

1. Test Platform Introduction

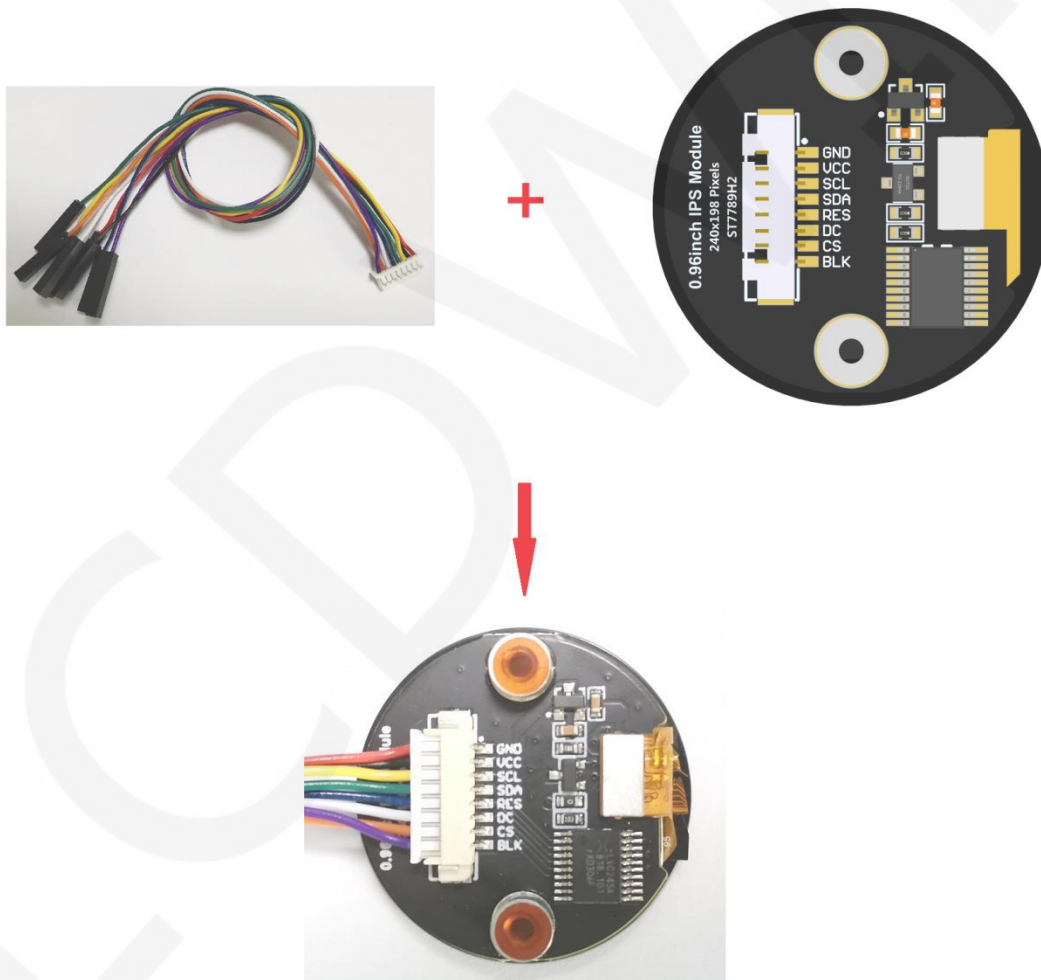
Development board: Raspberry Pi development board for each model

