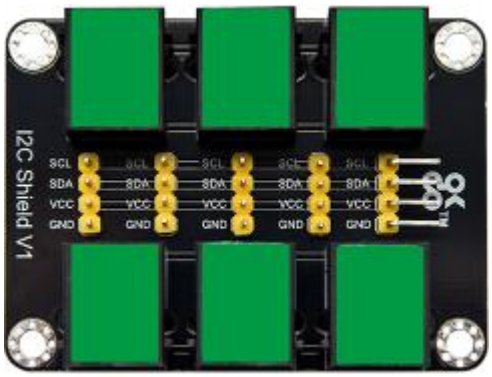
I2C Interface Expansion Board (000x0000 Article Number) (TS2174)

DESIGN THE WORLD

Product Details



This is a TelePort I2C interface conversion shield, compatible with both the TelePort interface and 2.54mm pitch header interface.

When the MCU communicates with other sensors/modules, I2C communication mode is required, and I2C communication can be performed simultaneously with multiple sensors/modules by changing the address.

Generally MCU only introduces an I2C communication interface, which is very inconvenient. Yet, this trouble can be solved easily with this I2C shield.

The interfaces are extended to 6 communication ports, ensuring that the MCU can perform I2C communication with five sensors/modules at the same time.

Features and Benefits

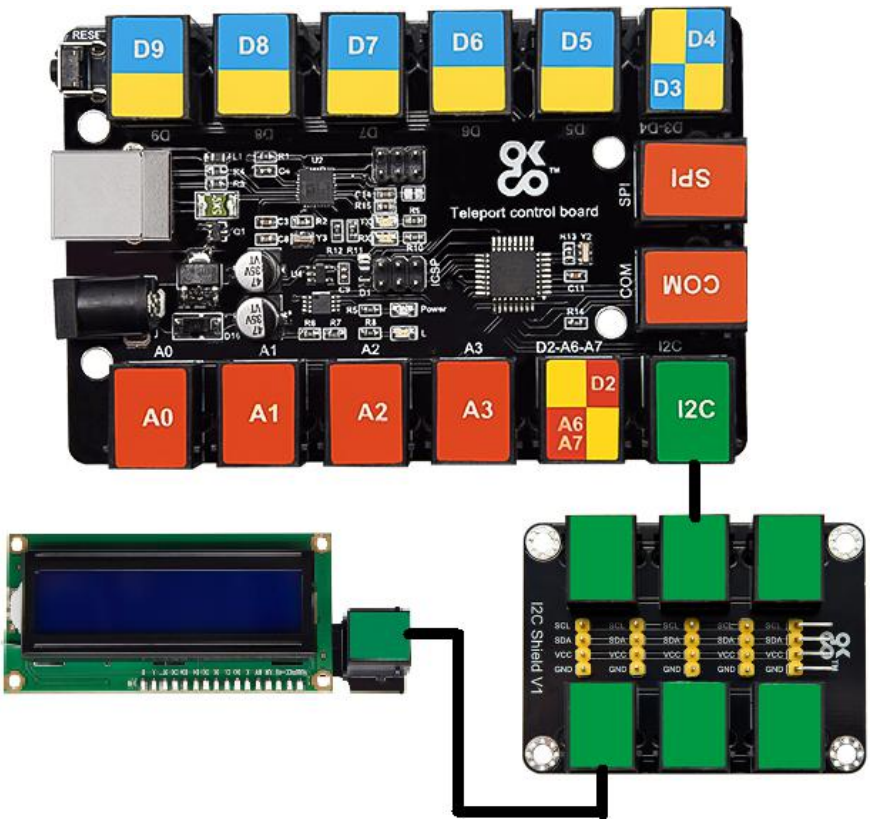
- Compatible with RJ11 6P6C OKdo TelePort Control boards and expansion shields.
- Use this breakout board to add up to 10 extra sensors and output devices to your project using the I2C bus. Each I2C device has a unique address allowing it to be on the same bus.
- This shield is compatible with TelePort and 2.54mm headers for convenience.

Technical Specifications

Sensor type	Adapter
Working voltage	DC 5V
Interface	I2C
Pin pitch	2.54mm
Dimensions	60mm*44.4mm*17.5mm
Weight	20.2g

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

If you added libraries, just ignore the following instruction.

Before compiling test code, remember to place libraries like **LiquidCrystal_I2C** and **Wire** folders into the libraries of Arduino IDE.

Copy the unzipped **LiquidCrystal_I2C** and **Wire** folders to the libraries of Arduino IDE.

