Using an I2C OLED Display Module with the Raspberry Pi

27

BY MATT ON APRIL 8, 2018 I2C, TUTORIALS & HELP

Miniature OLED display modules are a great way to add a small screen to your Raspberry Pi projects. They are available in various sizes but common sizes include 128×32 and 128×64 pixels. The cheaper ones have single colour pixels that are either white, yellow or blue. My device has white pixels and uses an I2C interface which only requires four wires to be connected to the Pi.

In this tutorial I'll explain how I setup my 0.96" OLED display module using the Pi's I2C interface. Once setup it is easy to use Python to place text, draw shapes or even display simple images and animations.

The OLED Module

My OLED display module is a 0.96" I2C IIC SPI Serial 128X64 OLED LCD LED Display Module.

Search ... Search

RECENT POSTS

NOVEMBER 13, 2019 1
Ntablet an Open-source Tablet

OCTOBER 21, 2019 2
Pi-Hole OLED Status Screen

SEPTEMBER 22, 2019 0
KKSB Raspberry Pi 4 Steel Case

JUNE 24, 2019 0
Introducing the Raspberry Pi 4

JUNE 19, 2019 2
Gameboy Zero 6 Button Board from Aliexpress

CATEGORIES

1-wire
3D Printing
Add-ons
BBC Micro:bit
BerryClip
Books
Camera Module
Cases
Events
General
Hardware
I2C
Infographics

It has four pins. Two are power (Vcc and Gnd) and two are for the I2C interface (SDA and SCL). The header may need to be soldered on before you can use it.

Update Operating System

As with all my projects I started off by creating an SD card with the latest Raspbian image. Then I made sure this was up-to-date by running the following commands :

```
sudo apt-get update
sudo apt-get upgrade
```

This step may take a few minutes if there are lots of packages to update but it usually saves some frustration in the future.

Display Module Setup

My screen had four pins, two for power and two for the I2C interface.

Follow on Twitter 18K
Subscribe on YouTube 1.4K
Like on Facebook
Follow on Instagram

TAGS

3D Printing audio battery berryclip Birthday bluetooth cambridge camera CamJam DigiMakers display games GPIO I2C interface Kickstarter LCD LED Linux media Minecraft Model A Model B motionEyeOS PCB photography photos Pi-Lite portable power python Raspberry Jam Raspberry Pi Bootcamp raspbian Retrogaming retroPie screen SD card security sensor SPI temperature ultrasonic video WiFi

RASPBERRY PI RELATED

I connected them directly to the Raspberry Pi's GPIO header using the following scheme :

OLED Pin	Pi GPIO Pin	Notes
Vcc	1 *	3.3V
Gnd	14 **	Ground
SCL	5	I2C SCL
SDA	3	I2C SCA

* You can connect the Vcc pin to either Pin 1 or 17 as they both provide 3.3V.
** You can connect the Gnd pin to either Pin 6, 9, 14 , 20, 25, 30, 34 or 39 as they all provide Ground.

Enable I2C Interface

The I2C interface is disabled by default so you need to enable it. You can do this within the raspi-config tool on the command line by running :

```
sudo raspi-config
```

For additional details on this step please see my how to Enable the I2C Interface on the Raspberry Pi post.

The following libraries may already be installed but run these commands anyway to make sure :

```
sudo apt install -y python3-dev
sudo apt install -y python-imaging python-smbus i2c-tools
sudo apt install -y python3-pil
sudo apt install -y python3-pip
sudo apt install -y python3-setuptools
sudo apt install -y python3-rpi.gpio
```

If you are using Python 2 then use these commands instead :

```
sudo apt install -y python-dev
sudo apt install -y python-imaging python-smbus i2c-tools
sudo apt install -y python-pil
sudo apt install -y python-pip
sudo apt install -y python-setuptools
```

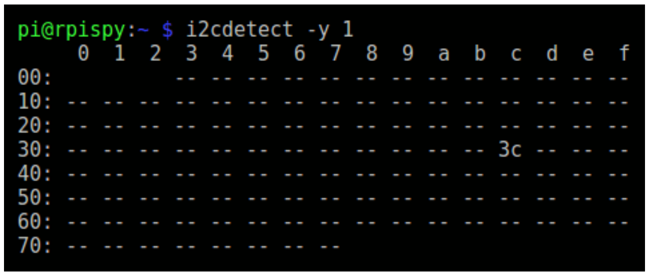
I would recommend using Python 3 unless you have a really good reason for using Python 2.

