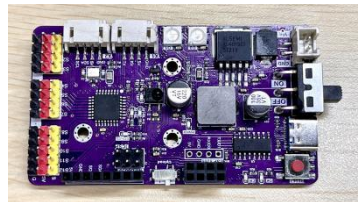


Lesson 6 Measure the Distance with an Ultrasonic Sensor

6.1 Overview

This course mainly introduces how to use ultrasonic sensors to measure distance, and combines Adeept Pixie Board and Arduino development environment to explain in detail the required components, working principles, circuit connections, code writing and testing process, helping learners master the application of ultrasonic ranging technology in practical projects.

6.2 Required Components

Components	Quantity	Picture
Adeept Pixie Board	1	
Type-C USB Cable	1	
Ultrasonic module	1	
OLED Screen	1	

6.3 Principle Introduction

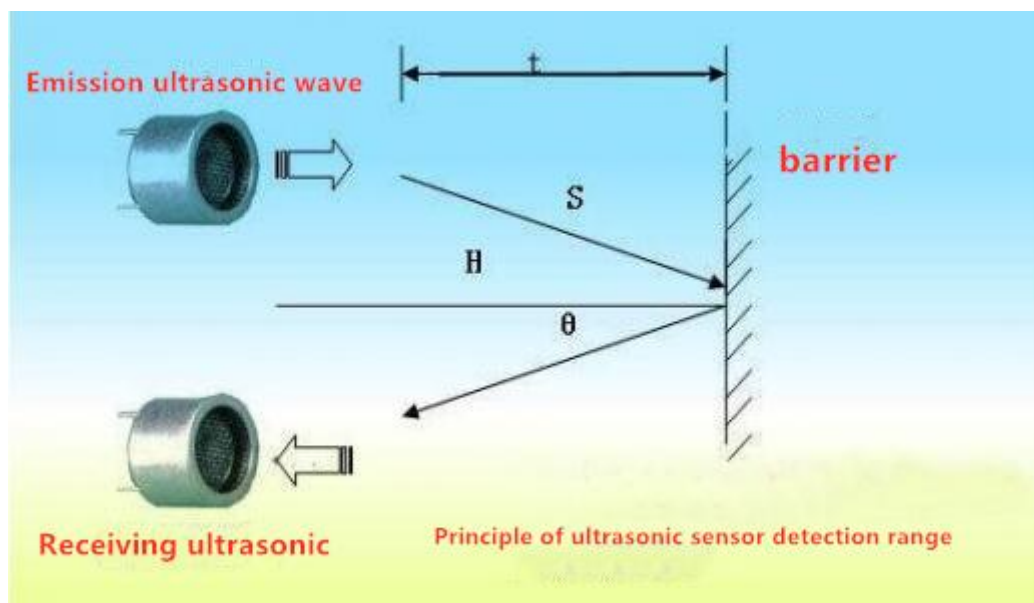
The ultrasonic module has four pins, namely VCC, GND, Echo and Trig. The HC-SR04 can provide a non-contact distance sensing function of 2cm-200cm, and the ranging accuracy can reach 3mm; The module includes an ultrasonic transmitter, receiver and control circuit. The basic working principle is as follows:

Use IO port TRIG to trigger distance measurement, and give a high level signal of at least 10us.

