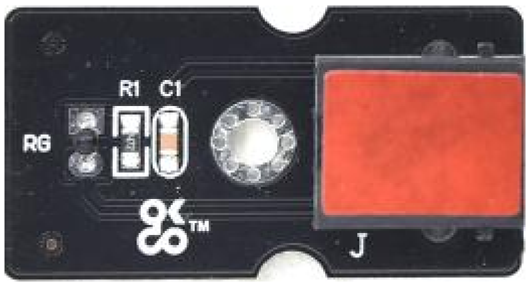
Thermistor Sensor (000x0000 Article Number) (TS2132)

DESIGN THE WORLD

Product Details

This is the TelePort thermistor sensor which adopts the thermistor component. When the the heat-sensitive material is surrounded by heat radiation, it will absorb radiant heat, increase temperature and make the resistance of the material change.



Features and Benefits

- Compatible with RJ11 6P6C OKdo TelePort Control boards and expansion shields.
- Analog temperature sensor, voltage varies according to temperature.

Technical Specifications

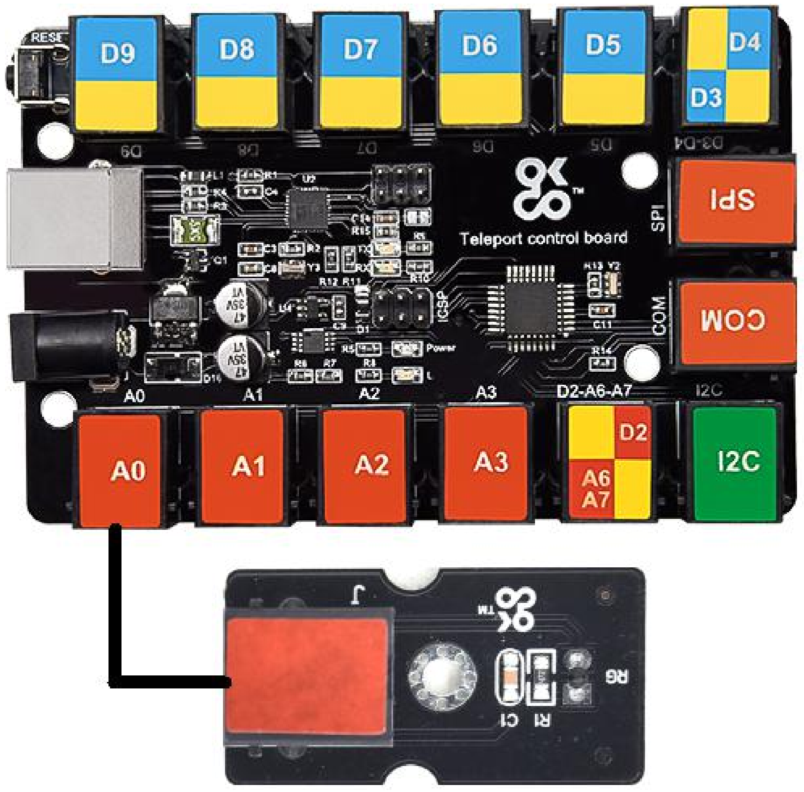
Sensor type	Analog input
Working voltage	3.3V-5V
Temperature range	-55℃ ~ 315℃
Dimensions	38mm*20mm*18mm
Weight	4.2g

Applications

- Thermal thermometer
- Temperature meter

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

➢ **Arduino Application**



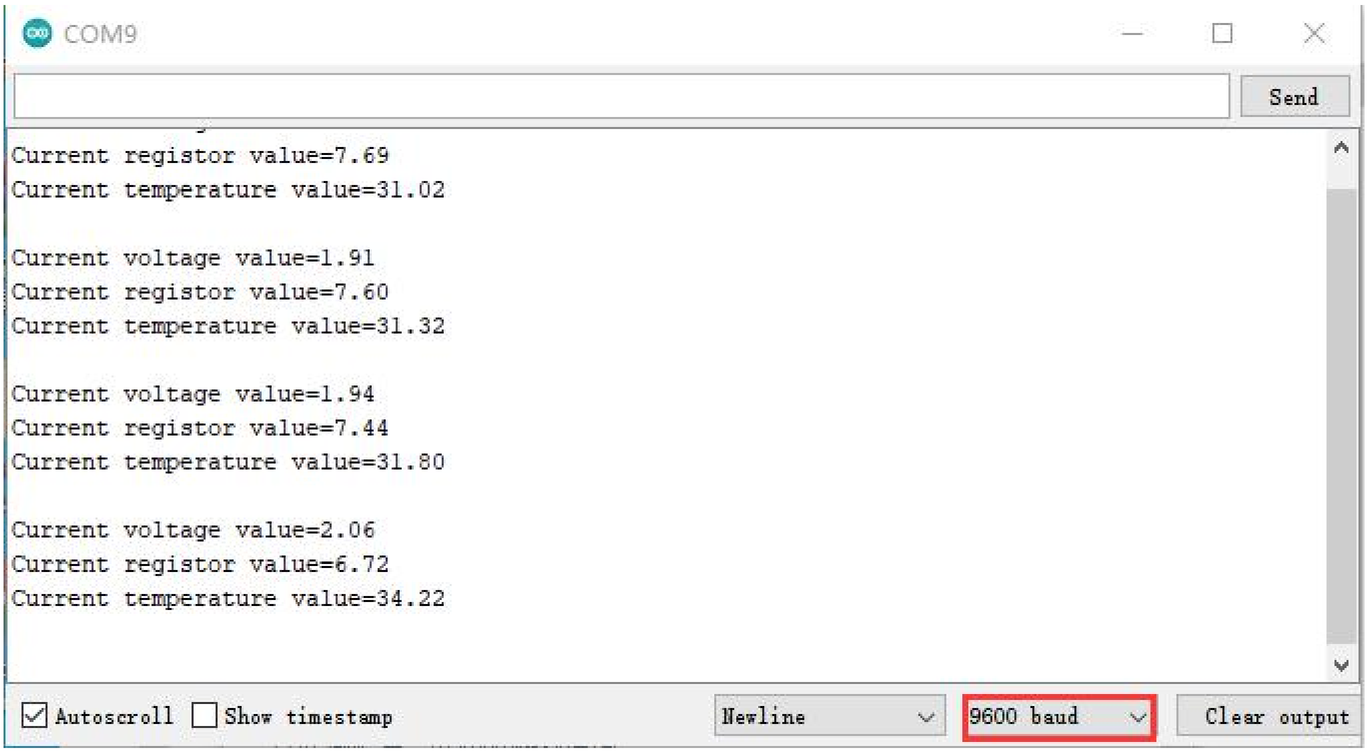
This module is compatible with the TS2178 TelePort control board.

