

# Lesson 6 Drawing Square

## 1. Working Principle

Call the related kinematics functions by calling “espmax” kinematics encapsulation library so that the robotic arm can realize the path planning for drawing “square”.

The path to the program of the source code is 7. Inverse Kinematics Basics and Application/Arduino Development/Lesson 6 Drawing Square/square/square.ino

```

1 #include "ESPMax.h"
2 #include "_espmax.h"
3
4 // 逆运动学画正方形例程
5
6 void setup() {
7     ESPMax_init();
8     go_home(2000); // 机械臂回到初始位置
9     Serial.begin(9600);
10    Serial.println("start...");
11 }
12
13 bool start_en = true;
14 void loop() {
15     if(start_en) {
16         float pos[3];
17         // set_position(pos,t), pos[0]: x轴坐标, pos[1]: y轴坐标, pos[2]: z轴坐标, t: 移动的总时间 (时间越长, 速度越慢)
18
19         // 机械臂运行到起始点
20         pos[0] = 50; pos[1] = -260; pos[2] = 80;
21         set_position(pos,2000);
22         delay(3000);
23
24         // 画上横边
25         for(int i=50; i > -50; i -= 5){
26             pos[0] = i; pos[1] = -260; pos[2] = 80;
27             set_position(pos,30);
28             delay(30);
29         }
30         delay(500);
31
32         // 画右竖边
33         for(int i=-260; i < -160; i += 5){
34             pos[0] = -50; pos[1] = i; pos[2] = 80-(26+i/10);
35             set_position(pos,30);
36             delay(30);
37         }
38         delay(500);

```

## 2. Preparation


### 2.1 Hardware

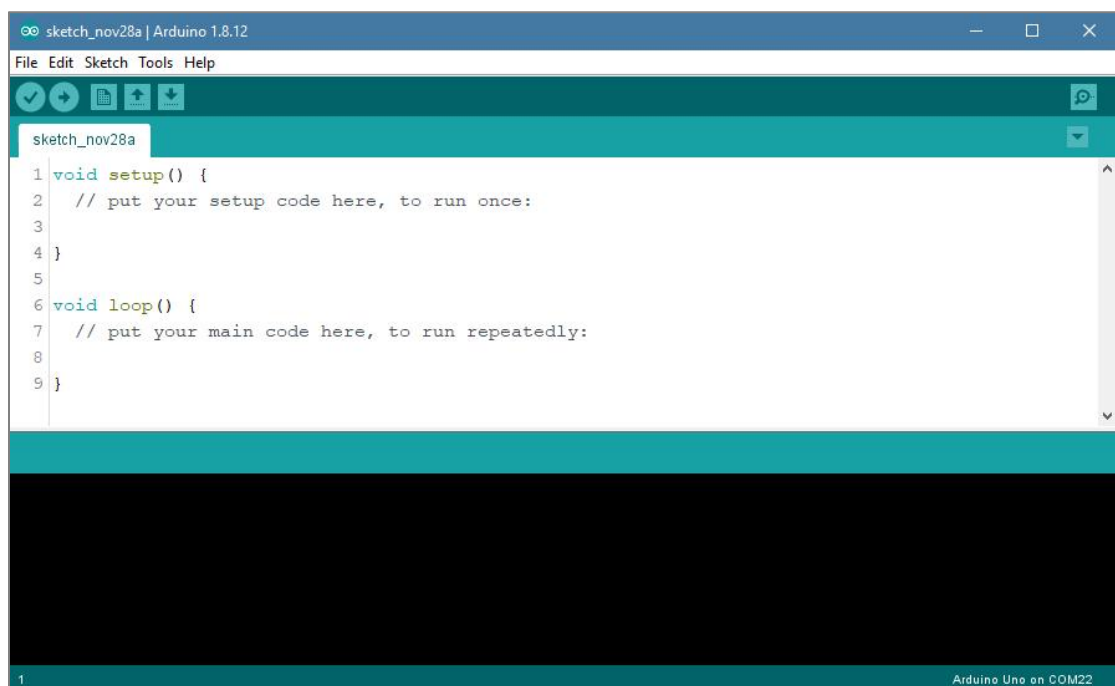
MaxArm robotic arm, power adapter, USB cable.

### 2.2 Software

Please connect MaxArm to the Arduino editor according to the tutorial in folder “4. MaxArm Underlying Program/Python Development/Lesson 1 Set Development Environment”.