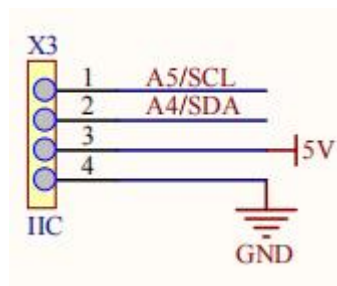
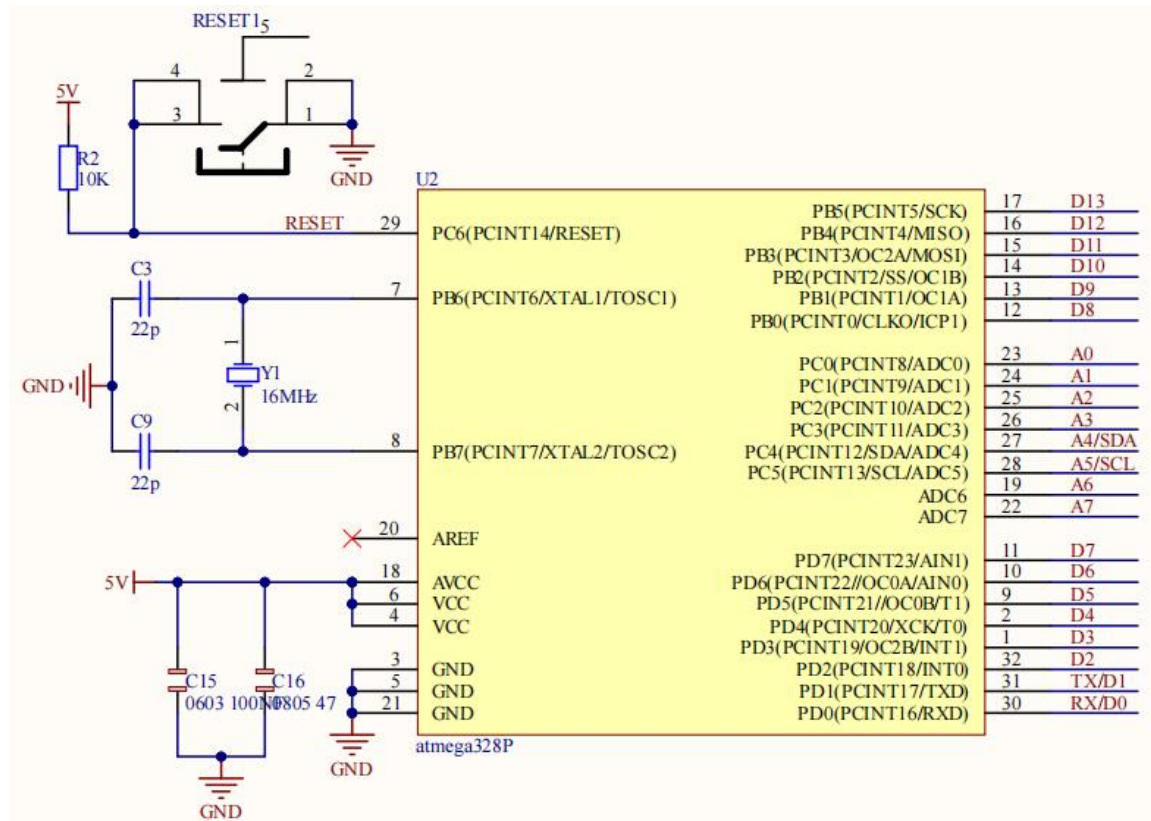
The module automatically sends eight 40khz square waves, and automatically detects whether there is a signal return.