System: Raspberry Pi OS

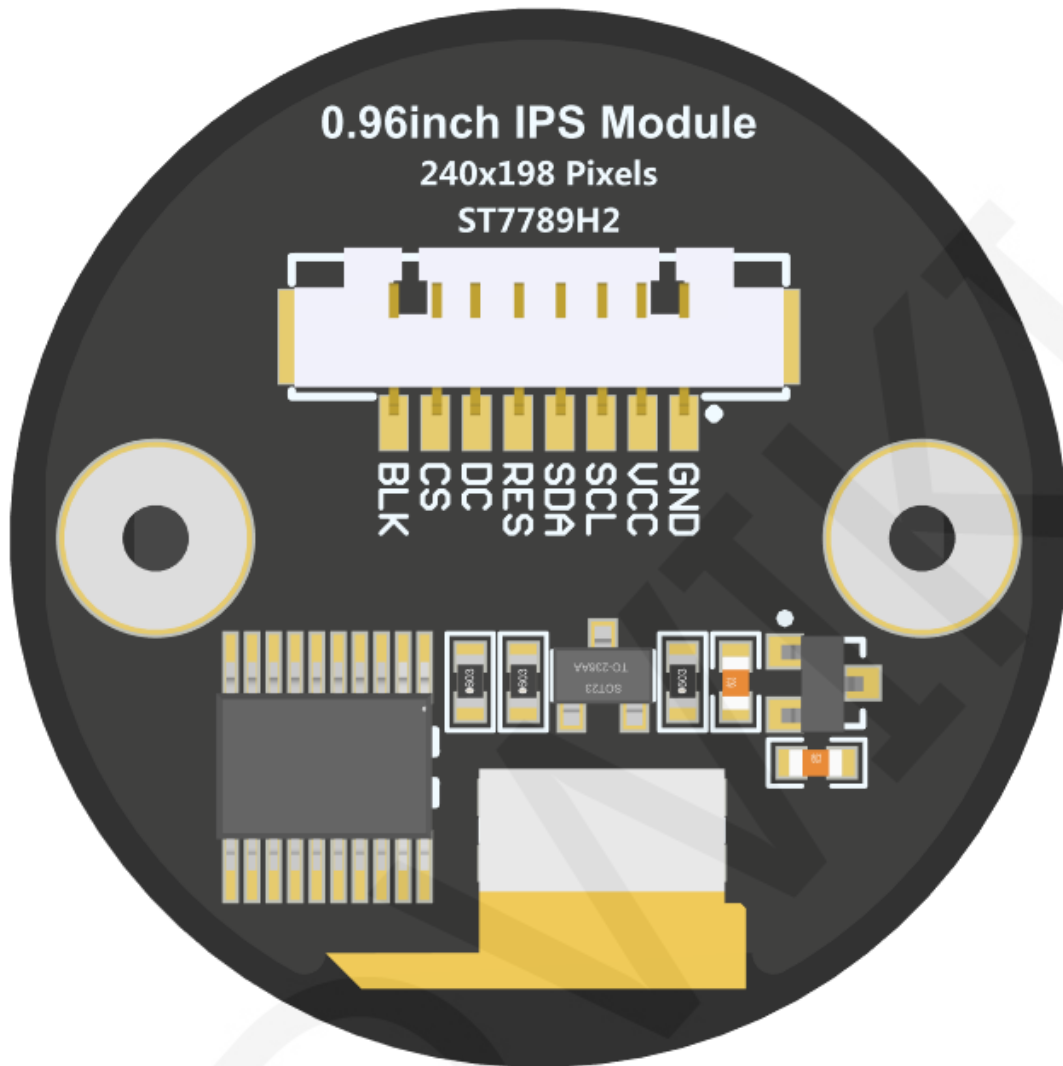
GPIO library: bcm2835、wiringpi

2. Wiring Instructions

The display module is connected to the microcontroller using a 1.25mm spacing 8P DuPont cable with connectors. The module connection is shown in the following figure:



Picture1 Module wiring



Picture2 Module Back Pin

| wiringPi 编码 | BCM 编码 | 功能名 | 物理引脚 BOARD编码 | | 功能名 | BCM 编码 | wiringPi 编码 |
|----------------|-----------|---------|-----------------|----|---------|-----------|----------------|
| | | 3.3V | 1 | 2 | 5V | | |
| 8 | 2 | SDA.1 | 3 | 4 | 5V | | |
| 9 | 3 | SCL.1 | 5 | 6 | GND | | |
| 7 | 4 | GPIO.7 | 7 | 8 | TXD | 14 | 15 |
| | | GND | 9 | 10 | RXD | 15 | 16 |
| 0 | 17 | GPIO.0 | 11 | 12 | GPIO.1 | 18 | 1 |
| 2 | 27 | GPIO.2 | 13 | 14 | GND | | |
| 3 | 22 | GPIO.3 | 15 | 16 | GPIO.4 | 23 | 4 |
| | | 3.3V | 17 | 18 | GPIO.5 | 24 | 5 |
| 12 | 10 | MOSI | 19 | 20 | GND | | |
| 13 | 9 | MISO | 21 | 22 | GPIO.6 | 25 | 6 |
| 14 | 11 | SCLK | 23 | 24 | CE0 | 8 | 10 |
| | | GND | 25 | 26 | CE1 | 7 | 11 |
| 30 | 0 | SDA.0 | 27 | 28 | SCL.0 | 1 | 31 |
| 21 | 5 | GPIO.21 | 29 | 30 | GND | | |
| 22 | 6 | GPIO.22 | 31 | 32 | GPIO.26 | 12 | 26 |
| 23 | 13 | GPIO.23 | 33 | 34 | GND | | |
| 24 | 19 | GPIO.24 | 35 | 36 | GPIO.27 | 16 | 27 |
| 25 | 26 | GPIO.25 | 37 | 38 | GPIO.28 | 20 | 28 |
| | | GND | 39 | 40 | GPIO.29 | 21 | 29 |

Picture3 Raspberry Pi GPIO map

Raspberry Pi Comprehensive test program wiring instructions

| Number | Module Pin | Corresponding to development board wiring pin | Remarks |
|--------|------------|---|---|
| 1 | GND | GND (Physical pin: 6,9,14,20,25,30,34,39) | LCD Power ground |
| 2 | VCC | 5V/3.3V (Physical pin: 1,2,4) | LCD power positive(It is recommended to connect to 5V. When connected to 3.3V, the backlight brightness will be slightly dim) |
| 3 | SCL | Physical pin: 23 BCM coding: 11 wiringPi coding: 14 | LCD SPI bus clock signal |

| | | | |
|---|------------|---|---|
| 4 | SDA | Physical pin: 19 BCM coding: 10 wiringPi coding: 12 | LCD SPI bus write data signal |
| 5 | RES | Physical pin: 5 BCM coding: 3 wiringPi coding: 9 | LCD reset control signal, Low level reset |
| 6 | DC | Physical pin: 3 BCM coding: 2 wiringPi coding: 8 | LCD command / data selection control signal High level: data, low level: command |
| 7 | CS | Physical pin: 24 BCM coding: 8 wiringPi coding: 10 | LCD selection control signal, Low level active |
| 8 | BLK | Physical pin: 12 BCM coding: 18 wiringPi coding: 1 | LCD backlight control signal (If you need control, please connect the pins. If you don't need control, you can skip it) |

Raspberry Pi RPi_Eye_Display test program wiring instructions

| Number | Module Pin | Corresponding to development board wiring pin | | Remarks |
|--------|------------|---|---|---|
| | | Display module 1 | Display module 2 | |
| 1 | GND | GND (Physical pin: 6,9,14,20,25,30,34,39) | | LCD Power ground |
| 2 | VCC | 5V/3.3V (Physical pin: 1,2,4) | | LCD power positive(It is recommended to connect to 5V. When connected to 3.3V, the backlight brightness will be slightly dim) |
| 3 | SCL | Physical pin: 23 BCM coding: 11 wiringPi coding: 14 | Physical pin: 40 BCM coding: 21 wiringPi coding: 29 | LCD SPI bus clock signal |
| 4 | SDA | Physical pin: 19 BCM coding: 10 wiringPi coding: 12 | Physical pin: 38 BCM coding: 20 wiringPi coding: 28 | LCD SPI bus write data signal |
| 5 | RES | Physical pin: 31 BCM coding: 6 wiringPi coding: 22 | | LCD reset control signal, Low level reset |
| 6 | DC | Physical pin: 29 | | LCD command / data |

| | | | | |
|---|-----|--|---|---|
| | | BCM coding: 5 wiringPi coding: 21 | | selection control signal High level: data, low level: command |
| 7 | CS | Physical pin: 24 BCM coding: 8 wiringPi coding: 10 | Physical pin: 36 BCM coding: 16 wiringPi coding: 27 | LCD selection control signal, Low level active |
| 8 | BLK | NC | | LCD backlight control signal (If you need control, please connect the pins. If you don't need control, you can skip it) |

