




Chapter 24 Ultrasonic Ranging

In this chapter, we learn a module which use ultrasonic to measure distance, HC SR04.

Project 24.1 Ultrasonic Ranging

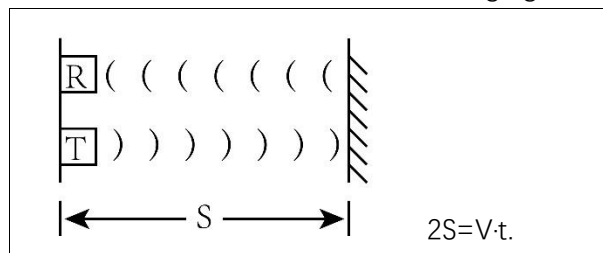
In this project, we use ultrasonic ranging module to measure distance, and print out the data in the terminal.

Component List

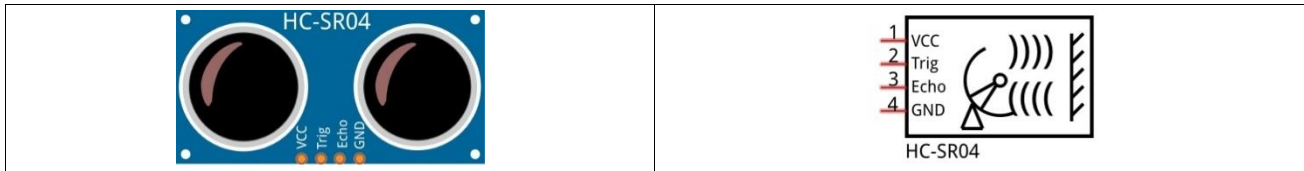
Raspberry Pi (with 40 GPIO) x1 GPIO Expansion Board & Ribbon Cable x1 Breadboard x1	HC SR04 x1 
Jumper Wire x4 	Resistor 1kΩ x1 

Component Knowledge

The Ultrasonic Ranging Module uses the principle that ultrasonic waves will be reflected 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:

VCC	power supply pin
Trig	trigger pin
Echo	Echo pin
GND	GND

Technical specs:

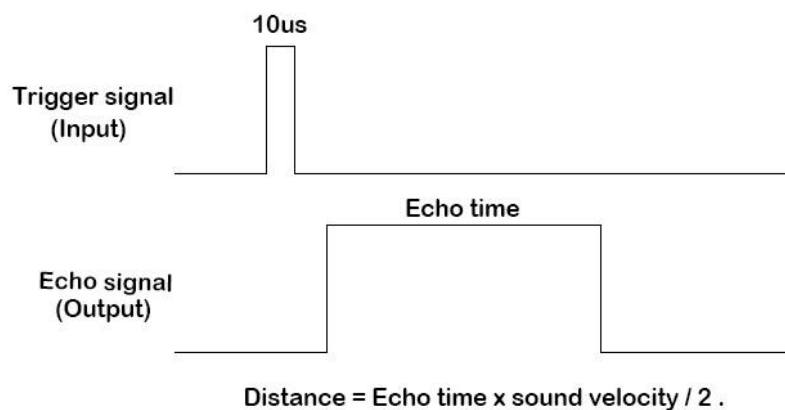
Working voltage: 5V

Working current: 12mA

Minimum measured distance: 2cm

Maximum measured distance: 200cm

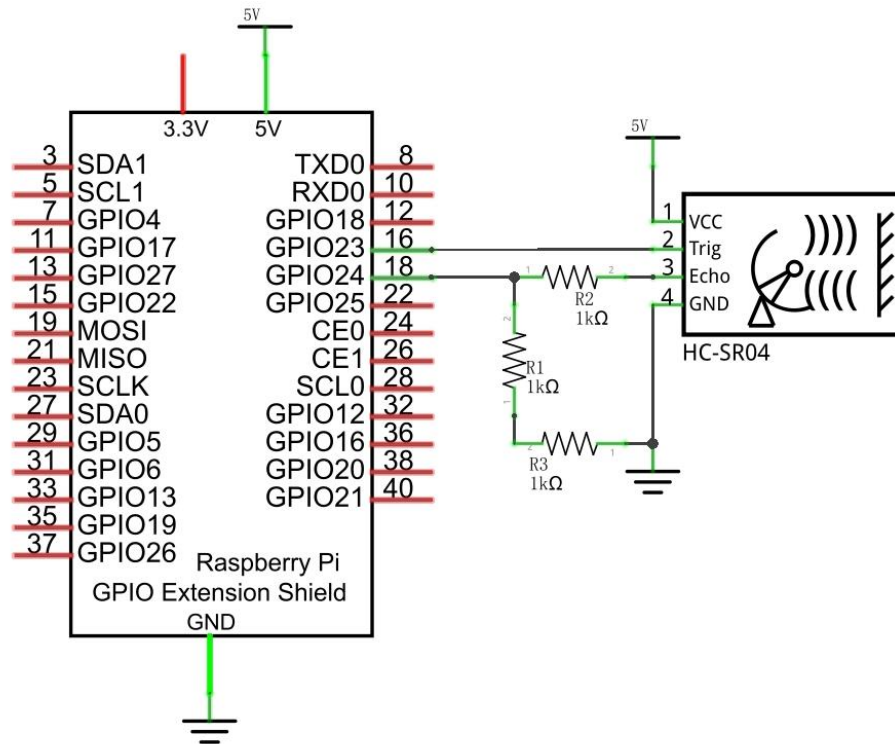
Instructions for Use: output a high-level pulse in Trig pin lasting for least 10uS, the module begins to transmit ultrasonic waves. At the same time, the Echo pin is pulled up. When the module receives the returned ultrasonic waves from encountering an obstacle, the Echo pin will be pulled down. The 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.



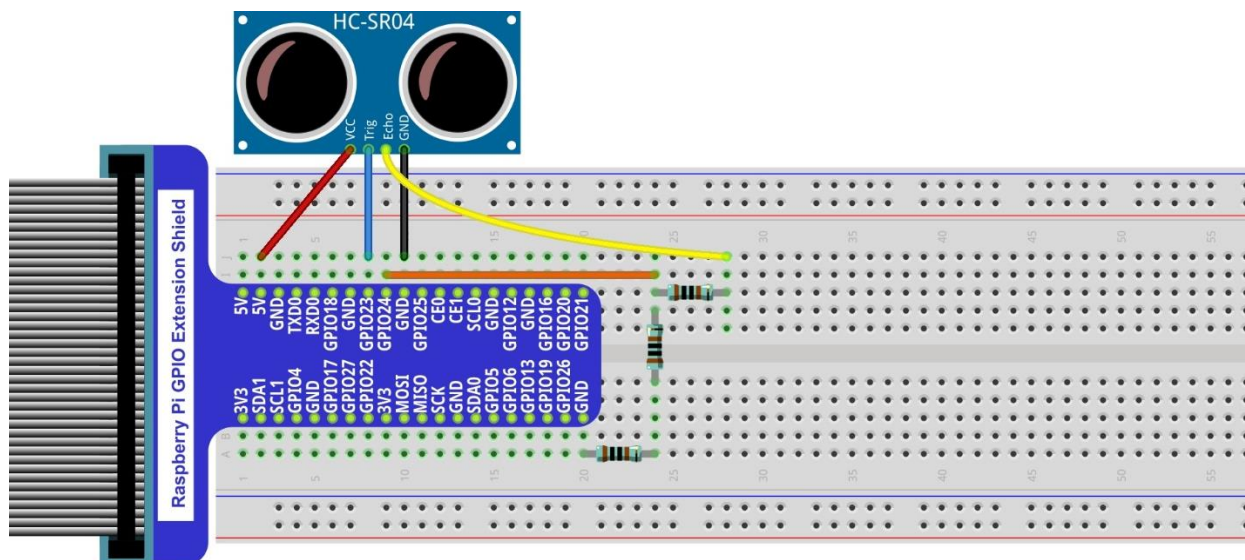
Circuit

Note that the voltage of ultrasonic module is 5V in this circuit.

Schematic diagram



Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com



Python Code 24.1.1 UltrasonicRanging

First, observe the project result, and then learn about the code in detail.

