Lesson 7 Control the Robot Arm

Table

1. Wiring and assembly inspection 2

1.1Wiring inspection 2

1.2 Structural inspection 3

2. Learn and use src files 3

2.1 Understand the src folder 3

2.2 How to use src files correctly 3

3. Compile and upload code 5

3.1 Open the Code 5

3.2 Select development board type and Port 5

3.4 Compile and upload code 6

4. How to control the robotic arm 7

5. Record more actions 11

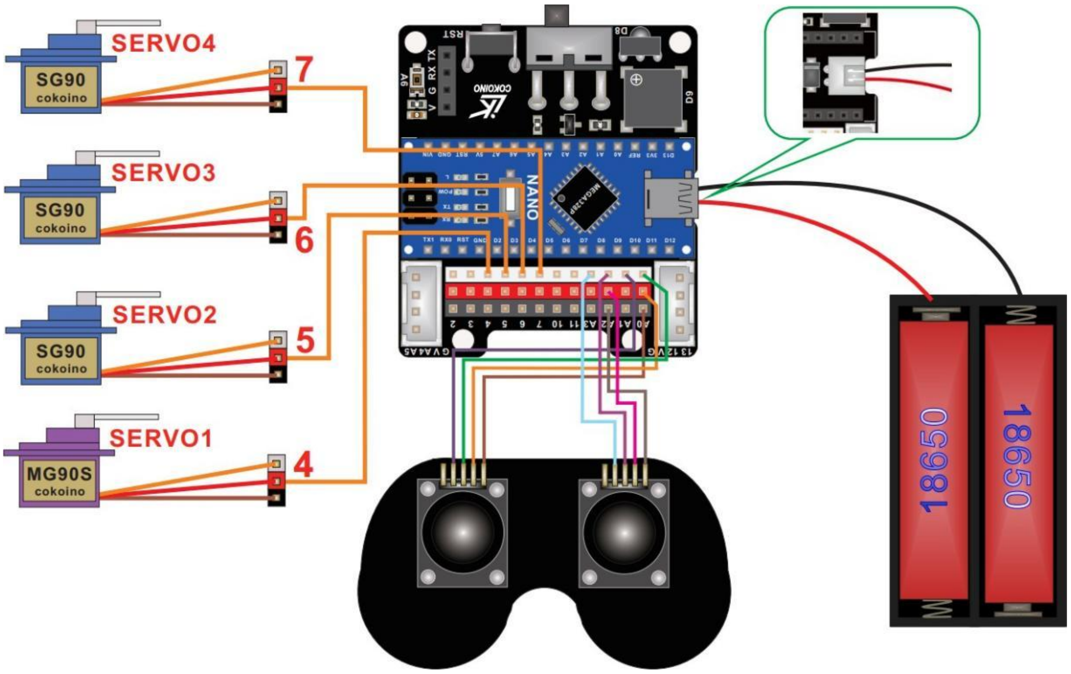
6. Any questions and suggestions are welcome 12

1. Wiring and assembly inspection

After assembling the robotic arm, do not immediately turn on the power or upload the code, but first check whether the wiring of the robotic arm and the assembly of each component are correct.

1.1Wiring inspection

Please refer to the following wiring diagram to check if the 4 servos are connected to the correct positions. Firstly, confirm that the signal pin of the servos is connected to the signal pin on the Nano Shield instead of GND. Secondly, confirm if each servos is connected to the corresponding pin position on the Nano Shield.



Wiring between the Modules and the Nano shield board			
Module		**Nano shield**	
Module Name	**Pin of module**	**Pin of Nano shield**	**Pin's color**
SERVO1	S	4	white
	V	VCC	red
	G	GND	black
SERVO2	S	5	white
	V	VCC	red

	G	GND	black
SERVO3	S	6	white
	V	VCC	red
	G	GND	black
SERVO4	S	7	white
	V	VCC	red
	G	GND	black
Left Stick	VRy	A1	white
	VRX	A0	white
	+5V	VCC	red
	GND	GND	black
Right Stick	VRy	A3	white
	VRX	A2	white
	+5V	VCC	red
	GND	GND	black

1.2 Structural inspection

Manually rotate the various axis parts of the robotic arm without turning on the power, and confirm that each axis rotates smoothly without any jamming or mechanical obstruction. If there are any problems, please carefully check where the assembly is wrong according to the assembly steps.