Test Code

```
#include<math.h>

const float voltagePower=5.0;
const float Rs=4.7;//The sampling resistance is 4.7 KHM
const int B=3950;
const double T1=273.15+25;//normal temperature
const double R1=10;//The resistance at room temperature is KΩ
void setup() {
  Serial.begin(9600);
}
void loop() {
  //get the voltage at A0
  double digitalValue=analogRead(0);
  double voltageValue=(digitalValue/1023)*5;
  Serial.print("Current voltage value=");
  Serial.println(voltageValue);

  //The resistance of thermistor is obtained by partial pressure ratio
  double Rt=((voltagePower-voltageValue)*Rs)/voltageValue;
  Serial.print("Current registor value=");
  Serial.println(Rt);
```

```
 //Convert to get the temperature value
 Serial.print("Current temperature value=");
 Serial.println(((T1*B)/(B+T1*log(Rt/R1)))-273.15);//
 Serial.println();

 //Output every 3 seconds. Change the frequency here
 delay(3000);
}
```

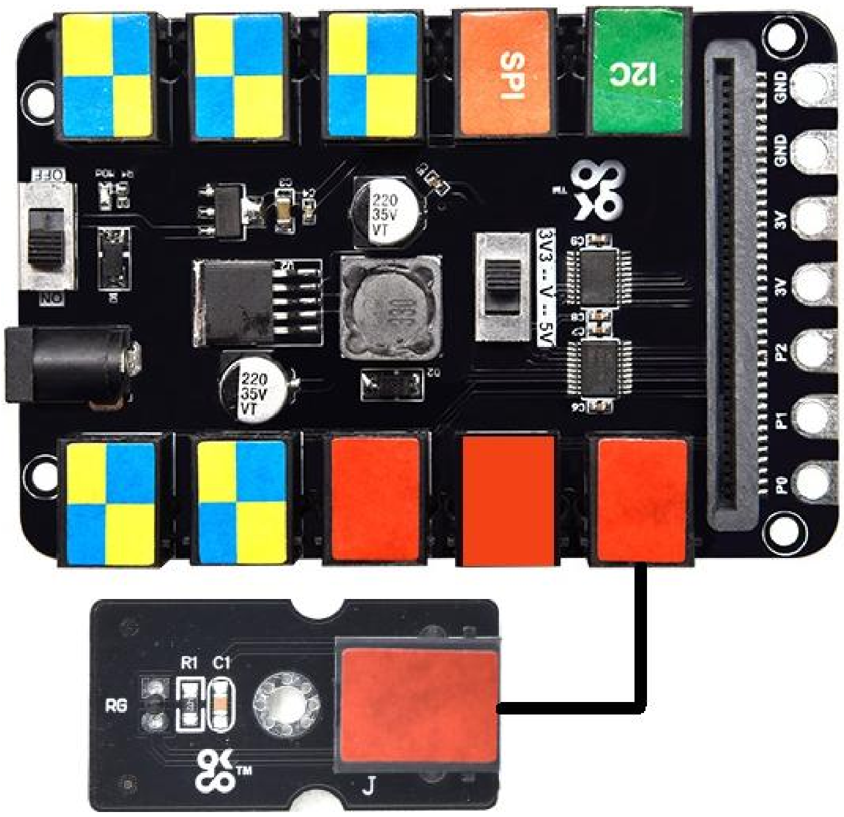
Test Result
Wire up, upload test code, power it up, open serial monitor and set baud rate to 9600. Then you can get the voltage value at Pin A0 and obtain the resistance and temperature value via partial pressure. As shown below;



```
Current registor value=7.69
Current temperature value=31.02

Current voltage value=1.91
Current registor value=7.60
Current temperature value=31.32

Current voltage value=1.94
Current registor value=7.44
Current temperature value=31.80

Current voltage value=2.06
Current registor value=6.72
Current temperature value=34.22
```

