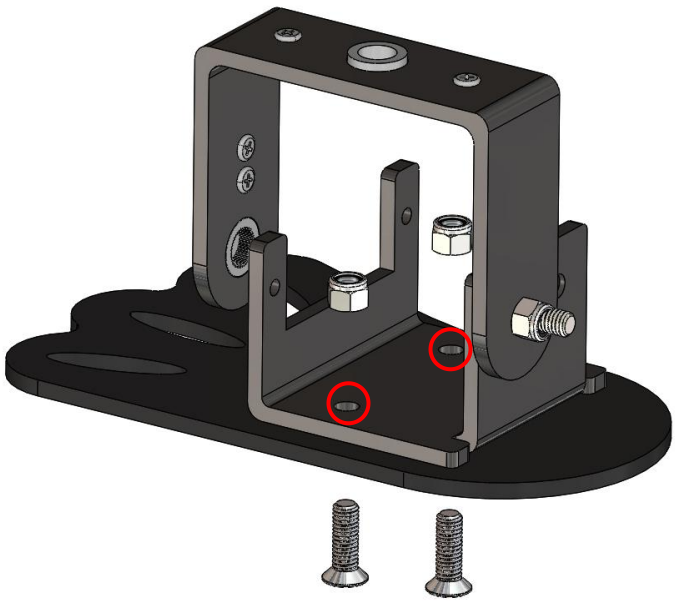
Use the same method to secure the other round hole on the 4th and 5th boards, as shown in the following figure:



Use two **M1.5*5 self-tapping screws** to secure the 1-arm rocker arm on the No.4 board.

Install another No.4 board in the same way.

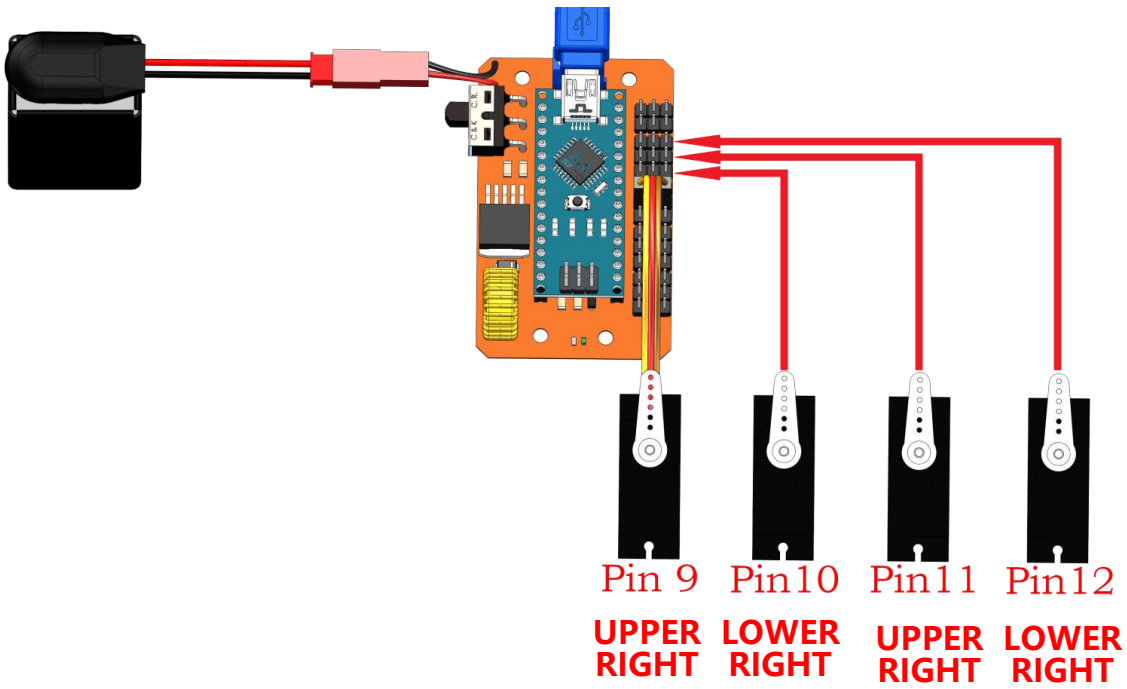
Turn the No. 6 board with the countersunk side down and secure the No. 6 board to the **right leg** described above with the **M3*8 countersunk screw** and the **M3 self-locking nut**.



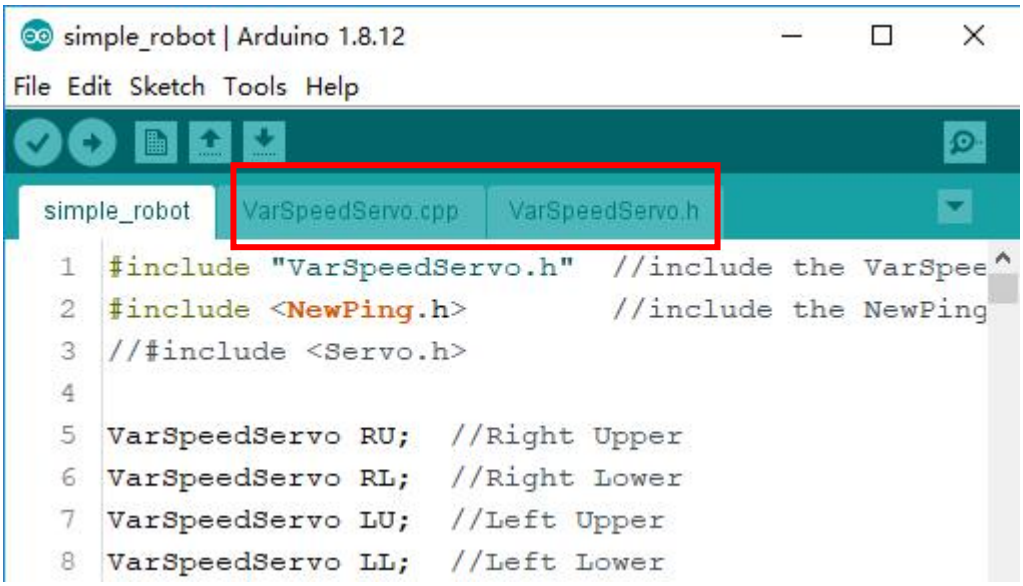
The same method can be used to secure the No.7 and the **left leg**. Observe the picture carefully. The left and right feet you have installed need to be exactly the same as that in the picture. Otherwise, the robot won't walk properly.

