Ultrasonic Sensor
(000x0000 Article Number)
(TS2181)

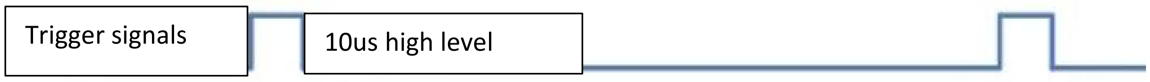
DESIGN THE WORLD

Product Details

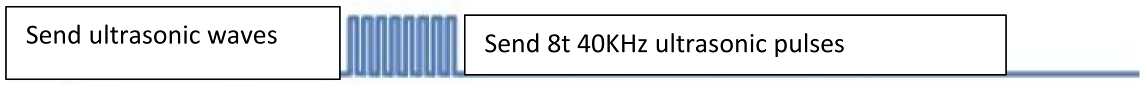
This module has an ultrasonic transmitter and an ultrasonic receiver so you can consider it as an ultrasonic transceiver. Familiar with sonar, when the 40KHz ultrasonic wave generated by the transmitter encounters the object, the sound wave will be emitted back, and the receiver can receive the reflected ultrasonic wave. It is only necessary to calculate the time from the transmission to the reception, and then multiply the speed of the sound in the air(340 m/s) to calculate the distance from the sensor to the object.

Working Principle：

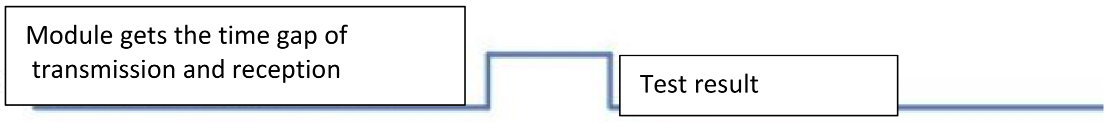
Use IO trigger ranging, at least 10us HIGH level signal; that is, first pull the Trip Low, then give a HIGH level signal of 10us.



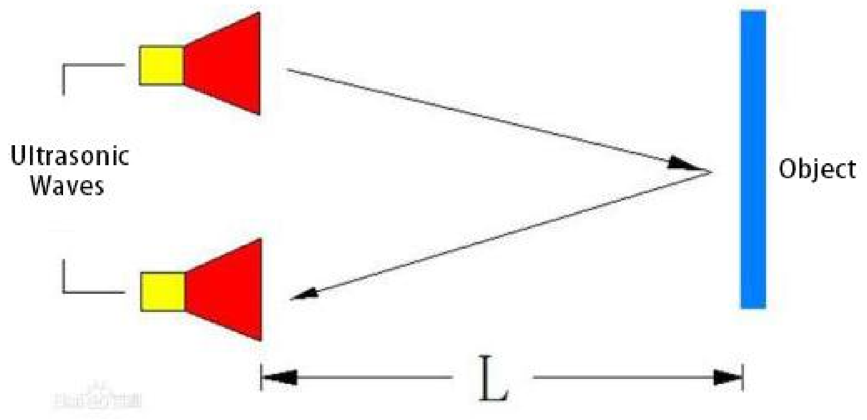
The module automatically sends eight square waves of 40khz to detect whether there is a signal returning back;



There is a back signal which outputs a High level through the IO, and the duration period of high level is the time of ultrasonic wave from emission to return.



Tested distance = (High level time * speed of sound (340M/S))/2



Features and Benefits

- Compatible with RJ11 6P6C OKdo TelePort control boards and expansion shields.
- Integrated ultrasonic transmitter, receiver and corresponding control circuit, easy to use.
- Very common in robotics projects and very useful for automation, interactive art and motion sensing.
- With four fixing holes, it can be easily fixed on other devices such as servo plastic platform.Compatible with all 40-pin Raspberry Pi motherboards.