Finding the OLED Display Module's Address

With the I2C libraries installed I used the i2cdetect command to find the module on the I2C bus.

```
i2cdetect -y 1
```

and I got the following result :



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

Adafruit Blog
Average Maker
Official RaspBerry Pi Site
Raspberry Pi Pod
RasPi.tv
RaspTut
Stuff About Code

TECH RESOURCES

MattsBits – Pi Resources
Microbit Spy
Technology Spy

ARCHIVES

Select Month

This was good news as it showed the device had been detected with an address of “0x3c”. This is the default hex address for this type of device. I’ve got no idea why the device PCB suggests the address is “0x78” when it is clearly “0x3c”.

If you’ve got an original Model B Rev 1 Pi then type the following command instead :

```
i2cdetect -y 0
```

Install OLED Python Library

In order to display text, shapes and images you can use the Adafruit Python library. It should work with all SSD1306 based displays including their own 128×32 and 128×64 devices.

To install the library we will clone the Adafruit git repository. Ensure git is installed by running :

```
sudo apt install -y git
```

Then clone the repository using the following command :

```
git clone https://github.com/adafruit/Adafruit_Python_SSD1306.git
```

Once that completes navigate to the library’s directory :

```
cd Adafruit_Python_SSD1306
```

and install the library for Python 2 :

```
sudo python setup.py install
```

and/or Python 3 :

```
sudo python3 setup.py install
```

This process will give you ability to include the library within your own Python scripts.

Example Python Scripts

Now we are ready to test some examples scripts. Navigate into the “examples” directory :

```
cd examples
```

In there you should find a number of example scripts such as :

- animate.py
- buttons.py
- image.py
- shapes.py
- stats.py

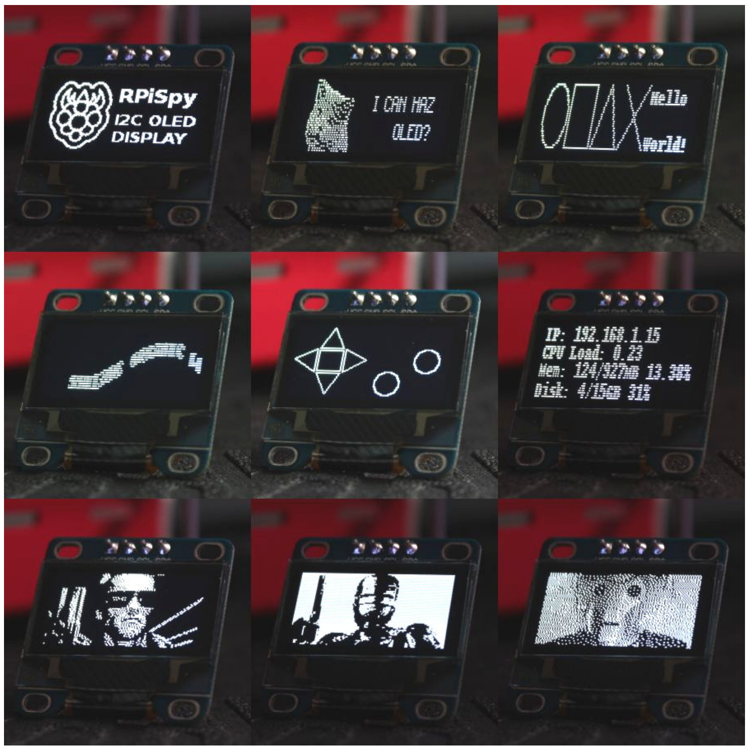
These examples can be run using :

```
python shapes.py
```

or using Python 3 :

```
python3 shapes.py
```

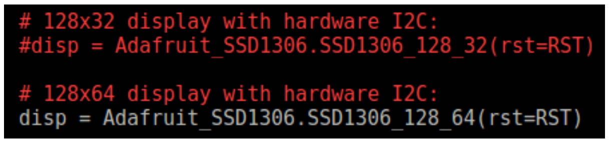
The examples should give you screens that appear in the examples below :



By modifying these scripts you can create your own graphics with shapes, images and text depending on your project. See if you can guess which ones were photos I downloaded to my Pi from Google Images!

Screen Size Adjustment

The Adafruit examples assume you have a 128×32 screen. They still run with a 128×64 pixel screen but it is better to change them before you move onto anything more complicated. To do this simply edit the scripts and disable the 128×32 config line by placing a # character at the front, and enable the 128×64 line by deleting the # character from the front. The section in the script should now look like this :



```
# 128x32 display with hardware I2C:
#disp = Adafruit_SSD1306.SSD1306_128_32(rst=RST)

# 128x64 display with hardware I2C:
disp = Adafruit_SSD1306.SSD1306_128_64(rst=RST)
```

This step becomes essential if you want to start creating your own images to display on the screen.

Creating New Images

So you tried image.py example and wondered how you can create your own images? It's fairly easy if you have an image editing application such as Photoshop or GIMP. I prefer using GIMP because it is free.

