I2C LCD2004

Introduction

As we all know, though LCD and some other displays greatly enrich the man-machine interaction, they share a common weakness. When they are connected to a controller, multiple IOs will be occupied of the controller which has no so many outer ports. Also it restricts other functions of the controller. Therefore, LCD2004 with an I2C bus is developed to solve the problem.

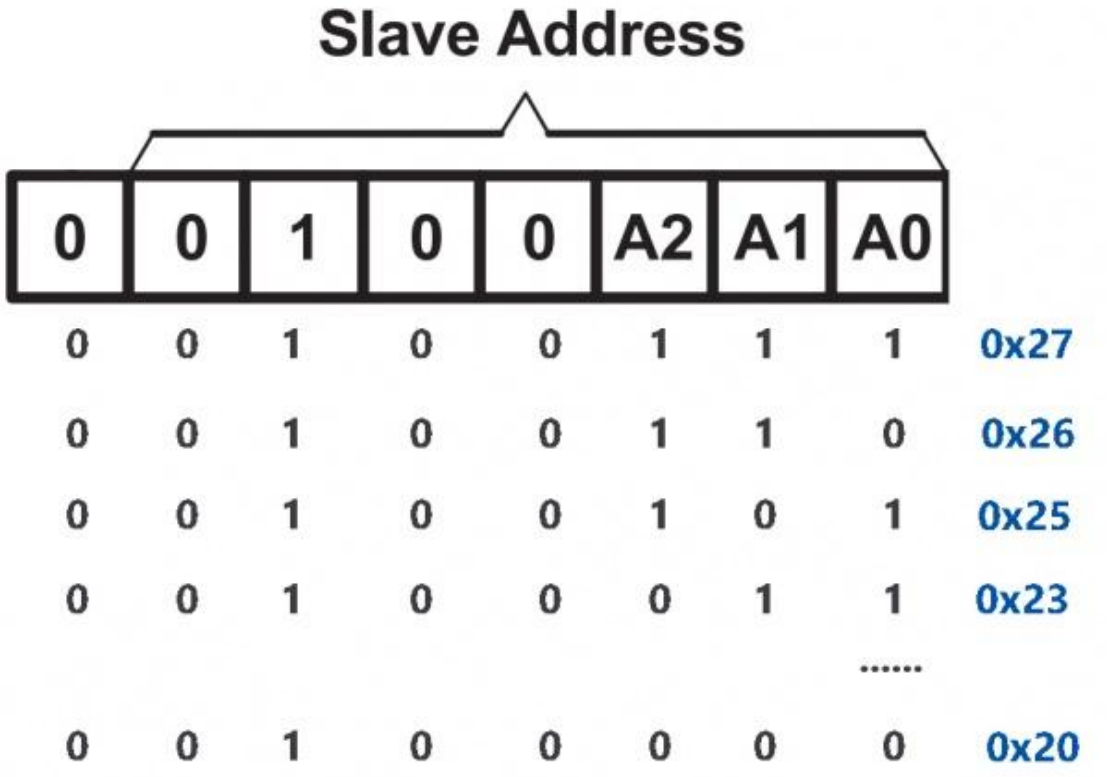
I2C bus is a type of serial bus invented by PHLIPS. It is a high performance serial bus which has bus ruling and high or low speed device synchronization function required by multiple host system. I2C bus has only two bidirectional signal lines, Serial Data Line (SDA) and Serial Clock Line (SCL). The blue potentiometer on the I2C LCD2004 is used to adjust backlight to make it easier to display on the I2C LCD2004.

- **GND**: Ground
- **VCC**: Voltage supply, 5V.
- **SDA**: Serial data line. Connect to VCC through a pull_up resistor.
- **SCL**: Serial clock line. Connect to VCC through a pull_up resistor.