2. Learn and use src files

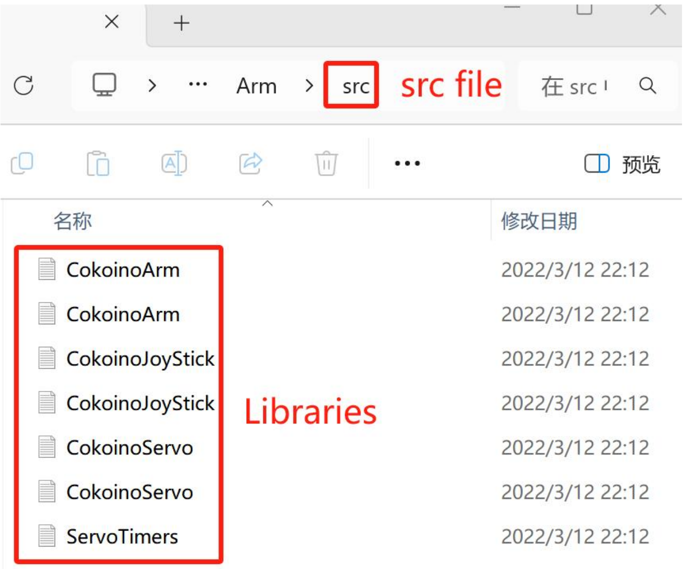
2.1 Understand the src folder

The src folder of Arduino is mainly used to store the source code files of the library. In the development process of Arduino library, the src folder is an essential part, which contains the main C++source code files (. cpp files) of the library. These source code files define the classes and functions of the library and are the core part of implementing the library's functionality.

By organizing the code in the src folder, modularization of the code can be achieved, making it clearer and easier to manage. In addition, the code in the src folder can be shared and reused by multiple Arduino projects, improving the reusability and development efficiency of the code.

2.2 How to use src files correctly

Our robotic arm uses four servos, and using Arduino's existing Servo library to program and control the actions of the four servos would result in complex and cumbersome code. To simplify the code, we wrote the CokoinoServo library and CokoinoArm library ourselves. Put them all in the src folder, as shown in the following figure:



名称	修改日期
CokoinoArm	2022/3/12 22:12
CokoinoArm	2022/3/12 22:12
CokoinoJoyStick	2022/3/12 22:12
CokoinoJoyStick	2022/3/12 22:12
CokoinoServo	2022/3/12 22:12
CokoinoServo	2022/3/12 22:12
ServoTimers	2022/3/12 22:12

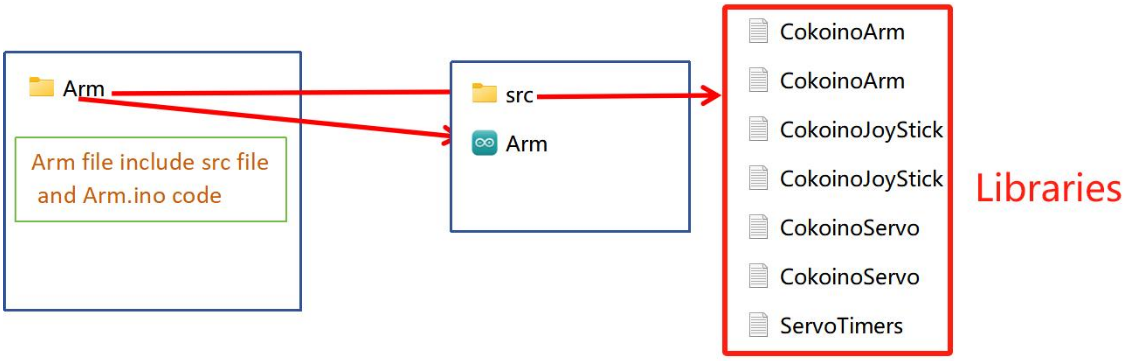
How to use the library written by oneself?

The first method is to install the library into the Arduino IDE and then use #include <xxx.h> in the code to reference it.

The second method is to directly use the functions in the library in the code, and then place the src folder containing the library in the folder where the code is located. This way, opening the code can directly call the library in the src folder.

The code required for this Lesson is Arm.ino

This code uses functions from the CokoinoServo and CokoinoArm libraries that we have written ourselves. When using them, we need to place the src folder containing the CokoinoServo and CokoinoArm libraries in the same folder as Arm'ino, as shown in the following figure:



