



# 0.91inch OLED Module

## User Manual

### OVERVIEW

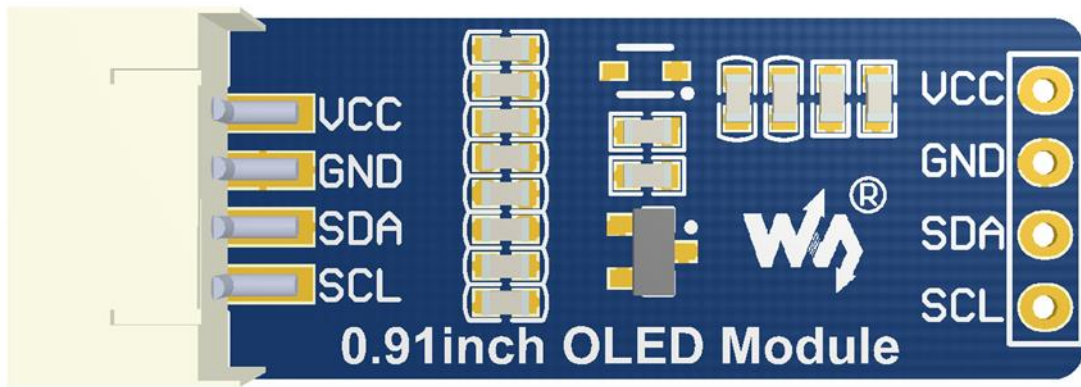
This is a general OLED display Module, 0.91inch diagonal, 128x32 pixels, with embedded controller, communicating via I2C interface.

### FEATURES

Controller:	SSD1306
Interface:	I2C
Resolution:	128*32
Display Size:	1.5inch
Display Color:	White
Operating Voltage:	2.2V/5V

### PINS

PIN	Description
VCC	Power
GND	Ground
SDA	Data input
SCL	Clock input

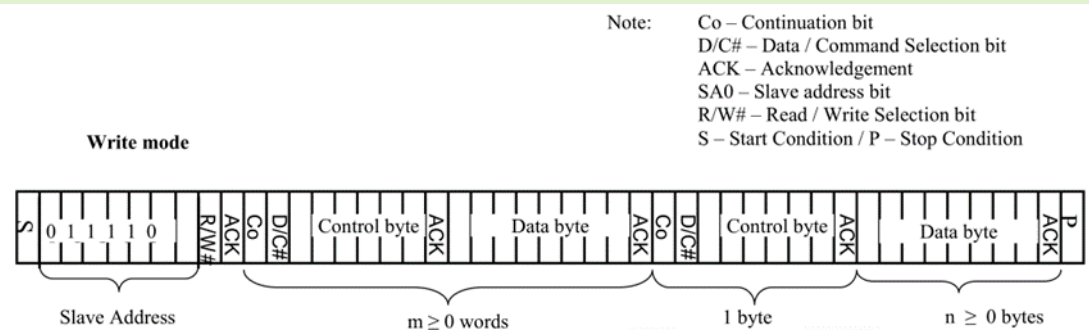


## WORKING PROTOCOL

SSD1306 is a controller for 128\*64 OLED. This OLED has only 128\*32 pixels, so it uses part of SSD1306' s buffer.

In theory, the OLED supports 8-bit 8080, 8-bits 6800, 3-wires SPI, 4-wires SPI and I2C, however, to save IO resources and because of the small size of OLED, we only pinout I2C interface.

## I2C COMMUNICATION



When working, MCU will first send a byte that the first 7bits are address of slave device and 1 bit write/read bit, and wait for response.

After received response from slave device, MCU will send a control byte, this byte defined the data following is command or data.

Slave response again, if sending command, MCU will send the command which is one byte. If sending data, MCU will sending data

For more details about I2C, please refer to Datasheet Page20 Figure 8-7

## HOW TO USE

We provide STM32, Arduino and Raspberry Pi demo code for this module. The demo code will release basic functions that: draw point, draw line, draw rectangle, draw circle.

