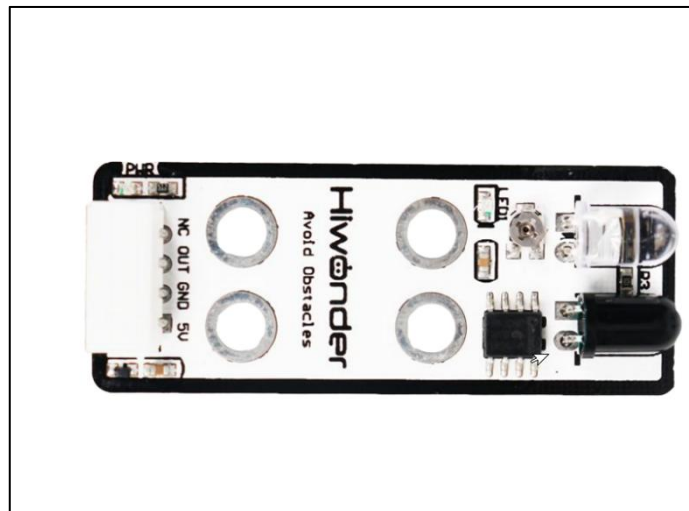


Lesson 2 Infrared Detection and Control

1. Working Principle

Infrared obstacle avoidance is a photoelectric sensor integrating IR transmitter and IR receiver. Featuring long detection distance and low interference from visible light , it is widely used in robot and assembly line piecework, etc.

This sensor detects obstacle by transmitting and receiving infrared. When the infrared transmitted by the sensor meets the obstacle ahead, the infrared will be reflected to the receiving terminal.



The path of the program file: 6.Secondary Development /Arduino Development/Sensor-extension Game/ Program Files/ Infrared Detection and Control/ InfraredSensor_contro/ InfraredSensor_control.ino.

```
34     if(sensor_state == 0.0){ ,
35         setBuzzer(100);
36         pos[0] = 0;pos[1] = -160;pos[2] = 100;
37         set_position(pos,1500);
38         delay(1500);
39         pos[0] = 0;pos[1] = -160;pos[2] = 85;
40         set_position(pos,800);
41         Pump_on();
42         delay(1000);
43         pos[0] = 0;pos[1] = -160;pos[2] = 200;
44         set_position(pos,1000);
45         delay(1000);
46         pos[0] = 70;pos[1] = -150;pos[2] = 200;
47         set_position(pos,800);
48         delay(800);
49         pos[0] = 70;pos[1] = -150;pos[2] = 90;
50         set_position(pos,800);
51         delay(800);
52         pos[0] = 130;pos[1] = -150;pos[2] = 88;
53         set_position(pos,500);
54         delay(500);
55         Valve_on();
56         pos[0] = 130;pos[1] = -150;pos[2] = 200;
57         set_position(pos,1000);
58         delay(1000);
59         Valve_off();
60         go_home(1500);
```

When the sensor detects this signal, it will send it to the microcontroller for processing. The closer the obstacle is, the stronger the reflection intensity; the farther the obstacle is, the weaker the reflection intensity. Different surface color has different reflection intensity. White is the strongest and black is the weakest.

Then, the object is detected by the infrared sensor and buzzer will make sound.

MaxArm will perform the corresponding action.

Finally, execute the function for controlling action, buzzer, air pump to suck and place the object.