3. Compile and upload code

3.1 Open the Code

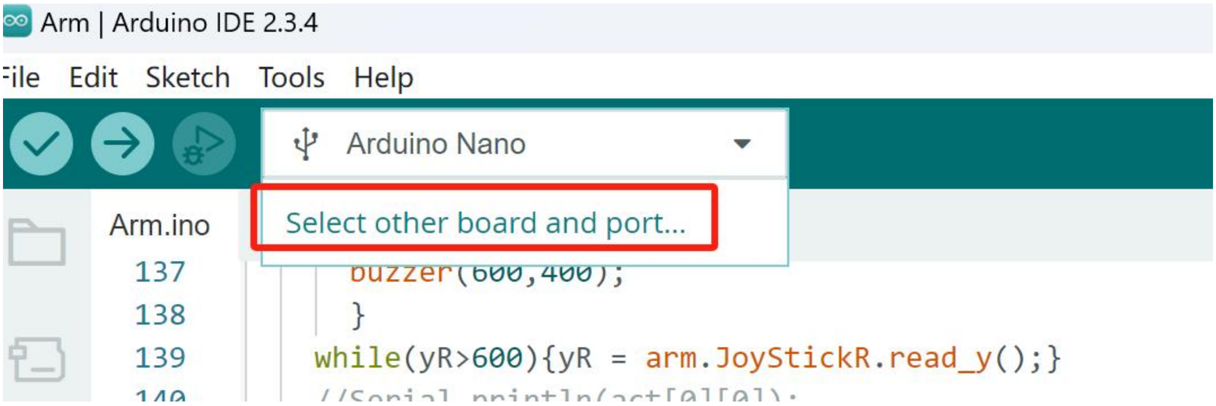
Find the "Arm" code from the following path, open it with the Arduino IDE and select the board type and com port of the IDE.

Note: The src folder cannot be moved, it must be placed in the Arm folder with the "Arm" code, otherwise, your code will not be uploaded.

3.1.1 Double-click the Arduino IDE shortcut on the desktop to open it

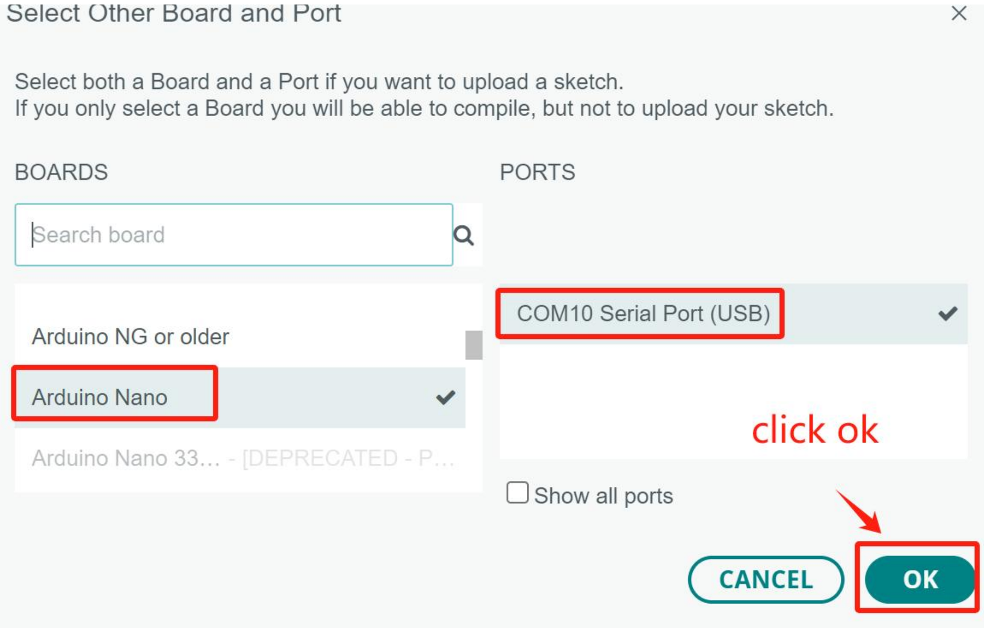
3.2 Select development board type and Port

3.2.1 Click the “Select Board”,then click “Select other board and port...”


Arm | Arduino IDE 2.3.4

```
137     buzzer(600,400);
138   }
139   while(yR>600){yR = arm.JoyStickR.read_y();}
140   //Serial.println(act[0][0]);
```

3.2.2 Click on the "BOARDS" dropdown menu to select Arduino Nano, and then select the corresponding COM port of Nano on the computer in the "PORTS" menu. This is COM10(COM port is commonly known as an input output port for a device normally PC which enables communication between Arduino and PC. You can check your arduino com number in device manager, the com port of our arduino board is recognized as COM10 in this tutorial)， Then click " OK"。