Servo INSTALL Test

Connect the 4 servos to pin **9**, **10**, **11** and **12** respectively again. This is designed to keep the servo angle of the upload code at 90°(internal angle) before the servo shaft is installed, in order to let the **Sloth** remain upright after assembly.



Open the program **simple_robot.ino** under the path of **DIY_4-DOF_Robot_Kit_-_Sloth\Code\simple_robot.** After opening, you can see the other 2 files: VarSpeedServo.cpp and VarSpeedServo.h are opened at the same time. This two files are set to adjust the angle of the servo.

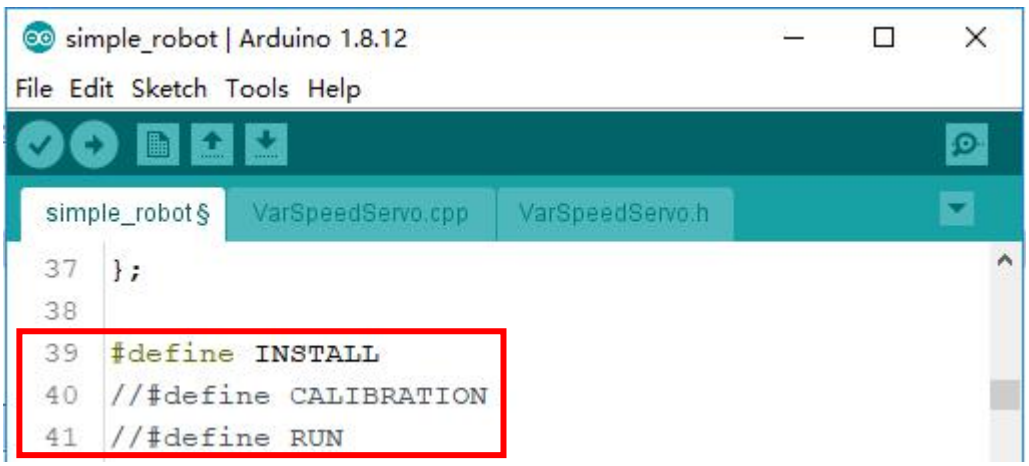


There are three **#define** statements in line **39-41.** Removing the respective comment signs "//" enables you to start their functions as shown.

#define INSTALL: Start the INSTALL mode, in which 4 servos will be fixed at 90°for assembly.

#define CALIBRATION: Start the calibration mode, in which the angles of 4 servos can be adjusted.

#define RUN: Start the RUN mode, in which the robot can go ahead and get round if it meets obstacles.



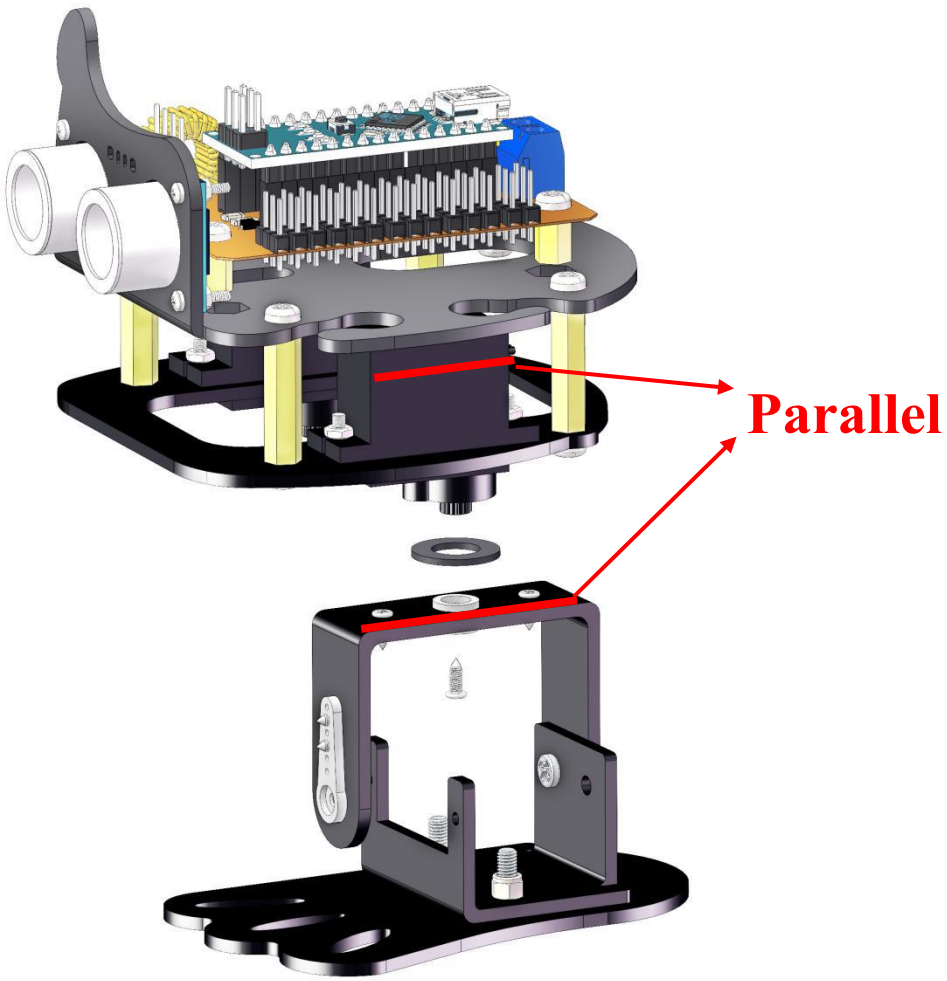
Note: Only one function can be used at the same time. Starting multiple functions might break down the robot.

