

Lesson 2 Ultrasonic Detection and Sucking

1. Working Principle

Ultrasonic sensors is a sensor that converts ultrasonic signals into other energy signals (usually electrical signals). There are two probes on ultrasonic sensor for receiving and transmitting ultrasound.

Firstly, import the corresponding library and initialize ultrasonic sensor, buzzer, servo and action groups.

Next, the object is detected by ultrasonic sensor and the measured distance is read by I2C protocol. After determining the distance, MaxArm will perform the corresponding action based on the determined result.

Then, execute the functions for controlling action group, starting buzzer and air pump to suck the object to the side.

The path to the source code of the program is 6.Secondary Development
/Arduino Development/Sensor Development/Program Files/ Ultrasonic
Detection and Suction/ Ultrasound_clamp/Ultrasound_clamp.ino

```
9 Ultrasound ultrasound;
10
11 void setup() {
12     Buzzer_init();
13     ESPMax_init();
14     Nozzle_init();
15     PWMServo_init();
16     Valve_on();
17     go_home(2000);
18     delay(2000);
19     SetPWMServo(1, 1500, 1000);
20     Valve_off();
21     Serial.begin(115200);
22     Serial.println("start...");
23     ultrasound.Breathing(30, 50, 60, 20, 30, 50);
```

2. Preparation


2.1 Hardware

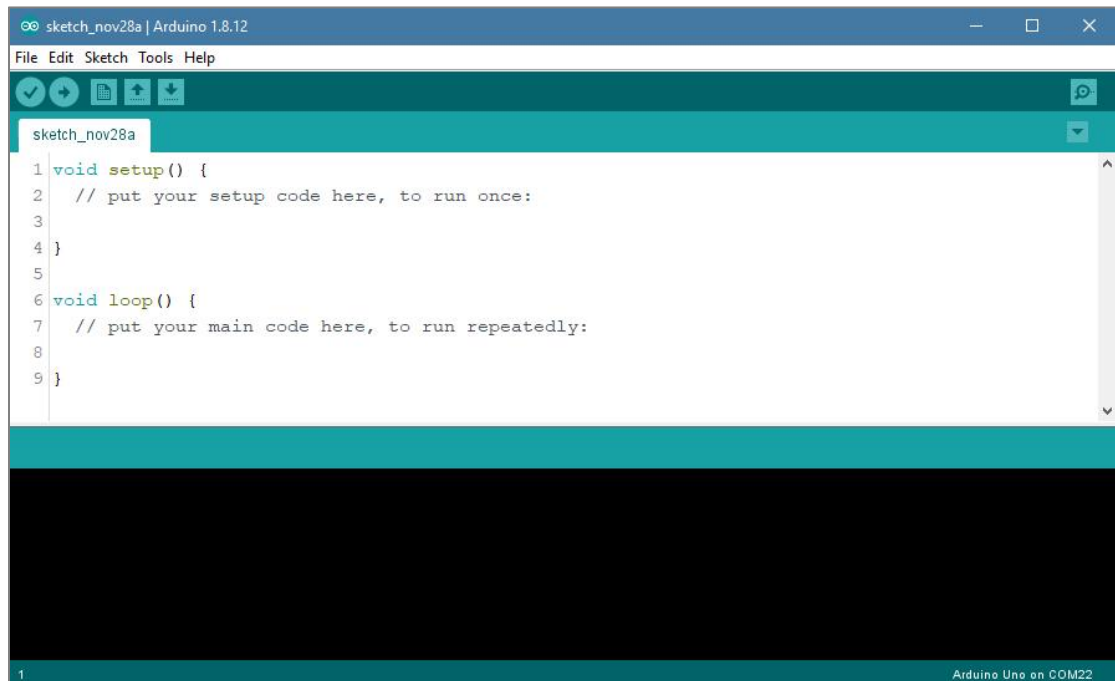
Please assemble the ultrasonic sensor to the corresponding position on MaxArm according to the tutorial in folder “Lesson 1 Sensor Assembly” under the same directory.