## STM32 DEMO CODE

### 1. Hardware configuration

Development board: XNUCLEO-F103RB

PIN	XNUCLEO-F103RB
VCC	3V3/5V
GND	GND
SDA	SDA/D14
SCL	SCL/D15

### 2. Project files:

Project is compiled in MDK-ARM v5, generated by STM32CubeMX

../Src:

Adafruit\_SSD1306.cpp: Bottom interface of OLED, provide functions that OLED initialize, basic display pixels and configure;

Adafruit\_GFX.cpp: Application function of OLED, provide display, drawing

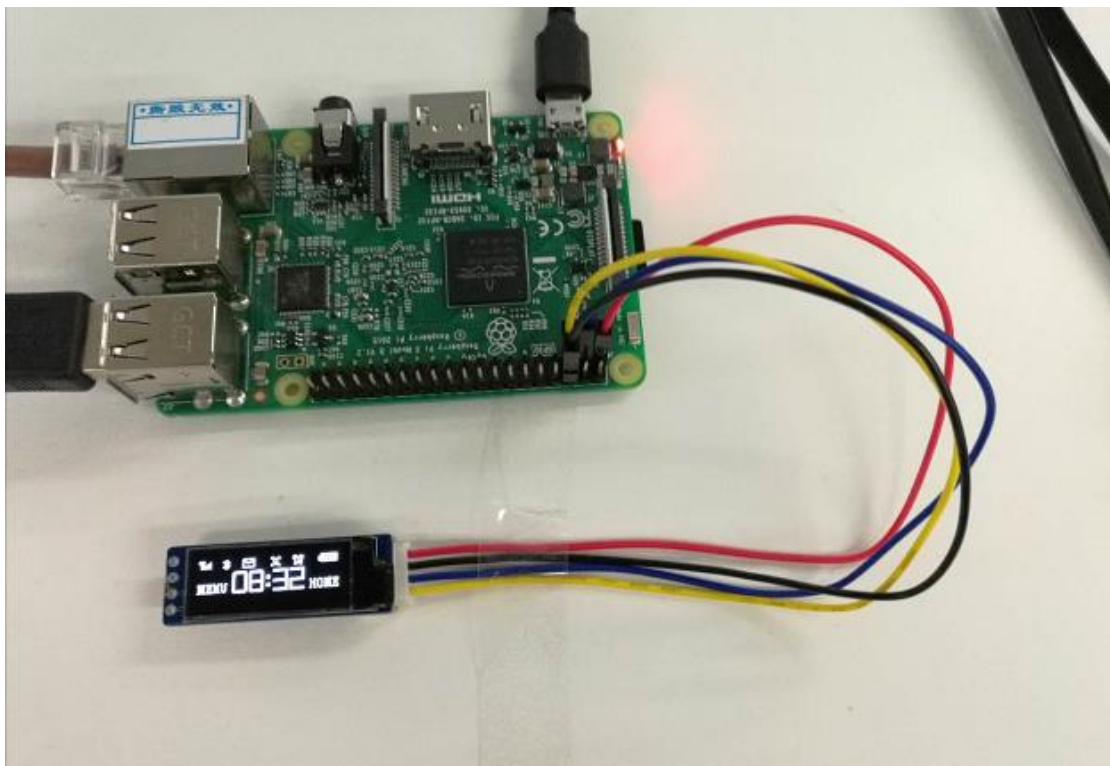
functions.

glcdfont.h: LCD font, provide English fonts which size 6\*8 and 8\*16

## RASPBERRY PI CODE

### 1. Hardware connection

PIN	Pi
VCC	3V3/5V
GND	GND
SDA	SDA
SCL	SCL



### 2. Enable I2C

```
sudo raspi-config
```

choose Interfacing Options->I2C ->Yes

```

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 location
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 SSH
P3 VNC         Enable/Disable graphical remote access to your Pi using RealVNC
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 connection
P7 1-Wire      Enable/Disable one-wire interface
P8 Remote GPIO Enable/Disable remote access to GPIO pins

<Select>                                <Back>

```

### 3. Libraries installation

About how to install I2C libraries, you can refer to Waveshare Wiki:

[https://www.waveshare.com/wiki/Libraries\\_Installation\\_for\\_RPi](https://www.waveshare.com/wiki/Libraries_Installation_for_RPi)