2. Preparation


2.1 Hardware

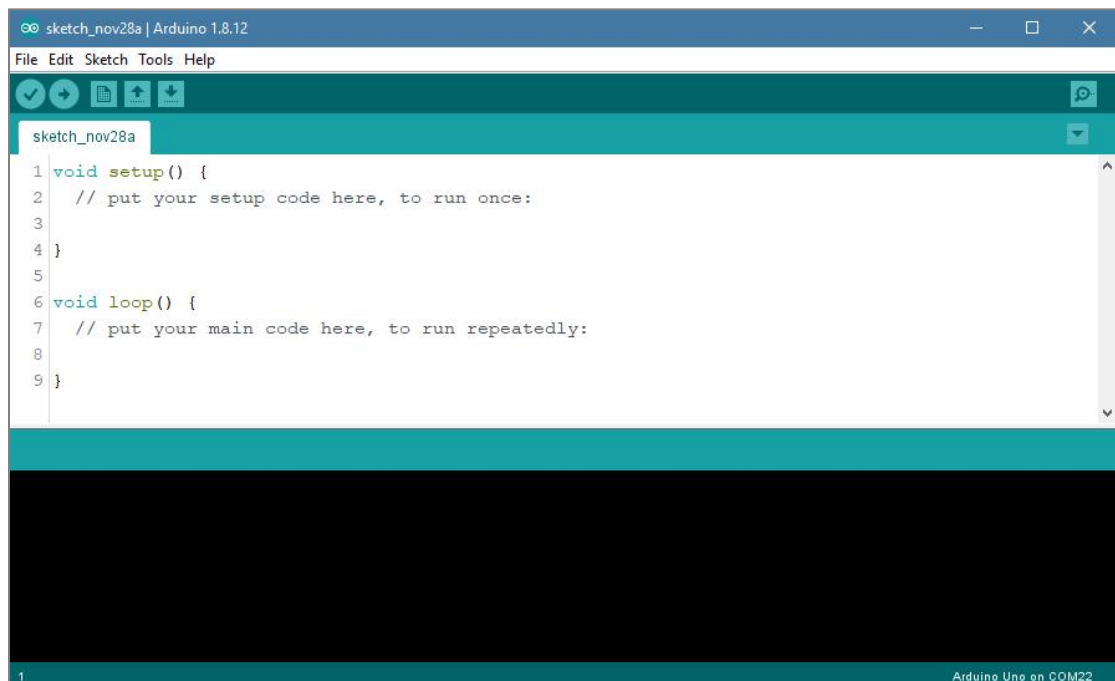
Please assemble the infrared sensor to the corresponding position on MaxArm according to the tutorial in folder “Lesson 1 Sensor Assembly” under the same directory.

1.2 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

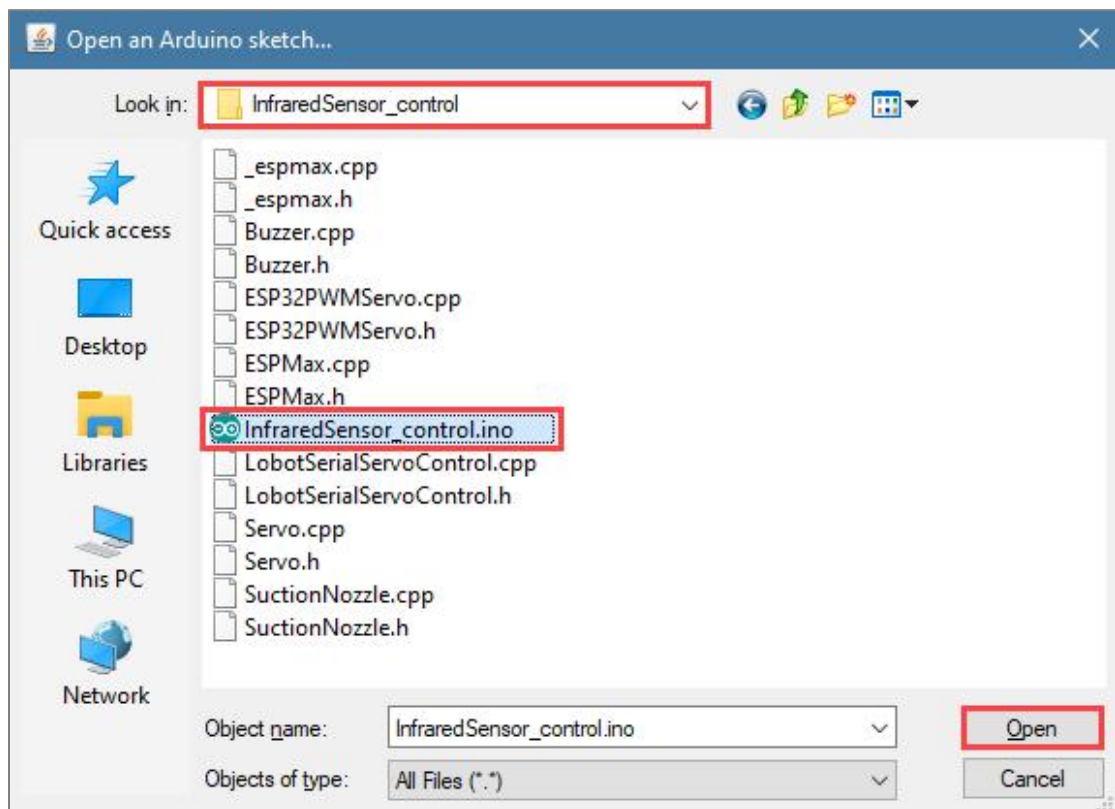
- 1) Click on  icon to open Arduino IDE.