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

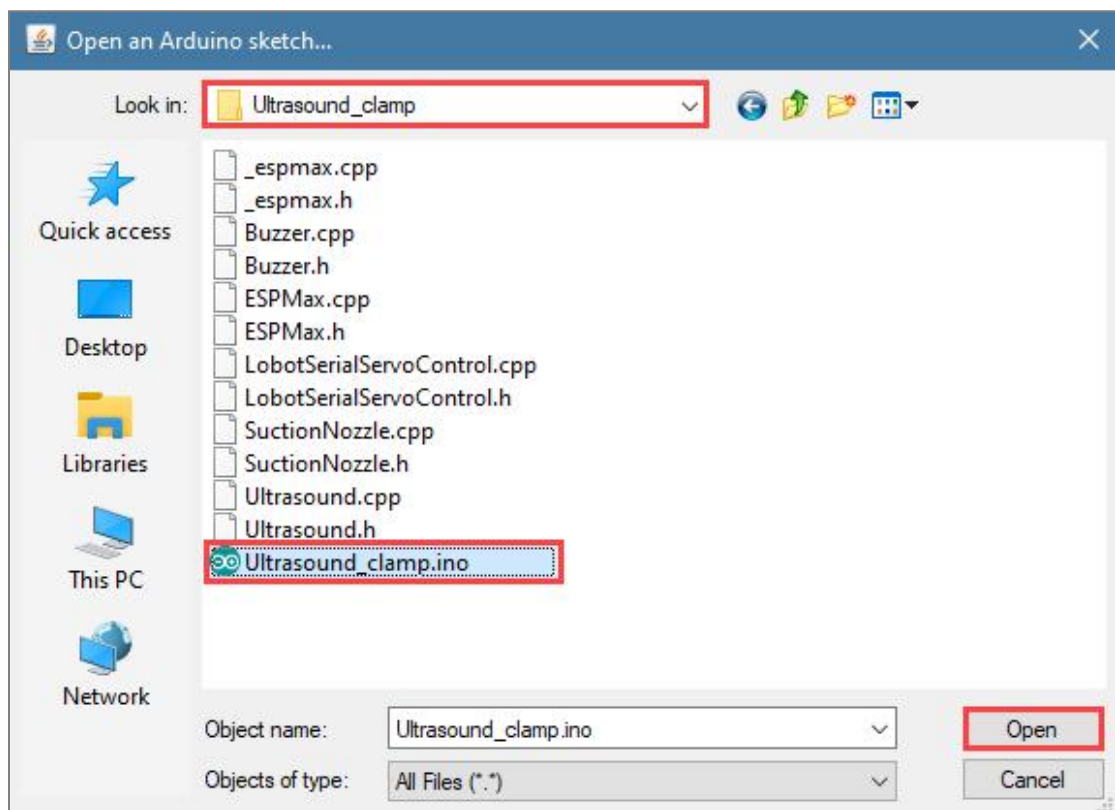
- 1) Click on  icon to open Arduino IDE.