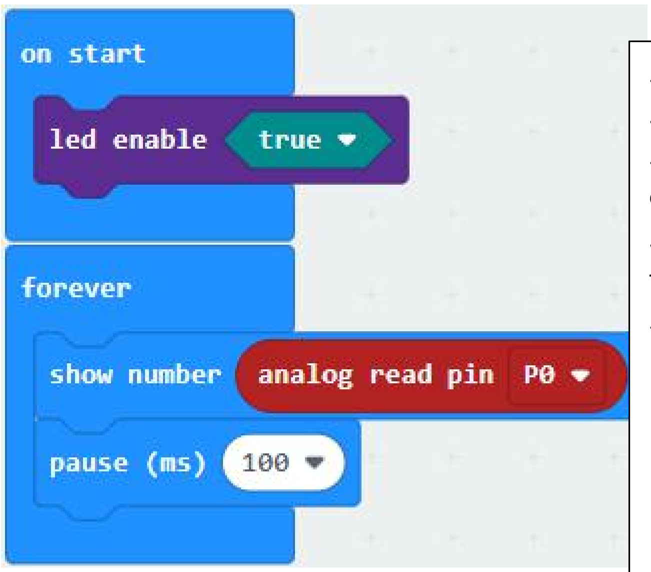
If you want to know more details about Arduino and the TelePort control board, you can refer to TS2178.

➢ <u>Micro:bit Application</u>



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

Test Code



```
on start
  led enable true
forever
  show number analog read pin P0
  pause (ms) 100
```

......................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

......................④Micro:bit shows the analog values detected by the thermistor.

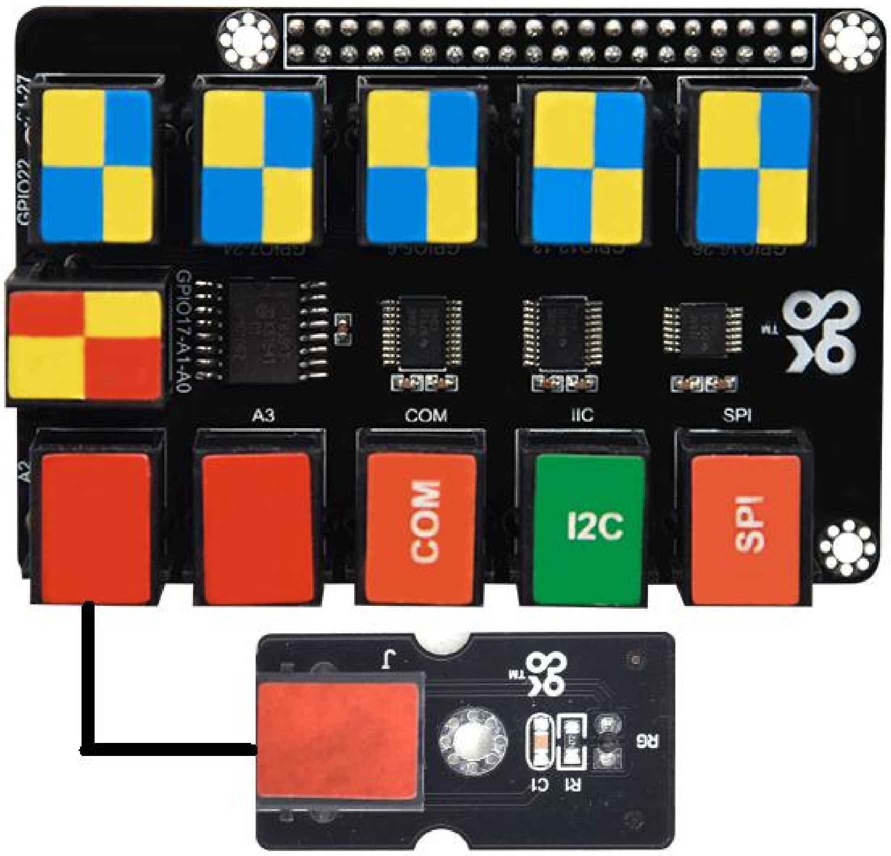
......................⑤delay in 100ms

Test Result

Wire up, insert the Micro:bit V2.0 into the shield, turn DIP switch to 3V3, upload test code and power it up. Then Micro:bit will show the analog values detected by the thermistor.

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

➢ **Raspberry Pi Application**



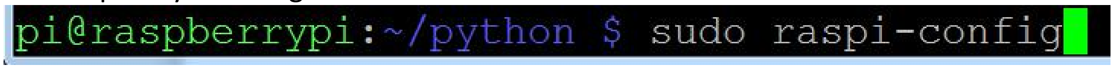
This module is compatible with the Raspberry Pi board and the TS2180 Raspberry Pi shield.

PCF8591 A/D Conversion：

The Raspberry Pi itself does not have AD/DA function; therefore an expansion board with this function is required when connected to external analog sensors. And here we use a PCF8591 A/D converter with I2C communication.

