

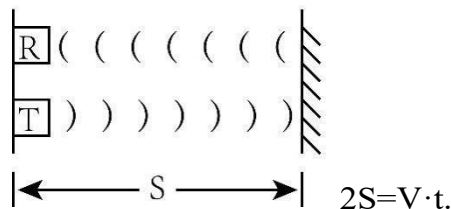
Experiment 4-Testing Ultrasonic Ranging Module

Table

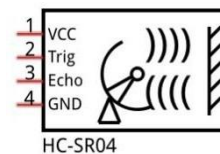
1.	Knowledge of Ultrasonic Ranging Module	1
2.	Component/Module List	2
3.	Build an experiment	3
4.	Upload code and test	6
5.	Make your suggestion and get support	9

1. Knowledge of Ultrasonic Ranging Module

The Ultrasonic Ranging Module uses the principle that ultrasonic waves will reflect when they encounter any obstacles. This is possible by counting the time interval between when the ultrasonic wave is transmitted to when the ultrasonic wave reflects back after encountering an obstacle. Time interval counting will end after an ultrasonic wave is received, and the time difference (delta) is the total time of the ultrasonic wave's journey from being transmitted to being received. Because the speed of sound in air is a constant, and is about $v=340\text{m/s}$, we can calculate the distance between the Ultrasonic Ranging Module and the obstacle: $s=vt/2$.



The HC-SR04 Ultrasonic Ranging Module integrates a both an ultrasonic transmitter and a receiver. The transmitter is used to convert electrical signals (electrical energy) into high frequency (beyond human hearing) sound waves (mechanical energy) and the function of the receiver is opposite of this. The picture and the diagram of the HC SR04



Ultrasonic Ranging Module are shown below:

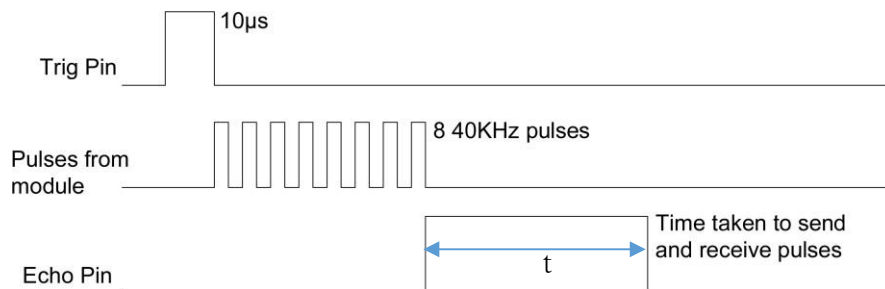
Pin description:

Pin name	Pin number	Description
Vcc	1	Positive electrode of power supply, the voltage is 5V
Trig	2	Triger pin

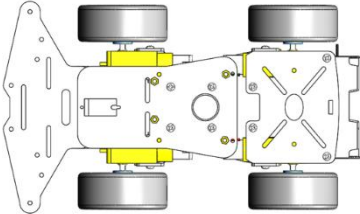
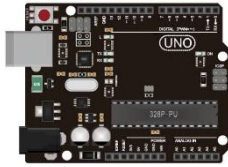


Echo	3	Echo pin
Gnd	4	Negative electrode of power supply



Instructions for use:

Output a high-level pulse in Trig pin lasting for least 10uS, the module begins to transmit ultrasonic waves. At the same time, the Echo pin is pulled **up**. When the module receives the returned ultrasonic waves from encountering an obstacle, the Echo pin will be pulled **down**. The duration of high level in the Echo pin is the total time of the ultrasonic wave from transmitting to receiving, $s=vt/2$. This is done constantly.