In the current step, use INSTALL mode. Then select the corresponding **Board, Processor and Port**. The code is then uploaded into the **SunFounder Nano** board. Don't forget to toggle the **power switch** to **ON**. When the servo control board is powered on, the servo will rotate to the position specified by the program.

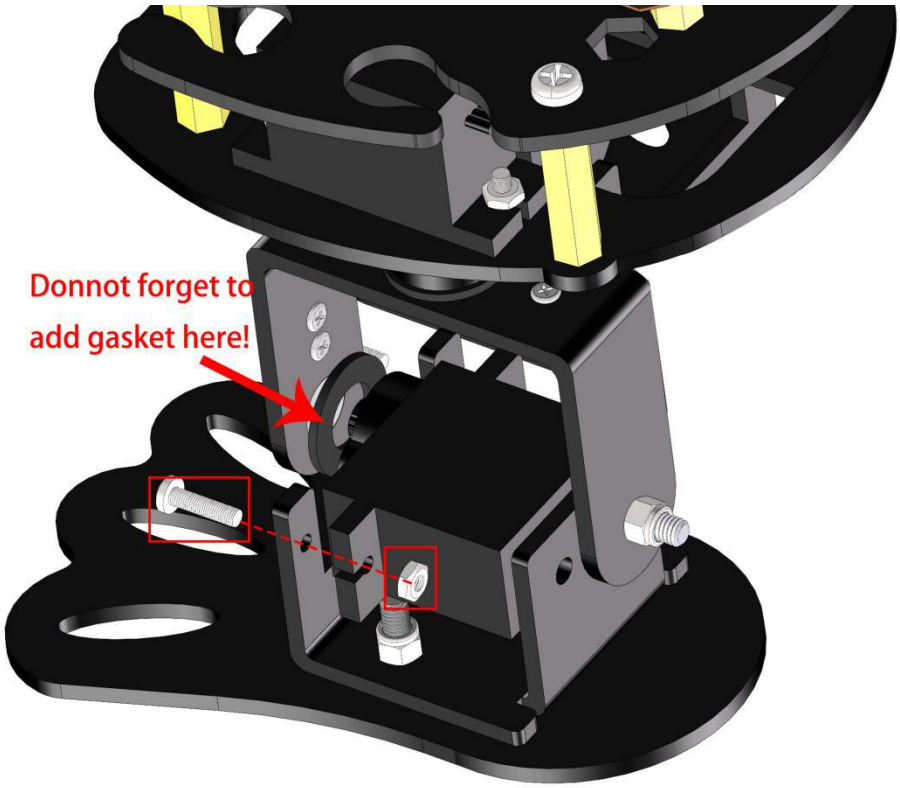
Foot Assembly

Note：Keep on the power until the whole step.

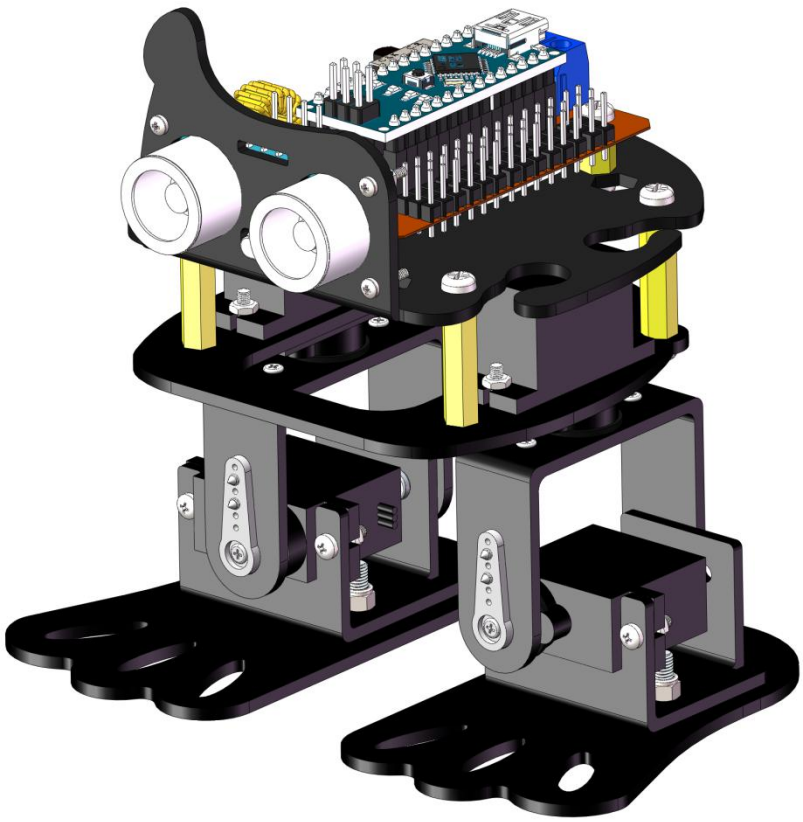
Assemble the left leg with the smallest screws in the packaged with servo, a gasket plate is needed between the servo and left leg. Try to keep the edges of the 4th board and the servo parallel to each other. If deviation are found at installation, it is normal and we will adjust them later when calibrating.