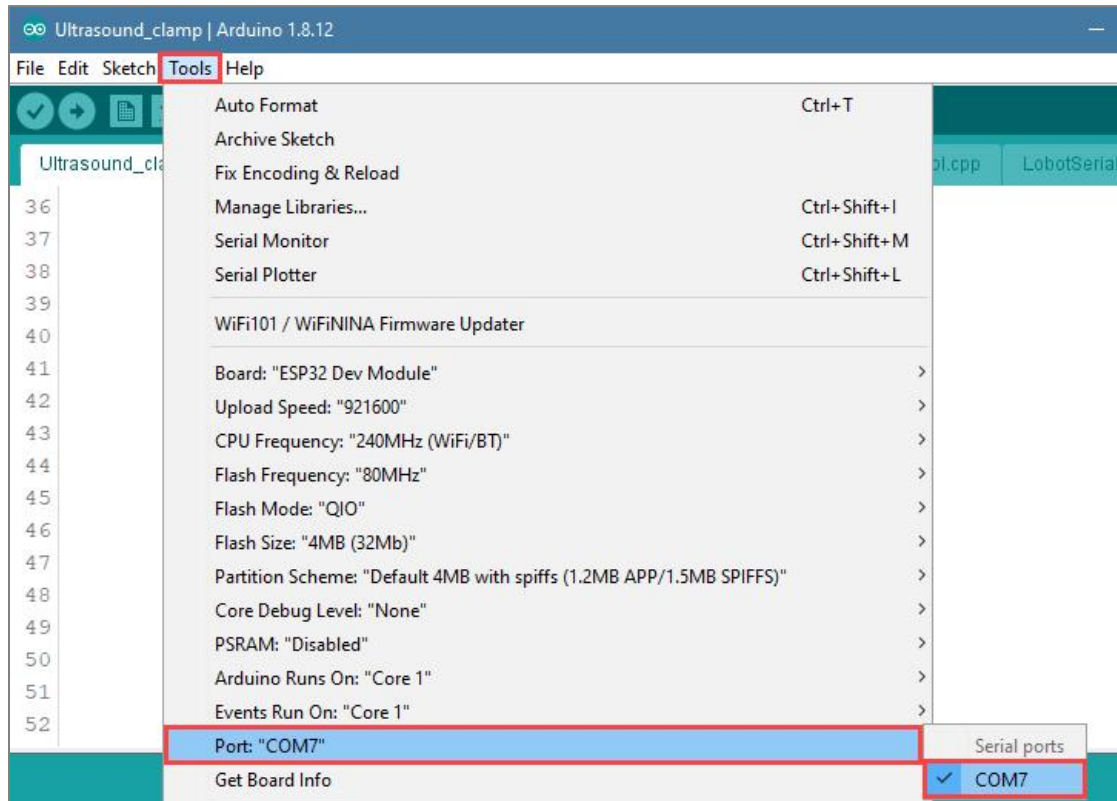
- 2) Click “File->Open” in turn.



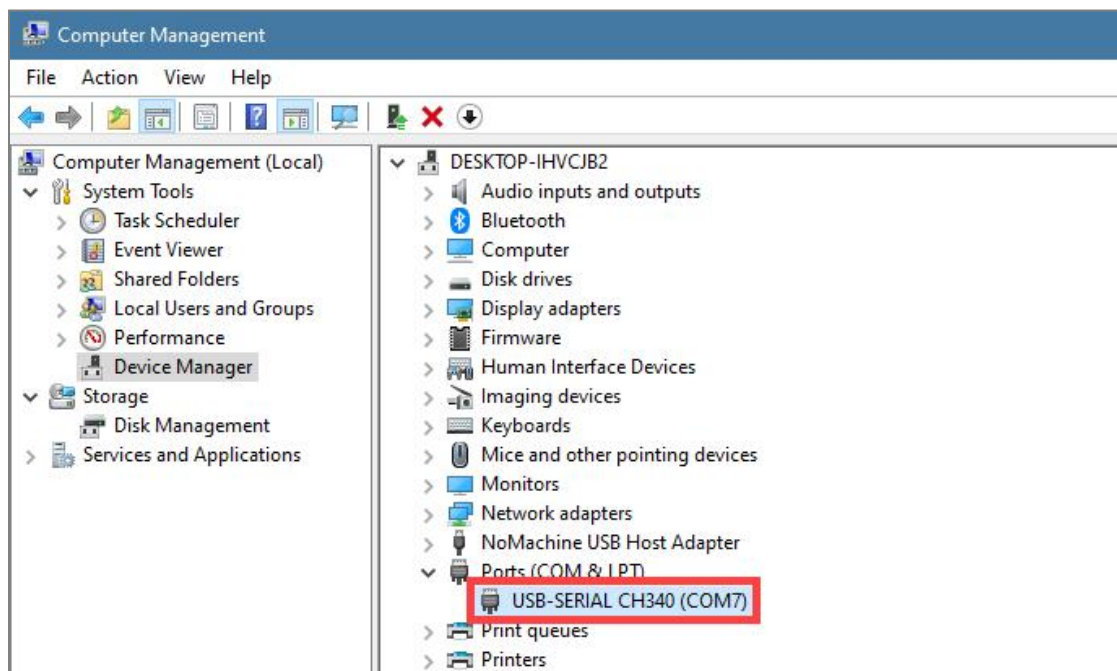
- 3) Select the program “Ultrasound_clamp.ino” in the folder “6.Secondary Development/ Arduino Development/Sensor-extension Game/ Program Files/Ultrasonic Detection and Suction/Ultrasound_clamp”.



