

Lesson 2 Touch Detection and Placement

1. Working Principle

This lesson uses touch sensor based on the principle of capacitive sensing. After supplying the touch sensor power, the touching will be sensed when our fingers or metal touch the metal sensing plate. In the meantime, the signal terminal OUT will output low level signal, vice verse. According to this characteristics, the robotic arm can be controlled to carry out the corresponding action.

The path to the source code of the program is 6.Secondary Development /Arduino Development/Sensor Development/Program Files/ Touch Detection and Placement/TouchSensor_put/TouchSensor_put.ino.

```
6 #include "LobotSerialServoControl.h"
7
8
9
10 #define sensor_pin 23
11
12 void setup() {
13
14     Buzzer_init();
15     ESPMax_init();
16     Nozzle_init();
17     PWMServo_init();
18     pinMode(sensor_pin, INPUT_PULLUP);
19     Serial.begin(9600);
20     Serial.println("start...");
21     setBuzzer(100);
22     go_home(2000);
23     SetPWMServo(1,1500,2000); //
```

Firstly, import the corresponding libraries and initialize buzzer, servo and action group.

Next, create the functions of buzzer and touch control. Set the buzzer to make

sound feedback and the robotic arm to perform the corresponding action when short press the touch sensor.

Then, execute the function for controlling action, buzzer, and air pump to suck the detected object to the side and place it.

2. Preparation


2.1 Hardware

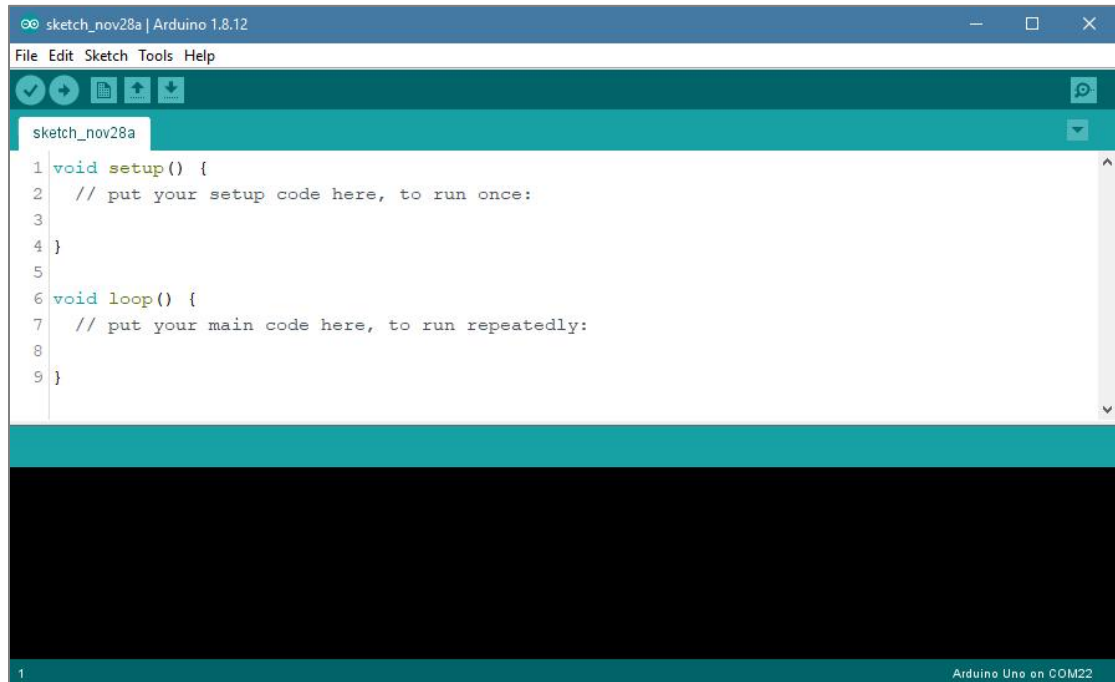
Please assemble the touch sensor to the corresponding position on MaxArm according to the tutorial in folder “Lesson 1 Sensor Assembly” under the same directory.