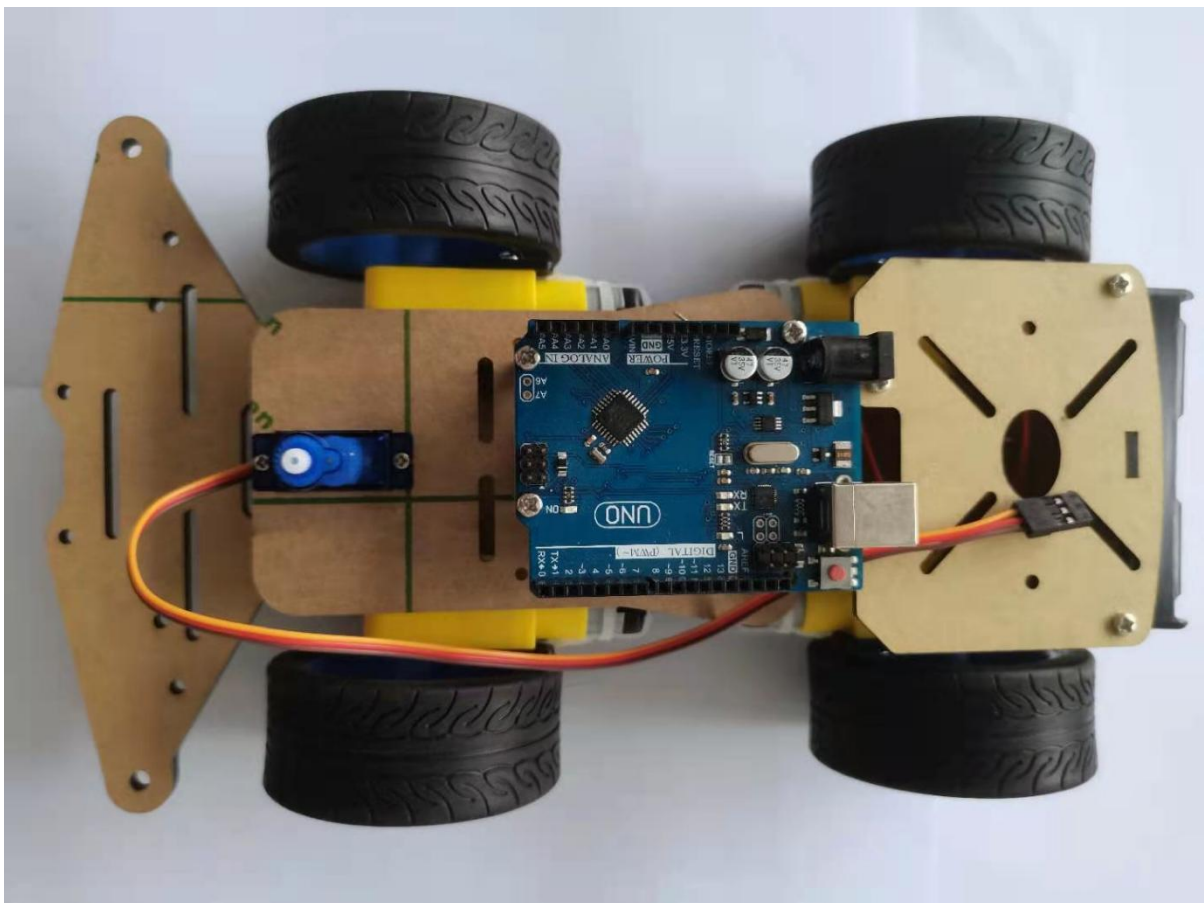
2. Component/Module List

Component/Module	QTY	Picture	Remark
4WD Car Chassis	1		Provided by the 4WD Car Chassis Kit
UNO R3 Board	1		You need to prepare these by yourself. These just as an example, you can DIY what is you want and prepared
MG90S Micro Servo	1		
Ultrasonic Ranging Module	1		

