

## 3.2 Arduino Control Routine

The document demonstrates the control of MaxArm's bus servo angles, setting of XYZ coordinates, control of nozzle functionality, and reading of bus servo angles and XYZ coordinates using an Arduino controller.

### 1. Working Principle

The source code of the program is located in “**10.Serial Communication/ Control Routine/ Arduino Control Routine/ Arduino Routine/ arduino\_and\_MaxArm/ arduino\_and\_MaxArm**”

The specific underlying implementation can be view in the “MaxArm\_ctl.cpp” program file stored in “**10.Serial Communication/ Control Routine/ Arduino Routine Control/ Arduino Routine /arduino\_and\_MaxArm/ arduino\_and\_MaxArm**”.

1) Set the debugging print serial port baud rate to 115200, call the **ma\_ctl.serial\_begin()** function from the **MaxArm\_ctl.h** library to initialize the software serial port communication at a baud rate of 9600.

```
void setup() {  
    Serial.begin(115200);  
    ma_ctl.serial_begin();  
}
```

2) By calling the **ma\_ctl.set\_angles()** function, you can control the angles of the bus servos. The example of “**ma\_ctl.set\_angles(angles , 1000)**” is involved in here.

The first parameter “**angles**” represents the rotation angles. This parameter is a one-dimensional array of length 3, corresponding to the rotation angle for the 3 bus servos. The rotation range is 0°-240°.

The second parameter "**1000**" represents the duration of operation (in milliseconds), with a duration of 1000 milliseconds in this case.

```
//设置总线舵机角度
Serial.println("set angles");
uint8_t angles[3] = {120,100,120};
//将总线舵机的角度分别设置为120/100/120度,运行时间为1000ms
ma_ctl.set_angles(angles , 1000);
delay(1500);
```

3) By calling the **ma\_ctl.set\_xyz()** function, the XYZ coordinates can be set. The example of "ma\_ctl.set\_xyz(xyz[0] , 1000)" is involved in here.

The first parameter "**xyz[0]**" represents the XYZ coordinates to be set, and it is a one-dimensional array of length 3.

The second parameter "**1000**" represents the duration of operation (in milliseconds), with a duration of 1000 milliseconds in this case.

```
//设置xyz坐标
Serial.println("set xyz");
int16_t xyz[2][3] = {{120,-180,85},{-120,-180,85}};
//将xyz坐标设置为120/-180/85,运行时间为1000ms
ma_ctl.set_xyz(xyz[0] , 1000);
delay(1500);
//将xyz坐标设置为-120/-180/85,运行时间为1000ms
ma_ctl.set_xyz(xyz[1] , 1000);
delay(1500);
```

4) The example of "**ma\_ctl.set\_pwmervo(90,1000)**" is involved in here.

The first parameter "**90**" represents the angle of rotation for the PWM servo.

The second parameter "**1000**" denotes the duration of operation (in milliseconds), set here to 1000 milliseconds.

```
//设置PWM舵机角度
Serial.println("set pwm servo");
//将PWM舵机角度设置为90度,运行时间为1000ms
ma_ctl.set_pwm servo(90,1000);
delay(1500);
//将PWM舵机角度设置为45度,运行时间为1000ms
ma_ctl.set_pwm servo(45,1000);
delay(1500);
//将PWM舵机角度设置为135度,运行时间为1000ms
ma_ctl.set_pwm servo(135,1000);
delay(1500);
```

5) Utilizing the **ma\_ctl.set\_SuctionNozzle()** function enables the control of the air pump and solenoid valve. Consider the code "ma\_ctl.set\_SuctionNozzle(1)" as an example, where the parameter "1" signifies the current operational mode as opening the pump. Given the brief duration of the venting operation, it is advisable to reduce the delay for the venting operation.

```
//设置吸嘴功能
Serial.println("set suction nozzle")
ma_ctl.set_SuctionNozzle(1); //打开气泵
delay(1500);
ma_ctl.set_SuctionNozzle(2); //打开电磁阀并关闭气泵
delay(200);
ma_ctl.set_SuctionNozzle(3); //关闭电磁阀
delay(1500);
```