Technical Specifications

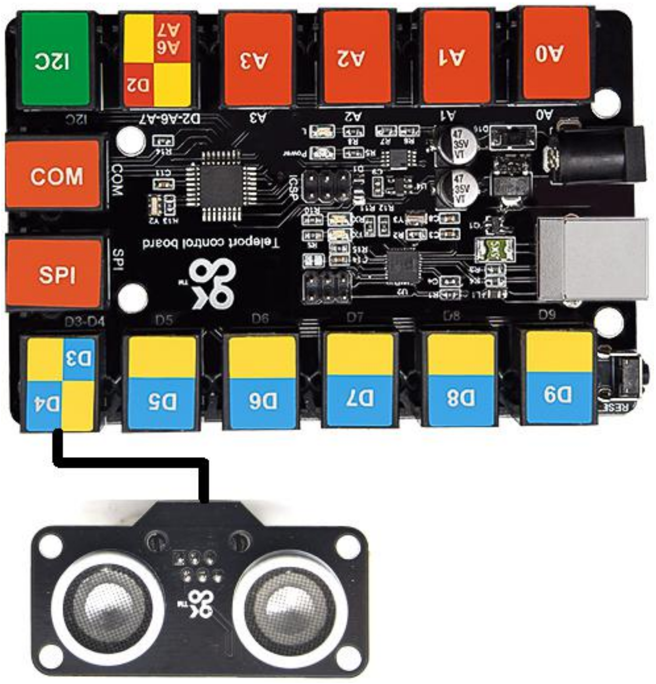
Sensor type	Digital output and input
Sensor type	I2C
Operating Voltage	DC 5V
Operating Current	15mA
Operating Frequency	40KHz
Max Range	3--5m
Min Range	2cm
Measuring Angle	15 degree
Trigger Input Signal	10μS TTL pulse
Dimensions	49mm*26mm*28mm
Weight	11.3g

Applications

- Distance measurement
- Ultrasonic detector
- Proximity alarm
- Smart car

This module is compatible with the TS2180-Raspberry Pi shield, the TS2179-Micro:bit shield and the TS2178-TelePort main board.

➢ **<u>Arduino Application</u>**



This module is compatible with the TS2178 TelePort control board.

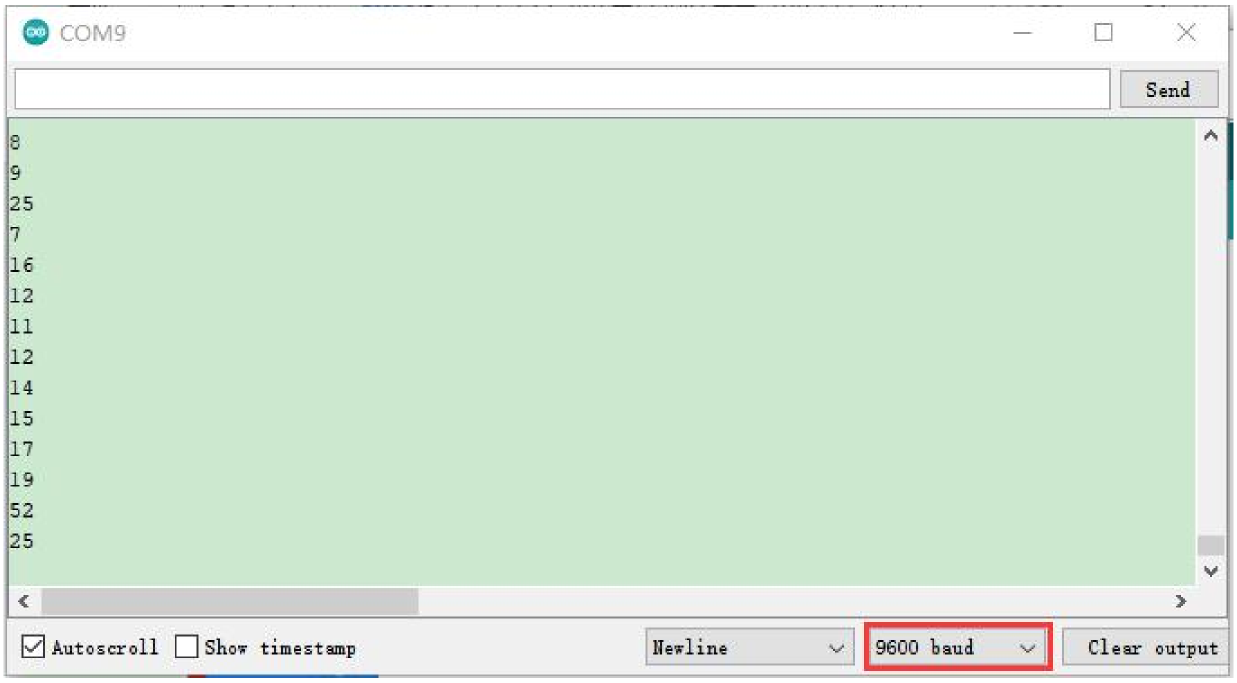
Test Code

```
#define echoPin 4 // Echo Pin
#define trigPin 3// Trigger Pin
#define LEDPin 13 // Onboard LED

int maximumRange = 200; // Maximum range needed
int minimumRange = 0; // Minimum range needed
long duration, distance; // Duration used to calculate distance

void setup() {
 Serial.begin (9600);
 pinMode(trigPin, OUTPUT);
 pinMode(echoPin, INPUT);
 pinMode(LEDPin, OUTPUT); // Use LED indicator (if required)
}

void loop() {
/* The following trigPin/echoPin cycle is used to determine the
 distance of the nearest object by bouncing soundwaves off of it. */
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);

 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
 duration = pulseIn(echoPin, HIGH);
  //Calculate the distance (in cm) based on the speed of sound.
 distance = duration/58.2;
  if (distance >= maximumRange || distance <= minimumRange){
```

```
/* Send a negative number to computer and Turn LED ON
to indicate "out of range" */
Serial.println("-1");
digitalWrite(LEDPin, HIGH);
}
else {
/* Send the distance to the computer using Serial protocol, and
turn LED OFF to indicate successful reading. */
Serial.println(distance);
digitalWrite(LEDPin, LOW);
}

//Delay 50ms before next reading.
delay(50);
}
```