I2C Address

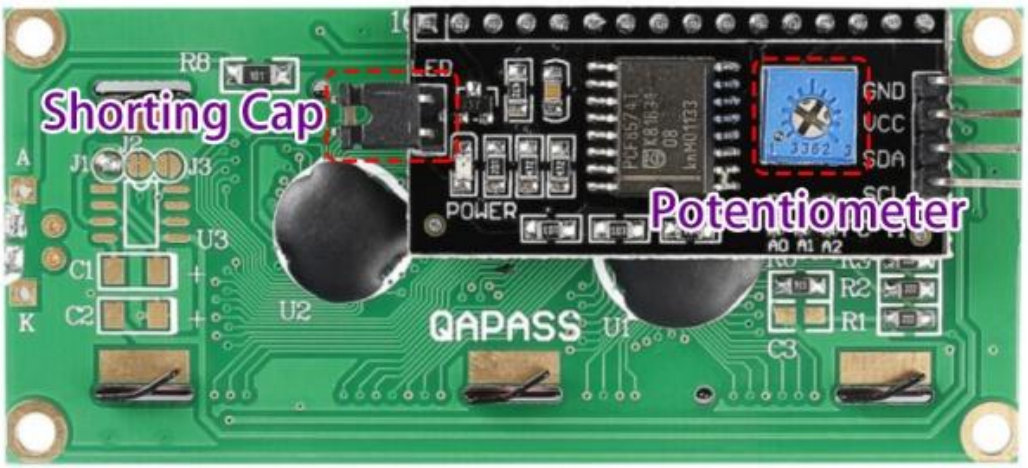
The default address is basically 0x27, in a few cases it may be 0x3F.

Taking the default address of 0x27 as an example, the device address can be modified by shorting the A0/A1/A2 pads; in the default state, A0/A1/A2 is 1, and if the pad is shorted, A0/A1/A2 is 0.



Backlight/Contrast

Backlight can be enabled by jumper cap, unplug the jumper cap to disable the backlight. The blue potentiometer on the back is used to adjust the contrast (the ratio of brightness between the brightest white and the darkest black).

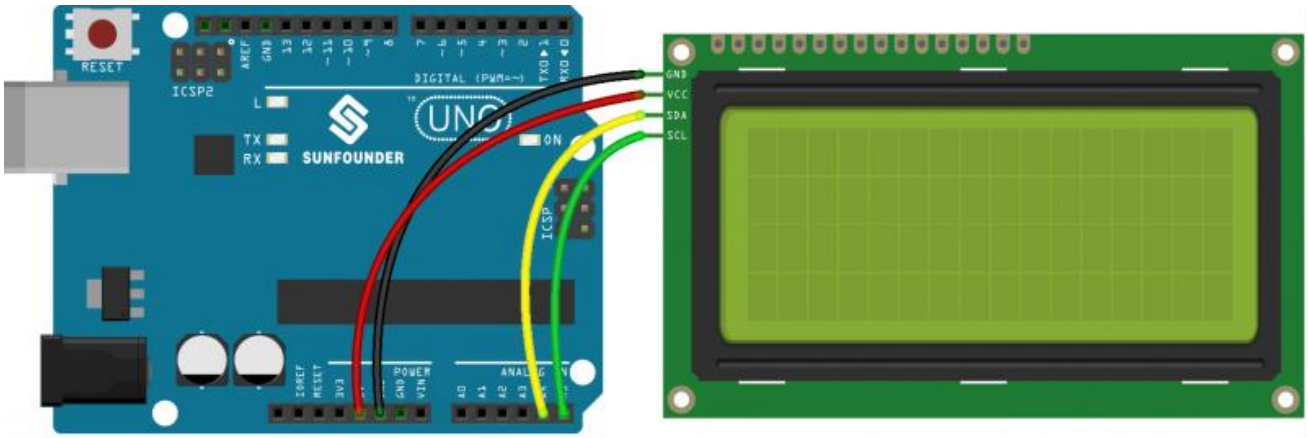


- **Shorting Cap**: Backlight can be enabled by this cap， unplug this cap to disable the backlight.
- **Potentiometer**: It is used to adjust the contrast (the clarity of the displayed text), which is increased in the clockwise direction and decreased in the counterclockwise direction.