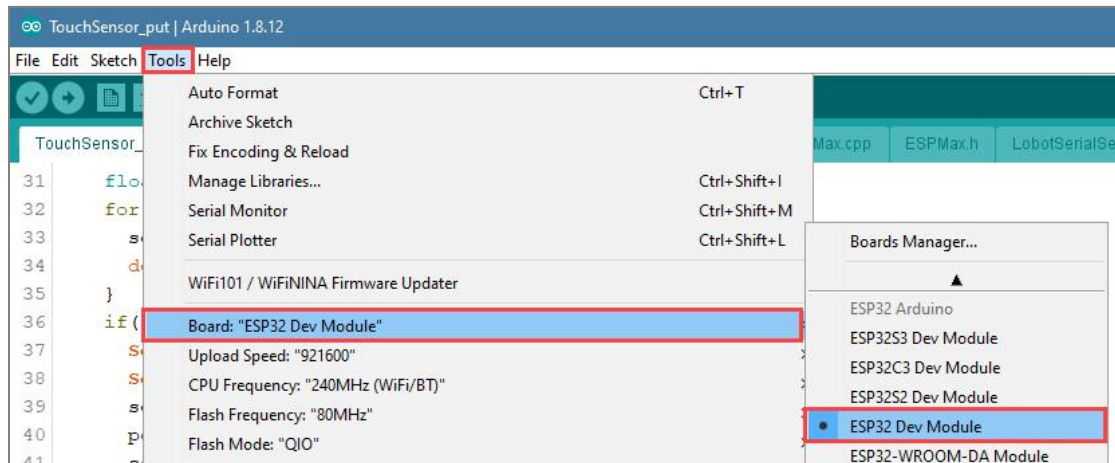
- 2) Click “File->Open” in turn.



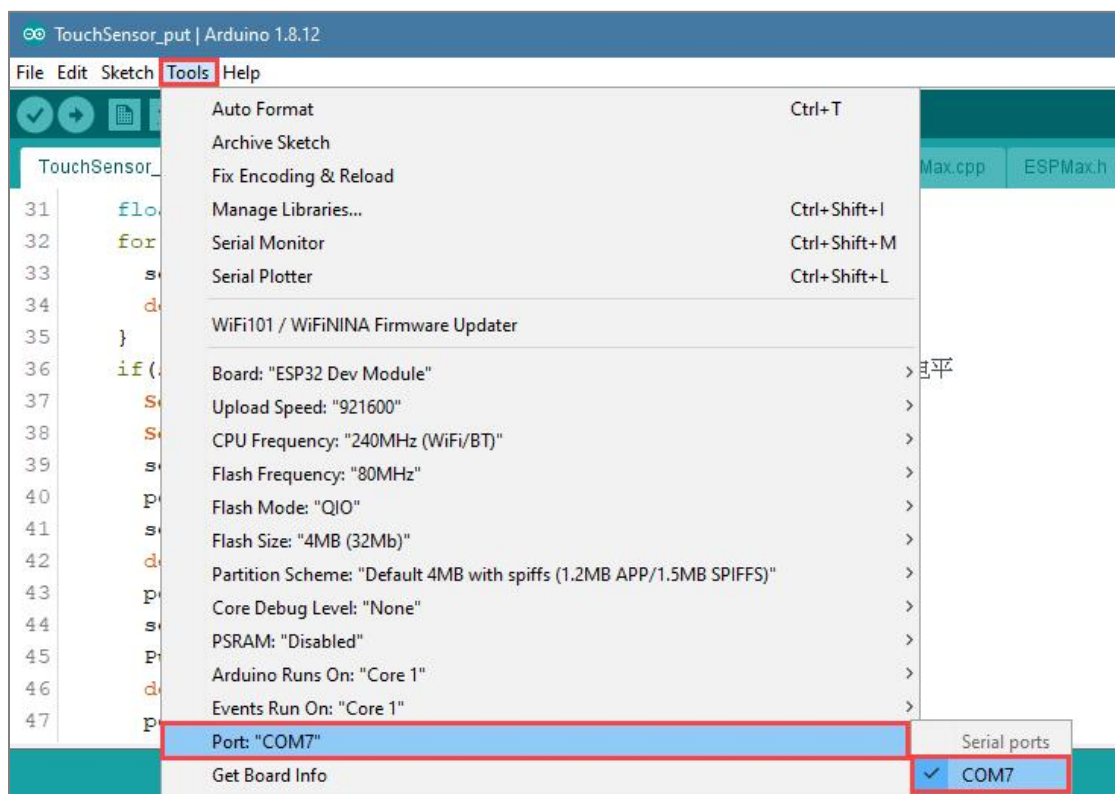
- 3) Select the program “InfraredSensor_control.ino” in the folder “6.Secondary Development / Arduino Development/Sensor-extension Game/Program Files/Infrared Detection and Control/InfraredSensor_control”.