6) By invoking the **ma\_ctl.read\_angles()** function, you can retrieve the angle information of the three current bus servos on MaxArm. Store the read servo angles into the "read\_a" array and then print them out via the serial port.

```
//读取并打印出当前状态的总线舵机角度
Serial.println("read angles");
int16_t read_a[3] = {0,0,0};
if(ma_ctl.read_angles(read_a))
{
    Serial.print("angles: ");
    Serial.print(read_a[0]);
    Serial.print(" ");
    Serial.print(read_a[1]);
    Serial.print(" ");
    Serial.print(read_a[2]);
    Serial.println(" ");
}
delay(100);
```

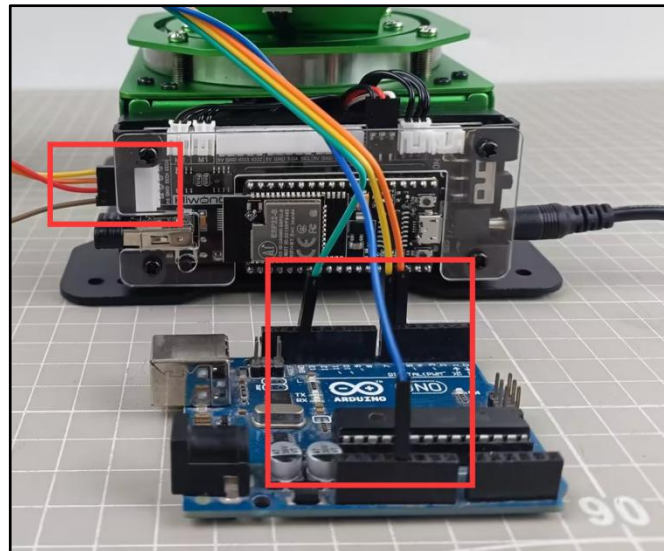
7) Utilizing the **ma\_ctl.read\_xyz()** function allows you to obtain the current XYZ coordinates of MaxArm. Store the retrieved XYZ coordinates into the **"read\_x"** array and then print them out via the serial port.

```
//读取并打印出当前状态的xyz坐标
Serial.println("read xyz");
int16_t read_x[3] = {0,0,0};
if(ma_ctl.read_xyz(read_x))
{
    Serial.print("xyz: ");
    Serial.print(read_x[0]);
    Serial.print(" ");
    Serial.print(read_x[1]);
    Serial.print(" ");
    Serial.print(read_x[2]);
    Serial.println(" ");
}
delay(100);
```

## 2. Preparation

### 2.1 Hardware

MaxArm has two types of serial communication interfaces: one is the 4-pin interface, which is suitable for devices with pin interfaces for UART communication, commonly used for Arduino development boards, STM32 development boards. The other is the microUSB interface, suitable for devices with USB host interfaces, such as Raspberry Pi, Jetson Nano, etc. Here, we need to connect to MaxArm's 4-pin interface for communication. The communication connection for 4Pin interfaces is as follow:



The specific pin connections are as follow:

MaxArm's IO32 pin serves as the RX signal pin, connected to pin 6 of the Arduino development board.

MaxArm's IO33 pin serves as the TX signal pin, connected to pin 7 of the Arduino development board.