How to Use in Arduino?

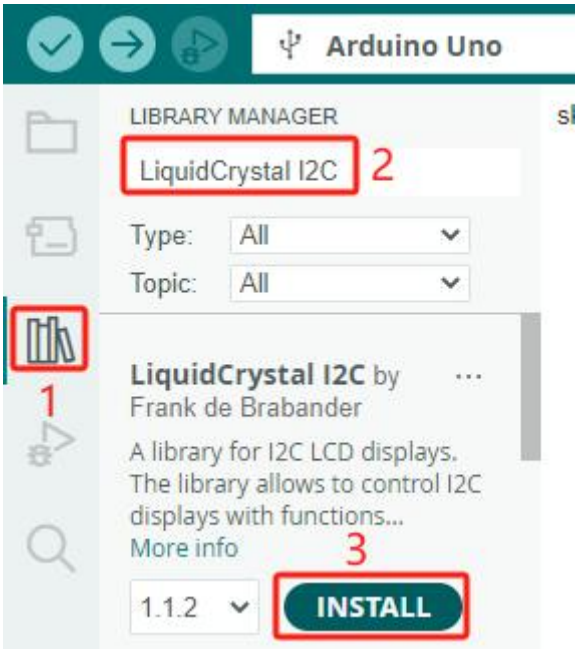
Build the Circuit

I2C LCD2004	Arduino Board
GND	GND
VCC	5V
SDA	A4 /pin 20 mega2560
SCL	A5 /pin 21 mega2560



Install the Library

Open the **Library Manager**, enter **LiquidCrystal I2C**, and click **INSTALL** when it appears.



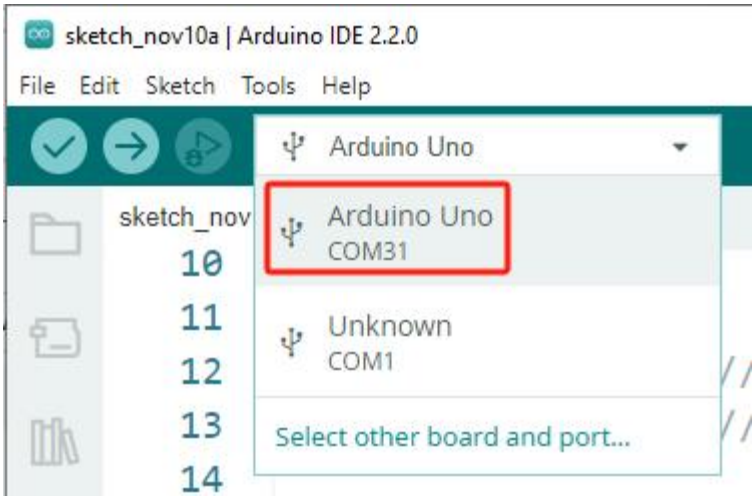
Copy and Upload the Code

1. Copy and paste the following code to the Arduino IDE.

```
/*******************************/
// Include necessary libraries
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// Initialize the LCD object, set the LCD I2C address to 0x27 for a 20x4 display
LiquidCrystal_I2C lcd(0x27, 20, 4);

/*********************************************************/
void setup()
{
  lcd.init();                      // Initialize the LCD
  lcd.backlight();                 // Turn on the backlight

  // Set cursor to the top left corner and print the string on the first row
  lcd.setCursor(0, 0);
  lcd.print("       Hello, world!       ");

  // Move to the second row and print the string
  lcd.setCursor(0, 1);
  lcd.print("     IIC/I2C LCD2004    ");

  // Move to the third row and print the string
  lcd.setCursor(0, 2);
  lcd.print("    20 cols, 4 rows      ");

  // Move to the fourth row and print the string
  lcd.setCursor(0, 3);
  lcd.print(" www.sunfounder.com ");
}
/*********************************************************/
```

```
void loop()
{
  // Empty loop
}
/**********************************************************/
```