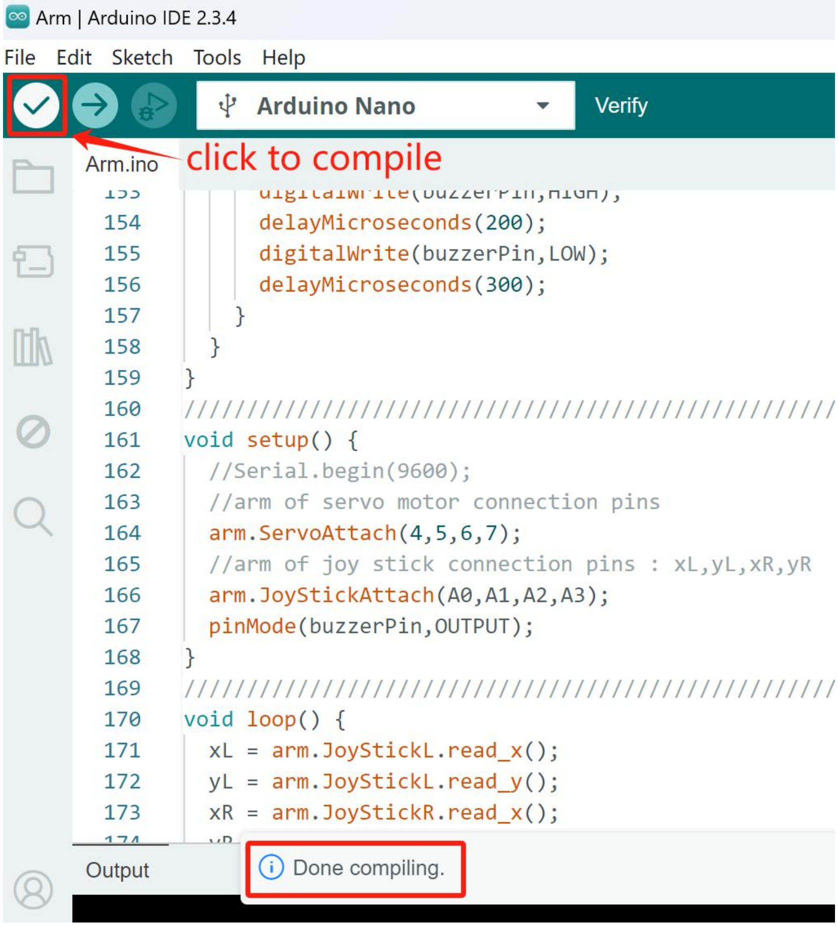


Select Other Board and Port

Select both a Board and a Port if you want to upload a sketch.
If you only select a Board you will be able to compile, but not to upload your sketch.

3.4 Compile and upload code

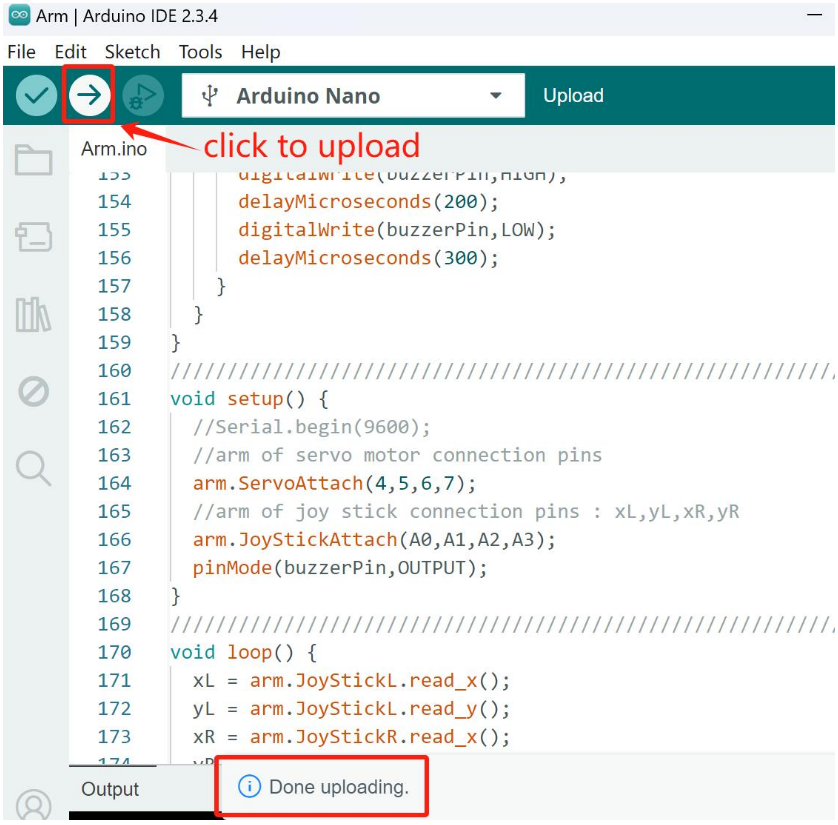
3.4.1 Click compile button ✓ , successfully compiled the code will display “Done compiling”



```
      digitalWrite(buzzerPin,HIGH);
      delayMicroseconds(200);
      digitalWrite(buzzerPin,LOW);
      delayMicroseconds(300);
    }
  }
}
////////////////////////////////////////////////////
void setup() {
  //Serial.begin(9600);
  //arm of servo motor connection pins
  arm.ServoAttach(4,5,6,7);
  //arm of joy stick connection pins : xL,yL,xR,yR
  arm.JoyStickAttach(A0,A1,A2,A3);
  pinMode(buzzerPin,OUTPUT);
}
////////////////////////////////////////////////////
void loop() {
  xL = arm.JoyStickL.read_x();
  yL = arm.JoyStickL.read_y();
  xR = arm.JoyStickR.read_x();
```

