13.I2C_LCD1602

Introduction

LCD1602 is a character type liquid crystal display, which can display 32 (16*2) characters at the same time.

Hardware Required

- ✓ 1 * Raspberry Pi
- ✓ 1 * T-Extension Board
- ✓ 1 * I2C LCD1602
- ✓ 1 * 40-pin Cable
- ✓ Several Jumper Wires
- ✓ 1 * Breadboard

Principle

I2C LCD1602

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, LCD1602 with an I2C bus is developed to solve the problem.

I2C communication

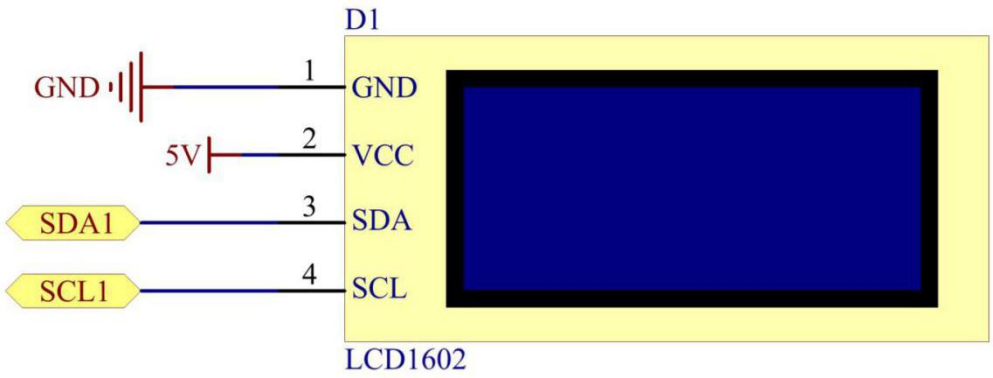
I2C(Inter-Integrated Circuit) bus is a very popular and powerful bus for communication between a master device (or master devices) and a single or multiple slave devices.

I2C main controller can be used to control IO expander, various sensors, EEPROM,

ADC/DAC and so on. All of these are controlled only by the two pins of host, the serial data (SDA1) line and the serial clock line(SCL1).

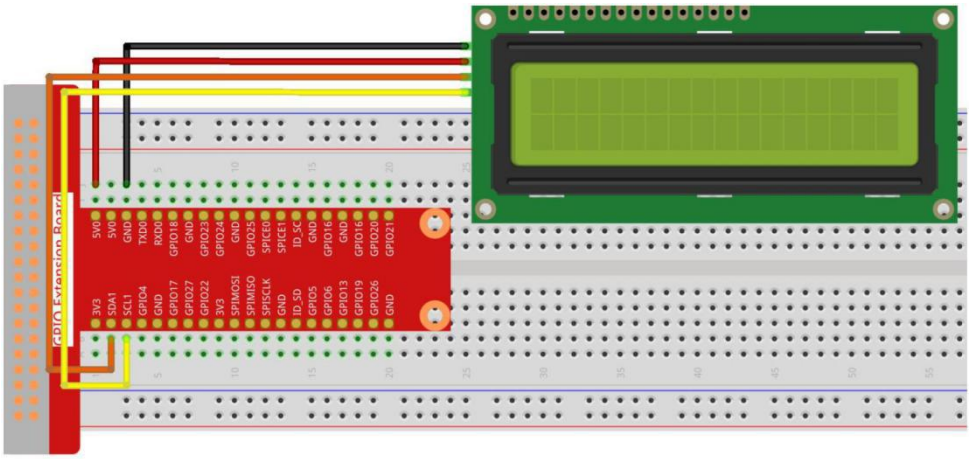
Schematic Diagram

T-Board Name	physical
SDA1	Pin 3
SCL1	Pin 5



Experimental Procedures

Step 1: Build the circuit.



Step 2: Setup I2C(see Appendix. If you have set I2C, skip this step.)

For C Language Users

Step 3: Change directory.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/13.I2C_LCD1602
```

Step 4: Compile the code.

```
gcc 13.I2C_LCD1602.c -o I2C_LCD1602.out -lwiringPi
```

Step 5: Run the executable file

```
sudo ./I2C_LCD1602.out
```

After the code runs, you can see ”Hello!”,”From Rexqualis” displaying on the LCD.

Code

Note: None of the following functions with ellipses is complete. You can view the complete code by using the command, nano 13.I2C_LCD1602.c in the Bash interface.