If you have any concerns, please contact us via: support@freenove.com

1. Use cd command to enter 24.1.1_UltrasonicRanging directory of Python code.

```
cd ~/Freenove_Kit/Code/Python_Code/24.1.1_UltrasonicRanging
```

2. Use Python command to execute code "UltrasonicRanging.py".

```
python UltrasonicRanging.py
```

After the program is executed, aim the Ultrasonic Ranging Module's detectors ("eyes") perpendicular to the surface of an object (try using your hand). The distance between the ultrasonic module and the object will be displayed in the terminal. As is shown below:

```
The distance is : 198.75 cm
The distance is : 199.22 cm
The distance is : 198.42 cm
The distance is : 198.74 cm
The distance is : 198.37 cm
The distance is : 198.47 cm
The distance is : 198.41 cm
```

The following is the program code:

```
1  import RPi.GPIO as GPIO
2  import time
3
4  trigPin = 16
5  echoPin = 18
6  MAX_DISTANCE = 220          #define the maximum measured distance(cm)
7  timeOut = MAX_DISTANCE*60   #calculate timeout(μs) according to the maximum measured
8  distance
9
10 def pulseIn(pin, level, timeOut): # function pulseIn: obtain pulse time of a pin
11     t0 = time.time()
12     while(GPIO.input(pin) != level):
13         if((time.time() - t0) > timeOut*0.000001):
14             return 0;
15     t0 = time.time()
16     while(GPIO.input(pin) == level):
17         if((time.time() - t0) > timeOut*0.000001):
18             return 0;
19     pulseTime = (time.time() - t0)*1000000
20     return pulseTime
21
22 def getSonar():              #get the measurement results of ultrasonic module,with unit: cm
23     GPIO.output(trigPin,GPIO.HIGH)      #make trigPin send 10us high level
24     time.sleep(0.00001)                #10us
25     GPIO.output(trigPin,GPIO.LOW)
26     pingTime = pulseIn(echoPin,GPIO.HIGH,timeOut)  #read plus time of echoPin
27
```

```

28     distance = pingTime * 340.0 / 2.0 / 10000.0    # the sound speed is 340m/s, and
29     calculate distance (cm)
30     return distance
31
32 def setup():
33     print ('Program is starting...')
34     GPIO.setmode(GPIO.BOARD)        #numbers GPIOs by physical location
35     GPIO.setup(trigPin, GPIO.OUT)   # set trigPin to output mode
36     GPIO.setup(echoPin, GPIO.IN)    # set echoPin to input mode
37
38 def loop():
39     while(True):
40         distance = getSonar()
41         print ("The distance is : %.2f cm"%(distance))
42         time.sleep(1)
43
44 if __name__ == '__main__':        #program start from here
45     setup()
46     try:
47         loop()
48     except KeyboardInterrupt:
49         GPIO.cleanup()

```

First, define the pins and the maximum measurement distance.

```

trigPin = 16
echoPin = 18
MAX_DISTANCE = 220    # define the maximum measured distance 220cm

```

If the module does not return high level, we cannot wait for this forever, so we need to calculate the time period for the maximum distance (200cm). Then **timOut= 2*MAX_DISTANCE/100/340*1000000**. The result of the constant part in this formula is approximately 58.8.

```

timeOut = MAX_DISTANCE*60

```

Subfunction **getSonar()** function is used to start the Ultrasonic Module to begin measurements, and return the measured distance in cm units. In this function, first let trigPin send 10us high level to start the Ultrasonic Module. Then use **pulseIn()** to read the Ultrasonic Module and return the duration time of high level. Finally, the measured distance according to the time is calculated.

```
def getSonar():    #get the measurement results of ultrasonic module, with unit: cm
    GPIO.output(trigPin, GPIO.HIGH)    #make trigPin send 10us high level
    time.sleep(0.00001)    #10us
    GPIO.output(trigPin, GPIO.LOW)
    pingTime = pulseIn(echoPin, GPIO.HIGH, timeOut)    #read plus time of echoPin
    distance = pingTime * 340.0 / 2.0 / 10000.0    # the sound speed is 340m/s, and
    calculate distance
    return distance
```

Finally, in the while loop of main function, get the measurement distance and display it continually.

```
while(True):
    distance = getSonar()
    print ("The distance is : %.2f cm"%(distance))
    time.sleep(1)
```

About function **def pulseIn(pin, level, timeOut):**

def pulseIn(pin,level,timeOut):

Return the length of the pulse (in microseconds) or 0 if no pulse is completed before the timeout (unsigned long).