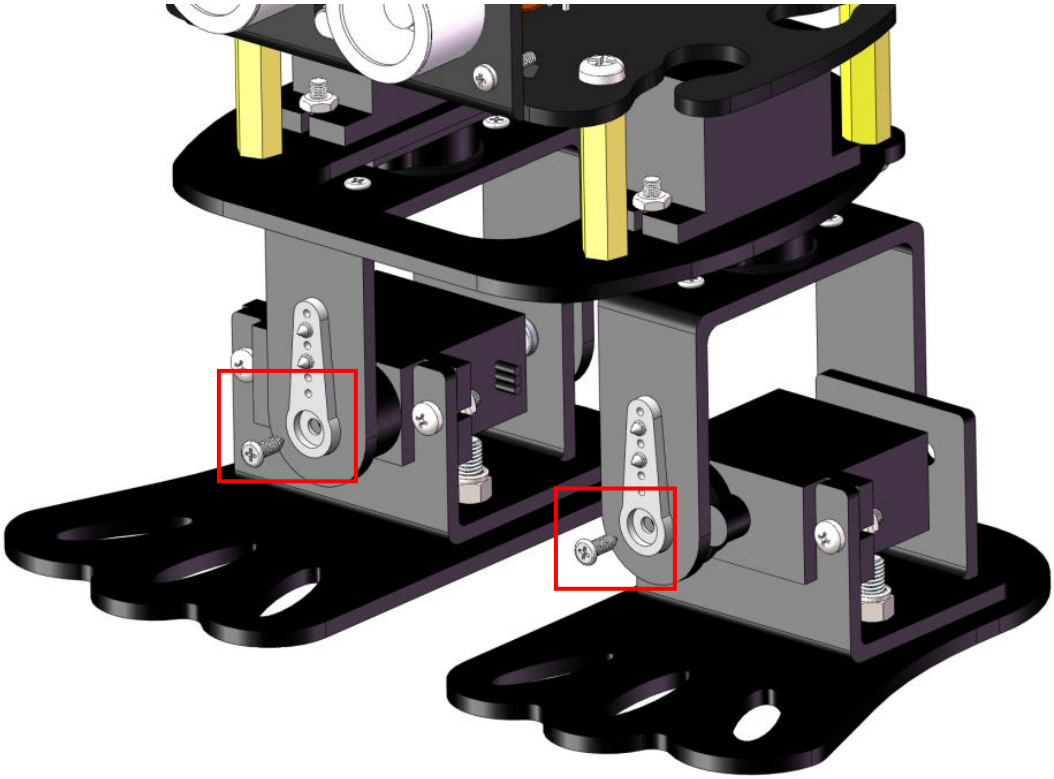
Insert a servo (in working condition) into the servo shaft of the left foot. Besides 2 M2*8 screws and 2 M2 nuts, a gasket plate is needed between the servo and left leg.



Assemble the right leg in the same way.

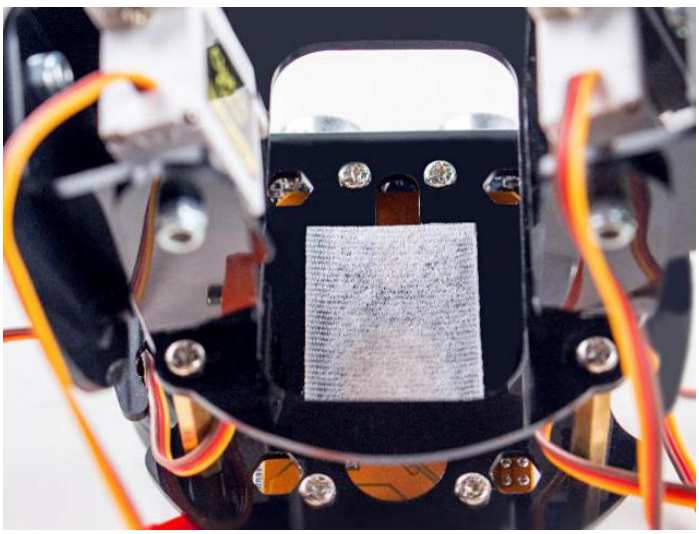


Secure the 2 legs with the smallest screws in the packaged with servo.