## 3. Program Download

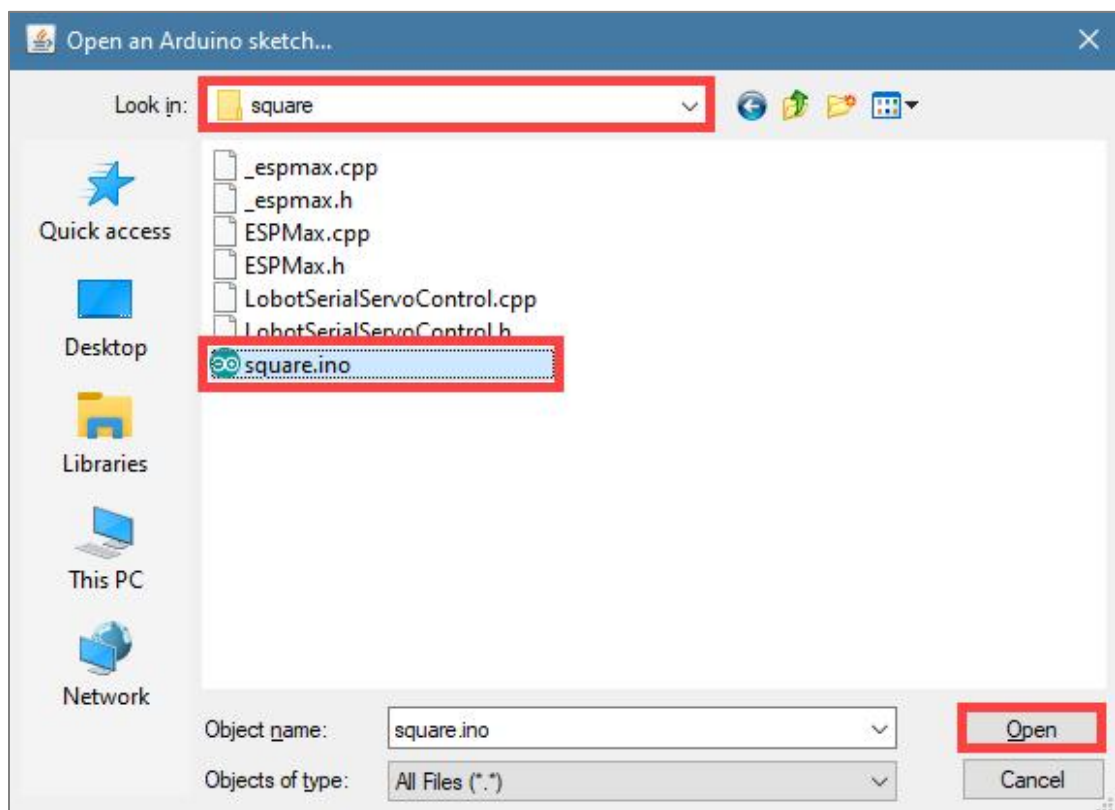
- 1) Click on  icon to open Arduino IDE.



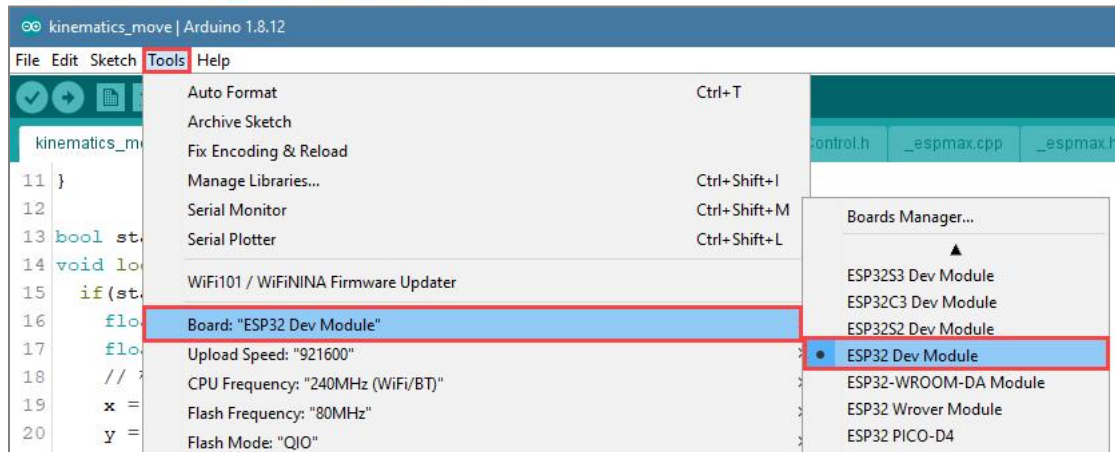
- 2) Click “File->Open” in turn.