Done compiling.

3.4.2 Click upload button , successfully uploading the code will display “Done uploading”.

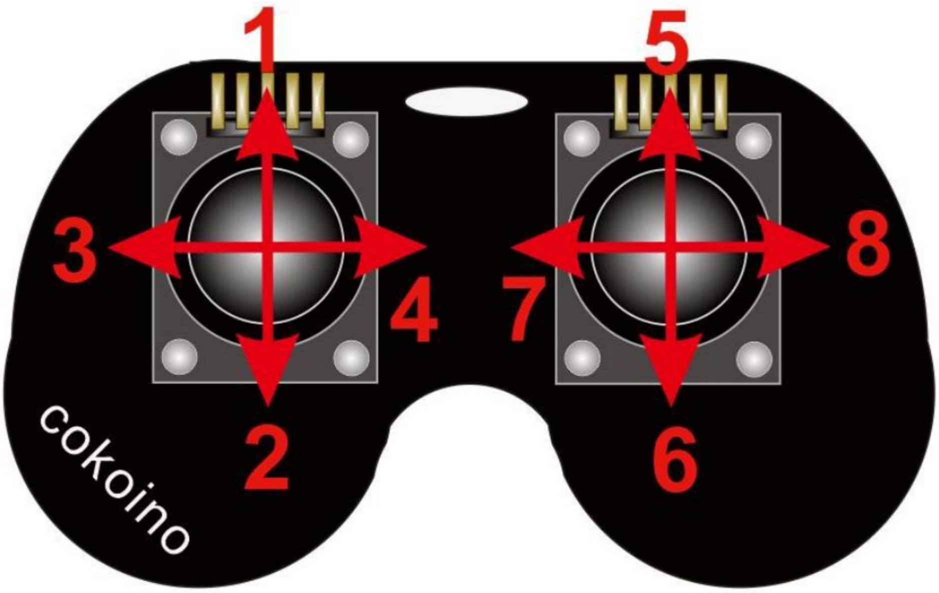


```
      digitalWrite(buzzerPin,HIGH);
      delayMicroseconds(200);
      digitalWrite(buzzerPin,LOW);
      delayMicroseconds(300);
    }
  }
}
//////////////////////////////////////////////////////////
void setup() {
  //Serial.begin(9600);
  //arm of servo motor connection pins
  arm.ServoAttach(4,5,6,7);
  //arm of joy stick connection pins : xL,yL,xR,yR
  arm.JoyStickAttach(A0,A1,A2,A3);
  pinMode(buzzerPin,OUTPUT);
}
//////////////////////////////////////////////////////////
void loop() {
  xL = arm.JoyStickL.read_x();
  yL = arm.JoyStickL.read_y();
  xR = arm.JoyStickR.read_x();
```