After pasting them, then reboot the compiler

For instance: C:\Program Files\Arduino\libraries

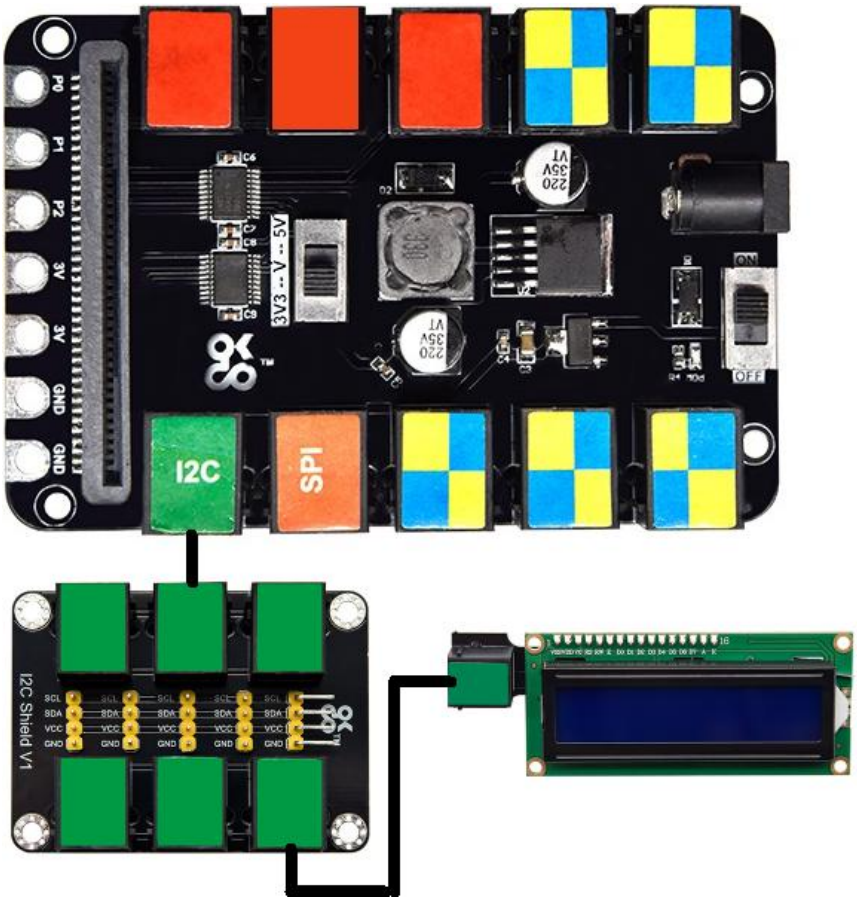
```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2); // set the address of LCD to 0x27
void setup()
{
lcd.init(); // initialize lcd
lcd.init();
lcd.backlight();
}
void loop()
{
lcd.setCursor(3,0);
lcd.print("Hello, world!");
lcd.setCursor(2,1);
lcd.print("OKDO!");
}
```

Test Result

Wire up, upload test code and power it up. I2C 1602 LCD will display Hello, world! and OKDO !

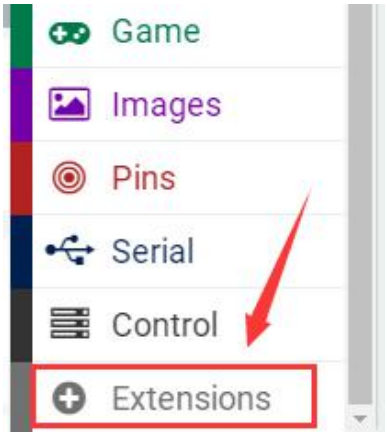
If you want to know more details about Arduino and the TelePort control board, you can refer to TS2178.

➢ **Micro:bit Application**

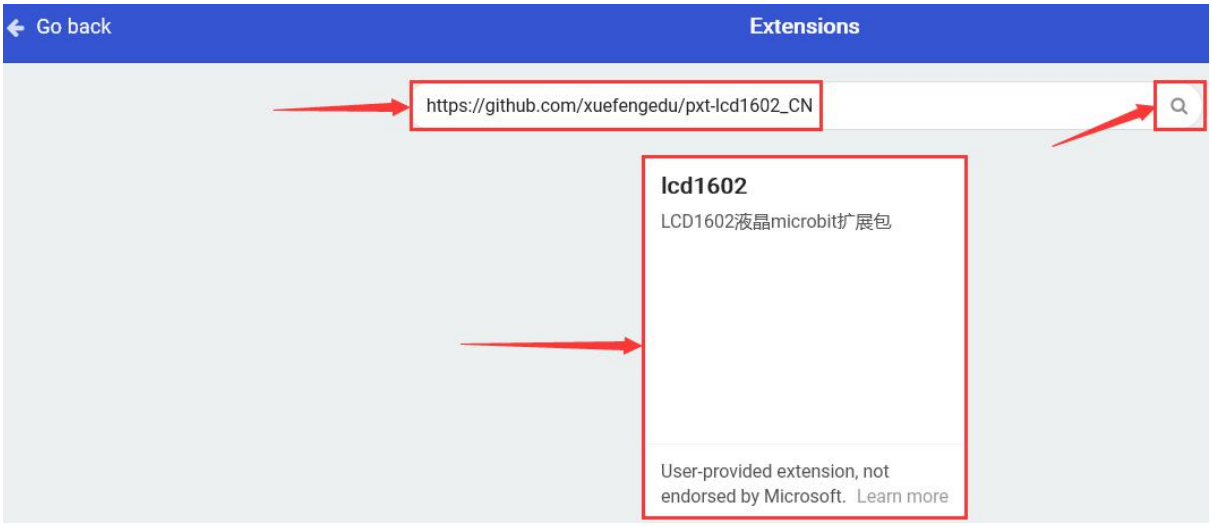


It is compatible with the Micro:bit board and the TS2179 Micro:bit expansion board.