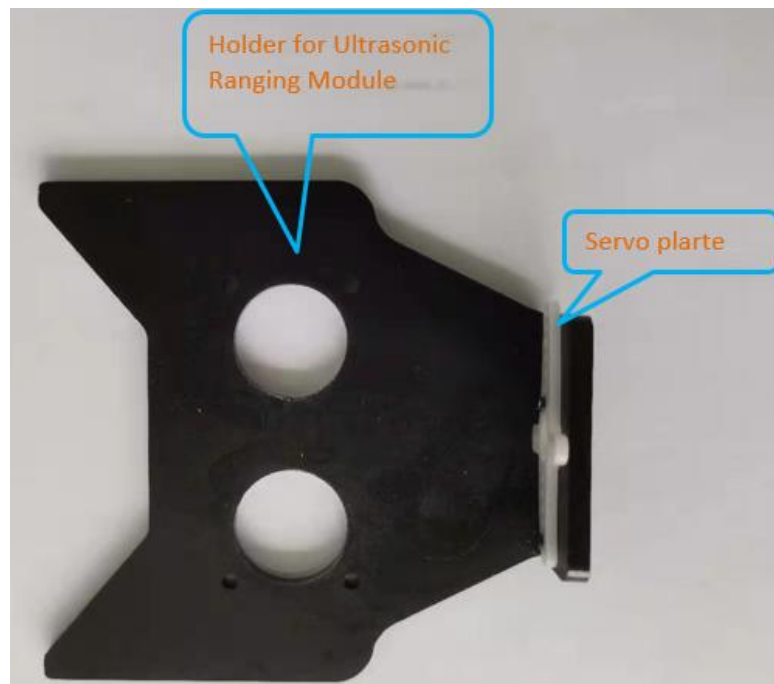
Holder for Ultrasonic Ranging Module	1		
Dupont line	Some		

3. Build an experiment