NOTE:

- A. Physical pin refers to the GPIO pin number of the RaspBerry Pi development board
- B. BCM encoding refers to the GPIO pin encoding when using the BCM2835 GPIO library
- C. WiringPi encoding refers to the GPIO pin encoding when using the wiringPi GPIO library
- D. Which GPIO library is used in the code, and the corresponding GPIO library code needs to be used for pin definition. Please refer to the Raspberry Pi GPIO map table in the above figure for details

3. Demo Function Description

This set of testing program is applicable to various models of Raspberry Pi development boards, including the bcm2835 library, wiringPi GPIO library, and Python testing program. Each testing program has hardware SPI functional testing. The test program is located in **Demo_RaspberryPI** directory, as shown in the following figure:



✧ Description of sample program content

The bcm2835, wiringPi, and Python testing programs include the following content:

- A. Main interface display;
- B. Menu interface display;
- C. Simple screen swiping;
- D. Drawing and filling of rectangles;
- E. Draw and fill a circle;
- F. Triangle drawing and filling;
- G. English display;
- H. Chinese display;
- I. Image display;
- J. Dynamic digital display;
- K. Rotating display;

RPi_Eye_ The Display testing program requires the use of two display modules, which display the dynamic rotation of a pair of eyes.

✧ Example program display direction switching instructions

When using the bcm2835 library or wiringPi GPIO library to test the program, find the macro definition **USE_HORIZONTAL** in the `source\include\lcd.h` file, as shown in the following figure:

```
////////////////////////////////////User configuration area////////////////////////////////////  
#define USE_HORIZONTAL 0 //Define the clockwise rotation direction of LCD screen:  
// 0-0 degree rotation, 1-90 degree rotation, 2-180 degree rotation, 3-270 degree rotation
```

Modify **USE_HORIZONTAL** macro according to the following definition:

```
USE_HORIZONTAL 0 //0°rotate  
USE_HORIZONTAL 1 //90°rotate  
USE_HORIZONTAL 2 //180°rotate  
USE_HORIZONTAL 3 //270°rotate
```

When using Python to test program, Find the **USE_HORIZONTAL** definitions in **source\lcd.py** file, as shown in the following figure:

```
USE_HORIZONTAL=0 #Define the clockwise rotation direction of LCD screen:  
| | | | | #// 0-0 degree rotation, 1-90 degree rotation, 2-180 degree rotation, 3-270 degree rotation
```

Modify the **USE_HORIZONTAL** values as follows:

```
USE_HORIZONTAL=0 //0°rotate  
USE_HORIZONTAL=1 //90°rotate  
USE_HORIZONTAL=2 //180°rotate  
USE_HORIZONTAL=3 //270°rotate
```

4. Demo Usage Instructions

✧ Establishing a development environment

Firstly, you need to download the Raspberry Pi system image file from the official website, and then use a microSD card to burn the image file. Please refer to the specific burning method online by yourself.

Official website address: <https://www.raspberrypi.org/downloads/raspbian/>

✧ Enable Raspberry Pi OS kernel SPI kernel drivers

After the system image file is burned, insert the microSD card into the Raspberry Pi, and then connect the display module to the Raspberry Pi using DuPont cables according to pin definitions. Next, plug the Raspberry Pi into the internet cable, and finally power up the Raspberry Pi. Open the terminal software (such as putty) and log in to Raspberry Pi using SSH (ensure that Raspberry Pi and

PC are on the same network segment). Enter the following command in the terminal software:

sudo raspi-config

select **Interfacing Options ->SPI ->YES** in the pop-up interface,After completing the selection, press the **Esc** key continuously to save and exit the graphical interface, At this point, the kernel SPI kernel driver is turned on.

✧ Installing GPIO software library

A. Install the bcm2835 library

There are three methods for downloading software libraries:

1) If Raspberry Pi cannot connect to the internet, it can be downloaded from a PC and then copied to Raspberry Pi OS using an SD card or FTP tool (such as FileZilla).