### 4. Using

Copy demo code which you can download from Wiki to Raspberry Pi. The demo code we described are all copied to /home/pi

#### a) BCM2835

(1) Install bcm2835 libraries

(2) use **ls** command to list the files:

```
pi@raspberrypi:~/0in91/bcm2835 $ ls
bin  Fonts  Makefile  obj  oled_0in91
```

bin: ./o files

Fonts: Include five fonts files

Obj: project files are saved here, include main.c, OLED\_Driver.c,

OLED\_Config.c, OLED\_GUI.c and their header files.

mian.c: main function

OLED\_Config.c: Hardware configuration, define pins and communication type

OLED\_Driver.c: Hardware (OLED) driver.

OLED\_GUI.c: Application functions, included functions that draw point, line, rectangle, display string, pictures and so on.

Show\_Pic.h: Pictures data which are used to display. You should convert your pictures to data. (description in net chapter)

oled\_0in91: executable files, generated by command **make**

To run this code, you can execute the command: **sudo ./oled\_0in91**

## b) WiringPi

(1) Install WiringPi libraries

(2) use **ls** command to list the files:

```
pi@raspberrypi:~/0in91/wiringpi $ ls  
bin  Fonts  Makefile  obj  oled_0in91
```

The folders included are similar to BCM2835' s. The only differences are that:

- WiringPi oprates by read/write the device files of Linux OS. and the

bcm2835 is library function of Raspberry Pi' s CPU, it operates registers

directly. Thus, if you have used bcm2835 libraries firstly, the usage of WiringPi code will be failed. In this case, you just need to reboot the system and try again.

- Due to the first difference, they underlying configuration are different. In

DEV\_Config.c, use wiringpiPi and the corresponding wiringPiSPI to provide

underlay interfaces.

To run the code, use the command: **sudo ./oled\_0in91**

c) Python

(1) use **ls** command to list the files:

```
pi@raspberrypi:~/0in91/python $ ls
Adafruit_Python_SSD1306  stats.py
```

(2) Here we used Adafruit libraries

before run code, you need to install libraries as below:

```
sudo apt-get update
```

```
sudo apt-get install build-essential python-dev python-pip
```

```
sudo pip install RPi.GPIO
```

```
sudo apt-get install python-imaging python-smbus
```

(3) Enter directory of python code, execute commands:

```
sudo python Adafruit_Python_SSD1306/setup.py install
```

```
sudo python stats.py
```

d) Auto-run

To make the code run automatically after booting, you can configure

/etc/rc.local file:

```
sudo vim /etc/rc.local
```

Add a statement in front of exit 0:

```
sudo python /home/pi/0in91/python/stats.py &
```