2.2 Software

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

3. Program Download

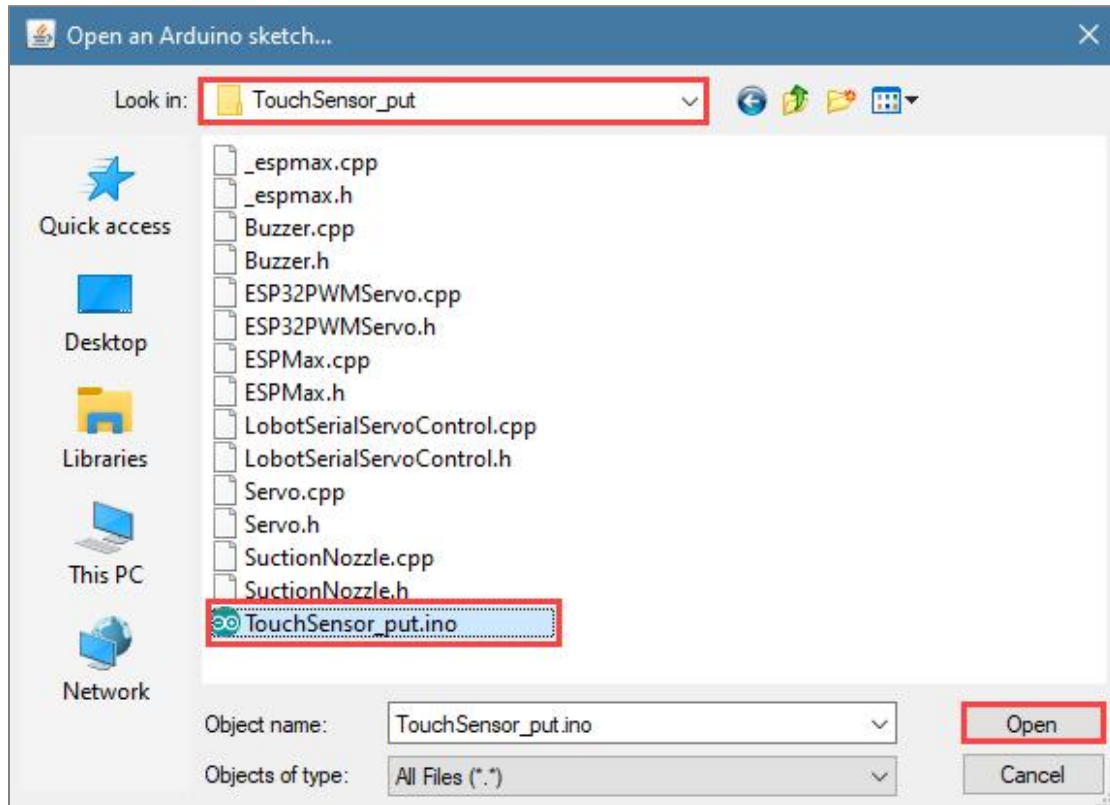
- 1) Click on  icon to open Arduino IDE.



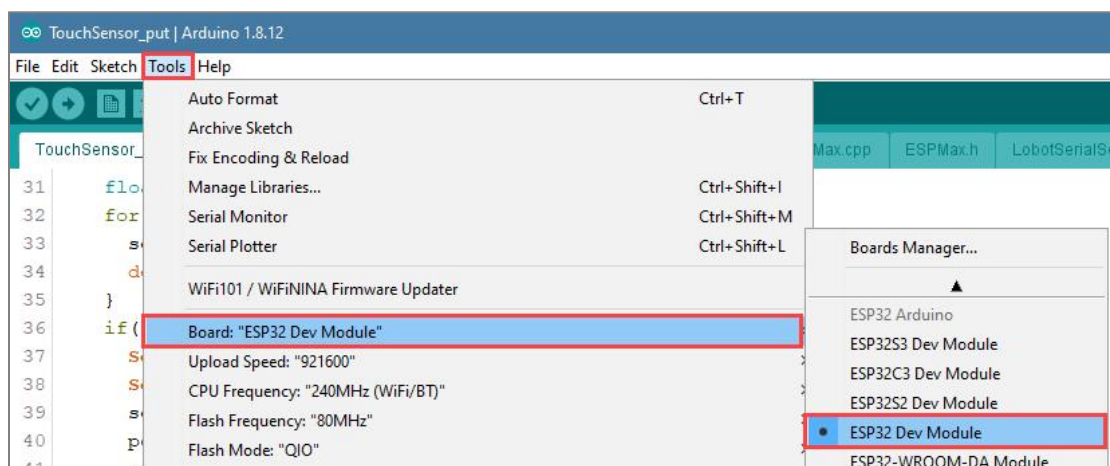
2) Click “File->Open” in turn.



3) Select the program “TouchSensor_put.ino” in the folder “6.Secondary Development / Arduino Development/Sensor-extension Game /Program Files/Button Detection and Placement/TouchSensor_put”.