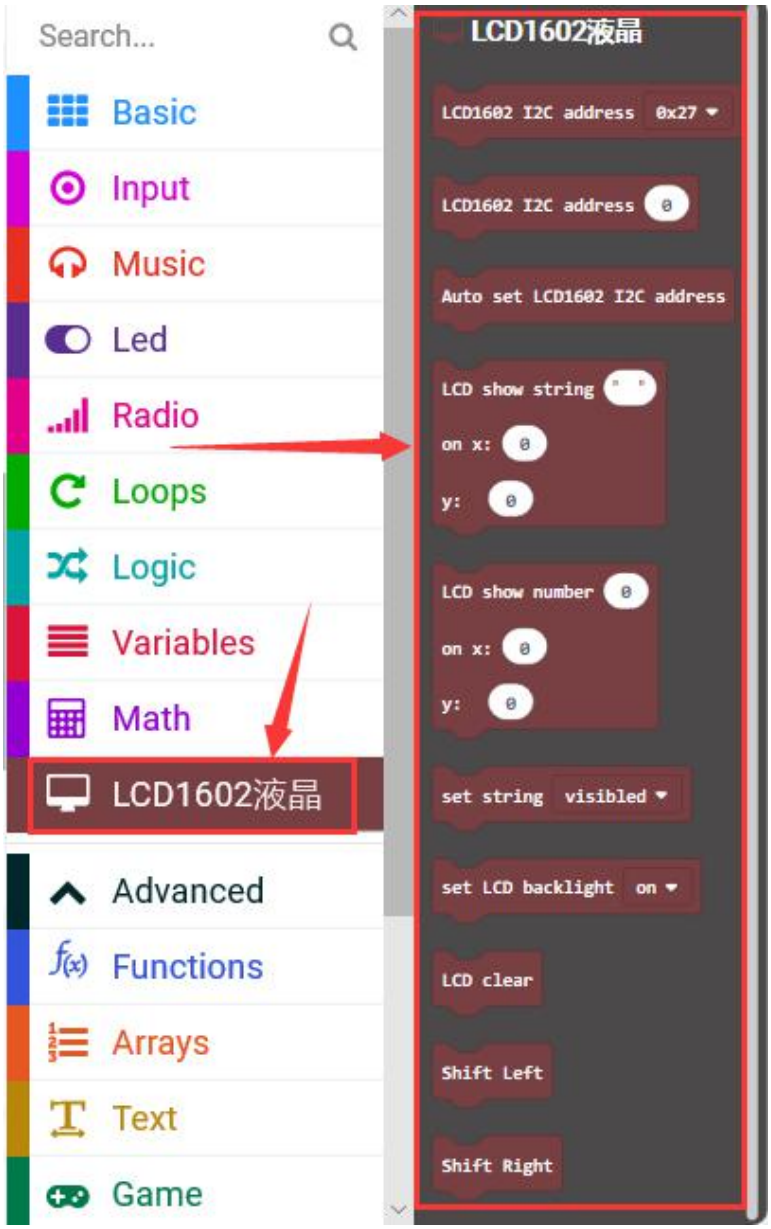
Add the library of the 1602 LCD display, as shown below;
Use the library file to set code, click“Extensions”



Enter https://github.com/xuefengedu/pxt-lcd1602_CN to search, as shown below, click the library file and download it automatically.



After the library of the 1602 LCD display is installed, then you can view the corresponding block in the blocks list.



Test Code

.....①Run the "on start" block to boot the program

.....②Turn off LED dot matrix

.....③set the address of LCD1602 I2C to 0x27

.....④LCD shows okdo at the coordinate(x:3 y:0)

.....⑤display characters

.....⑥set backlight on

.....⑦set Value to 0

.....⑧The program is run circularly under the command of "forever" block

.....⑨Value Change Value by 1

.....⑩display Value at the coordinate(x:7 y:1)