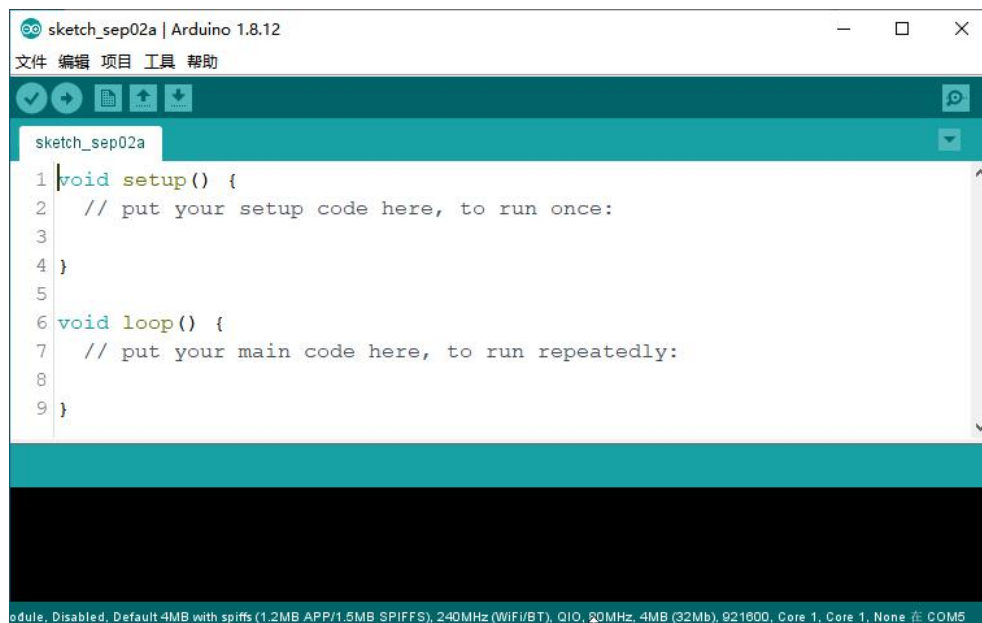
MaxArm's GND pin is connected to the GND pin of the Arduino development board.

## 2.2 Software

Connect MaxArm robotic arm to Arduino editor according to “4. Underlying Program Learning/ Arduino Development/ Set Arduino Development Environment”.

## 3. Program Download

- 1) Double click to open Arduino IDE .