Download website: <http://www.airspayce.com/mikem/bcm2835/>

Enter the website and click on the link shown below to download.

C library for Broadcom BCM 2835 as used in Raspberry Pi

This is a C library for Raspberry Pi (RPI). It provides access to GPIO and other IO functions on the Broadcom BCM 2835 chip, as used in the RaspberryPi, allowing access to devices.

It provides functions for reading digital inputs and setting digital outputs, using SPI and I2C, and for accessing the system timers. Pin event detection is supported by poll. Works on all versions up to and including RPI 4. Works with all versions of Debian up to and including Debian Buster 10. Reported to be working on Bullseye (Raspbian v

It is C++ compatible, and installs as a header file and non-shared library on any Linux-based distro (but clearly is no use except on Raspberry Pi or another board with BC

The version of the package that this documentation refers to can be downloaded from <http://www.airspayce.com/mikem/bcm2835/bcm2835-1.73.tar.gz> You can find the

Several example programs are provided.

2) If Raspberry Pi can connect to the internet, it can be downloaded by entering the following command in Raspberry Pi OS through terminal software:

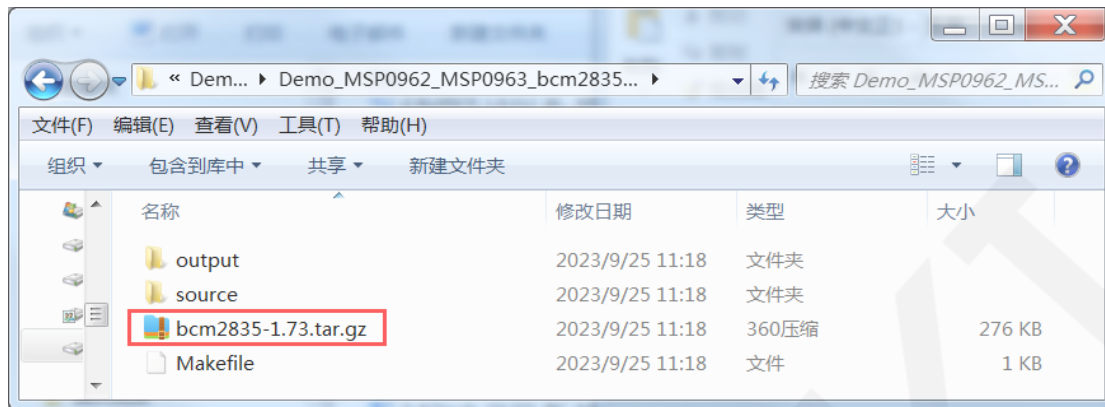
wget http://www.airspayce.com/mikem/bcm2835/bcm2835-X.XX.tar.gz

Among them, **X.XX** is the software library version number, as shown in the above figure. The version number is 1.73, so you can use the following command to download version 1.73:

wget http://www.airspayce.com/mikem/bcm2835/bcm2835-1.73.tar.gz

3) Directly use the bcm2835 library in the sample program directory (as shown in the following figure), and copy it to Raspberry Pi OS through an SD card or FTP

tool (such as FileZilla).



After the bcm2835 library is successfully downloaded or copied, enter the following command in the terminal software to decompress, compile, and install:

```
tar -zxvf bcm2835-X.XX.tar.gz
cd bcm2835-X.XX
./configure
sudo make
sudo make check
sudo make install
```

Among them, X.XX is the version number of the bcm2835 library, which needs to be filled in according to the actual situation, such as 1.73

B. Install wiringPi GPIO library

There are four methods for downloading software libraries:

1) If Raspberry Pi cannot connect to the internet, it can be downloaded from a PC and then copied to Raspberry Pi OS using an SD card or FTP tool (such as FileZilla).

Download website: <https://project-downloads.drogon.net/wiringpi-latest.deb>

Enter the website address in the browser or click on it to download.