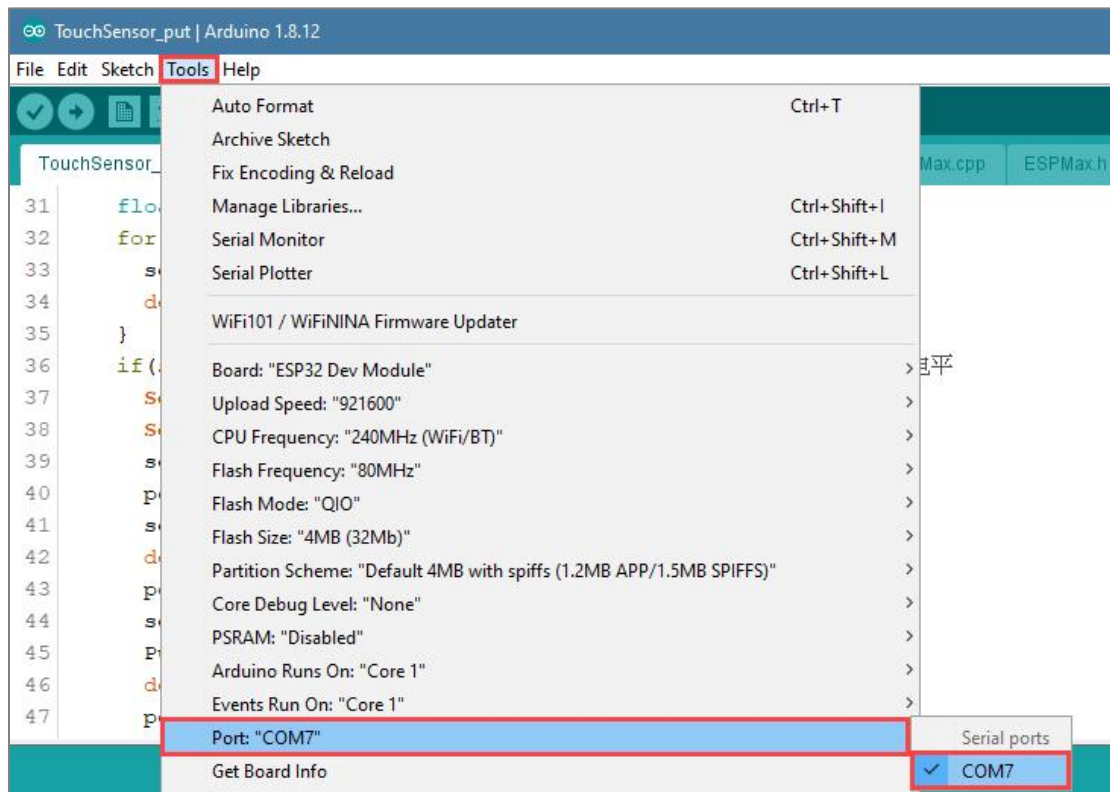


- 4) Select the model of the development board. Click “Tools-> Board” and select “ESP 32 Dev Module” (If the model of the development board has been configured when setting the development environment, you can skip this step).