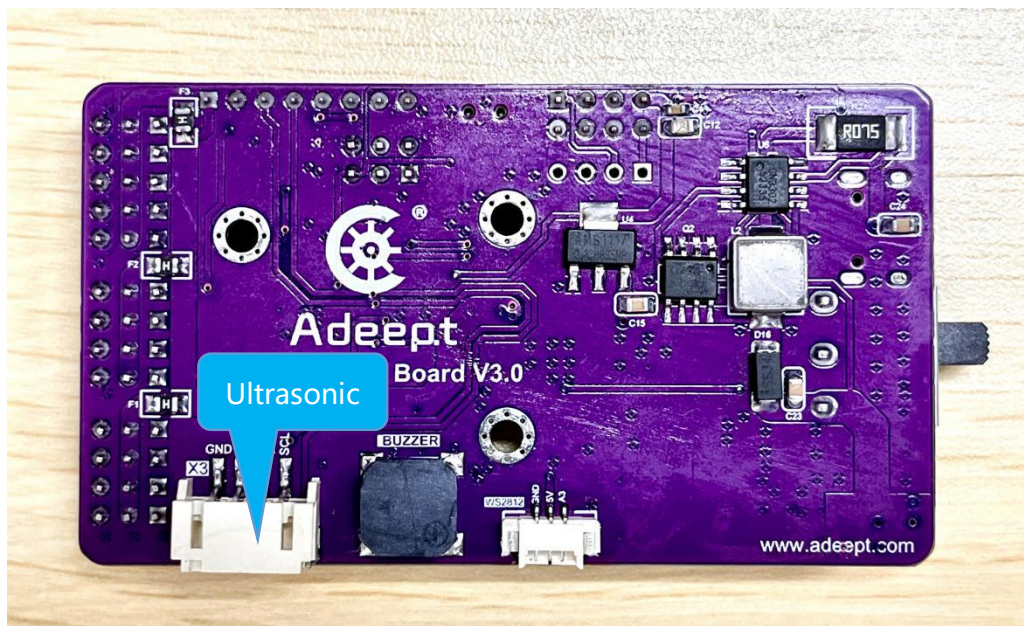
There is a signal return, and a high level is output with the IO port ECHO. The duration of the high level is the time from emission to return of the ultrasonic wave.

The principle of distance detection by ultrasonic ranging sensor: the method of detecting distance by ultrasonic is called echo detection method, that is, the ultrasonic transmitter emits ultrasonic waves in a certain direction, and the timer starts timing at the same time as the launch time. The ultrasonic waves propagate in the air and encounter obstacles on the way. When the object surface (object) is blocked, it will be reflected back immediately, and the ultrasonic receiver will immediately stop timing when the reflected ultrasonic wave is received. The propagation speed of ultrasonic waves in the air is 340m/s. According to the time t recorded by the timer, the distance s from the launch point to the obstacle surface can be calculated, namely: $s = 340t/2$. Using this principle of ultrasound, the ultrasonic ranging module is widely used in practical applications, such as car reversing radar, unmanned aerial vehicle, and smart car.