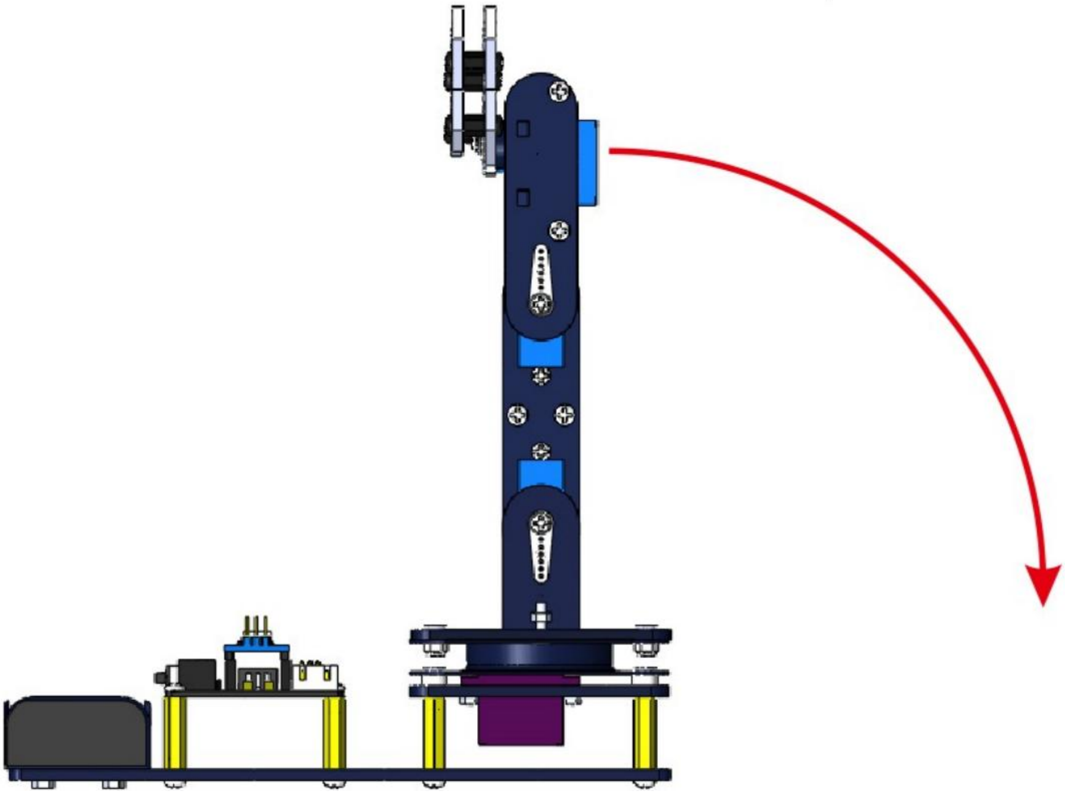
ⓘ Done uploading.

4. How to control the robotic arm

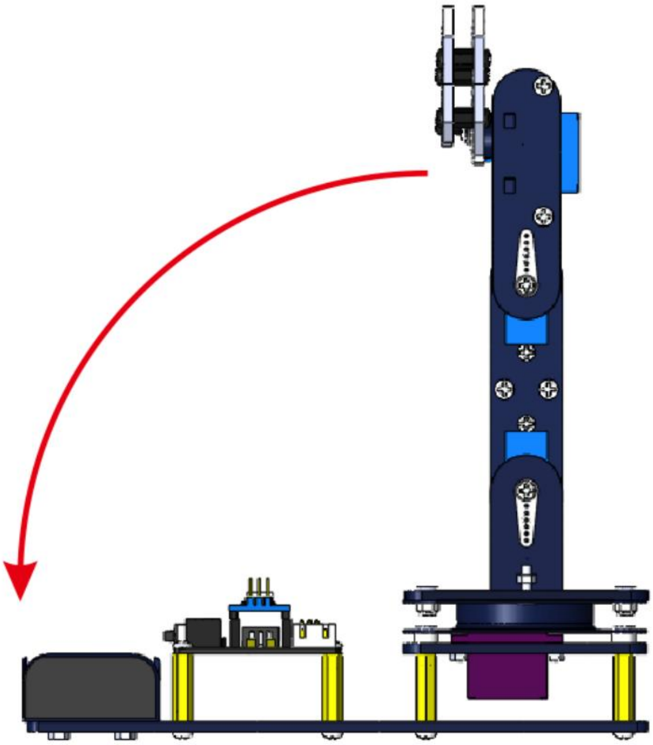
The following explains how the joystick controller controls the robot work by marking the direction of the joystick controller with numbers:



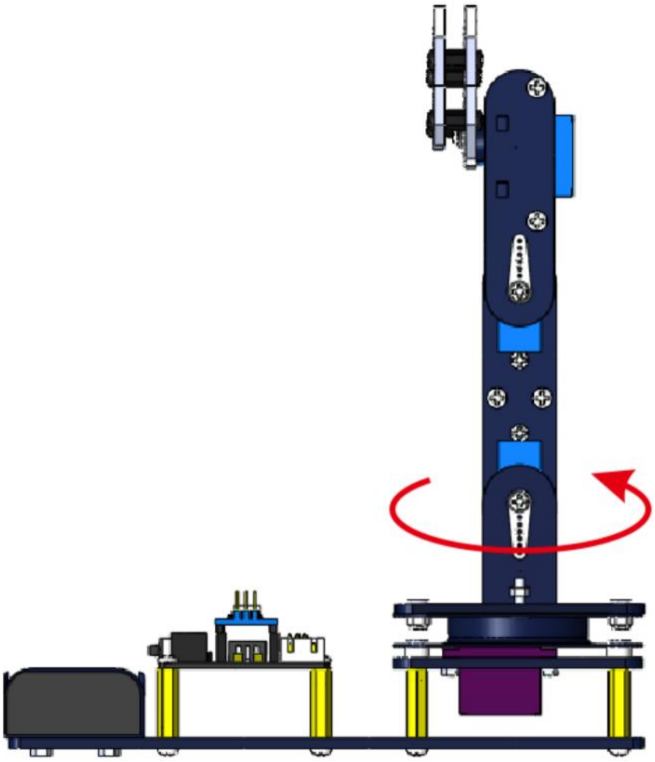
Number 1：The robotic arm stretches forward



