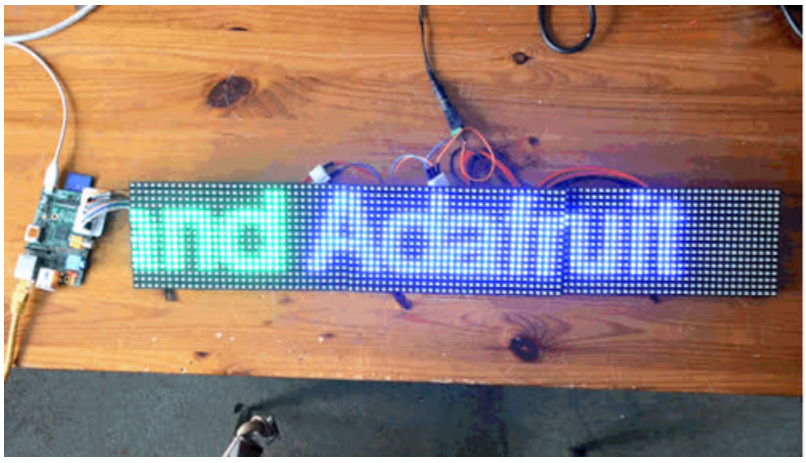
adafruit learning system

16x32 RGB Display with Raspberry Pi - part 2

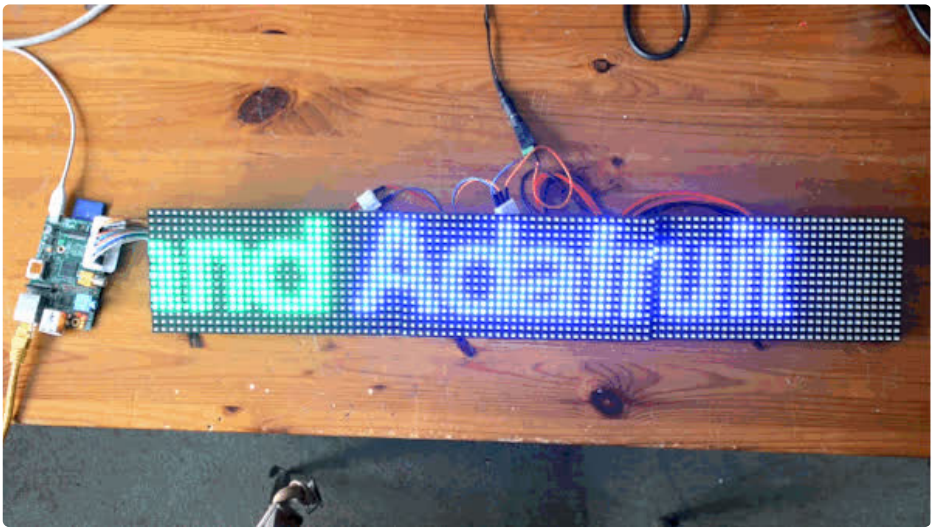
Created by Simon Monk



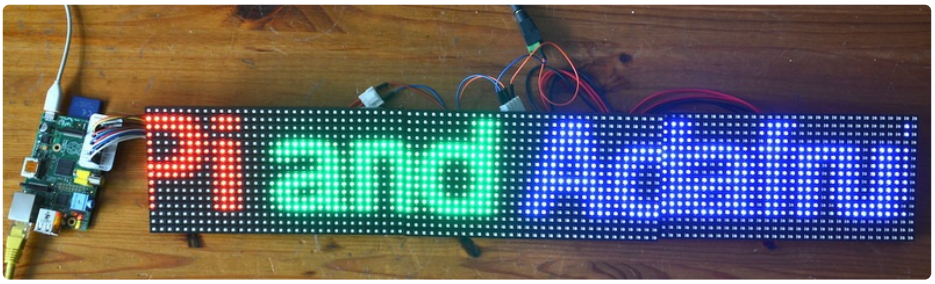
Last updated on 2021-10-22 11:03:23 AM EDT

Guide Contents

Overview



In the first tutorial (https://adafru.it/dfF) using the 16x32 RGB display, we used just one of these displays with the software developed by Henner Zeller (https://adafru.it/dcp)
In this tutorial, we will look at daisy chaining together two or three of these displays and also wrapping Henner's fast and efficient C code in some Python that will create images on the fly and populate them with text before handing them over to the C code to handle the refreshing of the display.