Note that if you put the code to different directory, you need to change the

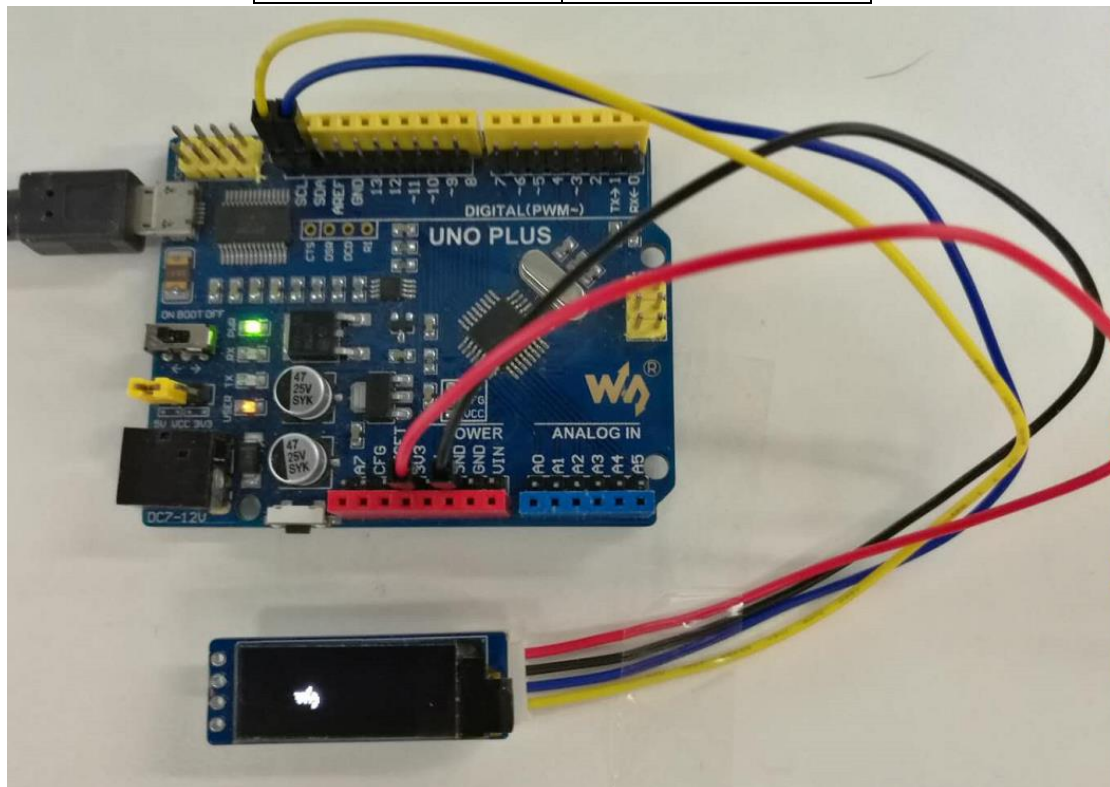
path: **/home/pi/** to the correct one. & is necessary at the end, otherwise, you may cannot login to Raspberry Pi and need to re-burn image)

## ARDUINO CODE

### 1. Hardware Connection

Development board: UNO PLUS

PIN	UNO PLUS
VCC	3V3/5V
GND	GND
SDA	SDA
SCL	SCL



### 2. Files Description:

../oled:



oled.ino: Project file of Arduino, double click to open

Project directory:

Adafruit\_SSD1306.cpp: Bottom interfaces of OLED, includes OLED initialize, basic display and configuration functions.

Adafruit\_GFX.cpp: Application functions of OLED

Adafruit\_SSD1306.h、Adafruit\_GFX.h: Header files

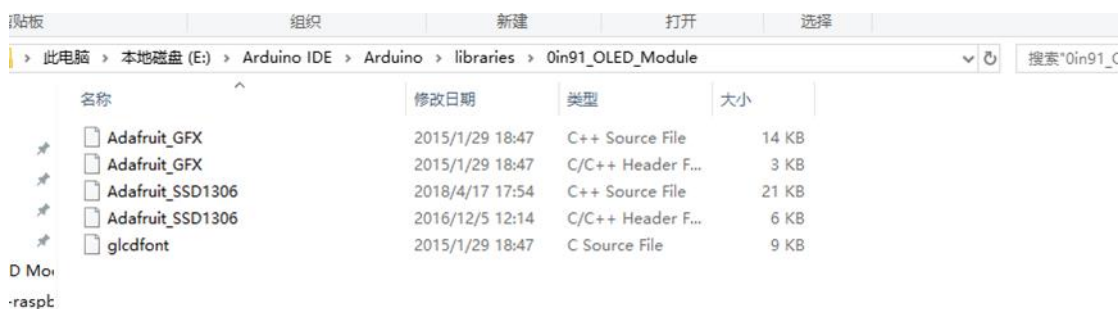
glcdfont.h: LCD font, provide English fonts which size 6\*8 and 8\*16

### 3. Running

Before running the code, you should copy the libraries files of this project to the

**libraries** directory of IDE, which is under the installation directory of Arduino IDE.

Note that you cannot put files directly to the libraries directory, you need to save them on a folder, for example 0in91\_OLED\_Module as below:



Then, open olde.ino then download the code

## IMAGE DATA

Use software Image2Lcd to open picture (Monochrome picture) and configure:

输出数据类型 (Data types) : C 语言数据(\*.c)

扫描方式(Scanning way): 数据水平 (data horizontal), 字节垂直 (byte vertical)

输出灰度(gray scale): 单色 (monochrome)

最大宽度和高度(height and width): 128 32 (Resolution of OLED)

And then check the option that color invert.