- 4) 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.)




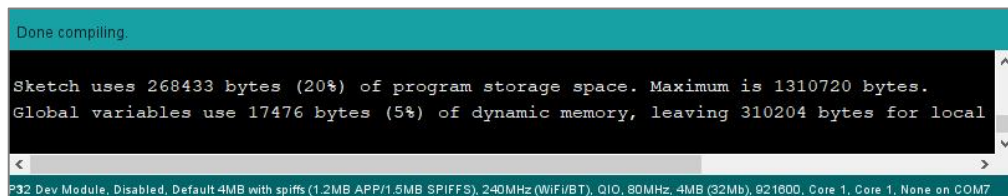
- 5) If you're not sure about the port number, please open the "This PC" and click "Properties->Device Manager" in turns to check the corresponding port number (the device is with CH340). Then select the correct port on Arduino editor.




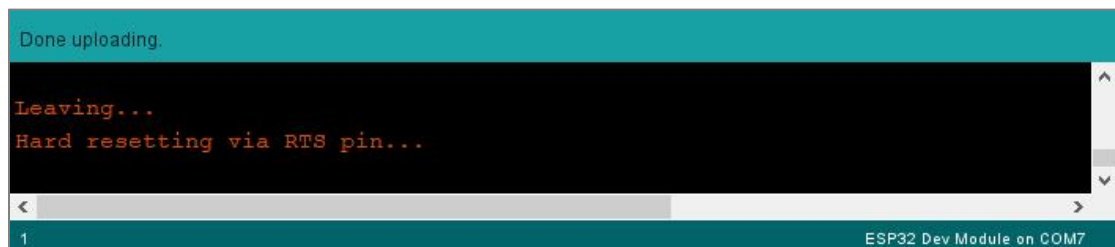
- 6) After selecting, confirm the board “ESP32 Dev Module” in the lower right corner and the port number “COM5” (it is an example here, please refer to the actual situation).



- 7) 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.



- 8) 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 putting the block on the fixed detection position, the buzzer will beep after the block is detected by the ultrasonic sensor. MaxArm will move to the front of the block and start the air pump to suck it, and moves to the left. Then placing the block and stopping the air pump. Finally, MaxArm is reset.

5. Program Instruction

5.1 Import Library File

Firstly, the ultrasonic sensor, PWM servo, buzzer and other related library files are called.

```
1 #include "ESPMMax.h"
2 #include "Buzzer.h"
3 #include "Ultrasound.h"
4 #include "SuctionNozzle.h"
5 #include "ESP32PWMServo.h"
```

5.2 Ultrasonic Detection

Then read the measured distance by setting the variable. To reduce errors, take the average.

```
26 void loop() {
27     float pos[3];
28     int distance = 0;
29     for(int i=0; i<5; i++){
30         distance += ultrasound.GetDistance();
31         delay(200);
32     }
33     int dis = int(distance/5);
```

5.3 Detection Feedback

Set a distance range with another if judgement statement. When the measured distance meets the set conditions, the buzzer will sound for 100ms.

```
35     if(60 < dis & dis < 80){
36         setBuzzer(100); //设置
```

5.4 Control Robotic Arm

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


```
37     pos[0] = 0;pos[1] = -160;pos[2] = 100;
38     set_position(pos,1500);
39     delay(1500);
40     pos[0] = 0;pos[1] = -160;pos[2] = 85;
41     set_position(pos,800);
42     Pump_on();
43     delay(1000);
44     pos[0] = 0;pos[1] = -160;pos[2] = 200;
45     set_position(pos,1000);
46     delay(1000);
47     pos[0] = 70;pos[1] = -150;pos[2] = 200;
48     set_position(pos,800);
49     delay(800);
50     SetPWMServo(1, 2200, 500);
51     delay(200);
52     pos[0] = 70;pos[1] = -150;pos[2] = 90;
53     set_position(pos,800);
54     delay(800);
55     pos[0] = 130;pos[1] = -150;pos[2] = 88;
56     set_position(pos,500);
57     delay(500);
58     Valve_on();
```

This section uses set_position() function to control the robotic arm. 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.

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