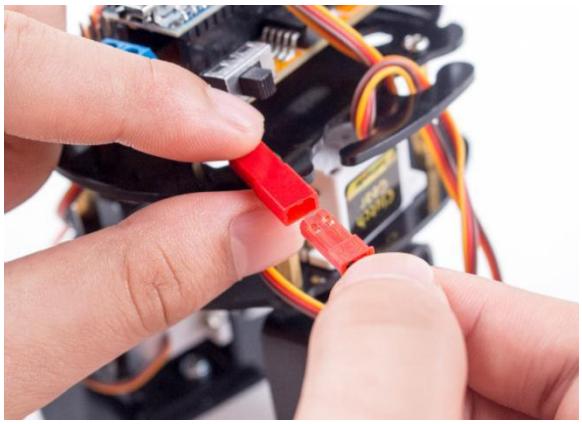


Battery Assembly

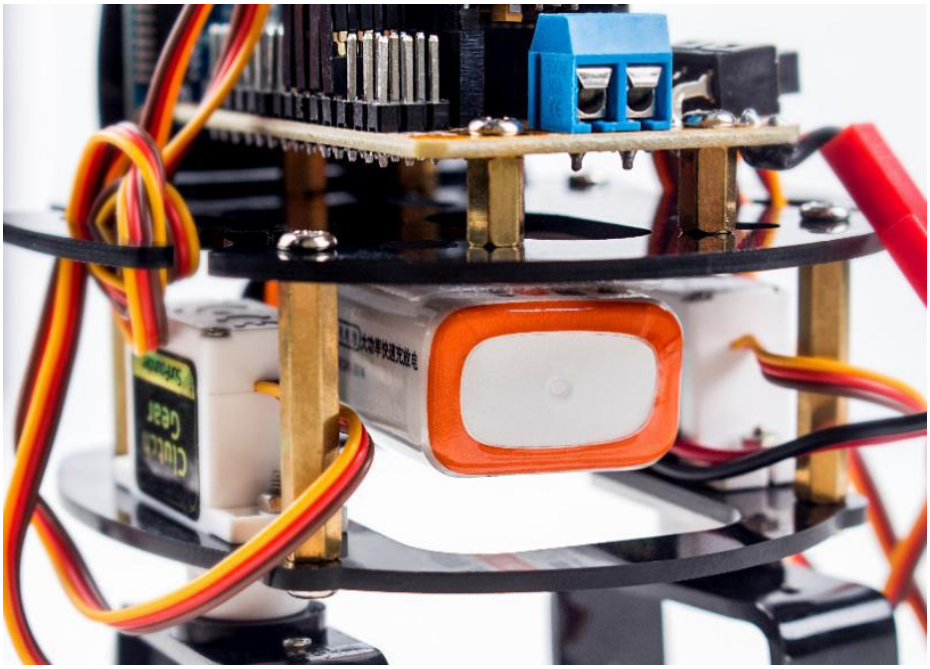
Attach one side of velcro tape to the bottom of the No. 1 board and the other side to the battery.



Insert the battery into the battery cable and plug the other end into the expansion board.

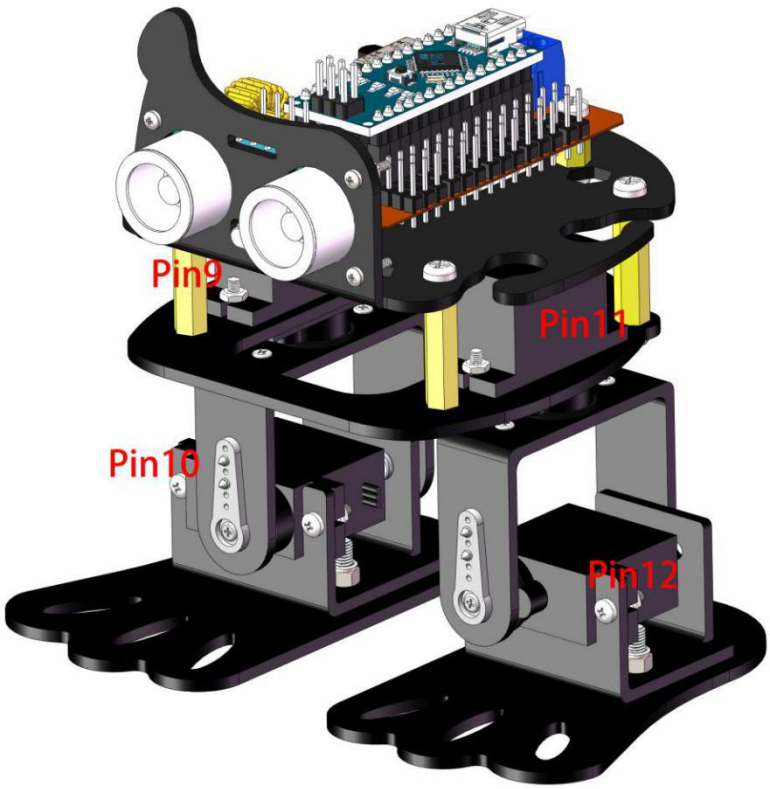


Lastly, paste the battery on the No. 1 board.

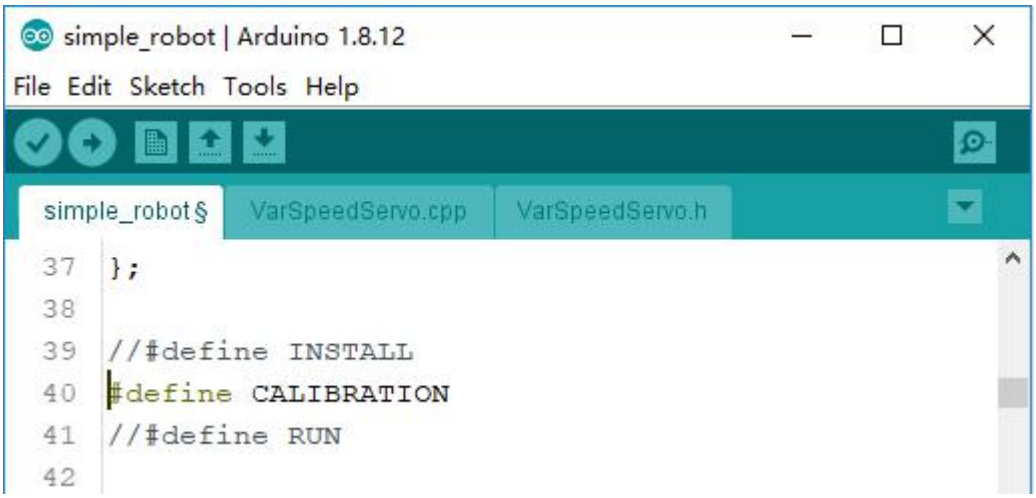


Servo CALIBRATION Test

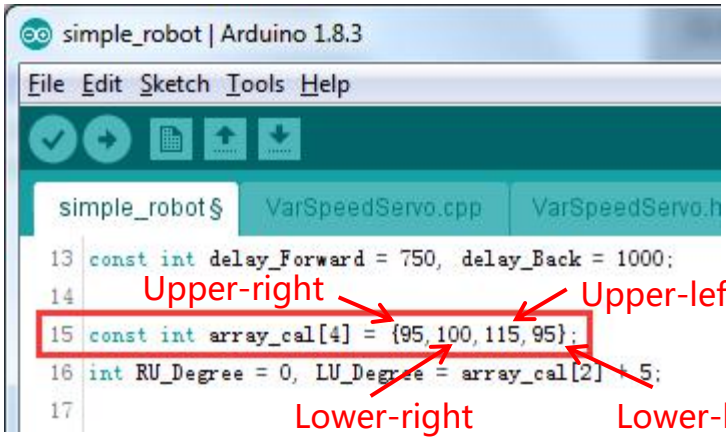
Check the assembly of the 4 servos according to the picture as shown.