Test Result

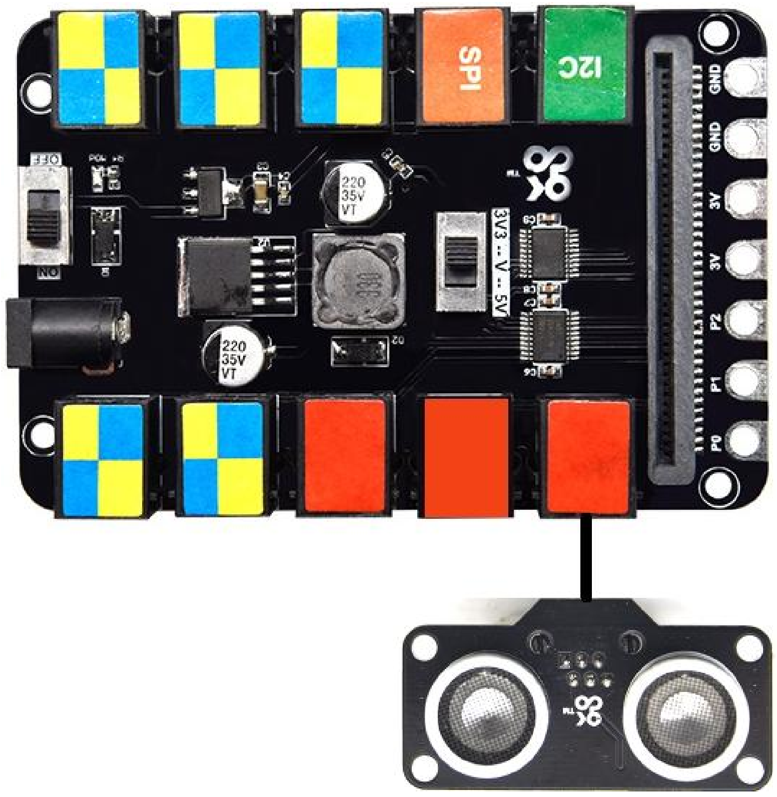
Hook components up, upload test code, power it up, open serial monitor and set baud rate to 9600. You can view the distance value between the ultrasonic sensor and the obstacle, as shown below:



```
8
9
25
7
16
12
11
12
14
15
17
19
52
25
```