.....⑪delay in 500ms

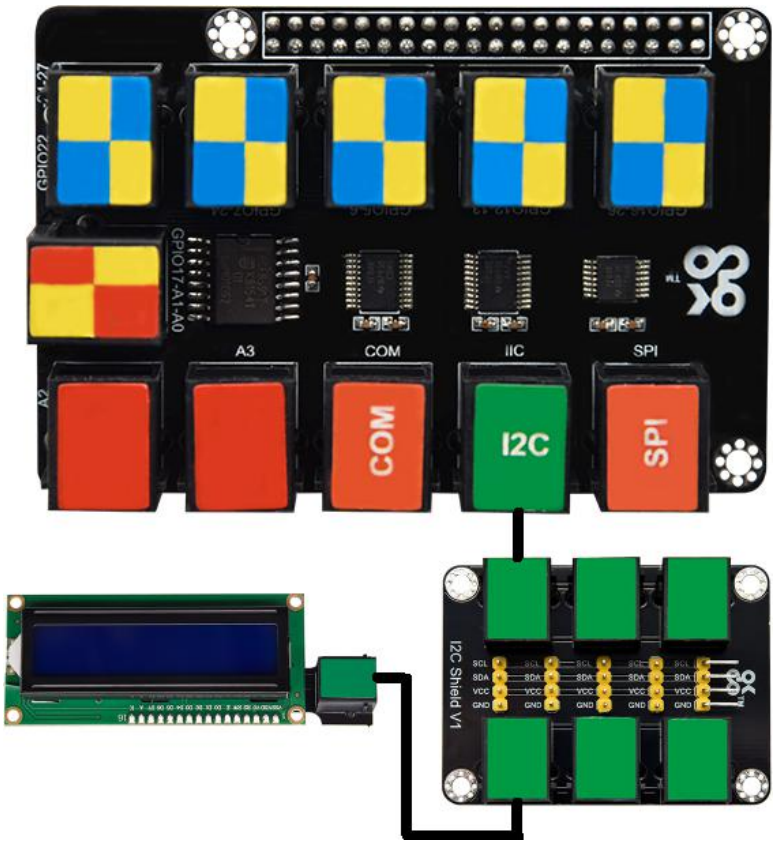
Test Result

Wire up, insert the Micro:bit V2, upload test code, turn DIP switches to 5V and ON and power it up(above 5V).

After rotating the potentiometer at the back of the 1602 LCD module, "okdo"will be displayed at the first row, and numbers will be shown at the second row, the numbers will increase by 1 for each 0.5 second.

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:

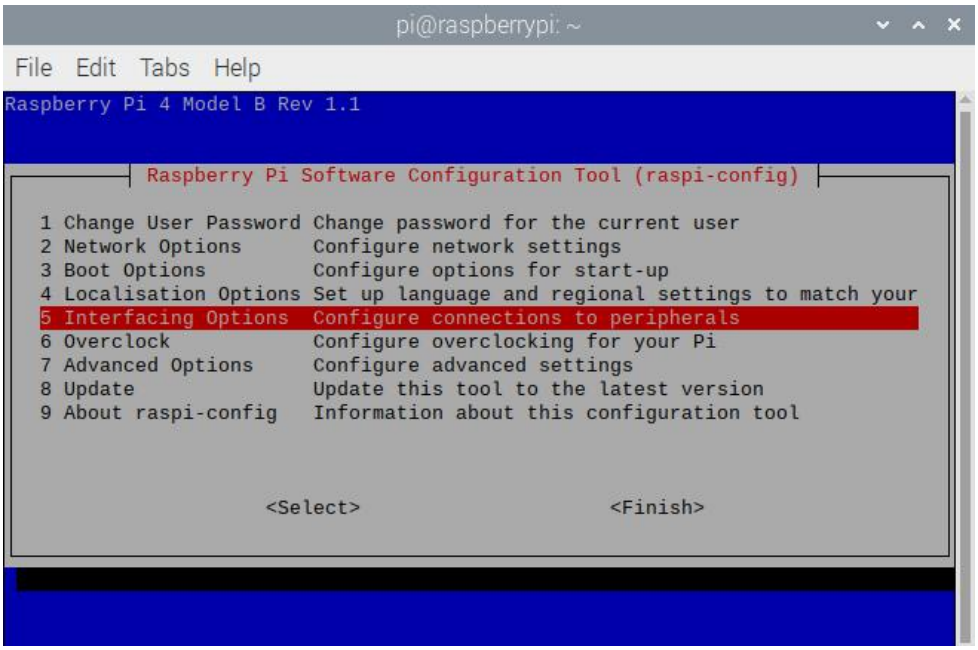
The Raspberry Pi itself does not have AD/DA function; therefore an expansion board with this function is required when connected to external analog sensors. And here we use a PCF8591 A/D converter with I2C communication.