6.4 Wiring Diagram

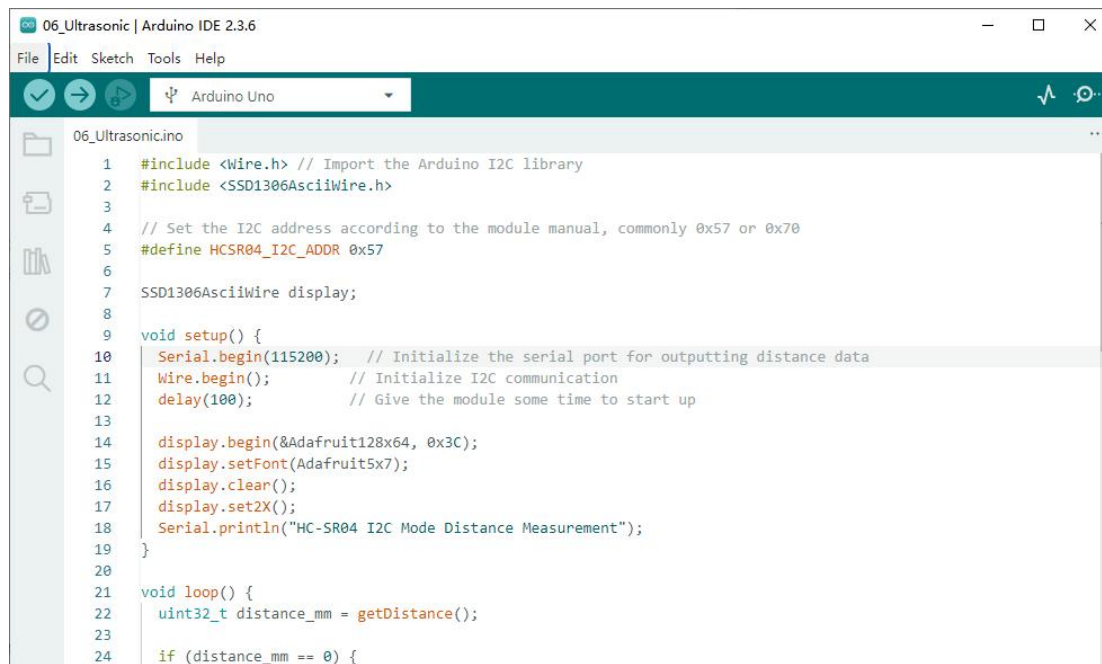




PINS of Adeept Pixie Board	Ultrasonic
5V	VCC
A5	Trig/SCL
A4	Echo/SDA
GND	GND

6.5 Demonstration

1. Connect your computer and Adeept Pixie Board (Arduino Board) with a USB cable.
2. Open "**06_Ultrasonic**" folder in **"/Code"** , double-click "**06_Ultrasonic.ino**" .



```

06_Ultrasonic.ino
1  #include <Wire.h> // Import the Arduino I2C library
2  #include <SSD1306AsciiWire.h>
3
4  // Set the I2C address according to the module manual, commonly 0x57 or 0x70
5  #define HCSR04_I2C_ADDR 0x57
6
7  SSD1306AsciiWire display;
8
9  void setup() {
10     Serial.begin(115200); // Initialize the serial port for outputting distance data
11     Wire.begin(); // Initialize I2C communication
12     delay(100); // Give the module some time to start up
13
14     display.begin(&Adafruit128x64, 0x3C);
15     display.setFont(Adafruit5x7);
16     display.clear();
17     display.set2X();
18     Serial.println("HC-SR04 I2C Mode Distance Measurement");
19 }
20
21 void loop() {
22     uint32_t distance_mm = getDistance();
23
24     if (distance_mm == 0) {

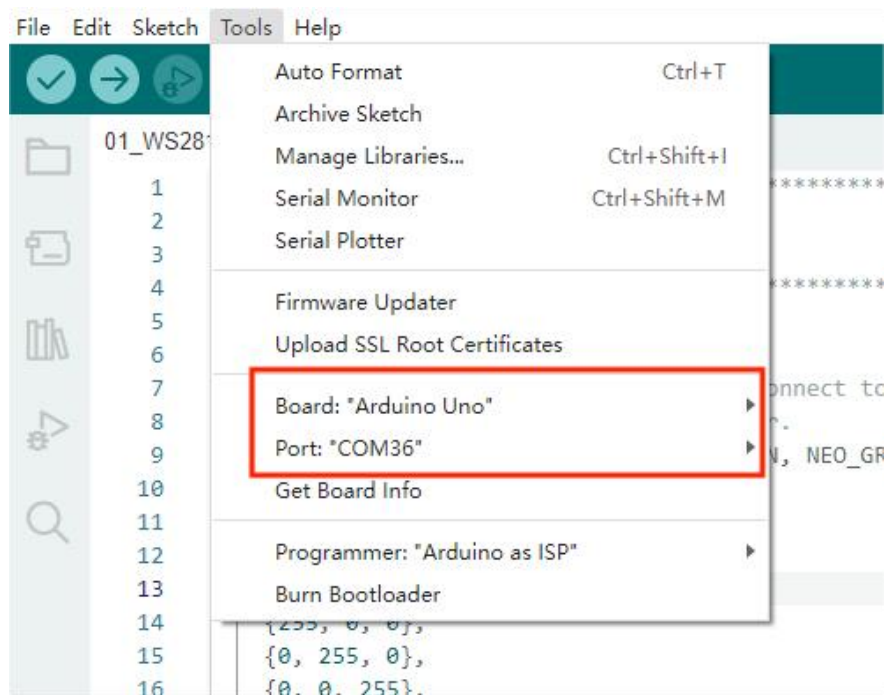
```

3. Select development board and serial port.

Board: **Tools--->Board--->Arduino AVR Boards--->Arduino Uno**

Port: **Tools --->Port--->COMx**

Note: The port number will be different in different computers.