2) If Raspberry Pi can connect to the internet, it can be downloaded by entering the following command in Raspberry Pi OS through terminal software:

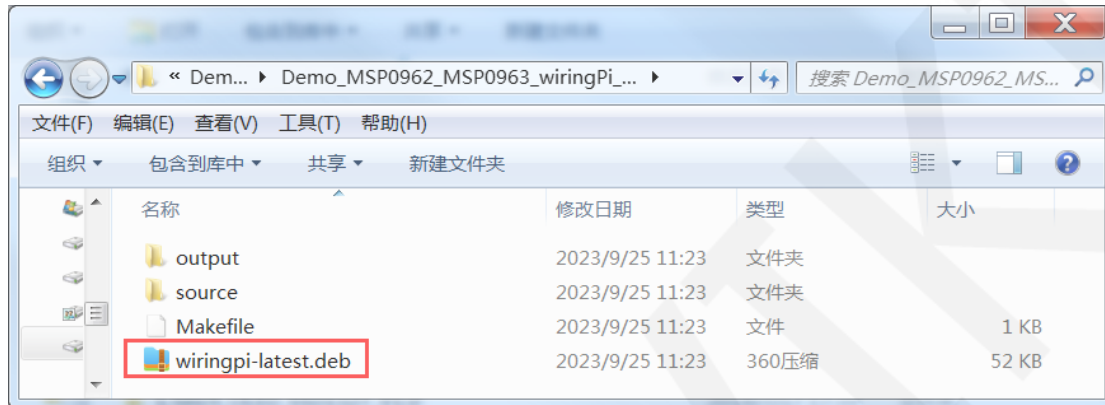
```
wget https://project-downloads.drogon.net/wiringpi-latest.deb
```

3) If Raspberry Pi can connect to the internet, it can also be downloaded from github by entering the following command in Raspberry Pi OS through terminal

software:

```
git clone https://github.com/WiringPi/WiringPi
```

- 4) Directly use the wiringPi GPIO library in the sample program directory (as shown in the following figure), and copy it to Raspberry Pi OS through an SD card or FTP tool (such as FileZilla).



If the software library is not downloaded or copied from GitHub, enter the following command in the terminal software for compilation and installation:

```
sudo dpkg -i -B wiringpi latest.deb
```

If the software library is downloaded from Github, enter the following command in the terminal software for compilation and installation:

```
cd Wiring Pi
```

```
./build
```

After the installation of the wiringPi GPIO library is completed, you can enter the following command in the terminal software to check if the installation is successful

```
gpio -v
```

```
gpio readall
```