Open the program **simple_robot.ino** and go to **Line 39**. Set #define CALIBRATION as able and disable the other two. Then select the correct board and port, and upload the sketch.



If the robot is not fully upright, the angle can be manually calibrated. Go to **Line 15** to rectify it.



The basic principle of calibration: increased values can rotate the servo shaft clockwise and vice versa. For example, if the right leg is toe out, you need to decrease the upper-right servo's angle; if it is toe in, you need to increase the angle.

Tips for calibration:

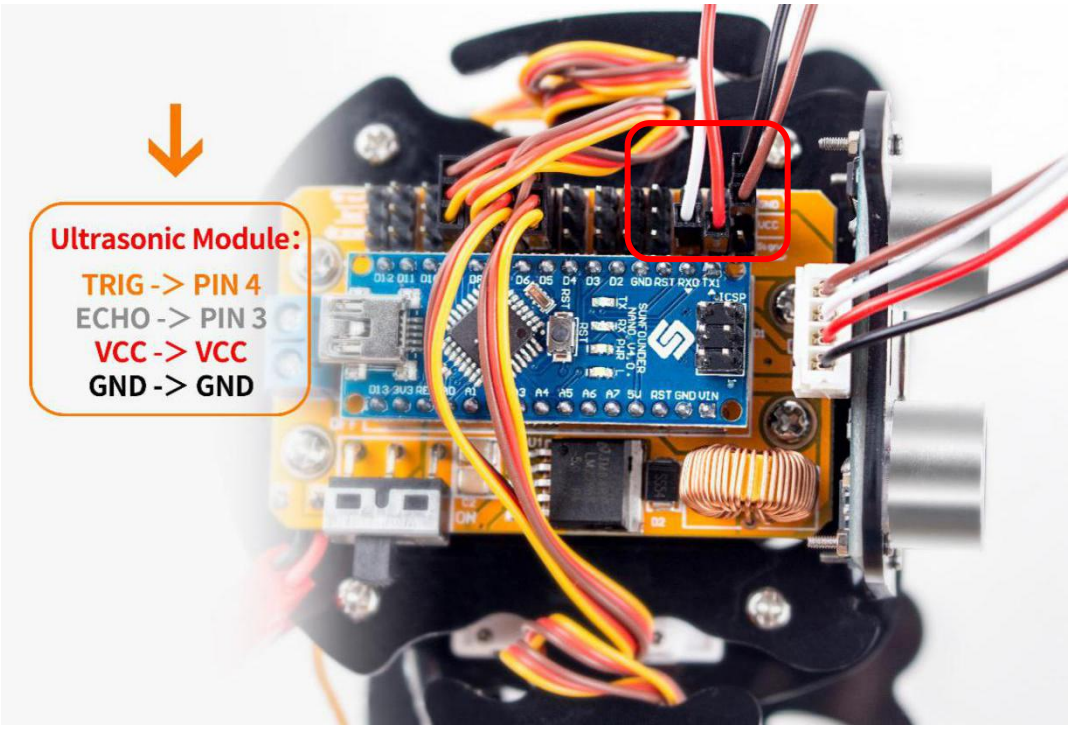
① The calibration method for the left leg works the opposite way for right leg.

② If the right foot's sole faces outward, you need to decrease the lower-right servo's angle; if its sole faces inward, you need to increase the angle.

③ The calibration method for the left foot works the opposite way for right foot.

Ultrasonic Connecting

Connect **pin TRIG** of the ultrasonic to **pin 4** of the board, **ECHO** to **pin 3**, **VCC** to **VCC** and **GND** to **GND**.



Wire Arrangement

Twine the servo wire and 4-Pin anti-reverse cable on the No. 1 board.

So far the robot has been assembled successfully, it's easy if you follow our steps closely. Hope you enjoy the fun of the bot, thanks for watching.

Example

Here, we provide you with two sample programs to play Sloth:

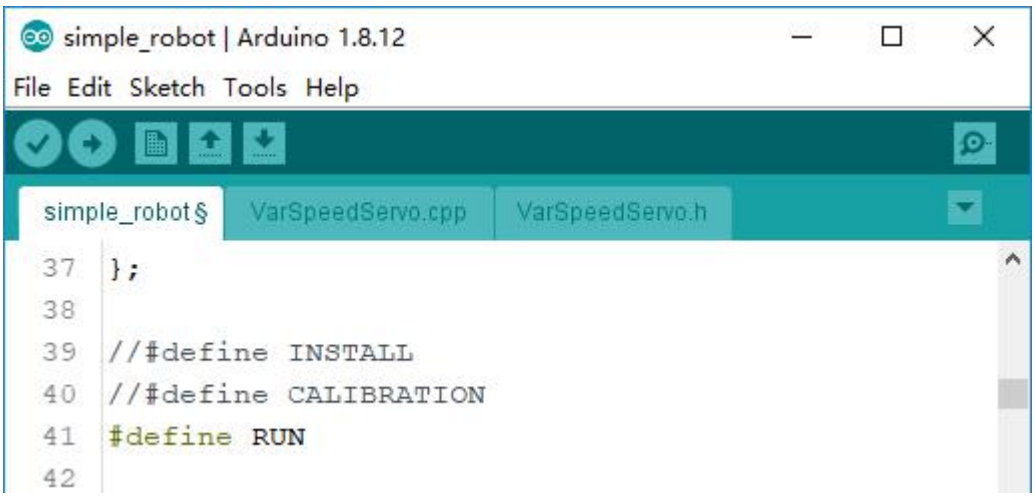
Simple Robot

In this code we write mobile obstacle avoidance for the robot. After the program is burned, sloth will go straight ahead. If it senses an obstacle ahead, it will step back and turn to find a new direction.