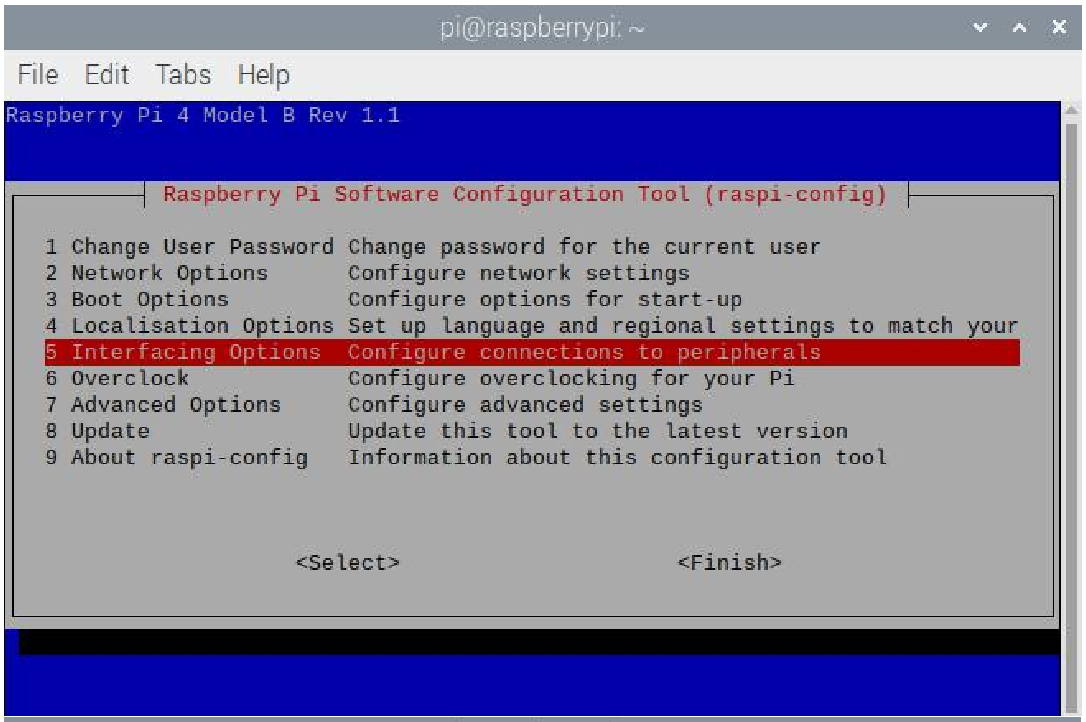
Enable the I2C communication function of the Raspberry Pi as follows:

a. Raspberry Pi does not enable the I2C function by default. Enter sudo raspi-config in the terminal to enter the Raspberry Pi configuration interface.

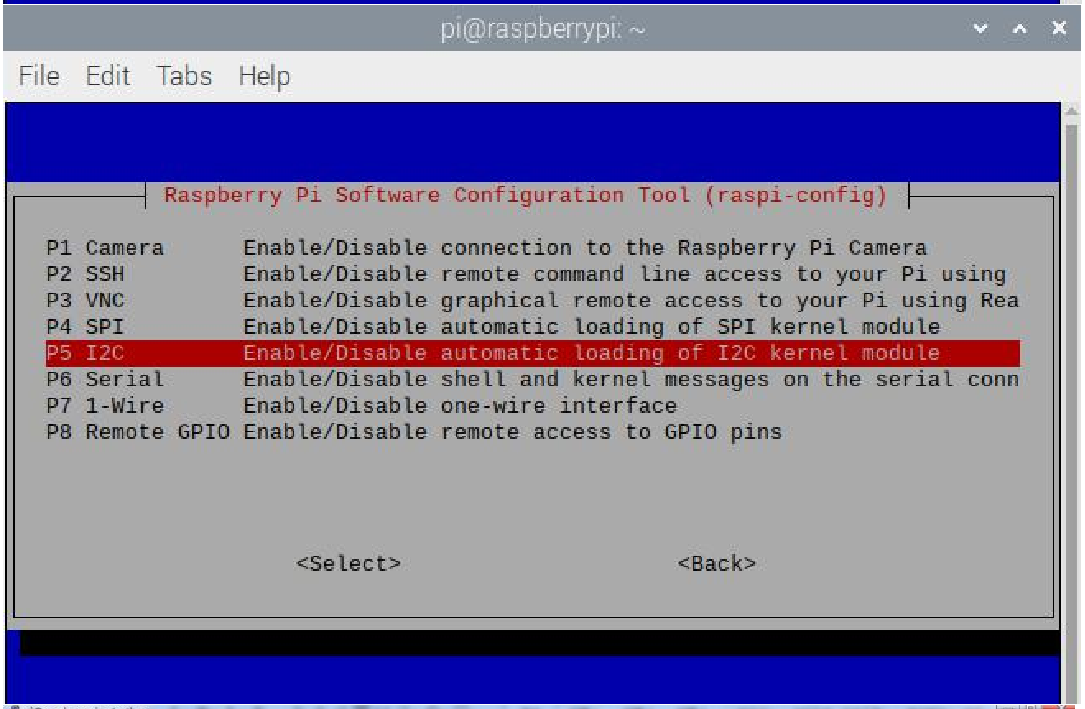


```
pi@raspberrypi:~/python $ sudo raspi-config
```