As shown in the following figure, the red box displays the version number and GPIO encoding number of the wiringPi library. If these contents do not appear, it indicates that the installation was not successful.

```

pi@raspberrypi:~ $ gpio -v
gpio version: 2.52
Copyright (c) 2012-2018 Gordon Henderson
This is free software with ABSOLUTELY NO WARRANTY.
For details type: gpio -warranty

Raspberry Pi Details:
Type: Pi 4B, Revision: 04, Memory: 8192MB, Maker: Sony
* Device tree is enabled.
*--> Raspberry Pi 4 Model B Rev 1.4
* This Raspberry Pi supports user-level GPIO access.
pi@raspberrypi:~ $ gpio readall
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| BCM | wPi |   Name   | Mode | V | Physical | V | Mode |   Name   | wPi | BCM |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|     |     |   3.3v   |      |   | 1 | 2 |     |   5v     |     |     |
|  2  |  8  | SDA.1    | OUT  | 1 | 3 | 4 |     |   5v     |     |     |
|  3  |  9  | SCL.1    | OUT  | 1 | 5 | 6 |     |   0v     |     |     |
|  4  |  7  | GPIO. 7  | IN   | 1 | 7 | 8 | 1 | IN  | TxD    | 15  | 14  |
|     |     |   0v     |      |   | 9 | 10 | 1 | IN  | RxD    | 16  | 15  |
| 17  |  0  | GPIO. 0  | IN   | 0 | 11 | 12 | 1 | OUT | GPIO. 1 | 1   | 18  |
| 27  |  2  | GPIO. 2  | IN   | 0 | 13 | 14 |   |     |   0v   |     |     |
| 22  |  3  | GPIO. 3  | IN   | 0 | 15 | 16 | 0 | IN  | GPIO. 4 | 4   | 23  |
|     |     |   3.3v   |      |   | 17 | 18 | 0 | IN  | GPIO. 5 | 5   | 24  |
| 10  | 12  | MOSI     | ALT0 | 0 | 19 | 20 |   |     |   0v   |     |     |
|  9  | 13  | MISO     | ALT0 | 0 | 21 | 22 | 0 | IN  | GPIO. 6 | 6   | 25  |
| 11  | 14  | SCLK     | ALT0 | 0 | 23 | 24 | 1 | OUT | CE0    | 10  |  8  |
|     |     |   0v     |      |   | 25 | 26 | 1 | ALT0 | CE1    | 11  |  7  |
|  0  | 30  | SDA.0    | IN   | 1 | 27 | 28 | 1 | IN  | SCL.0  | 31  |  1  |
|  5  | 21  | GPIO.21  | IN   | 1 | 29 | 30 |   |     |   0v   |     |     |
|  6  | 22  | GPIO.22  | IN   | 1 | 31 | 32 | 0 | IN  | GPIO.26 | 26  | 12  |
| 13  | 23  | GPIO.23  | IN   | 0 | 33 | 34 |   |     |   0v   |     |     |
| 19  | 24  | GPIO.24  | IN   | 0 | 35 | 36 | 0 | IN  | GPIO.27 | 27  | 16  |
| 26  | 25  | GPIO.25  | IN   | 0 | 37 | 38 | 0 | IN  | GPIO.28 | 28  | 20  |
|     |     |   0v     |      |   | 39 | 40 | 0 | IN  | GPIO.29 | 29  | 21  |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| BCM | wPi |   Name   | Mode | V | Physical | V | Mode |   Name   | wPi | BCM |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+

```

C. Installing Python libraries

At present, the latest Raspberry Pi OS is compatible with Python 2 using the Python 3 library, so only the Python 3 library needs to be installed. Before installation, execute the following command on the terminal software to check if the required Python 3 libraries have been installed on the system. If all of them have been installed, there is no need to install them again.

```
dpkg -l | grep -e python3-pip -e python3-pil -e python3-numpy -e spider
```

As shown in the figure below, it indicates that all have been installed and no further installation is required.

```

pi@raspberrypi:~ $ dpkg -l | grep -e python3-pip -e python3-pil -e python3-numpy -e spider
ii python3-numpy      1:1.19.5-1          armhf      Fast
ii python3-pil:armhf  8.1.2+dfsg-0.3+deb11u1 armhf      Pytho
ii python3-pip        20.3.4-4+rpt1+deb11u1 all        Pytho
ii python3-spidev     20200602~200721-1  armhf      Bindi

```

If not installed, execute the following command to install the terminal software:

```
sudo apt get update
sudo apt get install python3 pip
sudo apt get install python3 pil
sudo apt get install python3 numpy
sudo pip3 install RPi. GPIO
sudo pip3 install spider
```

If the Raspberry Pi OS running is using Python 2, execute the following command to install on the terminal software:

```
sudo apt get update
sudo apt get install python pip
sudo apt get install python pil
sudo apt get install python numpy
sudo pip install RPi. GPIO
sudo pip install spider
```

✧ Compile and Run Programs

A. Compile and run the bcm2835 library test program

Use an SD card or FTP tool (such as FileZilla) to copy the

Demo_MSP0962_MSP0963_bcm2835_Hardware_SPI folder which is in the

data package **Demo_RaspberryPI** directory to Raspberry Pi OS, and then

execute the following command in the terminal software to compile and run the program:

```
cd Demo_MSP0962_MSP0963_bcm2835_Hardware_SPI
sudo make
sudo ./0.96_IPS_LCD
```

```

pi@raspberrypi:~ $
pi@raspberrypi:~ $ cd Demo MSP0962 MSP0963 bcm2835 Hardware SPI/
pi@raspberrypi:~/Demo MSP0962 MSP0963 bcm2835 Hardware SPI $ sudo make
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/src/delay.c -o /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/include
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/src/gui.c -o /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/include
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/src/lcd.c -o /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/include
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/src/main.c -o /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/include
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/src/spi.c -o /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/include
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/src/test.c -o /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/source/include
gcc -g -O0 /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/output/delay.o /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/output/lcd.o /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/output/main.o /home/pi/Demo MSP0962 MSP0963 bcm2835 Hardware SPI/output/test.o -o 0.96_IPS_LCD -lbcm2835
pi@raspberrypi:~/Demo MSP0962 MSP0963 bcm2835 Hardware SPI $ sudo ./0.96_IPS_LCD