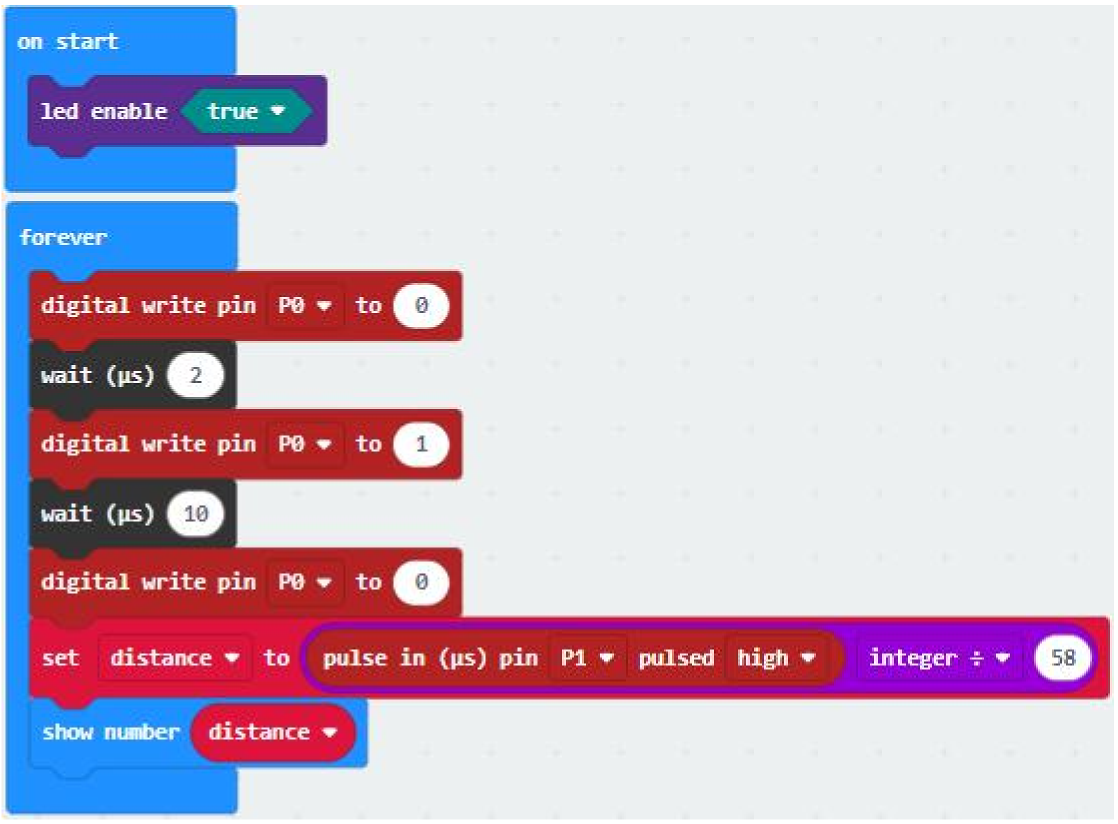
If you want to know more details about Arduino and the TelePort control board, you can refer to TS2178.

➢ **Micro:bit Application**



It is compatible with the Micro:bit board and the TS2179 Micro:bit expansion board.

Test Code



```
on start
    led enable true
forever
    digital write pin P0 to 0
    wait (µs) 2
    digital write pin P0 to 1
    wait (µs) 10
    digital write pin P0 to 0
    set distance to pulse in (µs) pin P1 pulsed high integer ÷ 58
    show number distance
```

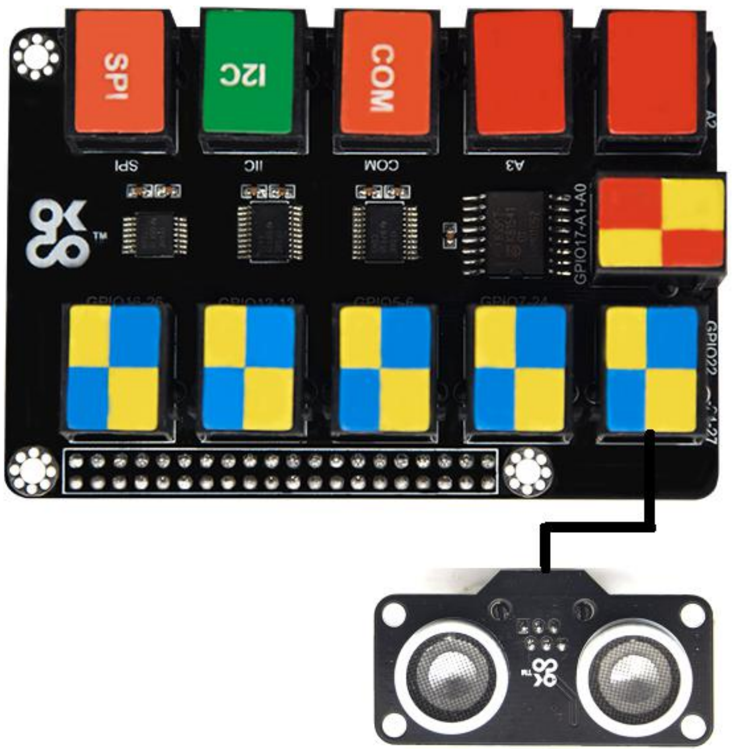
..............①Run the "on start" block to boot the program ....②Open the LED matrix of the Micro:bit ....③The program is run circularly under the command of "forever" block ....④Set P0 to low level(0) ....⑤delay in 2ms ....⑥Set P0 to high level(1) ....⑦delay in 10ms ....⑧Set P0 to low level(0) ....⑨Distance=the duration time of pulse÷58 ....⑩The Micro:bit shows the distance value between the ultrasonic sensor and the obstacle	