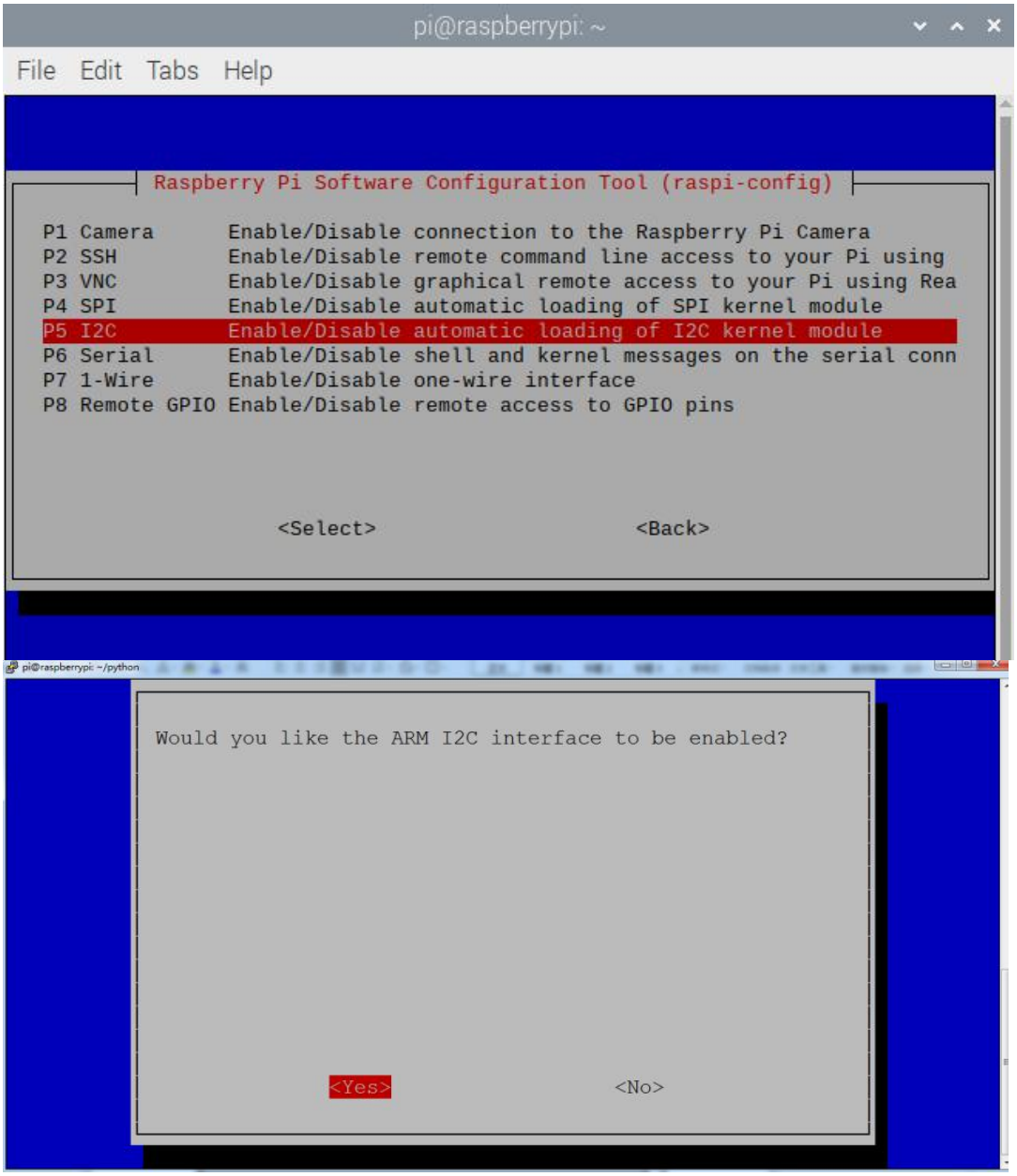
Enable the I2C communication function of the Raspberry Pi as follows:

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
```

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



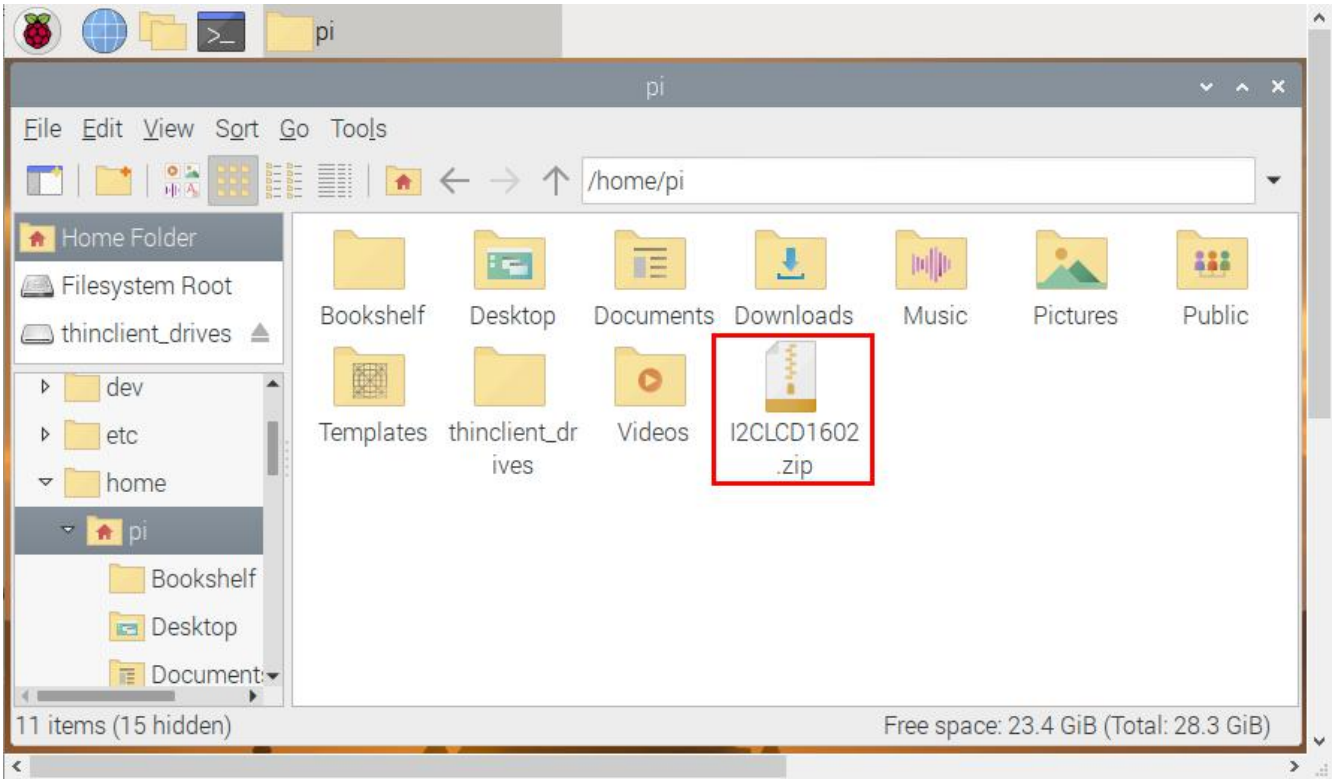


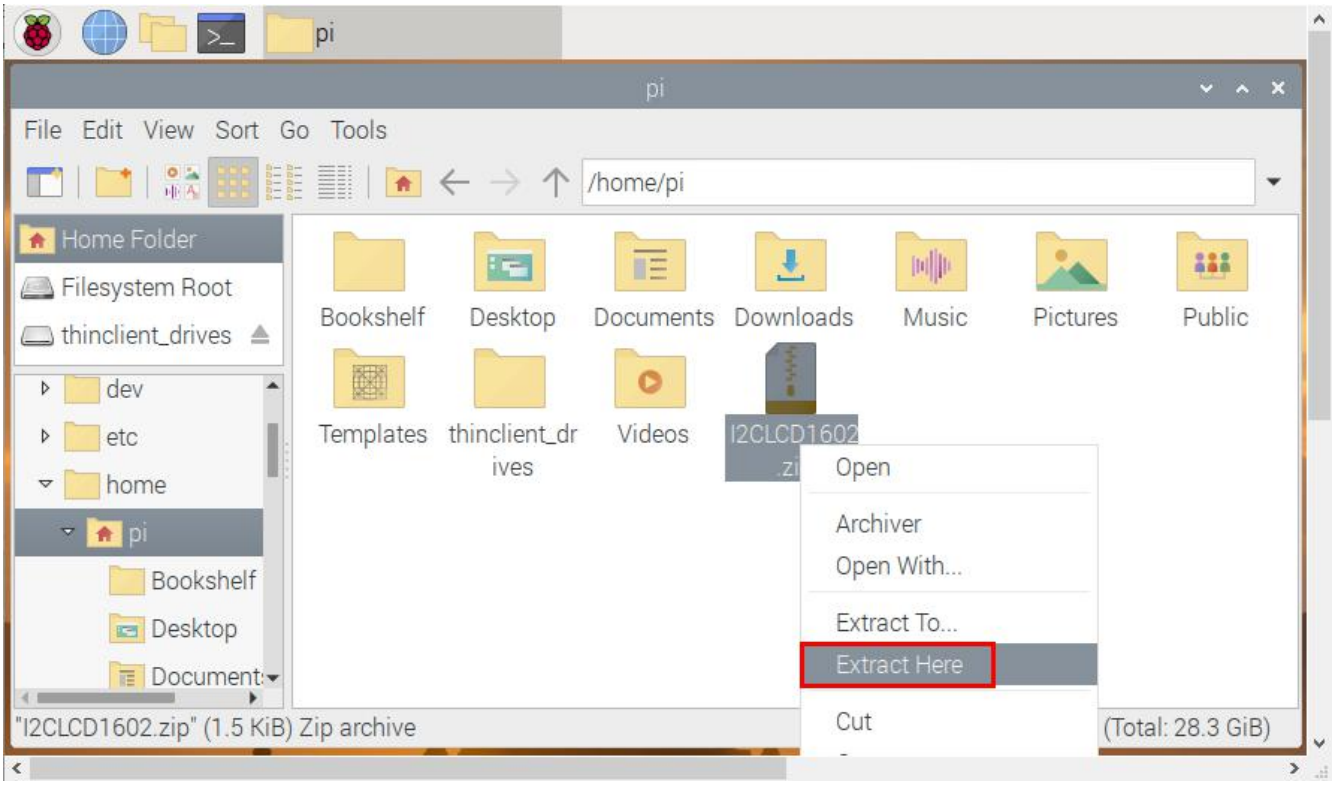
Check the address of the I2C module (PCF8591) connected to the Raspberry Pi, enter the command i2cdetect -y 1, and then press **Enter**.

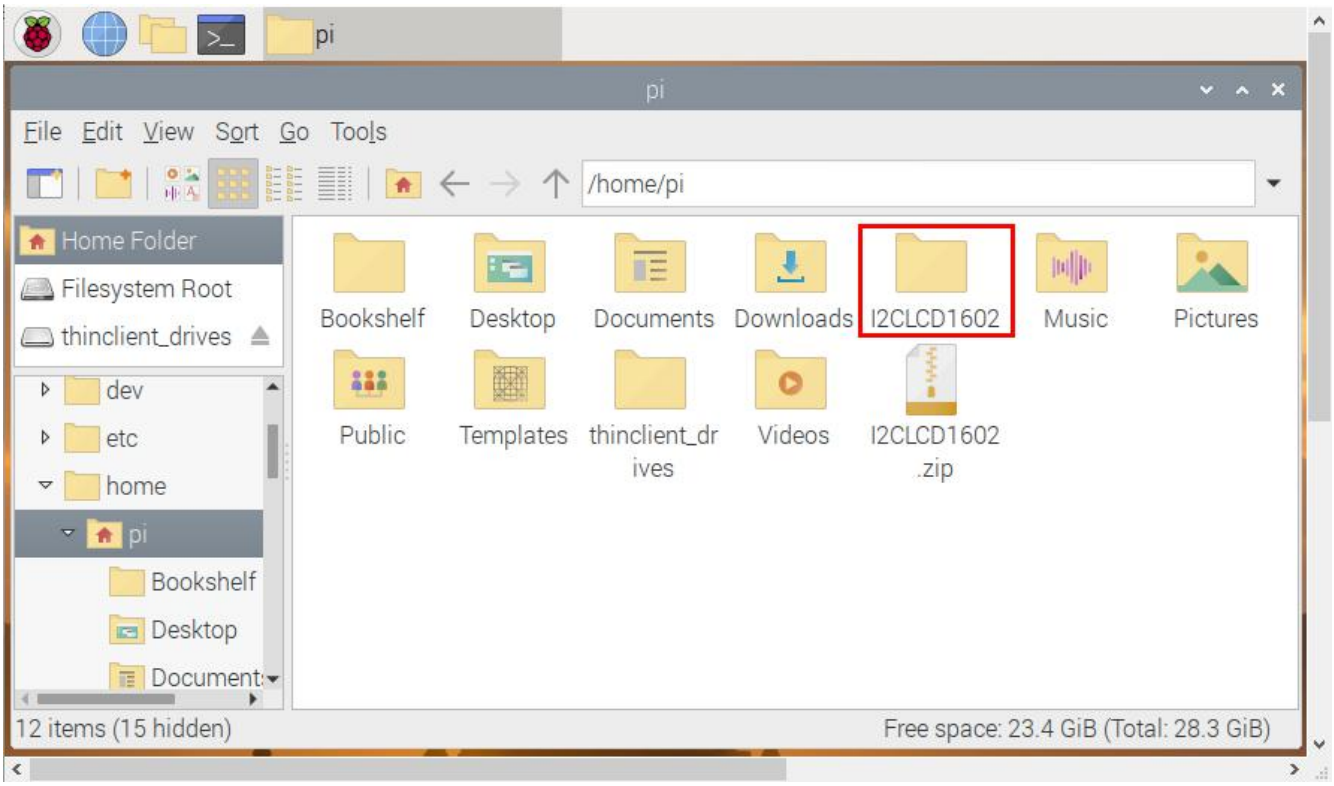
From the below picture, it is known that the I2C address of PCF8591 is 0x48 .

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 I2CLCD1602.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"