You Will Need

To build this project, you will need the following items:

Raspberry Pi (we're using the Model B) (http://adafru.it/998)

Adafruit 16x32 RGB LED Matrix (http://adafru.it/420) - we're using 3 of them!

5V 4A (http://adafru.it/1466) (2 panels any design, or 3 panels with text or otherwise not filling up all the pixels with bright color) or
5V 10A Power Supply (any number of panels, any designs) (http://adafru.it/658)

2.1mm to Screw Jack Adapter - to connect the 5V power adapter to the panels (http://adafru.it/368)

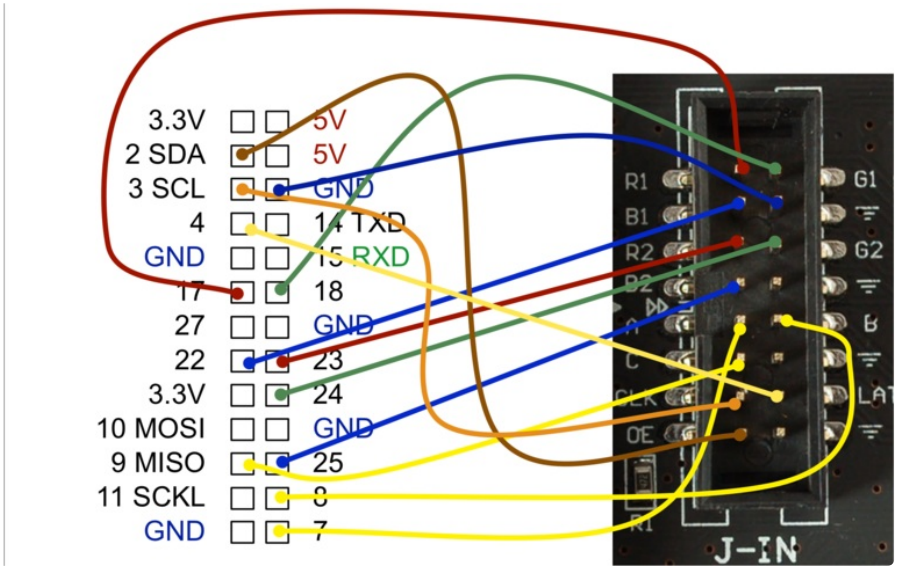
14 x Female-Female jumper wires. You can use 6" ones (http://adafru.it/266) or 12" ones. (http://adafru.it/793)

6 x Male-Male jumper wires - you can also just use wire.

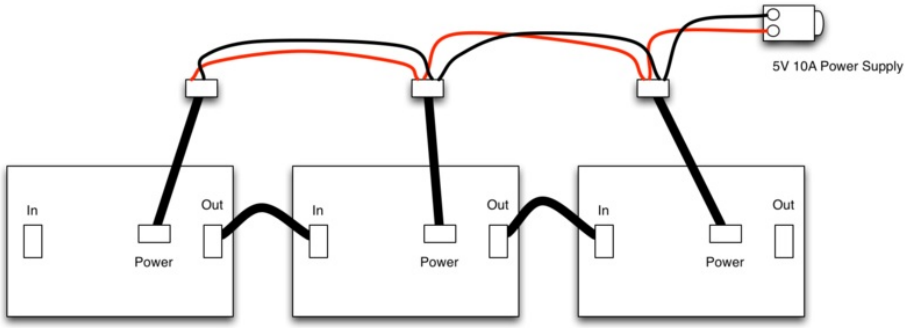
Raspberry Leaf - helps make wiring easier (http://adafru.it/1772)

Wiring the Displays

If you have not already done so, refer back to this tutorial (https://adafru.it/dfF) to wire up the first of the displays as shown below.



Each of the displays comes with both data and power connectors. Use the data connectors to bridge from the connector labelled OUTPUT on the first display to INPUT of the next display. If you have three displays then also bridge the data to the final display.



Attach the power connectors for all the displays to the displays and then use male to male jumper wires as shown above to connect all the red +5V connectors together and all the black negative connections together. Note that as long as you are connecting red-wired terminals on the power leads to other red-wired connections and black connections to the other black connections, it does not matter which of the connections on the four pin power connectors you use.