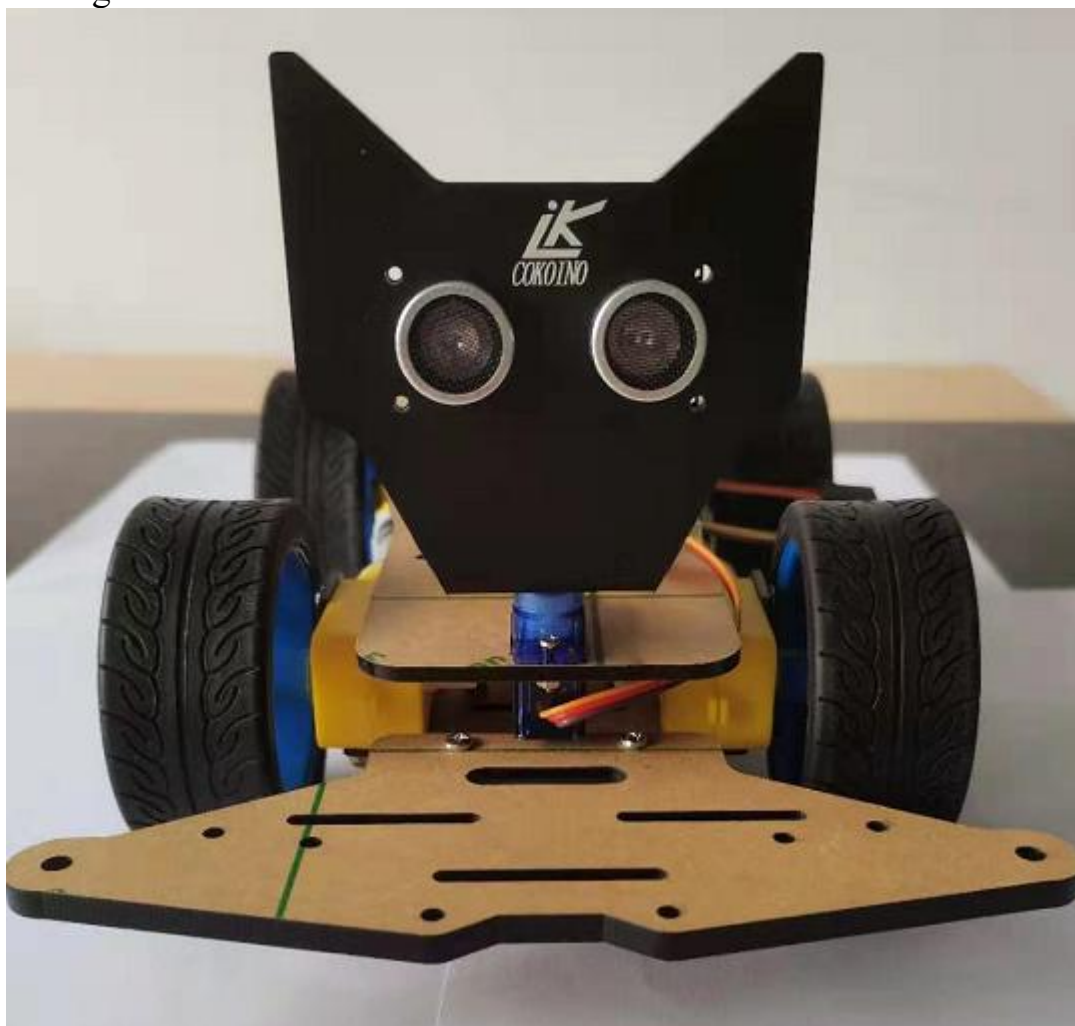
3.1 Assemble the Micro MG90S Servo and UNO R3 Board to the 4WD body as shown below