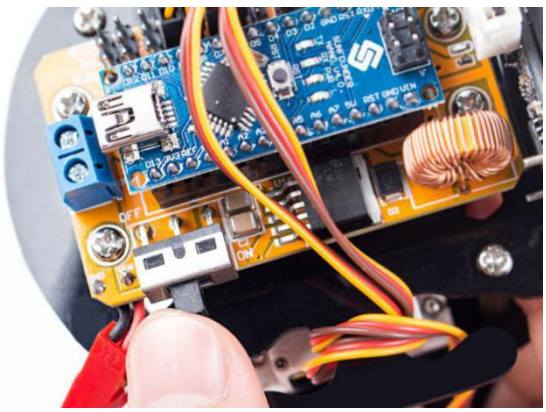
Open the program **simple_robot.ino** under the path of **DIY_4-DOF_Robot_Kit_-_Sloth\Code\simple_robot.**

(This is also the program what we use to install and calibrate the servo.)

Go to **Line 39** again, set #define RUN as able and disable the other two, then upload the code to the SunFounder Nano board.



After burning successfully, unplug the USB cable and slide the power switch to ON.



You will see the robot moving forward. When encountering an obstacle, it will make a turn and then go forward again.

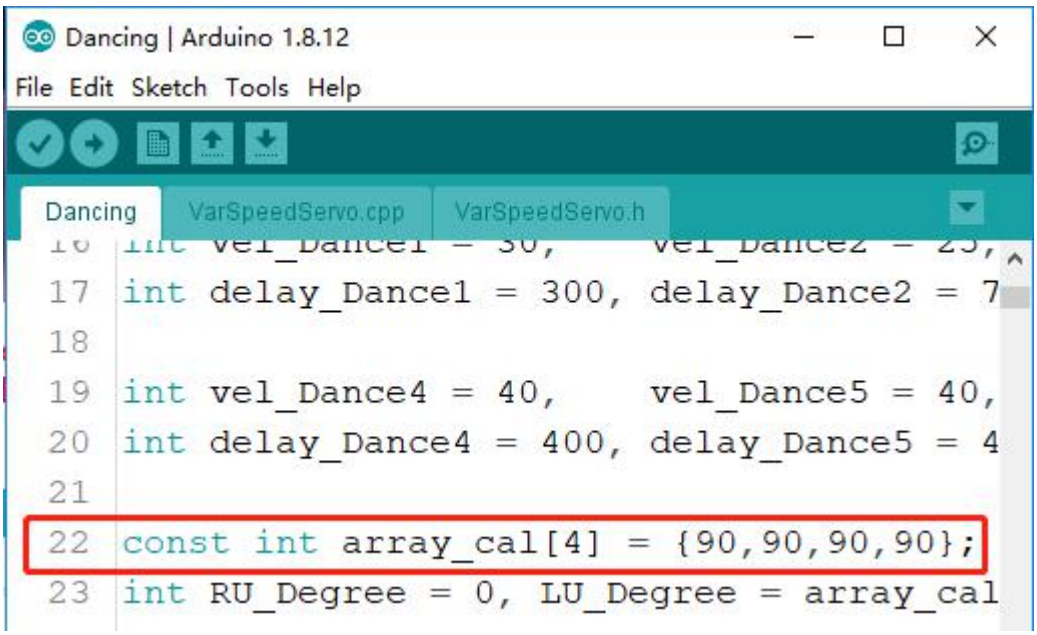
Dancing

In this code, we write the basic actions of sloth and compose them into a dance.

Open the program **Dancing.ino** under the path of **DIY_4-DOF_Robot_Kit_-_Sloth\Code\Dancing.** Go to **Line 196**, select the RUN function by rectifying #define.

After burning successfully, unplug the USB cable and press the power button on the servo control board. You will see the robot dancing.

Note: The program also needs to be calibrated in the same way as Servo CALIBRATION Test in Assembly. If there has been a precise calibration, you can modify the parameters in line 22 directly.



Q&A

Q1: How can we know the servo is damaged?

A1: In Servo Test step, if the servo rocker arm shake, get stuck or can not rotate smoothly, with an abnormal sound, we can judge it as a damaged one.

Q2: Why the Sloth reboots in running?

A2:

1) If the Sloth is in lower power, rebooting will happen **,**please charge the battery in time.