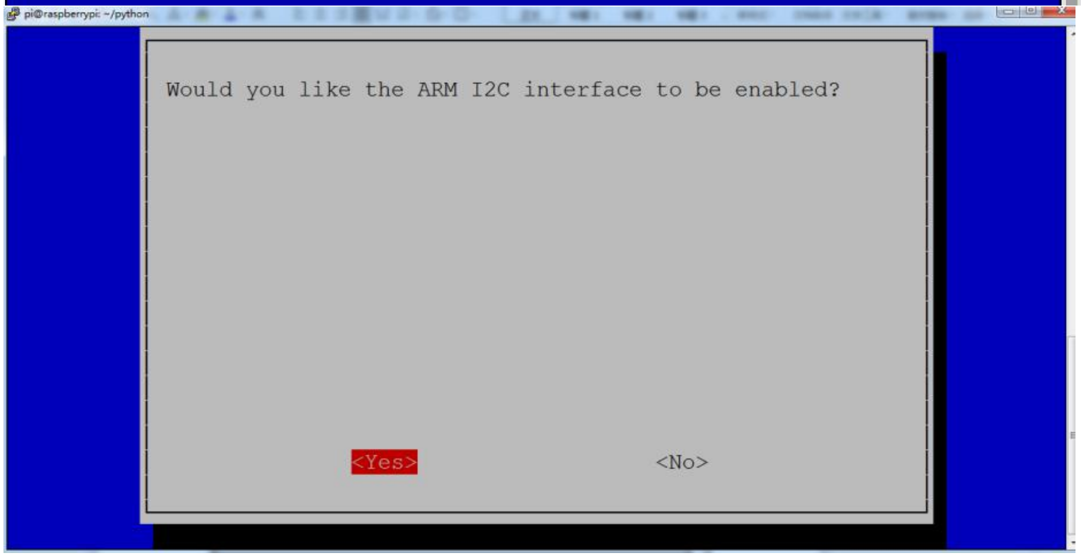
b. Follow the below instructions to enable the I2C function of Raspberry Pi:(press ←,↑,↓,→ then“Enter”)



```
Raspberry Pi 4 Model B Rev 1.1

Raspberry Pi Software Configuration Tool (raspi-config)

 1 Change User Password Change password for the current user
 2 Network Options      Configure network settings
 3 Boot Options         Configure options for start-up
 4 Localisation Options Set up language and regional settings to match your
 5 Interfacing Options  Configure connections to peripherals
 6 Overclock            Configure overclocking for your Pi
 7 Advanced Options     Configure advanced settings
 8 Update               Update this tool to the latest version
 9 About raspi-config   Information about this configuration tool

              <Select>                    <Finish>
```

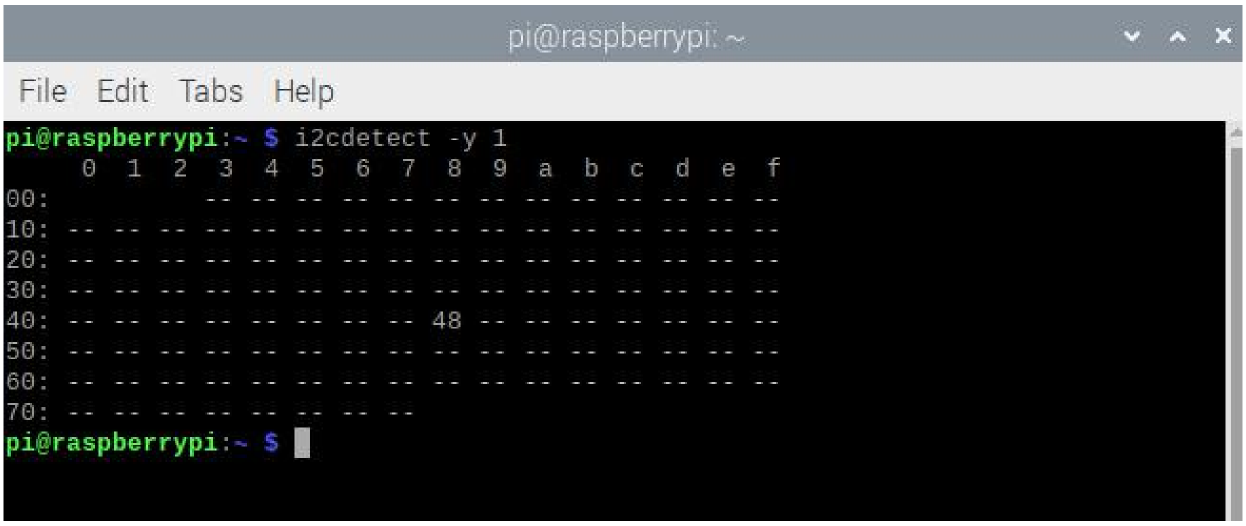


```
Raspberry Pi Software Configuration Tool (raspi-config)

 P1 Camera      Enable/Disable connection to the Raspberry Pi Camera
 P2 SSH         Enable/Disable remote command line access to your Pi using
 P3 VNC         Enable/Disable graphical remote access to your Pi using Rea
 P4 SPI         Enable/Disable automatic loading of SPI kernel module
 P5 I2C         Enable/Disable automatic loading of I2C kernel module
 P6 Serial      Enable/Disable shell and kernel messages on the serial conn
 P7 1-Wire      Enable/Disable one-wire interface
 P8 Remote GPIO Enable/Disable remote access to GPIO pins

              <Select>                    <Back>
```



```
Would you like the ARM I2C interface to be enabled?

         <Yes>                  <No>
```