3.2 Install the servo plate on the ultrasonic bracket, as shown below

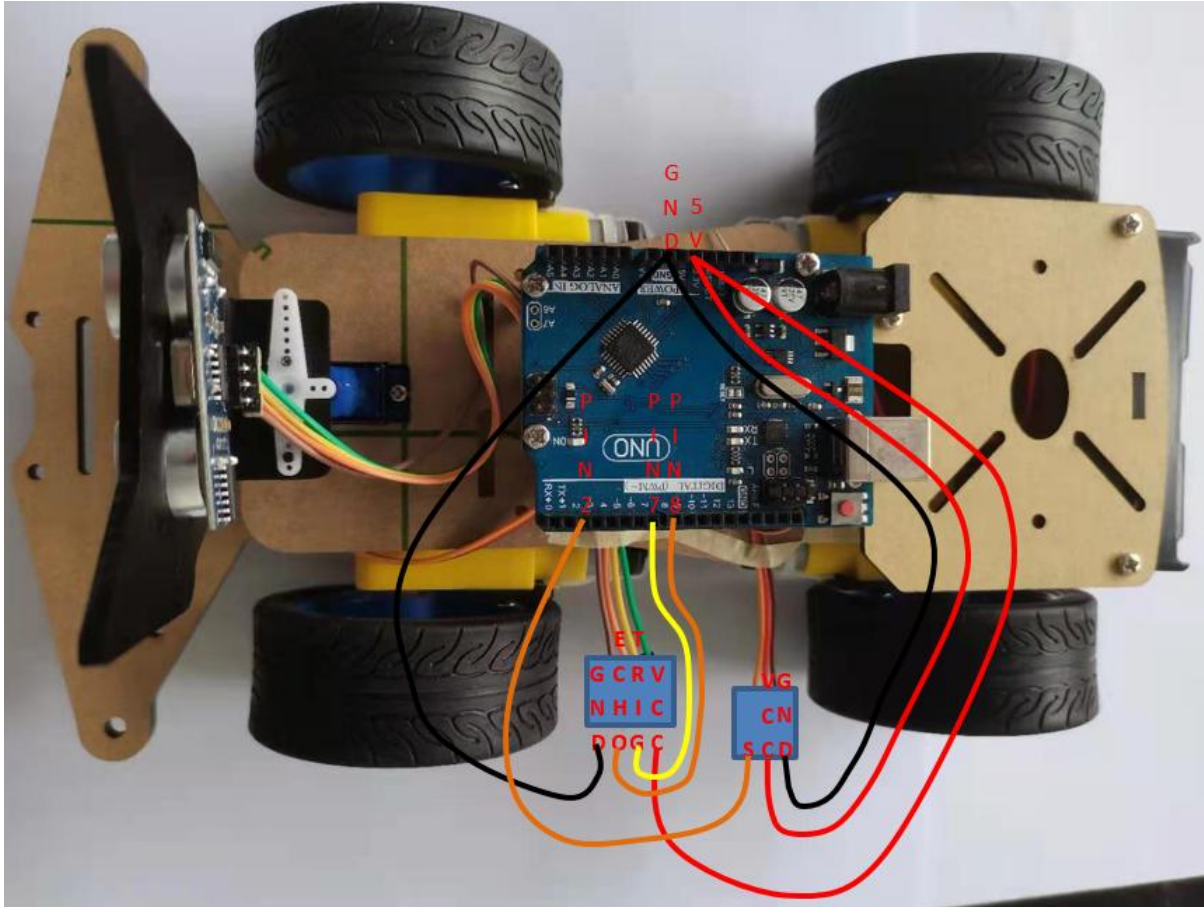


3.3 Install the ultrasonic on the bracket, and then install the bracket on the servo, as shown in the figure below



3.4 circuit connection

Connect the Signal pin of the servo to PIN2 of the UNO board, connect the GND of the servo to the GND of the UNO board, and connect the VCC of the servo to the 5V of the UNO board; connect the VCC of the ultrasonic module to the 5V of the UNO board, and the TRIG is connected to PIN7 of UNO board, ECHO of ultrasonic module is connected to PIN8 of UNO board, GND of ultrasonic module is connected to GND of UNO board. As shown below.