- 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).

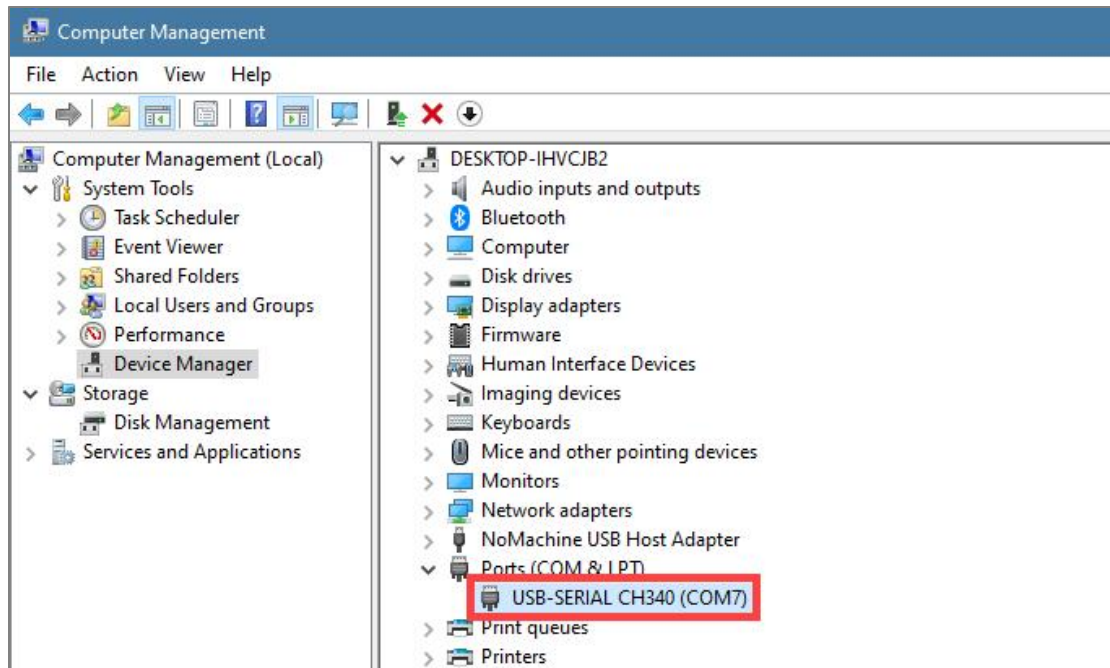


- 5) Select the corresponding port of Arduino controller in “Tools->Port”. (Here take the port “COM5” as example. Please select the port based on your computer. If COM1 appears, please do not select because it is the system communication port but not the actual port of the development port.)




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

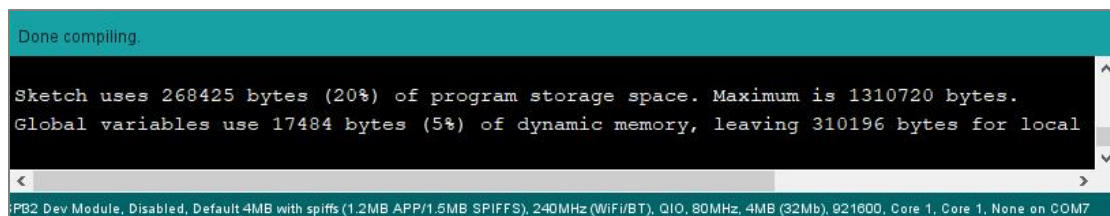
editor.