2. Choose the board and port you use.



3. Click the **Upload** button.

4. You will see relevant information such as Hello world displayed on the I2C LCD2004.

Read I2C Address

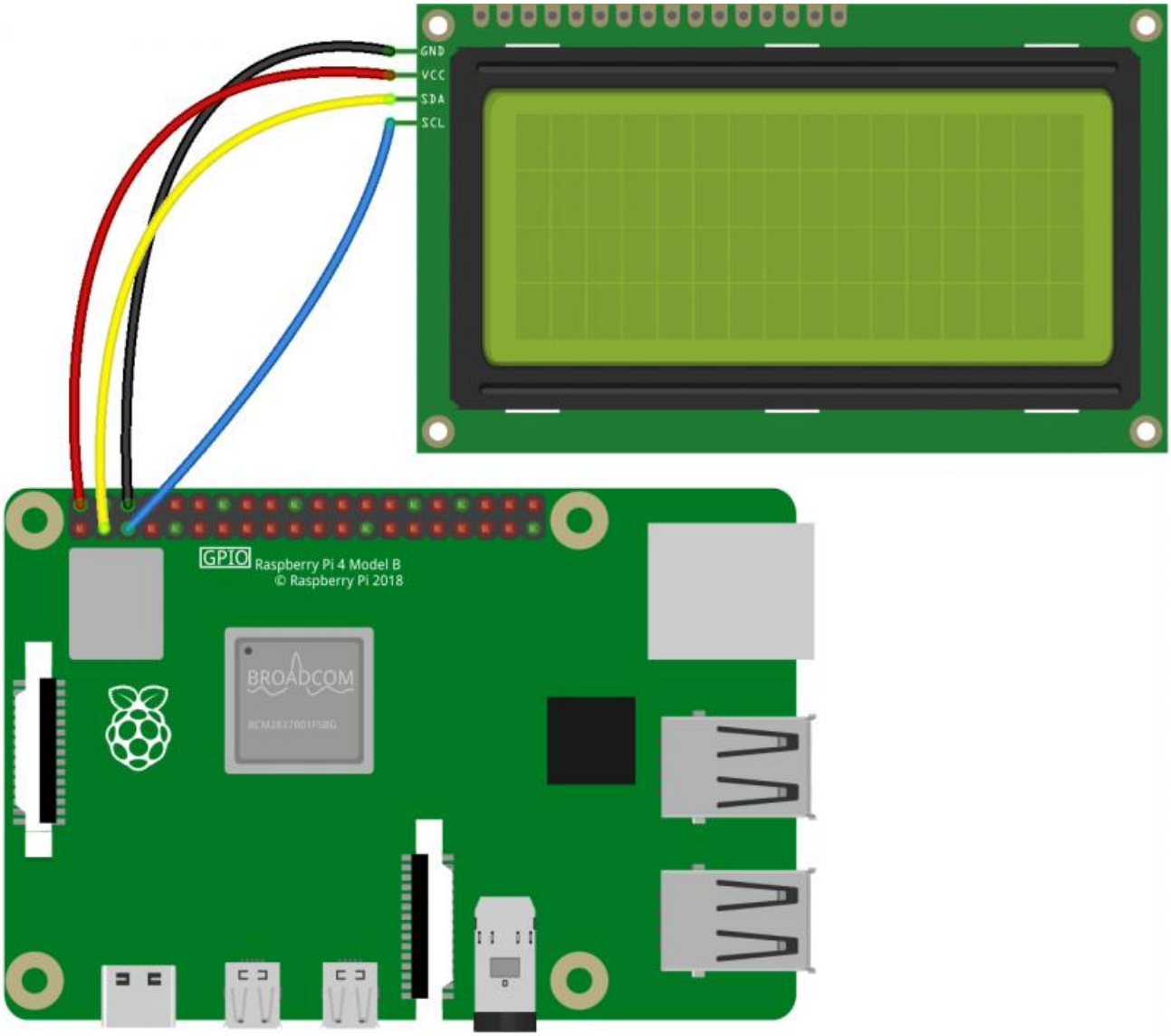
If the display is functioning properly but only shows 16 black rectangles on the line, it's likely that the I2C address is not set to 0x27. In this case, you should run the following code to determine the correct address and then replace '0x27' with the address you find.

```
/*********************************************************
 * Name: I2C_Address
 * Function: Read the address of the I2C LCD1602
 * Connection:
 * I2C                          Arduino UNO
 * GND                           GND
 * VCC                           5V
 * SDA                           A4 (pin 20 in Mega2560)
 * SCL                           A5 (pin 21 in Mega2560)
 *********************************************************/

#include <Wire.h>    // Include Wire library for I2C communication

void setup() {
  Wire.begin();                          // Initialize I2C communication
  Serial.begin(9600);                    // Start serial communication at 9600 baud rate
  Serial.println("\nI2C Scanner");    // Print a message to the serial monitor
}

void loop() {
  byte error, address;    // Declare variables for storing error status and I2C address
  int nDevices;              // Variable to keep track of number of devices found

  Serial.println("Scanning...");    // Print scanning message
  nDevices = 0;                          // Initialize the device count to 0

  // Loop through all possible I2C addresses (1 to 126)
  for (address = 1; address < 127; address++) {
```

```
      Wire.beginTransmission(address);    // Start a transmission to the I2C address
      error = Wire.endTransmission();     // End the transmission and get the status

      // Check if device responded without error (acknowledged)
      if (error == 0) {
        Serial.print("I2C device found at address 0x");    // Notify device found
        if (address < 16) Serial.print("0");                    // Print leading zero for
addresses less than 16
        Serial.print(address, HEX);                                 // Print the address in
hexadecimal
        Serial.println(" !");
        nDevices++;                                                  // Increment the device
count
      } else if (error == 4) {                                  // If there was an unknown error
        Serial.print("Unknown error at address 0x");    // Notify about the error
        if (address < 16) Serial.print("0");                // Print leading zero for addresses
less than 16
        Serial.println(address, HEX);                           // Print the address in
hexadecimal
      }
    }

    // After scanning, print the results
    if (nDevices == 0)
      Serial.println("No I2C devices found\n");    // No devices found
    else
      Serial.println("done\n");    // Scanning done

    delay(5000);    // Wait 5 seconds before the next scan
}
```