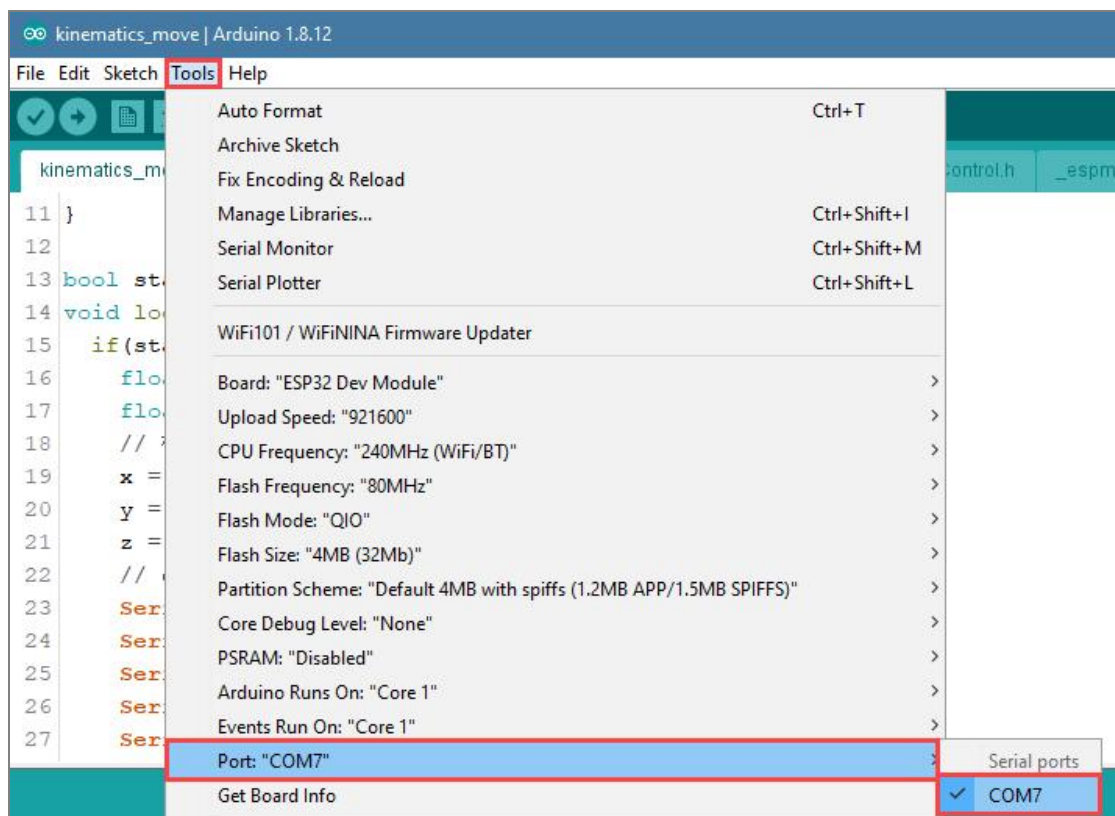
- 3) Select the program “square.ino” in the folder “7. Inverse Kinematics Basics and Application/Arduino Development/Lesson 5 Drawing Square/Program File/square”.