- 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).

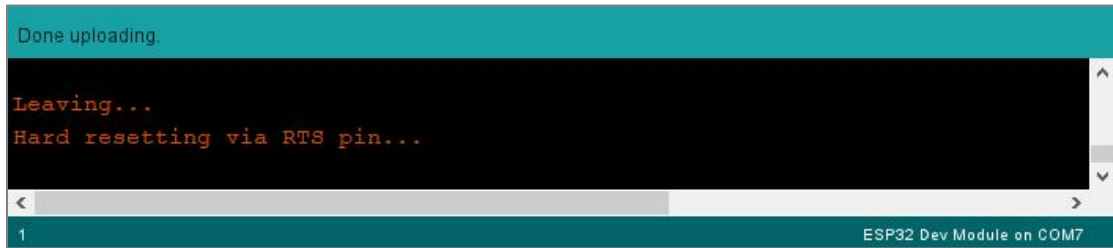


- 8) Then click on  icon to verify the program. If no error, the status area will display “Compiling->Compile complete” in turn. After compiling, the information such as the current used bytes, and occupied program storage space will be displayed.



- 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.



The screenshot shows a serial monitor interface with a teal header bar. The text "Done uploading." is displayed in the header. The main black area shows the text "Leaving..." and "Hard resetting via RTS pin..." in red. A scrollbar is visible on the right. The bottom status bar is teal and contains the text "1" on the left and "ESP32 Dev Module on COM7" on the right.

4. Project Outcome

When the colored block is detected by infrared sensor, the robotic arm will move to the block and suck it, then transfer and place the block to the placement area. Finally, turn off the air pump.

5. Program Instruction

5.1 Import and Initialize

The path to the source code of the program is 6.Secondary Development /Arduino Development/ Sensor-extension Game/ Program Files/ Infrared Detection and Control/ InfraredSensor_control/ InfraredSensor_control.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.

Before running the program, the buzzer, kinematics, kinematics encapsulation library, PWM servo, suction nozzle and other related library files 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"
```

Then, initialize the library file and the robotic arm.

```
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);
24 }
```

5.2 Infrared Detection

Use the `digitalRead()` function to read the detected value of the infrared sensor and then use the `for()` function to detect several times.

When the object is detected by infrared sensor, `sensor_pin` is 0, otherwise, it is 1. Then the value of the detected value plus `sensor_state` is assigned to `sensor_state`. If the `sensor_state` is equal to 0.0 after five rounds, the object is detected, otherwise, no object is detected.

```
26 void loop() {
27     float pos[3];
28
29     float sensor_state = 0.0;
30     for(int i=0; i<5; i++){
31         sensor_state += digitalRead(sensor_pin);
32         delay(50);
33     }
```


5.3 Control Robotic Arm

When the object is detected by infrared sensor, MaxArm will perform the corresponding action.

```
34     if(sensor_state == 0.0){ ,
35         setBuzzer(100);
36         pos[0] = 0;pos[1] = -160;pos[2] = 100;
37         set_position(pos,1500);
38         delay(1500);
39         pos[0] = 0;pos[1] = -160;pos[2] = 85;
40         set_position(pos,800);
41         Pump_on();
42         delay(1000);
43         pos[0] = 0;pos[1] = -160;pos[2] = 200;
44         set_position(pos,1000);
45         delay(1000);
46         pos[0] = 70;pos[1] = -150;pos[2] = 200;
47         set_position(pos,800);
48         delay(800);
49         pos[0] = 70;pos[1] = -150;pos[2] = 90;
50         set_position(pos,800);
51         delay(800);
52         pos[0] = 130;pos[1] = -150;pos[2] = 88;
53         set_position(pos,500);
54         delay(500);
55         Valve_on();
56         pos[0] = 130;pos[1] = -150;pos[2] = 200;
57         set_position(pos,1000);
58         delay(1000);
59         Valve_off();
60         go_home(1500);
```

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

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

The robotic arm is controller by set_position() function. 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” is the running time and the unit is ms.

Use Pump_on() function to turn on the air pump, Valve_on() function to turn off the air pump, and Valve_off() function to turn off the solenoid valve.

Finally, use go_home() function to make robotic arm return to the initial posture.

Take the code “go_home(1500)” as example.

The parameter “1500” is the time for the robotic arm to reset and the unit is ms.