4. After opening, click  to upload the code program to the Arduino. If there is no error warning in the console below, it means that the Upload is successful.

```

Output
Sketch uses 2952 bytes (9%) of program storage space. Maximum is 32256 bytes.
Global variables use 82 bytes (4%) of dynamic memory, leaving 1966 bytes for local variables. Maximum is 2048 bytes.

```

5. After successfully running the program, the ultrasonic sensor will continuously measure the distance and display the measurement results in real-time on the OLED, in centimeters.

6.6 Code

```

01 #include <Wire.h> // Import the Arduino I2C library
02 #include <SSD1306AsciiWire.h>
03
04 // Set the I2C address according to the module manual, commonly 0x57 or 0x70
05 #define HCSR04_I2C_ADDR 0x57
06
07 SSD1306AsciiWire display;
08
09 void setup() {
10     Serial.begin(115200); // Initialize the serial port for outputting distance data
11     Wire.begin(); // Initialize I2C communication
12     delay(100); // Give the module some time to start up
13
14     display.begin(&Adafruit128x64, 0x3C);
15     display.setFont(Adafruit5x7);
16     display.clear();
17     display.set2X();
18     Serial.println("HC-SR04 I2C Mode Distance Measurement");
19 }
20
21 void loop() {
22     uint32_t distance_mm = getDistance();
23
24     if (distance_mm == 0) {
25         Serial.println("Measurement error or timeout.");
26     } else {
27         display.setCursor(0, 0); // Center display
28         display.print("Distance: ");
29         display.setCursor(0, 3); // Center display
30         display.print(distance_mm); // Output distance in millimeters
31         display.println(" cm");
32     }
33     delay(200); // At least 200ms interval between two measurements[1](@ref)
34 }

```

```
35
36 // Function to get distance via I2C
37 uint32_t getDistance() {
38     // 1. Send measurement command
39     Wire.beginTransmission(HCSR04_I2C_ADDR);
40     Wire.write(0x01); // Usually write 0x01 command to start ranging[1,4](@ref)
41     byte error = Wire.endTransmission();
42
43     if (error != 0) {
44         Serial.print("I2C transmission error: ");
45         Serial.println(error);
46         return 0;
47     }
48
49     // 2. Wait for the measurement to complete (the module needs time to measure)
50     delay(200); // Blocking delay, simple handling. Can be changed to non-blocking if needed
51
52     // 3. Read data. Usually 3 bytes[1,4](@ref)
53     Wire.requestFrom(HCSR04_I2C_ADDR, 3);
54     if (Wire.available() >= 3) {
55         byte highByte = Wire.read(); // High byte of distance data
56         byte midByte = Wire.read(); // Middle byte of distance data
57         byte lowByte = Wire.read(); // Low byte of distance data
58
59         // Combine the 3 bytes into a complete distance value (usually in millimeters)
60
61         uint32_t dist = ( ((uint32_t)highByte << 16) + ((uint32_t)midByte << 8) + lowByte ) /1000;
62
63         return dist/10.0;
64     } else {
65         Serial.println("Incomplete data received.");
66         return 0;
67     }
68 }
```

Code explanation

Initialization Stage:

In the setup function, initialize I2C communication and initialize OLED to display the data received by ultrasonic waves.

Loop Control Process:

In the loop function, call the getDistance() to obtain the distance data. Use display.print to display the distance data.

Delay by 200ms after each measurement to avoid frequent measurements and ensure the stability of measurement results.