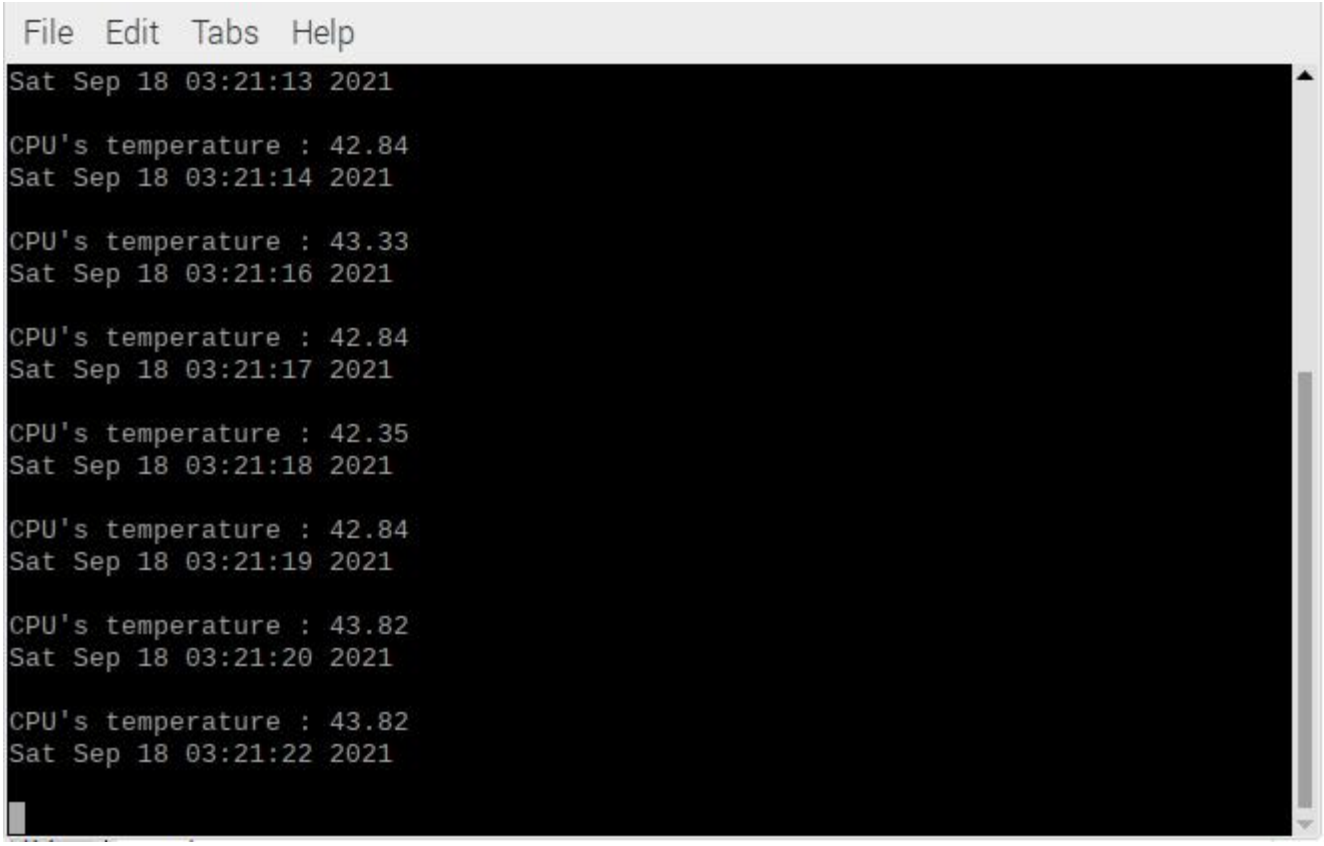
```
cd /home/pi/I2CLCD1602
gcc I2CLCD1602.c -o I2CLCD1602 -lwiringPiDev -lwiringPi
sudo ./I2CLCD1602
```

(3) Test Result:

Insert the shield into the Raspberry Pi board. After programming finishes, then the LCD1602 display will show the time and temperature of CPU of Raspberry Pi board.

Note: if you can't see the screen clearly, rotate the blue potentiometer at the back of LCD 1602 until you can see the displayed time and temperature.

Note: press Ctrl + C to exit code running



Test Code

File name: I2CLCD1602.c