```

B. Compile and run the wiringPi GPIO library test program

Use an SD card or FTP tool (such as FileZilla) to copy the

Demo_MSP0962_MSP0963_wiringPi_Hardware_SPI folder which is in the data package **Demo_RaspberryPI** directory to Raspberry Pi OS, and then execute the following command in the terminal software to compile and run the program:

```

cd Demo_MSP0962_MSP0963_wiringPi_Hardware_SPI
sudo make
sudo ./0.96_IPS_LCD

```

```

pi@raspberrypi:~ $
pi@raspberrypi:~ $ cd Demo MSP0962 MSP0963 wiringPi Hardware SPI/
pi@raspberrypi:~/Demo MSP0962 MSP0963 wiringPi Hardware SPI $ sudo make
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/src/delay.c -o /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/include
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/src/gui.c -o /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/include
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/src/lcd.c -o /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/include
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/src/main.c -o /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/include
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/src/spi.c -o /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/include
gcc -g -O0 -c /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/src/test.c -o /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/source/include
gcc -g -O0 /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/output/delay.o /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/output/lcd.o /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/output/main.o /home/pi/Demo MSP0962 MSP0963 wiringPi Hardware SPI/output/test.o -o 0.96_IPS_LCD -lwiringPi
pi@raspberrypi:~/Demo MSP0962 MSP0963 wiringPi Hardware SPI $ sudo ./0.96_IPS_LCD

```

C. Compile and run Python test programs

Use an SD card or FTP tool (such as FileZilla) to copy the

Demo_MSP0962_MSP0963_python_Hardware_SPI folder which is in the data package **Demo_RaspberryPI** directory to Raspberry Pi OS, and then execute the following command in the terminal software to compile and run the program.

Using Python 3, execute the following command:

```
cd Demo_MSP0962_MSP0963_python_Hardware_SPI/source
sudo python3 0.96_IPS_LCD.py
```

```
pi@raspberrypi:~$ cd Demo_MSP0962_MSP0963_python_Hardware_SPI/source/
pi@raspberrypi:~/Demo_MSP0962_MSP0963_python_Hardware_SPI/source$ sudo python3 0.96_IPS_LCD.py
```

Using Python 2, execute the following command:

```
cd Demo_MSP0962_MSP0963_python_Hardware_SPI/source
sudo python 0.96_IPS_LCD.py
```

```
pi@raspberrypi:~$ cd Demo_MSP0962_MSP0963_python_Hardware_SPI/source/
pi@raspberrypi:~/Demo_MSP0962_MSP0963_python_Hardware_SPI/source$ sudo python 0.96_IPS_LCD.py
```

D. Compile and run RPi_Eye_Display test programs

Use an SD card or FTP tool (such as FileZilla) to copy the **RPi_Eye_Display** folder which is in the data package

Demo_RaspberryPiDemo_MSP0962_MSP0963_RPi_Eye_Display_Hardware_SPI directory to Raspberry Pi OS, Then execute the following command on the terminal software to run the installation program.

```
sudo chmod -R 755 RPi_Eye_Display
cd RPi_Eye_Display
sudo ./rpi_eye.sh
```

```
pi@raspberrypi:~$ sudo chmod -R 755 RPi_Eye_Display/
pi@raspberrypi:~$ cd RPi_Eye_Display/
pi@raspberrypi:~/RPi_Eye_Display$ sudo ./rpi_eye.sh

Starting installation...
Updating package index files...
```

After the program installation is completed, it will automatically restart.

Afterwards, it will automatically run every time it is turned on.