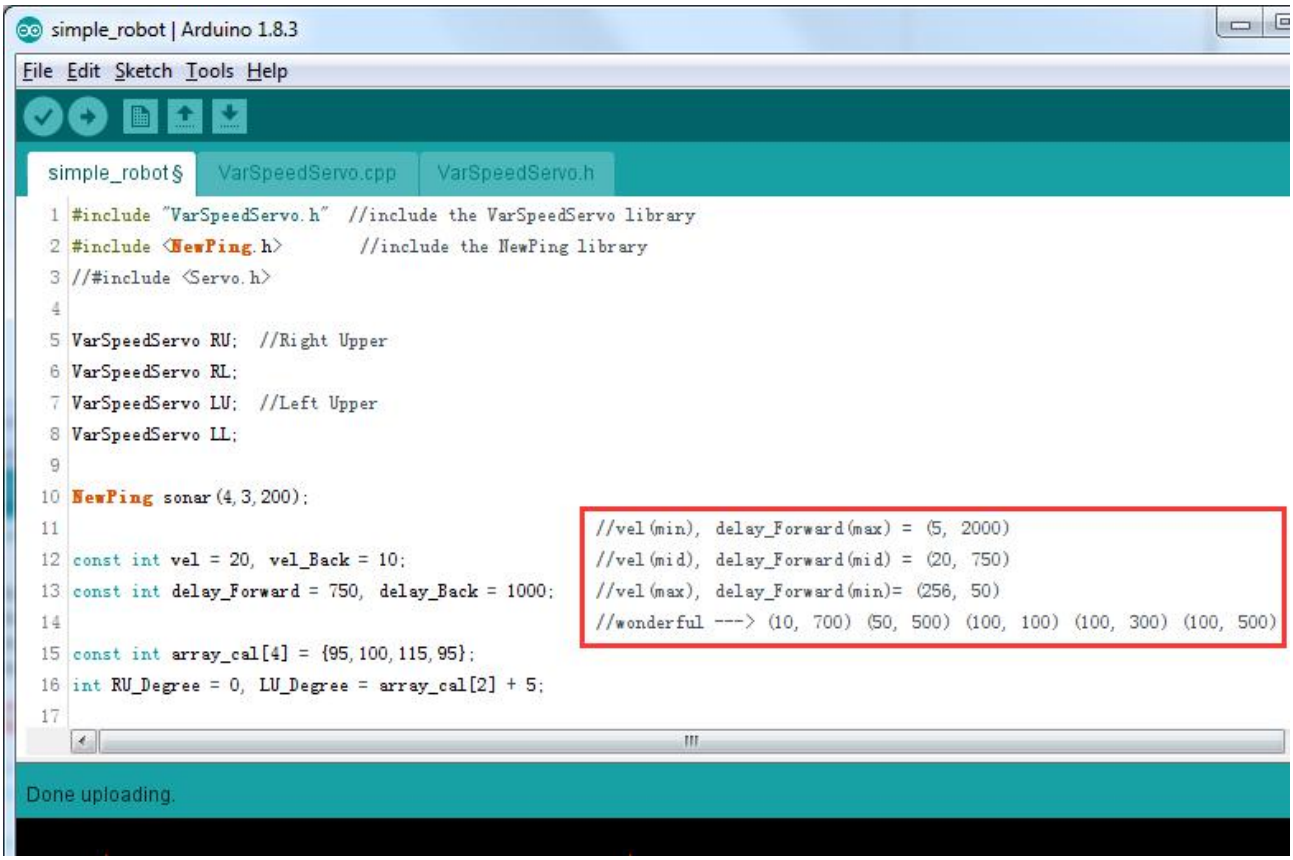
2) It could be the servos are lacking for power. Open the program and go to Line 12, 13. "**vel**" is the servos rotating speed in "initialization or moving forward"; "**vel_Back**" is the servos rotating speed in "moving backward"; "**delay_Forward**", "**delay_Back**" are the delays between two moving forward loops and moving backward loops.

   a) If rebooting happens in moving forward actions, you can decrease the value of "vel" or/ and increase the value of "delay_Forward". For example, decrease "vel" value to 10, and increase "delay_Forward" to 1500.

   b) If rebooting happens in moving backward actions, you can decrease "vel_Back" or/ and increase "delay_Backward". For instance, decrease "vel_Back" to 8, and increase "delay_Backward" to 1500. You can adjust to a proper value as you want. Then click **Upload**.

Q3: Sloth walks too slowly when it moves forward. How to solve this?

A3: Sloth ' s default speed is middle speed, the related sketch is " vel(mid), delay_Forward(mid) = (20, 750)". You can change the speed value as shown below to adjust the walking speed.



change the value of vel and delay_Forward in line12 and 13 to as shown:

vel = 50, delay_Forward = 500

Then click Upload.

Note: If you adjust the robot to a high walking speed, it may fall down and break. Thus it's better to do some protection for the Sloth.

Q4: Sloth walks too slowly when it moves backward. How to solve this?

A4: Considering the structure of Sloth, it's better do adjust a slow speed for backward walking. If you want to adjust the walking speed, refer to Q3 to adjust the value. DO NOT adjust a high speed for walking backward to avoid possible falling down.

Q5: How to make the sloth more stable in walking?

A5: Cut to get two paper cushion for the robot feet, and stick them on the Sloth soles to maintain enough friction for a stable walking.

Q6：What is macro definition (#define)?

A6: The **#define** creates a macro, which is the association of an identifier or parameterized identifier with a token string. After the macro is defined, the compiler can substitute the token string for each occurrence of the identifier in the source file.

You can use the **#ifdef** directives anywhere #if can be used. The #ifdef identifier statement is equivalent to #if 1 when identifier has been defined. It's equivalent to #if 0 when identifier hasn't been defined, or has been undefined by the #undef directive. These directives check only for the presence or absence of identifiers defined with #define, not for identifiers declared in the C or C++ source code.

In sloth code, we use #define and #ifdef to start corresponding functions.

Summary

In this manual, having learned the related components for building the robot kit, you've gone through the assembly of the mechanical parts and electrical modules with the knowledge of Arduino as well as a brief introduction of the key parts like servo, ultrasonic, etc. Also you've got a lot of software and coding, which lays a solid foundation for your futrue journey of exloring open-source field.

The SunFounder DIY 4-DOF Robot Kit is not only a toy, but more a meaningful development kit for Arduino. After all the study and hands-on practice of the kit, you should have a better understanding of Aduino. Now, get started to make better work!

Copyright Notice