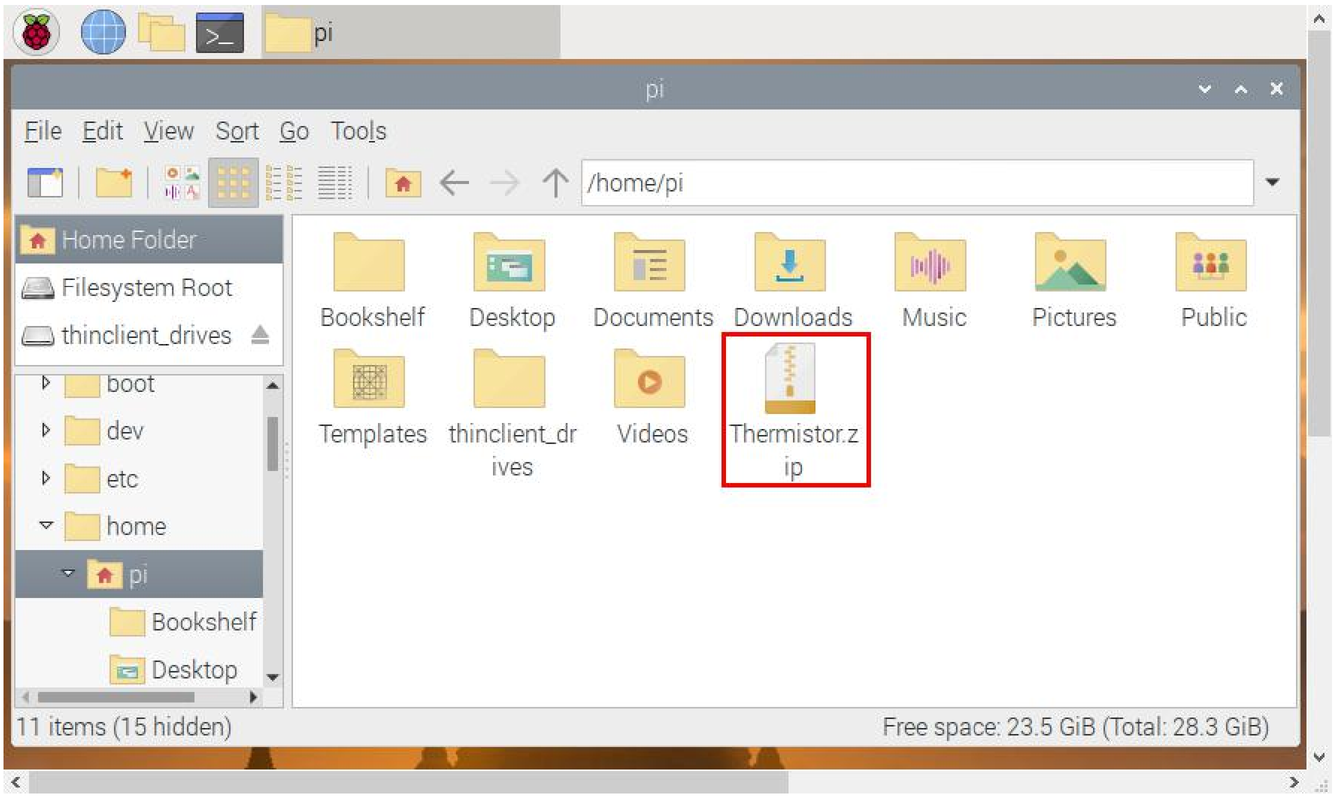
Check the address of the I2C module (PCF8591) connected to the Raspberry Pi, enter the command: i2cdetect -y 1, and then press **Enter**.
From below picture, it is known that the I2C address of PCF8591 is 0x48 .

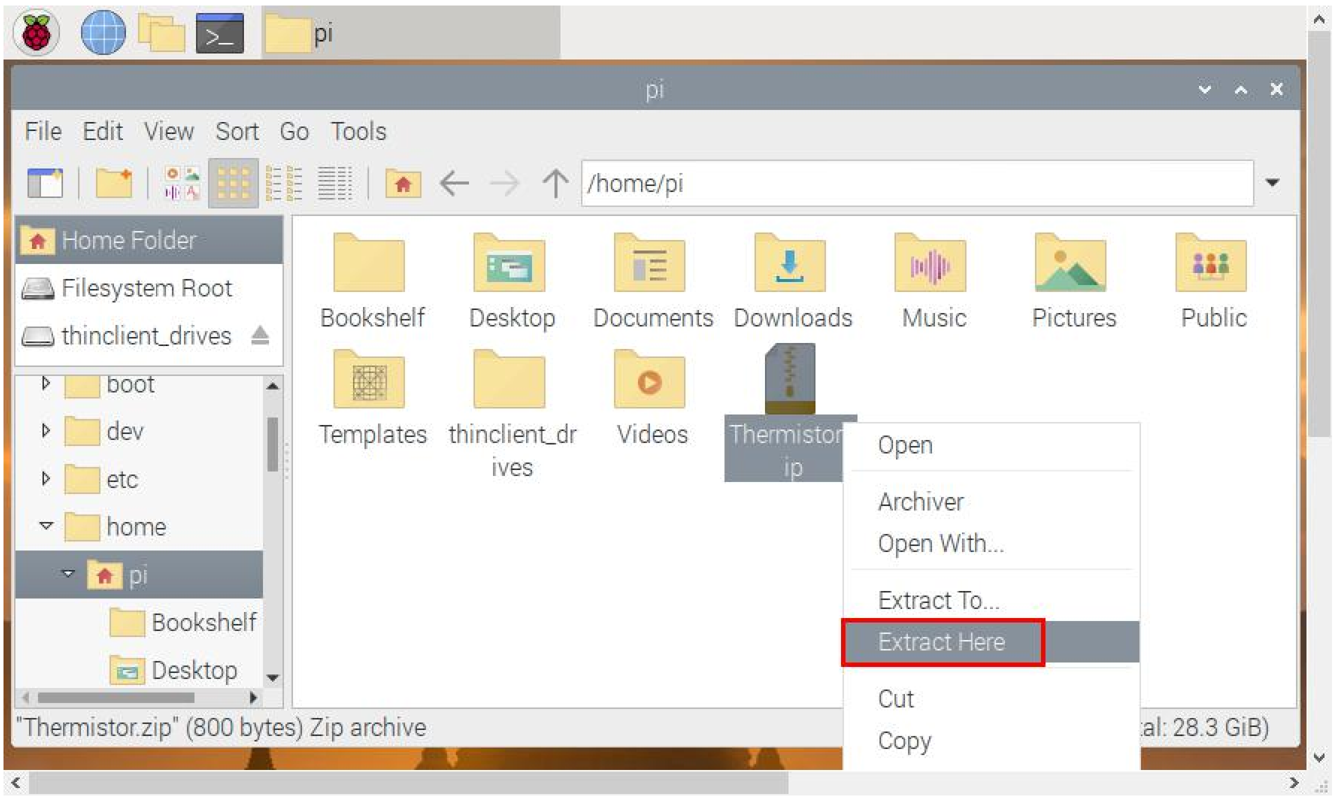


```
pi@raspberrypi:~ $ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- 48 -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
pi@raspberrypi:~ $
```