How to Use in Raspberry Pi?

Build the Circuit



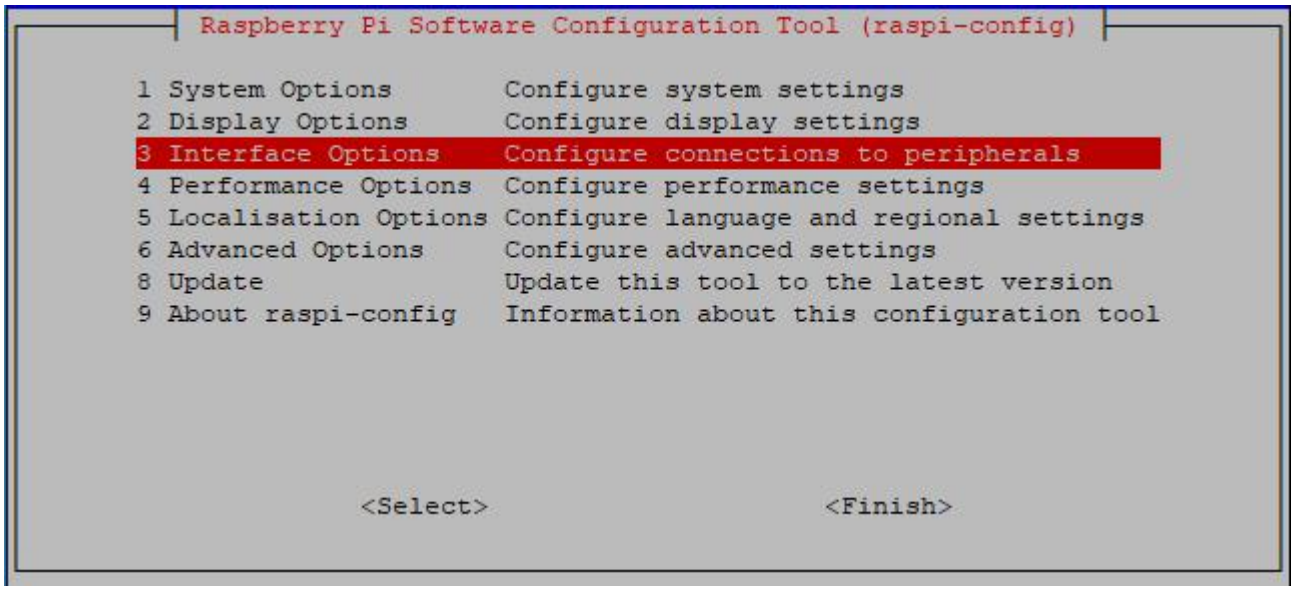
Setup I2C

Enable the I2C port of your Raspberry Pi (If you have enabled it, skip this; if you do not know whether you have done that or not, please continue).

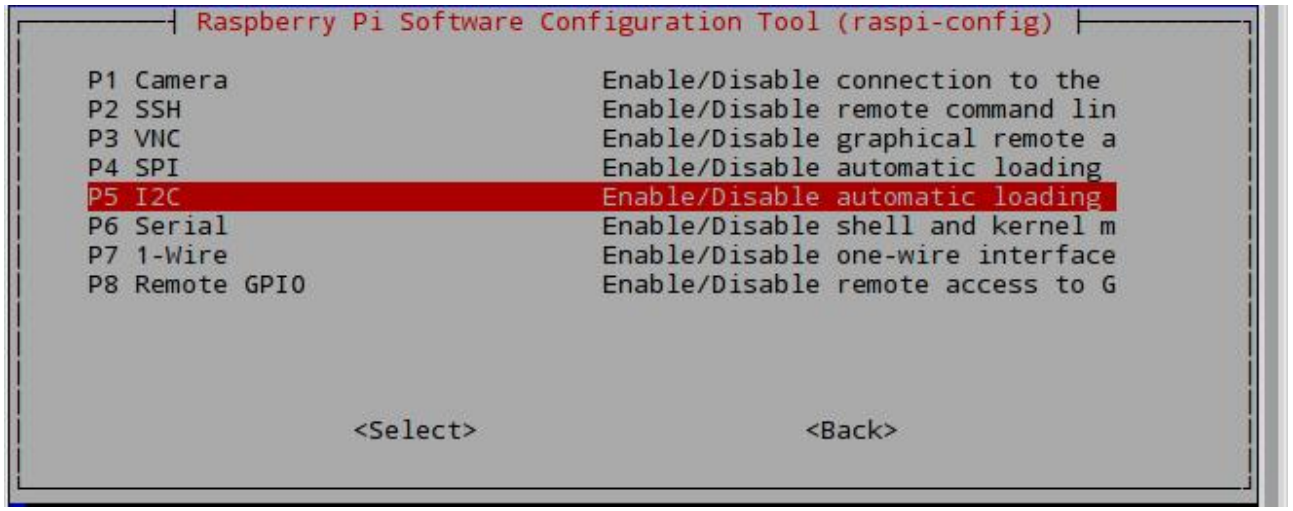
Step 1: Run the following command.

```
sudo raspi-config
```