Test Result

Wire up, insert the Micro:bit into the shield, turn DIP switch to 5V, slide the another switch to ON end and power up (above 5V). Then the Micro:bit will show the distance value between the ultrasonic sensor and the obstacle.

If you want to know more details about the Micro:bit board and Micro:bit shield, you can refer to TS2179.

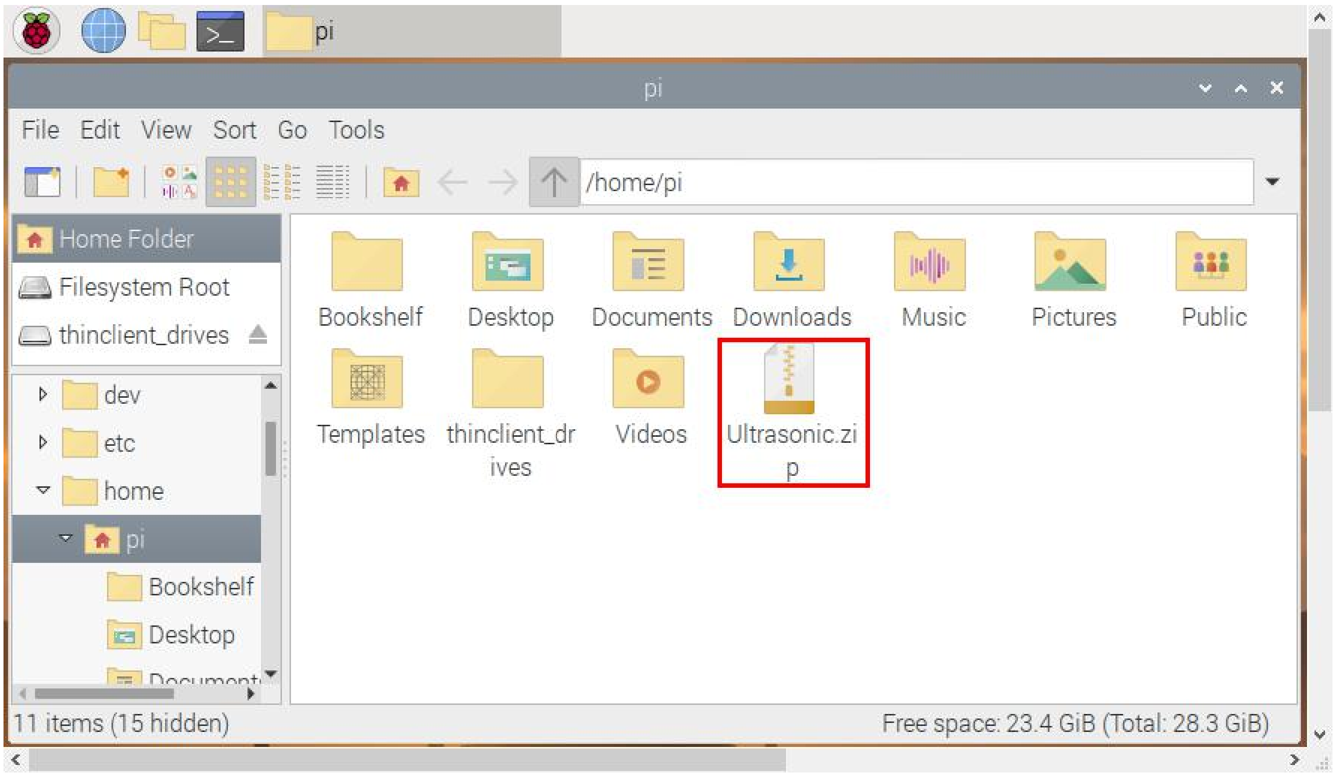
➤ **Raspberry Pi Application**

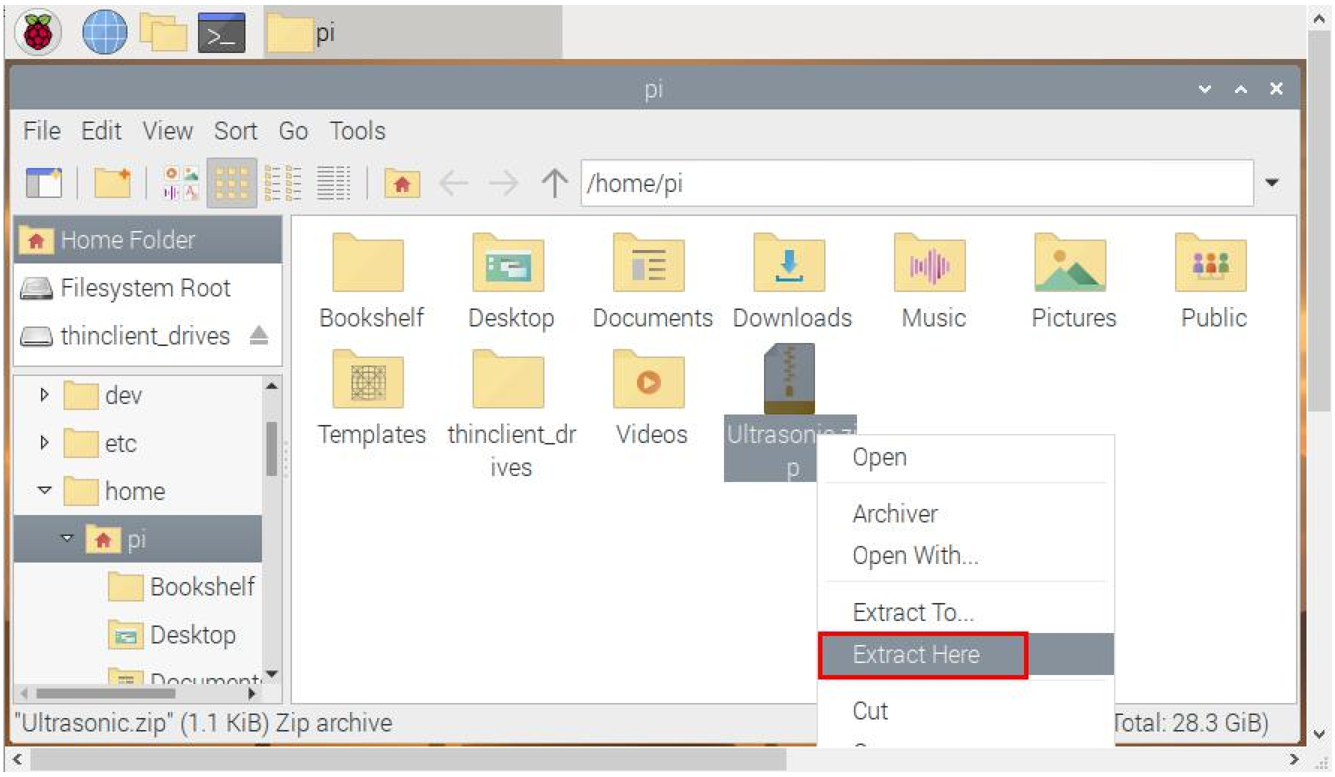


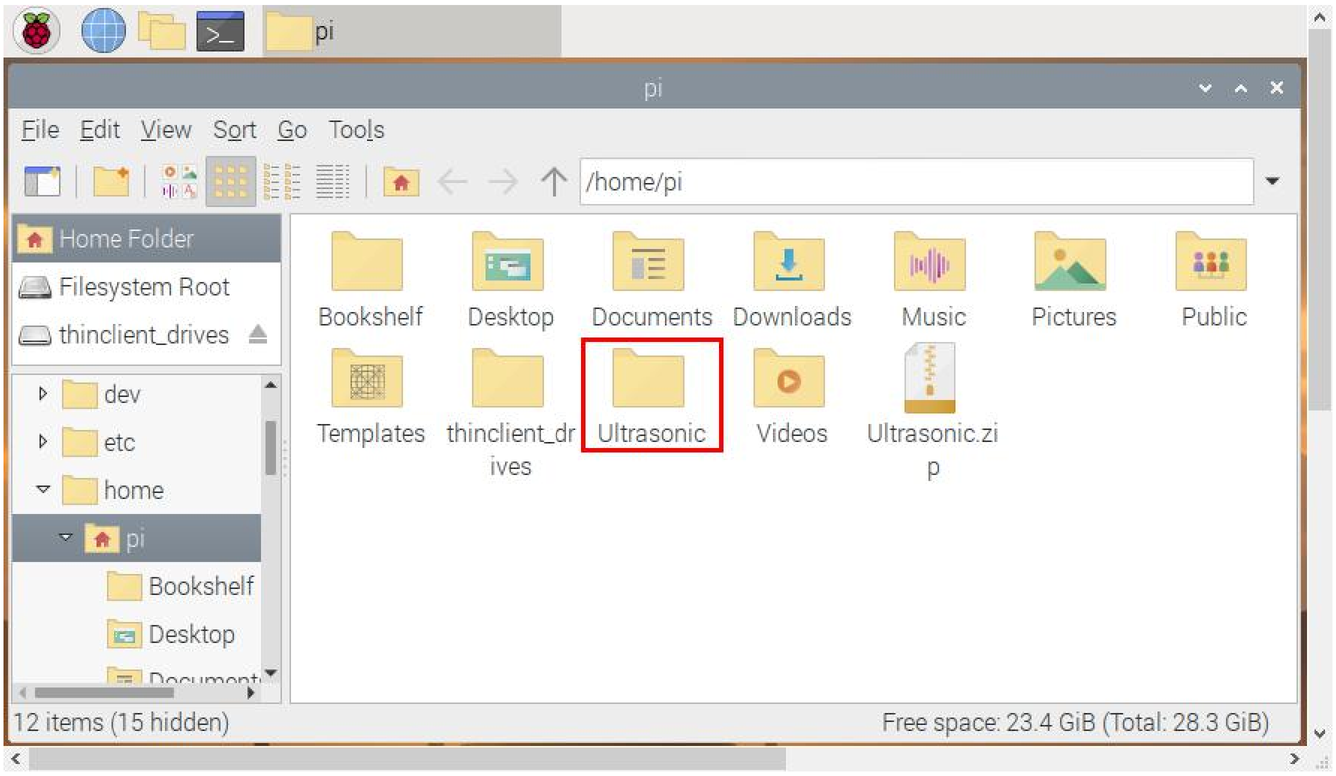
This module is compatible with the Raspberry Pi board and the TS2180 Raspberry Pi shield.

Copy the test code to Raspberry Pi system to run it

(1) Save the test code in the **pi** folder of Raspberry