```
#include <stdio.h>
#include <wiringPi.h>
#include <wiringPiI2C.h>
#include <string.h>

int LCDAddr = 0x27;//i2c address
int BLEN = 1;
int fd;  //the driver file

void write_word(int data){  //wrtite to the i2c data
    int temp = data;
    if ( BLEN == 1 )
        temp |= 0x08; // or get the 1 in 0000 1000
    else
        temp &= 0xF7;  // and  get the 1 in 1111 0111
    wiringPiI2CWrite(fd, temp);
}

void send_command(int comm){//send the command in twice
    int buf;
```

```
    // Send bit7-4 firstly
    buf = comm & 0xF0;
    buf |= 0x04;            // RS = 0, RW = 0, EN = 1
    write_word(buf);
    delay(2);
    buf &= 0xFB;            // Make EN = 0
    write_word(buf);

    // Send bit3-0 secondly
    buf = (comm & 0x0F) << 4;
    buf |= 0x04;            // RS = 0, RW = 0, EN = 1
    write_word(buf);
    delay(2);
    buf &= 0xFB;            // Make EN = 0
    write_word(buf);
}

void send_data(int data){ //send the data in twice
    int buf;
    // Send bit7-4 firstly
    buf = data & 0xF0;
    buf |= 0x05;            // RS = 1, RW = 0, EN = 1
    write_word(buf);
    delay(2);
    buf &= 0xFB;            // Make EN = 0
    write_word(buf);

    // Send bit3-0 secondly
    buf = (data & 0x0F) << 4;
```

```
    buf |= 0x05;            // RS = 1, RW = 0, EN = 1
    write_word(buf);
    delay(2);
    buf &= 0xFB;            // Make EN = 0
    write_word(buf);
}

void init(){
    send_command(0x33);     // Must initialize to 8-line mode at first
    delay(5);
    send_command(0x32);     // Then initialize to 4-line mode
    delay(5);
    send_command(0x28);     // 2 Lines & 5*7 dots
    delay(5);
    send_command(0x0C);     // Enable display without cursor
    delay(5);
    send_command(0x01);     // Clear Screen
    wiringPiI2CWrite(fd, 0x08); //end
}

void clear(){
    send_command(0x01);     //clear Screen
}

void write(int x, int y, char data[]){
    int addr, i;
    int tmp;
    //the position of string
    if (x < 0)    x = 0;
```

```
    if (x > 15) x = 15;
    if (y < 0)    y = 0;
    if (y > 1)    y = 1;

    // Move cursor
    addr = 0x80 + 0x40 * y + x;
    send_command(addr);    //send the x y data

    tmp = strlen(data);
    for (i = 0; i < tmp; i++){
        send_data(data[i]);
    }//send the data
}

void main(){
    fd = wiringPiI2CSetup(LCDAddr);    //the driver file
    init(); //init the lcd
    write(0, 0, "Hello!");//send the hello
    write(1, 1, "From Rexqualis");//send the From Rexqualis
}
```

Code Explanation

```
void write_word(int data){……}
void send_command(int comm){……}
void send_data(int data){……}
void init(){……}
void clear(){……}
void write(int x, int y, char data[]){……}
```

These functions are used to control I2C LCD1602 open source code. They allow us to easily use I2C LCD1602.

Among these functions, init() is used for initialization, clear() is used to clear the screen, write() is used to write what is displayed, and other functions support the above functions.

```
fd = wiringPiI2CSetup(LCDAddr);
```

This function initializes the I2C system with the specified device symbol. The prototype of the function:

```
int wiringPiI2CSetup(int devId);
```

Parameters devId is the address of the I2C device, it can be found through the i2cdetect command(see Appendix) and the devId of I2C LCD1602 is generally 0x27.

```
void write(int x, int y, char data[]){}
```

In this function, data[] is the character to be printed on the LCD, and the parameters x and y determine the printing position (line y+1, column x+1 is the starting position of the character to be printed).

For Python Language Users

Step 3: Change directory.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Python
```

Step 4: Run.

```
sudo python3 13.I2C_LCD1602.py
```

After the code runs, you can see ”Hello!”,”From Rexqualis ” displaying on the LCD.

Code

The code here is for Python3, if you need for Python2, please open the code with the suffix py2 in the attachment.

```
#!/usr/bin/env python3
import LCD1602
import time


def setup():
```

```
    LCD1602.init(0x27, 1)     # init(slave address, background light)
    LCD1602.write(0, 0, 'Hello!')
    LCD1602.write(1, 1, 'from Rexqualis')
    time.sleep(2)

def destroy():
    LCD1602.clear()

if __name__ == "__main__":
    try:
        setup()
    except KeyboardInterrupt:
        destroy()
```

Code Explanation

```
import LCD1602
```

This file is an open source file for controlling I2C LCD1602. It allows us to easily use I2C LCD1602.

```
    LCD1602.init(0x27, 1)
```

The function initializes the I2C system with the designated device symbol. The first parameter is the address of the I2C device, which can be detected through the i2cdetect command (see Appendix for details). The address of I2C LCD1602 is generally 0x27.

```
    LCD1602.write(0, 0, 'Hello!')
```

Within this function, 'Hello! ' is the character to be printed on the Row 0+1, column 0+1 on LCD.

Now you can see “Hello! From Rexqualis” displayed on the LCD.

13.I2C_LCD1602

Phenomenon Picture