Ideally you want the images to be :

- 128×64 resolution
- 1-bit colour (i.e. black and white)

By default the image.py example will convert the image to 1-bit but it assumes the resolution is correct.

You'll notice in the script an alternative line which resizes and converts an image so you can load images without worrying about their size and colour.

Load image and convert to 1-bit color :

```
image = Image.open('happycat_oled_64.ppm').convert('1')
```

Load an image, resize and convert to 1-bit :

```
image = Image.open('example.png').resize((disp.width, disp.height), Image.ANTIALIAS).convert('1')
```

Which technique you use is up to you. Resizing and converting takes extra processing time so in high performance applications you are better feeding the script images that have already been resized.

Resizing and Converting Images

If you want to load an image or photo then load it into your graphics application and perform the following steps :

- Load image
- Resize/scale to 128×64
- Convert to 1-bit colour (monchrome)
- Export as a ".pbm" or ".png" file
- Copy to your Pi in the same location as your Python script
- Update the Python script to use your new file

The Adafruit example image is a "ppm" file because it is colour although it is converted to monochrome at the point it is displayed on the screen. Adafruit use ppm as the library also supports their colour OLED modules. If you don't have a colour screen you can switch to pbm or png.

I prefer creating "pbm" files as they are black and white and much smaller files. It also means your Python script doesn't need to convert them. The library can handle both just make sure you use the correct filename and extension in your scripts.

Increasing I2C Bus Speed

If you are displaying multiple images per second it is worth increasing the bus speed of the interface as it can improve performance. Please see the Change Raspberry Pi I2C Bus Speed post .

Troubleshooting

If your screen isn't working you should start at the beginning of this tutorial and work through it. Here are some thing to consider :

- Did you enable I2C and instal "python-smbus" and "i2c-tools"?
- Are the four module connections correct? Did you get SDA and SCL mixed up?
- Did "i2cdetect -y 1" give you the address of the display on the I2C bus?
- If your screen is using an address other than 0x3c did you adjust the Python script?

Buy a Miniature OLED Screen

These screens are available from a number of retailers so take a look and pick one that is convenient for your location :

- Adafruit
- The PiHut
- ModMyPi
- eBay
- Amazon

Read the descriptions carefully as some OLED display modules use the SPI interface rather than I2C. Those are fine but you'll need to follow a different tutorial to use that style.

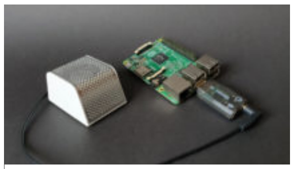
SHARE.

PREVIOUS ARTICLE
Raspberry Pi RetroPie Shutdown Button

NEXT ARTICLE
Cheap SD Cards from eBay are Fake

RELATED POSTS

OCTOBER 21, 2019 2
Pi-Hole OLED Status Screen


JUNE 4, 2019 4
Using a USB Audio Device with the Raspberry Pi

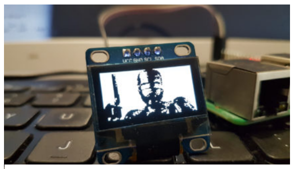
FEBRUARY 5, 2019 1
Setting up SSH Keys on the Raspberry Pi

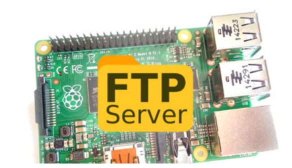
DECEMBER 15, 2018 3
Running Flask under NGINX on the Raspberry Pi

DECEMBER 8, 2018 3
Remote Access to a Raspberry Pi using MobaXterm

NOVEMBER 5, 2018 0
Raspberry Pi 7-Segment LED Display Module using Python

SEPTEMBER 4, 2018 0
Using a Level Shifter With the Raspberry Pi GPIO


JUNE 16, 2018 0
Create an I2C OLED Display Slideshow with Python


MAY 13, 2018 1
Creating a Simple FTP Server with a Raspberry Pi

27 COMMENTS

JES WOODLAND on APRIL 20, 2018 7:53 AM

Thanks Matt. This has been a really helpful tutorial. There is very little info regarding these little screens beyond setting them up. This is far and away the most concise and well explained tutorial I have read about displaying your own images.

REPLY

PIETRO on MAY 15, 2018 5:07 PM

Hello,
Thank you for the helpful tutorial.
I have a problem while installing the python library. When i execute the command

sudo python setup.py install

i get the following output:

Extracting in /tmp/tmpoaljab
Traceback (most recent call last):
File "setup.py", line 4, in
use_setuptools()
File "/home/pi/Adafruit_Python_SSD1306/ez_setup.py", line 128, in use_setuptools
return _do_download(version, download_base, to_dir, download_delay)
File "/home/pi/Adafruit_Python_SSD1306/ez_setup.py", line 108, in _do_download