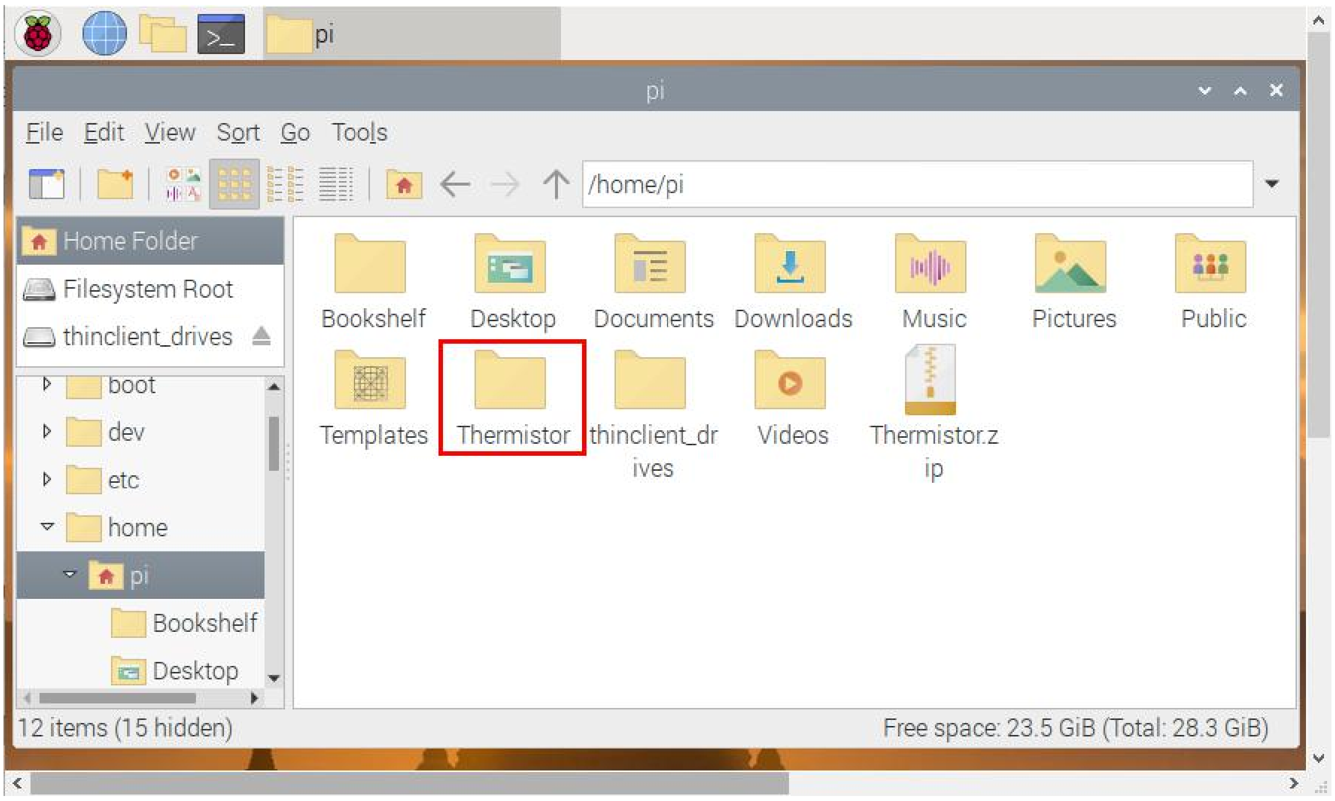
Copy the test code to Raspberry Pi system to run it

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





"Thermistor.zip" (800 bytes) Zip archive



12 items (15 hidden) Free space: 23.5 GiB (Total: 28.3 GiB)