Pi system. Then place the Ultrasonic.zip file we provide in the **pi** folder, right-click and click **Extract Here.** As shown below:







(2) Compile and run test code：

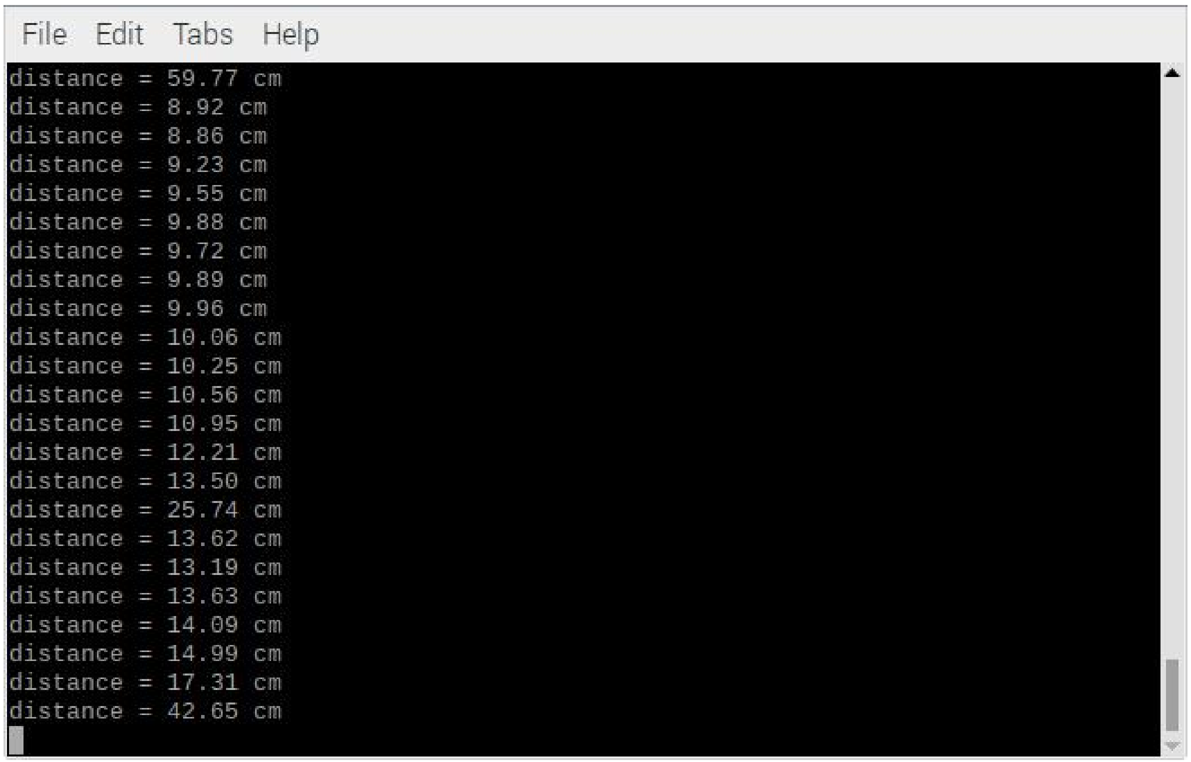
Input the following code and press“Enter”

```
cd /home/pi/Ultrasonic
gcc Ultrasonic.c -o Ultrasonic -lwiringPi
sudo ./Ultrasonic
```