Finally use male to male jumper wires on the final display to connect it to the screw terminals of the 2.1mm to Screw Jack Adapter.

The power supply must be 5V at 4A (mostly text) or 10A (all pixels used, bright graphics). DO NOT USE A POWER SUPPLY OF MORE THAN 5V.

Software

Follow the instructions here (https://adafru.it/dfG) to install the software, but before you type the command “make”, there are a couple of changes that need to be made to the code.

Change to the directory using the command:

```
$ cd ~/display16x32/rpi-rgb-led-matrix
```

Then edit the file led-matrix.h using nano:

```
$ nano led-matrix.h
```

Look for the line below, that sets the pixel width. Its near the top of the file. Change the value of width from 32 to either 64 (if you have two displays) or 96 if you have three.

```
  int width() const { return 96; }
```

Next, look a little further down the same file and modify the value of kChainedBoards to the number of boards you have. In this case 3.

```
enum {
  kDoubleRows = 8,     // Physical constant of the used board.
  kChainedBoards = 3,   // Number of boards that are daisy-chained.
  kColumns = kChainedBoards * 32,
  kPWMBits = 4          // maximum PWM resolution.
};
```

Having changed the code, you now need to rebuild the project, so type the command “make”.

You can now run the original demo program, to check that everything is working, using the command:

```
$ sudo ./led-matrix
```

The square pattern will disappear of the screen some of the time, as it was designed for the 32x32 matrix. You can also use scrolling image example by running:

```
$ sudo ./led-matrix 1 runtext.ppm
```

or use an image of your own creation.

Using Python

To be able to display any text we want on the displays without having to design an image, you can use the following Python program that first creates an image containing the text and then runs the “led-matrix” C program to display it.

This program uses the Python Imaging Library (PIL). To install PIL, enter the following commands.

```
$ sudo apt-get update
$ sudo apt-get install python-imaging
```

Type the command below to create the Python program file:

```
$ nano message.py
```

.. and then paste in the following code.

```
import os
from PIL import ImageFont
from PIL import Image
from PIL import ImageDraw

text = (("Raspberry Pi ", (255, 0, 0)), ("and ", (0, 255, 0)), ("Adafruit", (0, 0, 255)))


font = ImageFont.truetype("/usr/share/fonts/truetype/freefont/FreeSans.ttf", 16)
all_text = ""
for text_color_pair in text:
    t = text_color_pair[0]
    all_text = all_text + t

print(all_text)
width, ignore = font.getsize(all_text)
print(width)


im = Image.new("RGB", (width + 30, 16), "black")
draw = ImageDraw.Draw(im)

x = 0;
for text_color_pair in text:
    t = text_color_pair[0]
    c = text_color_pair[1]
    print("t=" + t + " " + str(c) + " " + str(x))
    draw.text((x, 0), t, c, font=font)
    x = x + font.getsize(t)[0]

im.save("test.ppm")

os.system("./led-matrix 1 test.ppm")
```

Now run the program and you should see the message scroll across your displays.

You can change the message and its use of colors by editing the variable "text". This contains a collection of entries, each in the form of a piece of text followed by the color for that text.

```
("Raspberry Pi ", (255, 0, 0))
```

You can change the colour of the text, by changing the color tuple which is set to (255, 0, 0) in the example above. There three values corresponding to the amount of red, green and blue in the color, with 0 being none 255 being full brightness. For white, use (255, 255, 255).

Although this example has three sections of differently colored text, you can add as many as you like, to create longer messages.

Next Steps

If you want use a different font to display your text, then you can find the fonts available on your Raspberry Pi using the command:

```
$ fc-list
```

To use the font, edit message.py changing the path to the font.

As well as text, the Python Imaging Library also allows you to draw shapes etc on the image. Take a look at the documentation for PIL (https://adafru.it/dfH) here for ideas on how to do this.

About the Author

As well as contributing lots of tutorials about Raspberry Pi, Arduino and now BeagleBone Black, Simon Monk writes books about open source hardware including 'The Raspberry Pi Cookbook' and 'Programming Rasppberry Pi'. You will find his books for sale here (https://adafru.it/caH) at Adafruit.