(2) Compile and run test code：

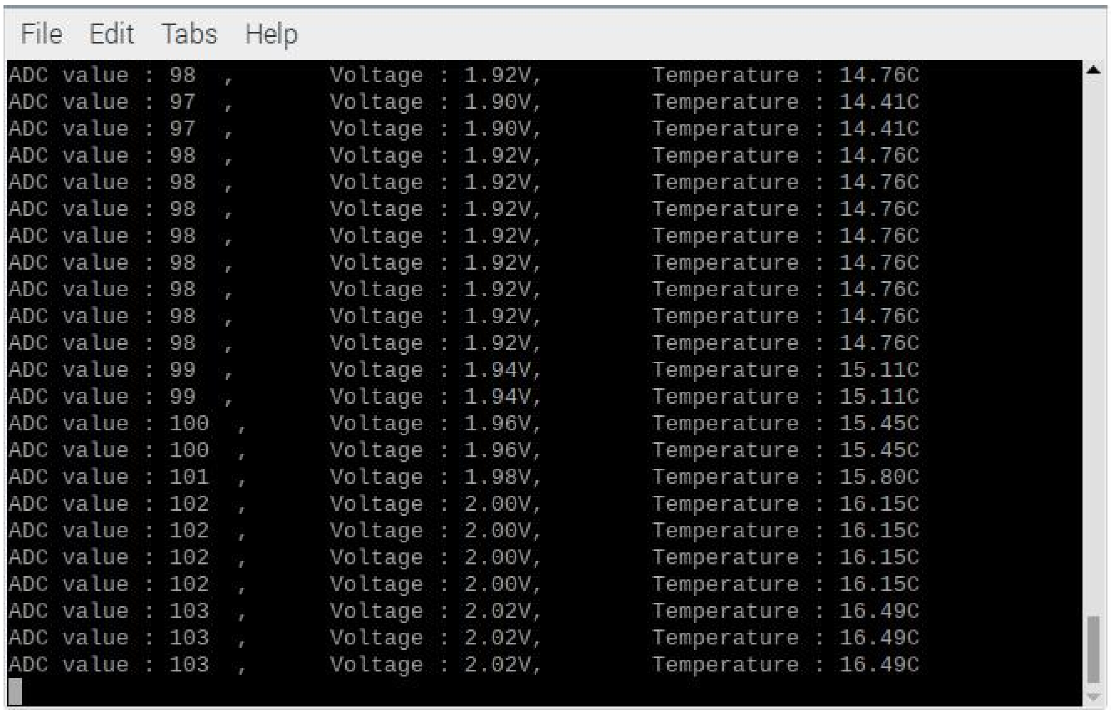
Input the following code and press“Enter”

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

(3) Test Result：

Insert the shield into the Raspberry Pi board. After programming finishes, then the terminal will show ADC value, voltage value and temperature value.

Note: press Ctrl + C to exit code running



```
ADC value : 98  ,       Voltage : 1.92V,        Temperature : 14.76C
ADC value : 97  ,       Voltage : 1.90V,        Temperature : 14.41C
ADC value : 97  ,       Voltage : 1.90V,        Temperature : 14.41C
ADC value : 98  ,       Voltage : 1.92V,        Temperature : 14.76C
ADC value : 98  ,       Voltage : 1.92V,        Temperature : 14.76C
ADC value : 98  ,       Voltage : 1.92V,        Temperature : 14.76C
ADC value : 98  ,       Voltage : 1.92V,        Temperature : 14.76C
ADC value : 98  ,       Voltage : 1.92V,        Temperature : 14.76C
ADC value : 98  ,       Voltage : 1.92V,        Temperature : 14.76C
ADC value : 98  ,       Voltage : 1.92V,        Temperature : 14.76C
ADC value : 98  ,       Voltage : 1.92V,        Temperature : 14.76C
ADC value : 99  ,       Voltage : 1.94V,        Temperature : 15.11C
ADC value : 99  ,       Voltage : 1.94V,        Temperature : 15.11C
ADC value : 100  ,      Voltage : 1.96V,        Temperature : 15.45C
ADC value : 100  ,      Voltage : 1.96V,        Temperature : 15.45C
ADC value : 101  ,      Voltage : 1.98V,        Temperature : 15.80C
ADC value : 102  ,      Voltage : 2.00V,        Temperature : 16.15C
ADC value : 102  ,      Voltage : 2.00V,        Temperature : 16.15C
ADC value : 102  ,      Voltage : 2.00V,        Temperature : 16.15C
ADC value : 102  ,      Voltage : 2.00V,        Temperature : 16.15C
ADC value : 103  ,      Voltage : 2.02V,        Temperature : 16.49C
ADC value : 103  ,      Voltage : 2.02V,        Temperature : 16.49C
ADC value : 103  ,      Voltage : 2.02V,        Temperature : 16.49C
```

Test Code

File name: Thermistor.c

```
#include <wiringPi.h>
#include <stdio.h>
#include <pcf8591.h>  //pcf8591 library
#include <math.h>

#define Address 0x48  //i2c address
#define BASE 64   //DAC write address
#define A0 BASE+0  //A0 analogRead  address
#define A1 BASE+1  //A1 analogRead  address
#define A2 BASE+2
#define A3 BASE+3

int main(void){
   wiringPiSetup();
   pcf8591Setup(BASE,Address);  //Initialize the pcf8591

   while(1){
      int value = analogRead(A2);  //read analog value A2 pin
      float voltage = (float)value / 255.0 * 5.0; // calculate voltage
      float Rt = 4.7 * (5.0 / voltage) - 4.7 ; //calculate resistance value of thermistor, 5.0 * (R / (Rt + R)) = voltage,>>>
```

```
        float tempK = 1/(1/(273.15 + 25) + log(Rt/4.7)/3950.0);  //calculate temperature (Kelvin)
        float tempC = tempK - 273.15;  //calculate temperature (Celsius)
        printf("ADC value : %d  ,\tVoltage : %.2fV, \tTemperature : %.2fC\n",value,voltage,tempC);
        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