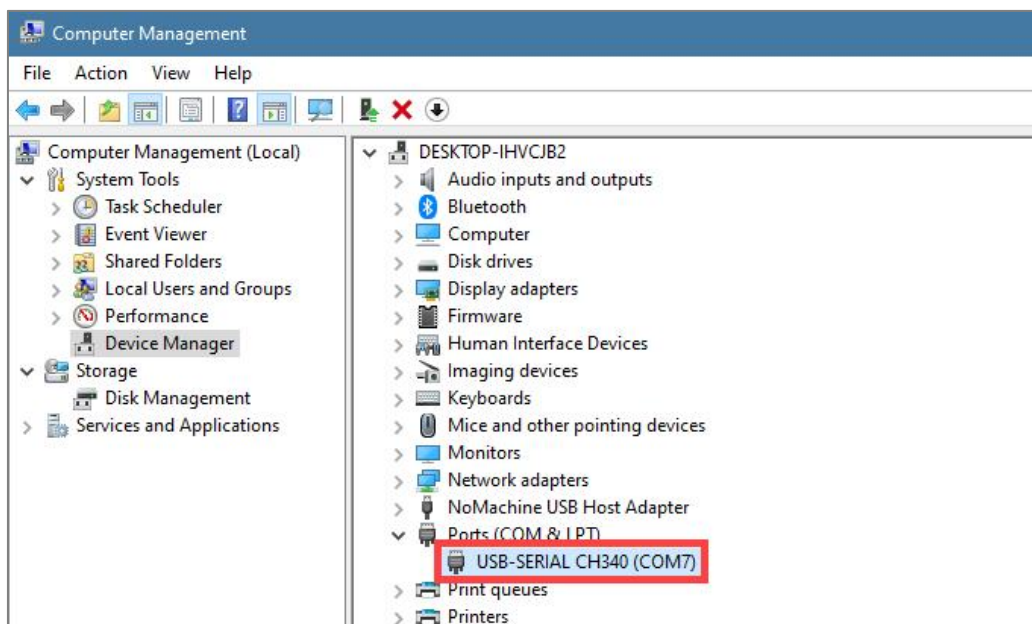


- 5) Select the corresponding port of Arduino controller in “Tools->Port”. (Here take the port “COM5” as example. Please select the port based on your computer. If COM1 appears, please do not select because it is the system

communication port but not the actual port of the development port.)




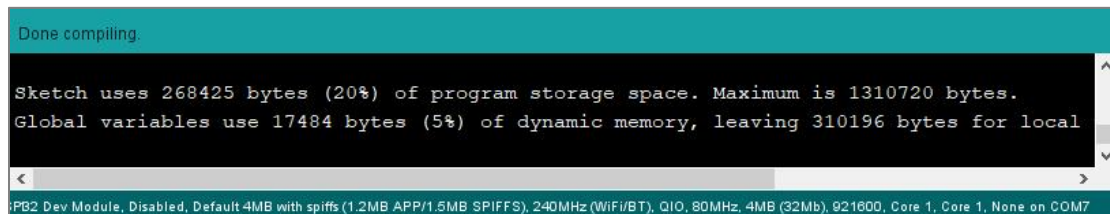
- 6) If you're not sure about the port number, please open the "This PC" and click "Properties->Device Manager" in turns to check the corresponding port number (the device is with CH340). Then select the correct port on Arduino editor.




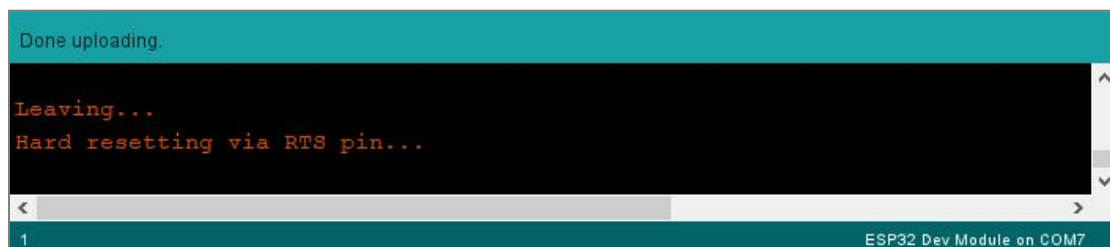
- 7) After selecting, confirm the board “ESP32 Dev Module” in the lower right corner and the port number “COM5” (it is an example here, please refer to the actual situation).