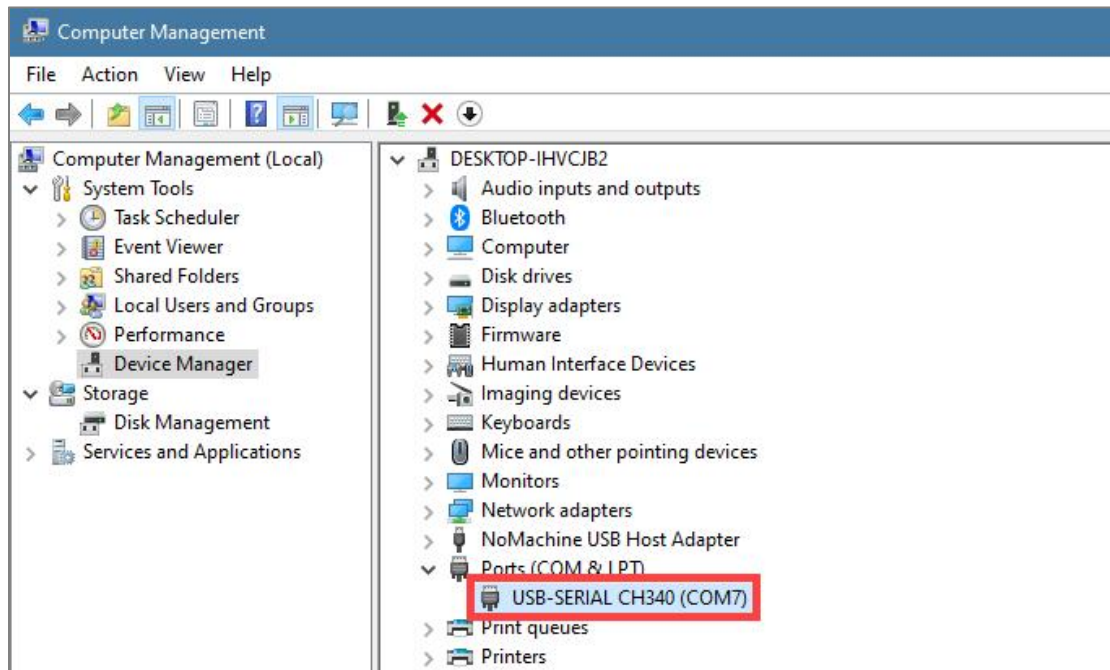
- 4) Select the model of the development board. Click “Tools-> Board” and select “ESP 32 Dev Module” (If the model of the development board has been configured when setting the development environment, you can skip this step).



- 5) Select the corresponding port of Arduino controller in “Tools->Port”. (Here take the port “COM5” as example. Please select the port based on your computer. If COM1 appears, please do not select because it is the system communication port but not the actual port of the development port.)




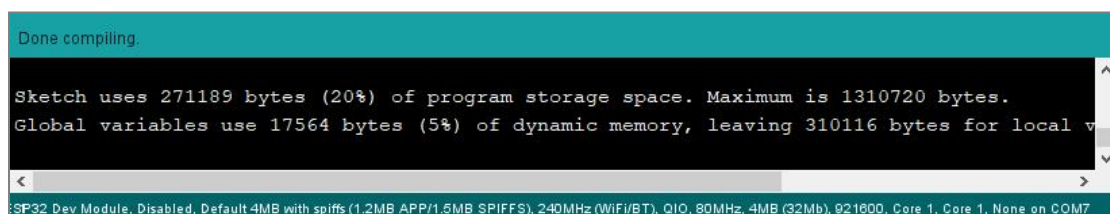
- 6) If you're not sure about the port number, please open the “This PC” and click “Properties->Device Manager” in turns to check the corresponding port number (the device is with CH340). Then select the correct port on Arduino editor.




- 7) After selecting, confirm the board “ESP32 Dev Module” in the lower right corner and the port number “COM5” (it is an example here, please refer to the actual situation).



- 8) Then click on  icon to verify the program. If no error, the status area will display “Compiling->Compile complete” in turn. After compiling, the information such as the current used bytes, and occupied program storage space will be displayed.



- 9) After compiling, click on  icon to upload the program to the development board. The status area will display “Compiling->Uploading->Complete” in turn. After uploading, the status area will stop printing the uploading information.



## 4. Project Outcome

When running program, the robotic arm will be controlled to draw “square”.  
After stopping, exit the program automatically.

## 5. Program Instruction

### 5.1 Import Function File

Before the robotic arm starts to move, the kinematics, bus servo and other related library function need to be called.

```
1 #include "ESPMaX.h"  
2 #include "_espmax.h"
```

### 5.2 Return to the Initial Position

Firstly, use go\_home() function in function library to control the robotic arm to move to the initial position. The specific program is as follow:

```
7   ESPMaX_init();  
8   go_home(2000);  
9   Serial.begin(9600);  
10  Serial.println("start...");  
11 }
```



### 5.3 Control Robotic Arm

Robotic arm starts to move on xyz axes.

```
20 pos[0] = 50; pos[1] = -260; pos[2] = 80;
21 set_position(pos,2000);
22 delay(3000);
23
24
25 for(int i=50; i > -50; i -= 5){
26     pos[0] = i; pos[1] = -260; pos[2] = 80;
27     set_position(pos,30);
28     delay(30);
```

The robotic arm is controlled by `set_position()` function. Take the code

“`set_position(pos,2000)`” as example:

The first parameter “pos” represents the position of the robotic arm on x, y and z axes. Among them, `pos[0]` represents the coordinate of x axis, `pos[1]` represents the coordinate of y axis, and `pos[2]` represents the coordinate of z-axis.

The second parameter “2000” represents the running time and the unit is ms.