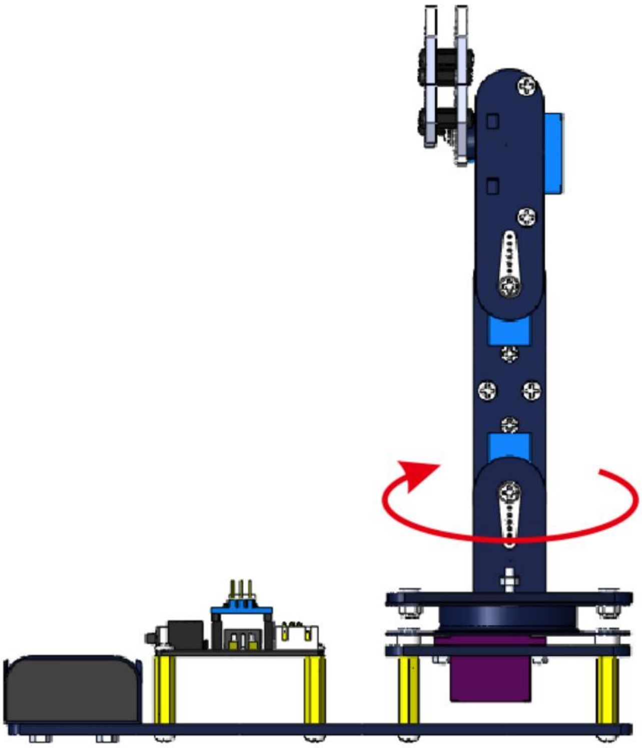
Number 2：Robot arm stretches back



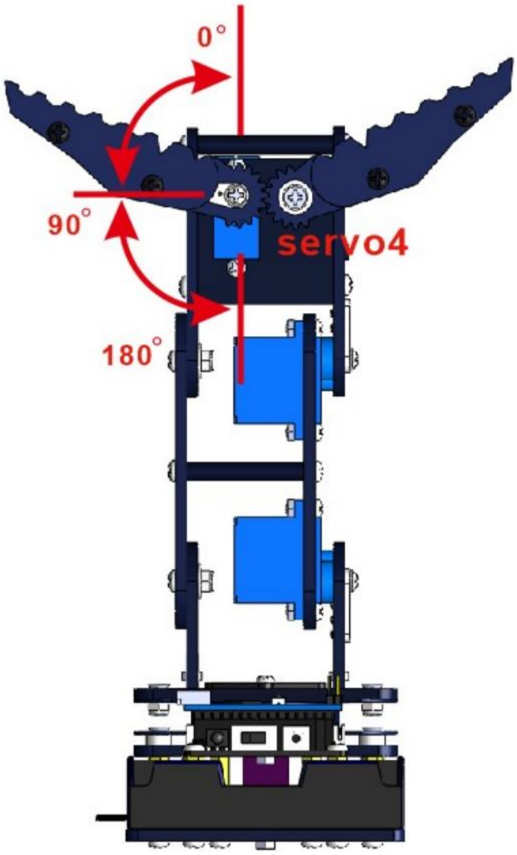
Number 3： The base of the robotic arm rotates to the left



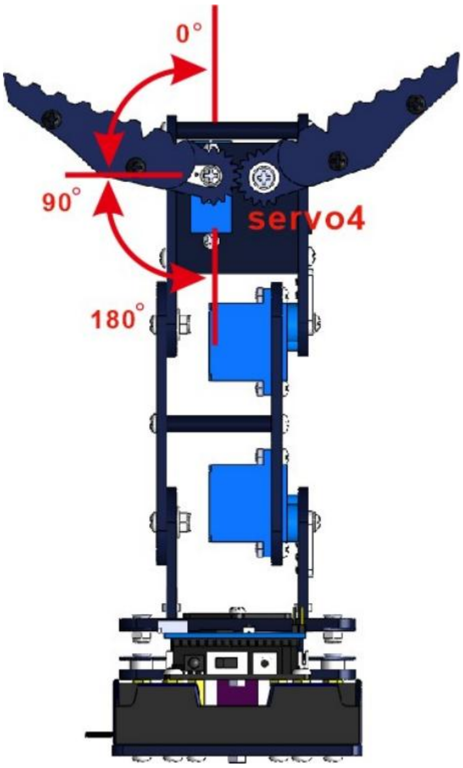
Number 4： The base of the robotic arm rotates to the right