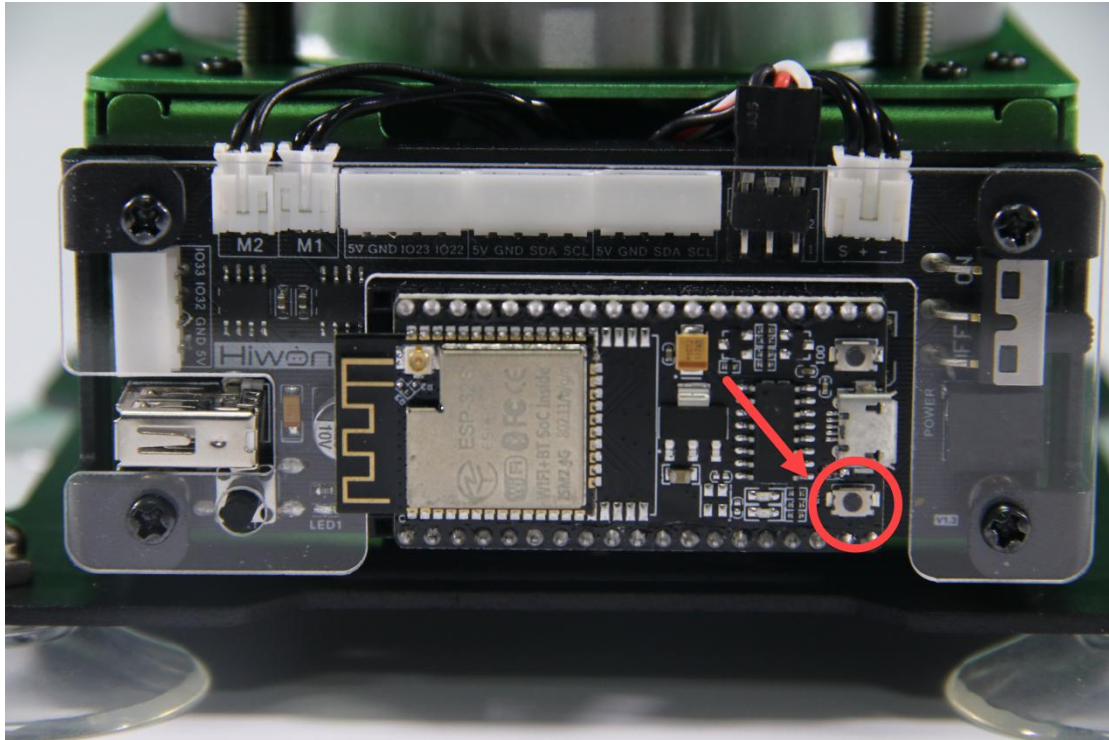


- 8) Then click on  icon to verify the program. If no error, the status area will display “Compiling->Compile complete” in turn. After compiling, the information such as the current used bytes, and occupied program storage space will be displayed.



- 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

After the buzzer beeps once, the robotic arm will suck and place the blocks in a line. When three blocks are placed completely, the buzzer will make “DiDi” sound as a feedback, and then a round of placement is finished.

5. Program Instruction

The path to the source code of the program is 6.Secondary Development /Arduino Development/Sensor-extension Game/Program Files/ Touch Detection and Placement/TouchSensor_put/TouchSensor_put.ino (The source code of the library files can be found under the same path).

If the program is modified, you can find the backup files in Appendix.

5.1 Import Function library

Before executing the program, the Python function libraries related to the buzzer, PWM servo, bus servo and air pump need to be imported.

```
1 #include "Buzzer.h"
2 #include "ESPMax.h"
3 #include "_espmax.h"
4 #include "ESP32PWMServo.h"
5 #include "SuctionNozzle.h"
6 #include "LobotSerialServoControl.h"
```

5.2 Define the Pin of Sensor

Firstly, define the P23 pin of the main chip as the access terminal of the touch sensor.

```
10 #define sensor_pin 23
```

5.3 Initialization Setting

Initialize the driver library and set the pin of sensor as pull-up mode.

```
12 void setup() {
13
14     Buzzer_init();
15     ESPMax_init();
16     Nozzle_init();
17     PWMServo_init();
18     pinMode(sensor_pin, INPUT_PULLUP);
19     Serial.begin(9600);
20     Serial.println("start...");
21     setBuzzer(100);
22     go_home(2000);
23     SetPWMServo(1,1500,2000);
```

5.2 Touch detection

When touching the metal surface of the sensor, the signal terminal will output low level and the buzzer will sound for 100ms.


```
36     if(sensor_state == 0.0){
37         Serial.print("num: ");
38         Serial.println(num+1);
39         setBuzzer(100);
```

The buzzer is controlled by `buzzer.setBuzzer()` function. Take the code “`buzzer.setBuzzer(100)`” as example.

The parameter “100” represents the sounding time of buzzer and the unit is ms.

5.5 Placement Setting

By setting the position parameter, MaxArm will transport the block to the corresponding position.

```
41     set_position(pos,1500);
42     delay(1500);
43     pos[0] = 0;pos[1] = -160;pos[2] = 85;
44     set_position(pos,800);
45     Pump_on();
46     delay(1000);
47     pos[0] = 0;pos[1] = -160;pos[2] = 180;
48     set_position(pos,1000);
49     delay(1000);
50     pos[0] = 120;pos[1] = (-20-60*num);pos[2] = 180;
51     set_position(pos,1500);
52     Serial.println(angle_pul[num]);
53     delay(100);
54     SetPWMServo(1,angle_pul[num],1000);
55     delay(500);
56     pos[0] = 120;pos[1] = (-20-60*num);pos[2] = (83+num);
57     set_position(pos,1000);
58     delay(1200);
59     Valve_on();
60     pos[0] = 120;pos[1] = (-20-60*num);pos[2] = 200;
61     set_position(pos,1000);
62     delay(1000);
63     Valve_off();
```

This section uses `arm.set_position()` function to control the robotic arm to move on the set axes. Take the code “`set_position(pos,1500)`” 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 “1500” represents the running time and its unit is ms.

The Pump_on() function is used to turn on the air pump, and the Valve_on() function is used to turn off the air pump.