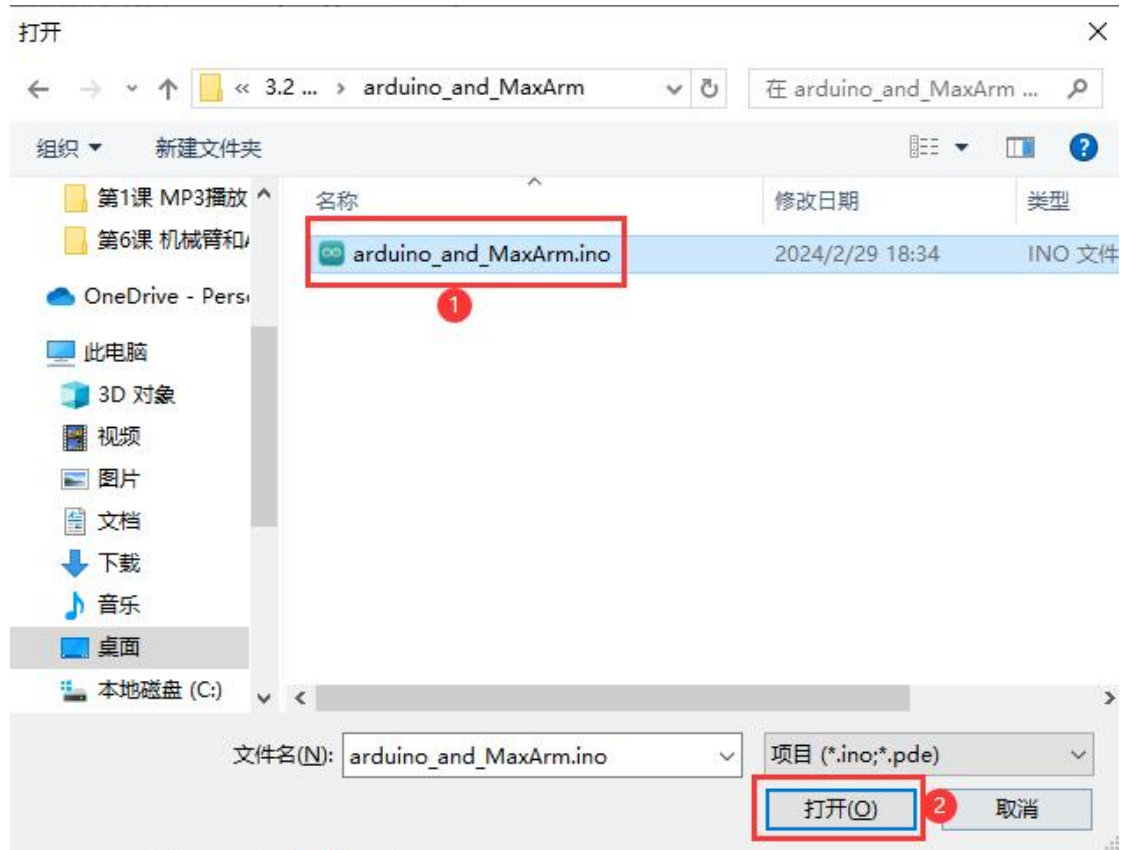


- 2) Click “File-> Open” in sequence.





3) Select the program “**arduino\_and\_MaxArm.ino**” in the folder “Serial Communication/ Control Routine/ Arduino Control Routine/ Arduino Routine/ **arduino\_and\_MaxArm**”, then click “Open”.



4) Select the correct model for development board. Click “**Tool-> Development Board**” to select “**arduino\_and\_MaxArm**”. (If the development board model and port number have already been configured in the development environment, you can skip the step.)



5) Choose the corresponding port number for ESP32 controller in “Tool -> Port”. (In this case, the port number of COM5 is used as example. It may vary on different computers, so it is necessary to select the correct one according to your own computer. Do not choose COM1 if it appears in here, as it is typically the system communication port instead of the actual port of the development board.)






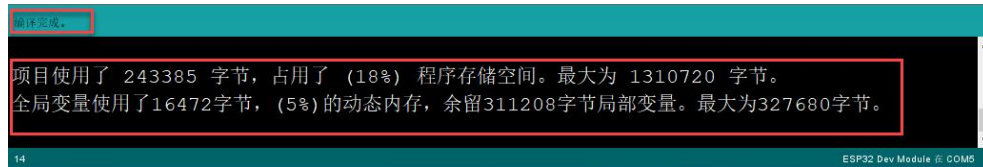
6) If you're unsure about the port number, you can right-click on "**This PC**," then click on "**Properties -> Device Manager**," and check the port number corresponding to the control board (the one with CH340 is our device).




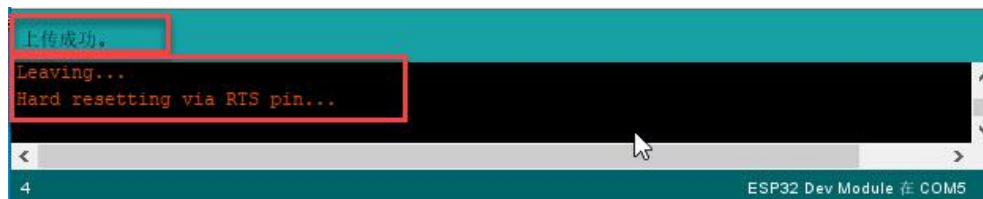
7) After the selection, confirm the development board model of "**ESP32 Dev Module**" and port number of "**COM5**" at bottom right corner.



8) Once confirmed, click on  to verify the program. If the program is error-free, the status area will sequentially display **"Compiling project -> Compilation complete"**. After compilation, it will display information such as the number of bytes used by the current project and program storage space.



9) After compiling without errors, we click on  to upload the program to the development board. The status area will sequentially display **"Compiling project -> Uploading -> Upload successful"**. Once the upload is successful, the status area will stop printing upload information.



## 4. Project Outcome

While the program is running:

- 1) Three bus servo rotate to 120, 100 and 120 degrees, respectively.
- 2) Change MaxArm's coordinates to 120/-180/85, and then change to -120/-180/85 after 1.5 seconds.
- 3) The PWM servo smoothly transitions its angle from 90 to 45 degrees, and then to 135 degrees.
- 4) After a delay of 1.5 seconds, the air pump is activated to start suctioning the object, and after 1.5 seconds, the object is released.

- 5) Finally, print the current angles of the bus servos and XYZ coordinates in the serial port.