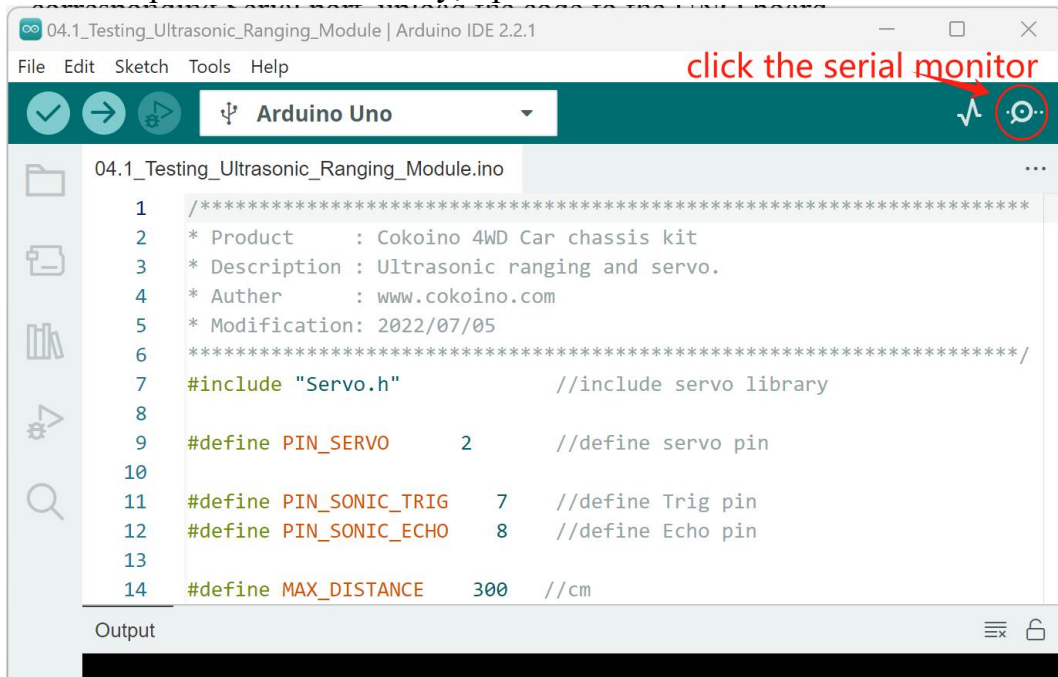
3.5 Circuit inspection

Before powering on, please carefully check whether the connected circuit is open or short-circuited, especially GND and 5V, GND and 3.3V, must not be short-circuited. A short circuit can cause high current in your circuit, create excessive component heat and cause permanent damage to your hardware!

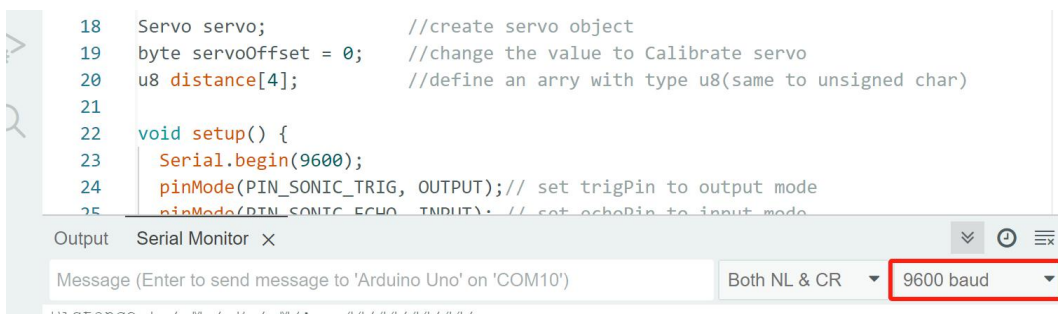
4. Upload code and test

Click "File"---"Open" in the IDE interface, select the code under the path "E:\CKK0011-main\Tutorial\Arduino\Sketches\ 04.1 _Testing _Ultrasonic Ranging Module". After the code is compiled successfully, connect the UNO board to the computer through the USB cable, and select the corresponding Serial port, upload the code to the UNO board.

After the code is uploaded successfully, open the serial monitor.



Set the baud rate to 9600



Then you will see servo sweep among 45°, 90°, 135°. And ultrasonic module detects different distance at different angles. All the distance values are printed below:

```
DistM / R / M2: 69/70/70/70/
Distance L / M / R / M2: 182/101/69/50/
Distance L / M / R / M2: 182/50/70/51/
Distance L / M / R / M2: 33/50/69/50/
Distance L / M / R / M2: 182/102/69/101/
Distance L / M / R / M2: 182/51/70/51/
Distance L / M / R / M2: 183/51/69/100/
Distance L / M / R / M2: 182/102/70/100/
Distance L / M / R / M2: 182/102/69/103/
```