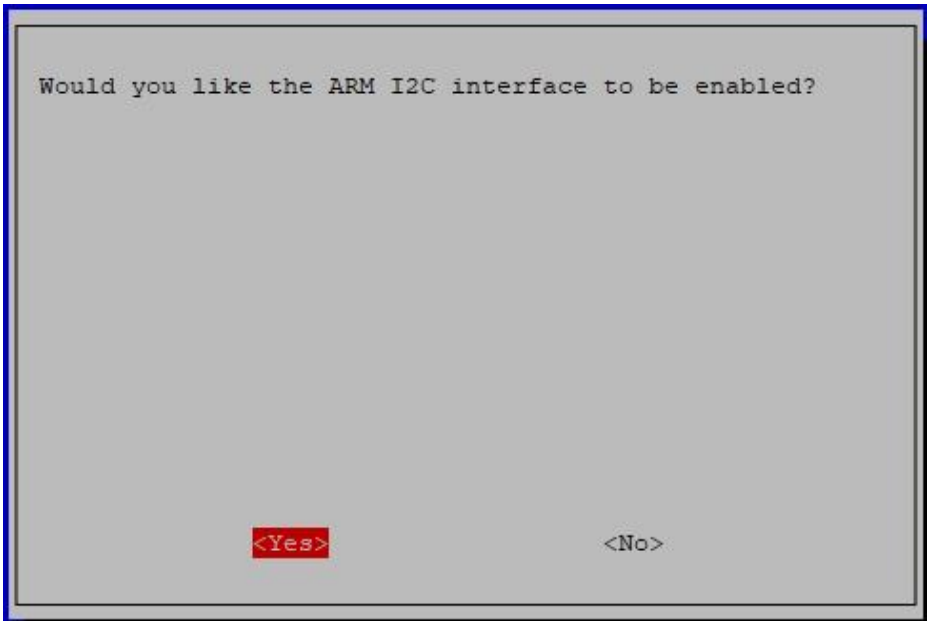
Step 2: **Interfacing options**.



Step 3: **I2C**.



Step 4: <Yes>, then <Ok> -> <Finish>.



Step 5: Check whether the i2c modules are loaded and active.

```
lsmod | grep i2c
```

Step 6: Then the following codes will appear (the number may be different).

```
i2c_dev                 6276  0
i2c_bcm2708             4121  0
```

Step 7: Install i2c-tools.

```
sudo apt-get install i2c-tools
```

Step 8: Check the address of the I2C device.

```
i2cdetect -y 1          # For Raspberry Pi 2 and higher version
i2cdetect -y 0          # For Raspberry Pi 1
```

```
pi@raspberrypi ~ $ i2cdetect -y 1
0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- 27 -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

If there is an I2C device connected, the address of the device will be displayed.

Step 9: Install *libi2c-dev* or *smbus2*.

For C language users

```
sudo apt-get install libi2c-dev
```

For Python users

```
sudo pip3 install smbus2
```

Download and Run the Code

Step 1: Download the code package.

```
wget http://wiki.sunfounder.cc/images/3/36/I2c_lcd2004_for_raspberry_pi.zip
```

Step 2: Extract the package

```
unzip I2c_lcd2004_for_raspberry_pi.zip
```

For C Language Users

Step 3: If you have not installed wiringPi, then you will need to install it first.

```
sudo apt-get update
git clone https://github.com/WiringPi/WiringPi
cd WiringPi
./build
```

Step 4: You can test whether the wiringPi library is installed successfully or not by the following instruction.

```
gpio -v
```

Step 5: Get into the folder of code.

```
cd ~/I2c_lcd2004_for_raspberry_pi/c
```

Step 6: Compile.

```
gcc lcd2004.c -o lcd2004 -lwiringPiDev -lwiringPi
```

Step 7: Run.

```
sudo ./lcd2004
```

For Python Users

Step 3: Get into the folder of code.

```
cd ~/I2c_lcd2004_for_raspberry_pi/python
```

Step 4: Run.

```
sudo python3 lcd2004_show.py
```