```
#include <stdlib.h>
#include <stdio.h>
#include <wiringPi.h>
#include <wiringPiI2C.h>
#include <pcf8574.h>
#include <lcd.h>
#include <time.h>

int pcf8574_address = 0x27;       // PCF8574T:0x27, PCF8574AT:0x3F
#define BASE 64         // BASE any number above 64
//Define the output pins of the PCF8574, which are directly connected to the LCD1602 pin.
#define RS      BASE+0
#define RW      BASE+1
#define EN      BASE+2
#define LED     BASE+3
#define D4      BASE+4
#define D5      BASE+5
#define D6      BASE+6
#define D7      BASE+7

int lcdhd;// used to handle LCD
void printCPUTemperature(){// sub function used to print CPU temperature
```

```
    FILE *fp;
    char str_temp[15];
    float CPU_temp;
    // CPU temperature data is stored in this directory.
    fp=fopen("/sys/class/thermal/thermal_zone0/temp","r");
    fgets(str_temp,15,fp);      // read file temp
    CPU_temp = atof(str_temp)/1000.0;   // convert to Celsius degrees
    printf("CPU's temperature : %.2f \n",CPU_temp);
    lcdPosition(lcdhd,0,0);     // set the LCD cursor position to (0,0)
    lcdPrintf(lcdhd,"CPU:%.2fC",CPU_temp);// Display CPU temperature on LCD
    fclose(fp);
}
void printDataTime(){//used to print system time
    time_t rawtime;
    struct tm *timeinfo;
    time(&rawtime);// get system time
    timeinfo = localtime(&rawtime);//convert to local time
    printf("%s \n",asctime(timeinfo));
    lcdPosition(lcdhd,0,1);// set the LCD cursor position to (0,1)
    lcdPrintf(lcdhd,"Time:%02d:%02d:%02d",timeinfo->tm_hour,timeinfo->tm_min,timeinfo->tm_sec); //Display sys
}
int detectI2C(int addr){
    int _fd = wiringPiI2CSetup (addr);
    if (_fd < 0){
        printf("Error address : 0x%x \n",addr);
        return 0 ;
    }
    else{
        if(wiringPiI2CWrite(_fd,0) < 0){
            printf("Not found device in address 0x%x \n",addr);
            return 0;
        }
        else{
            printf("Found device in address 0x%x \n",addr);
            return 1 ;
        }
    }
}
int main(void){
    int i;

    printf("Program is starting ...\n");
```

```
    wiringPiSetup();
    if(detectI2C(0x27)){
        pcf8574_address = 0x27;
    }else if(detectI2C(0x3F)){
        pcf8574_address = 0x3F;
    }else{
        printf("No correct I2C address found, \n"
"Please use command 'i2cdetect -y 1' to check the I2C address! \n"
"Program Exit. \n");
        return -1;
    }
    pcf8574Setup(BASE,pcf8574_address);//initialize PCF8574
    for(i=0;i<8;i++){
        pinMode(BASE+i,OUTPUT);     //set PCF8574 port to output mode
    }
    digitalWrite(LED,HIGH);     //turn on LCD backlight
    digitalWrite(RW,LOW);       //allow writing to LCD
            lcdhd = lcdInit(2,16,4,RS,EN,D4,D5,D6,D7,0,0,0,0);// initialize LCD and return “handle” used to handle LCD
    if(lcdhd == -1){
        printf("lcdInit failed !");
        return 1;
    }
    while(1){
        printCPUTemperature();//print CPU temperature
        printDataTime();        // print system time
        delay(1000);
    }
    return 0;
}
```

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

END