The code is below:

```
1  #include "Servo.h" //include servo library
2
3  #define PIN_SERVO 2 //define servo pin
4
5  #define PIN_SONIC_TRIG 7 //define Trig pin
6  #define PIN_SONIC_ECHO 8 //define Echo pin
7
8  #define MAX_DISTANCE 300 //cm
9  #define SONIC_TIMEOUT (MAX_DISTANCE*60) // calculate timeout
10 #define SOUND_VELOCITY 340 //sound Velocity: 340m/s
11
12 Servo servo; //create servo object
13 char servoOffset = 0; //change the value to Calibrate servo
14 u8 distance[4]; //define an array with type u8(same to unsigned char)
15
16 void setup() {
17     Serial.begin(9600);
18     pinMode(PIN_SONIC_TRIG, OUTPUT); // set trigPin to output mode
19     pinMode(PIN_SONIC_ECHO, INPUT); // set echoPin to input mode
20     servo.attach(PIN_SERVO); //initialize servo
21     servo.write(90 + servoOffset); // change servoOffset to Calibrate servo
22 }
23 void loop() {
24     servo.write(45);
25     delay(1000);
26     distance[0] = getSonar(); //get ultrasonic value and save it into distance[0]
27
28     servo.write(90);
29     delay(1000);
30     distance[1] = getSonar();
31
32     servo.write(135);
33     delay(1000);
34     distance[2] = getSonar();
35
36     servo.write(90);
37     delay(1000);
38     distance[3] = getSonar();
39
40     Serial.print("Distance L / M / R / M2: "); //Left/Middle/Right/Middle2
41     for (int i = 0; i < 4; i++) {
42         Serial.print(distance[i]); //print ultrasonic in 45°, 90°, 135°, 90°
43         Serial.print(", ");
```

```

44  }
45  Serial.print("\n"); //next content will be printed in new line
46  }
47
48  float getSonar() {
49      unsigned long pingTime;
50      float distance;
51      digitalWrite(PIN_SONIC_TRIG, HIGH); // make trigPin output high level lasting for 10µs to
52      trigger HC_SR04,
53      delayMicroseconds(10);
54      digitalWrite(PIN_SONIC_TRIG, LOW);
55      pingTime = pulseIn(PIN_SONIC_ECHO, HIGH, SONIC_TIMEOUT); // Wait HC-SR04 returning to the
56      high level and measure out this waiting time
57      if (pingTime != 0)
58          distance = (float)pingTime * SOUND_VELOCITY / 2 / 10000; // calculate the distance
59      according to the time
60      else
61          distance = MAX_DISTANCE;
62      return distance; // return the distance value
63  }

```

```
#define MAX_DISTANCE 300 //cm
```

```
#define SONIC_TIMEOUT (MAX_DISTANCE*60) // calculate timeout
```

```
#define SOUND_VELOCITY 340 //sound Velocity: 340m/s
```

First calculate t for max distance.

$340 (t/1000000) / 2 = \text{MAX_DISTANCE} / 100$

$t = 58.8 * \text{MAX_DISTANCE}$ (Unit of t is µs)

We set timeout to $\text{MAX_DISTANCE} * 60$, a little larger, since the maximum distance of ultrasonic module is much larger.

```
pingTime = pulseIn(PIN_SONIC_ECHO, HIGH, SONIC_TIMEOUT); // Wait HC-SR04 returning to the high level
```

If time of high level lasting time of echo is larger **SONIC_TIMEOUT**, it will return 0. Then pingTime=0. than

```
pulseIn(pin, value, timeout)
```

pin: the number of the Arduino pin on which you want to read the

pulse. value: type of pulse to be read: either HIGH or LOW.

timeout (optional): the number of microseconds to wait for the pulse to start; default is one second. For more details, please refer to:

<https://www.arduino.cc/reference/en/language/functions/advanced-io/pulsein/>

Array

An array is a collection of variables that are accessed with an index number.

Define an array in Arduino.

Data type **Array name** **[Number of elements]**

u8 distance[3]; define an array named distance, its data type is u8, which has 3 elements.

The index starts from 0.

u8= **unsigned char** Its range is $0 \sim 2^8$, namely 0~255.

Range of u16 is 0~65535.

For more details about array, please refer to:

<https://www.arduino.cc/reference/en/language/variables/data-types/array/>

5. Make your suggestion and get support

THANK YOU for participating in this learning experience!

If you find errors, omissions or you have suggestions and/or questions about this document, please feel free to contact us: cokoino@outlook.com

We will make every effort to make changes and correct errors as soon as feasibly possible and publish a revised version.

If you want to learn more about Arduino, Raspberry Pi, Smart Cars, Robotics and other interesting products in science and technology, please continue to visit our Amazon Store by search for "**LK COKOINO**" on Amazon. We will continue to launch fun, cost-effective, innovative and exciting products.

Thank you again for choosing Cokoino products.

LK COKOINO