(3) Test Result：

Insert the shield into the Raspberry Pi board. After programming finishes, then the terminal will display the distance detected by the ultrasonic sensor.

Note: press Ctrl + C to exit code running



```
distance = 59.77 cm
distance = 8.92 cm
distance = 8.86 cm
distance = 9.23 cm
distance = 9.55 cm
distance = 9.88 cm
distance = 9.72 cm
distance = 9.89 cm
distance = 9.96 cm
distance = 10.06 cm
distance = 10.25 cm
distance = 10.56 cm
distance = 10.95 cm
distance = 12.21 cm
distance = 13.50 cm
distance = 25.74 cm
distance = 13.62 cm
distance = 13.19 cm
distance = 13.63 cm
distance = 14.09 cm
distance = 14.99 cm
distance = 17.31 cm
distance = 42.65 cm
```

Test Code

File name: Ultrasonic.c

```
#include <wiringPi.h>
#include <stdio.h>
#include <sys/time.h>   //Import the time system header file

//define the pin
#define Trig    3   //BCM GPIO 22
#define Echo    2   //BCM GPIO 27

//set pin mode
void ultraInit(void)
{
        pinMode(Echo, INPUT);
        pinMode(Trig, OUTPUT);
}

//Write programs based on sequence diagrams
float disMeasure(void)
{
        struct timeval tv1;  //Create the Timeval structure tv1
        struct timeval tv2;  //Create the Timeval structure tv2
        long start, stop;
        float dis;

        digitalWrite(Trig, LOW);
        delayMicroseconds(2);

        digitalWrite(Trig, HIGH);
        delayMicroseconds(10);
  digitalWrite(Trig, LOW);

        while(!(digitalRead(Echo) == 1));  //Wait for the low level received by the Echo pin to pass
        gettimeofday(&tv1, NULL);   //function gettimeofday, The time it took the system to get here

        while(!(digitalRead(Echo) == 0));  //Wait for the high level received by the Echo pin to pass
        gettimeofday(&tv2, NULL);   //function gettimeofday, The time it took the system to get here

  //Tv1.tv_sec is the seconds obtained, tv1.TV_USec is the subtlety obtained
  //Calculate the first time
        start = tv1.tv_sec * 1000000 + tv1.tv_usec;

        //Calculate the second time
        stop  = tv2.tv_sec * 1000000 + tv2.tv_usec;
```

```
    //stop - start , the time difference is the high level time acquired by the echo pin
    //34000cm/s, speed of sound
    //Calculate the distance measured(cm)
        dis = (float)(stop - start) / 1000000 * 34000 / 2;

        return dis;
}

int main(void)
{
        float dis;

        if(wiringPiSetup() == -1){ //when initialize wiring failed,print messageto screen
                printf("setup wiringPi failed !");
                return 1;
        }

        ultraInit();

        while(1){
                dis = disMeasure();
                printf("distance = %0.2f cm\n",dis);
                delay(100);
        }

        return 0;
}
```

If you want to know how to utilize Raspberry Pi and the Raspberry Pi shield, you can refer to TS2180.

END