Number 5：Claw of robotic arm close up



Number 6：The claw of the robot arm opens



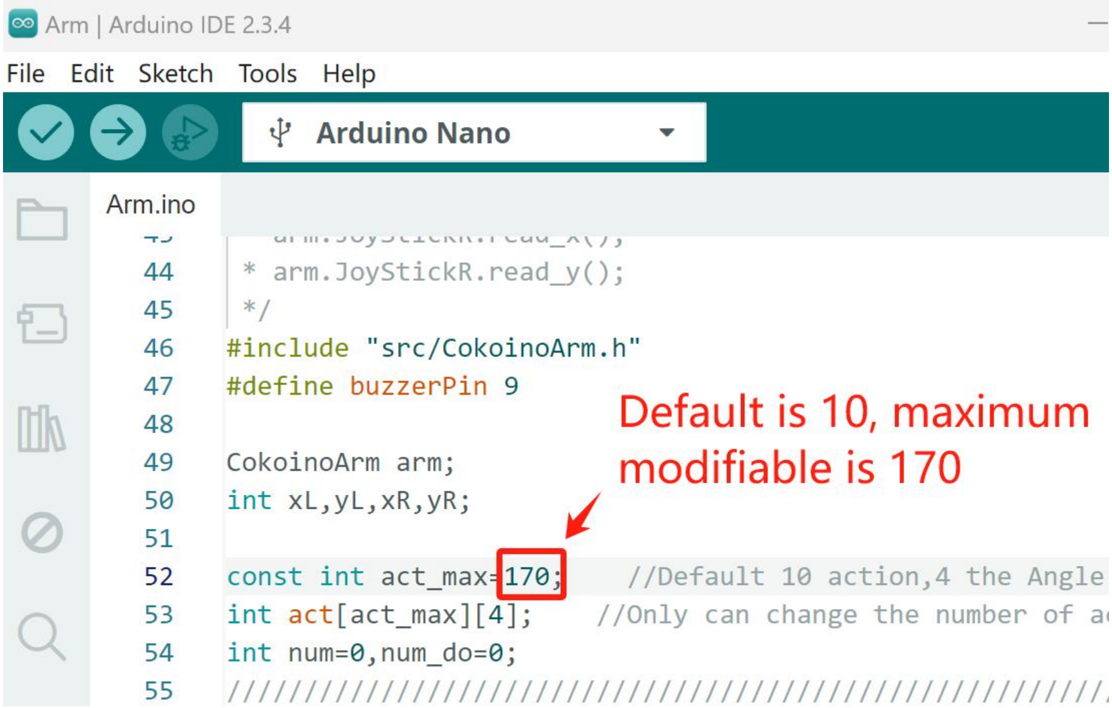
Number 7： Push the right joystick to the left direction once(number 7 direction), and an action will be recorded. In the program we provide, the robotic arm can record up to 10 actions continuously. When the robot records 10 actions, the buzzer will sound once, and then it will start to record 10 actions repeatedly. After 10 actions are completed, the buzzer will sound again once.

Number 8： Perform recorded actions

Push the right joystick to the right direction (number 8 direction), the buzzer will sound and the robot will perform the recorded action, after the action is completed, the buzzer will sound again.

5. Record more actions

By modifying the numbers in the base code, you can record up to 170 actions, as shown in the image below:



```
 * arm.JoyStickR.read_y();
 */
#include "src/CokoinoArm.h"
#define buzzerPin 9

CokoinoArm arm;
int xL,yL,xR,yR;

const int act_max=170;    //Default 10 action,4 the Angle
int act[act_max][4];    //Only can change the number of a
int num=0,num_do=0;
///////////////////////////////////////////////////////
```

Default is 10, maximum modifiable is 170

6. Any questions and suggestions are welcome

Thank you for reading this document!

If you find any errors and omissions in the tutorial, or if you have any suggestions and questions, please feel free to contact us:

cokoino@outlook.com

We will do our best to make changes and publish revisions as soon as possible.

If you want to learn more about Arduino, Raspberry Pi, Smart Cars, Robotics and other interesting products in science and technology, please continue to visit our Amazon Store by search for " **LK COKOINO**" on Amazon. We will continue to launch fun, cost-effective, innovative and exciting products.

LK COKOINO