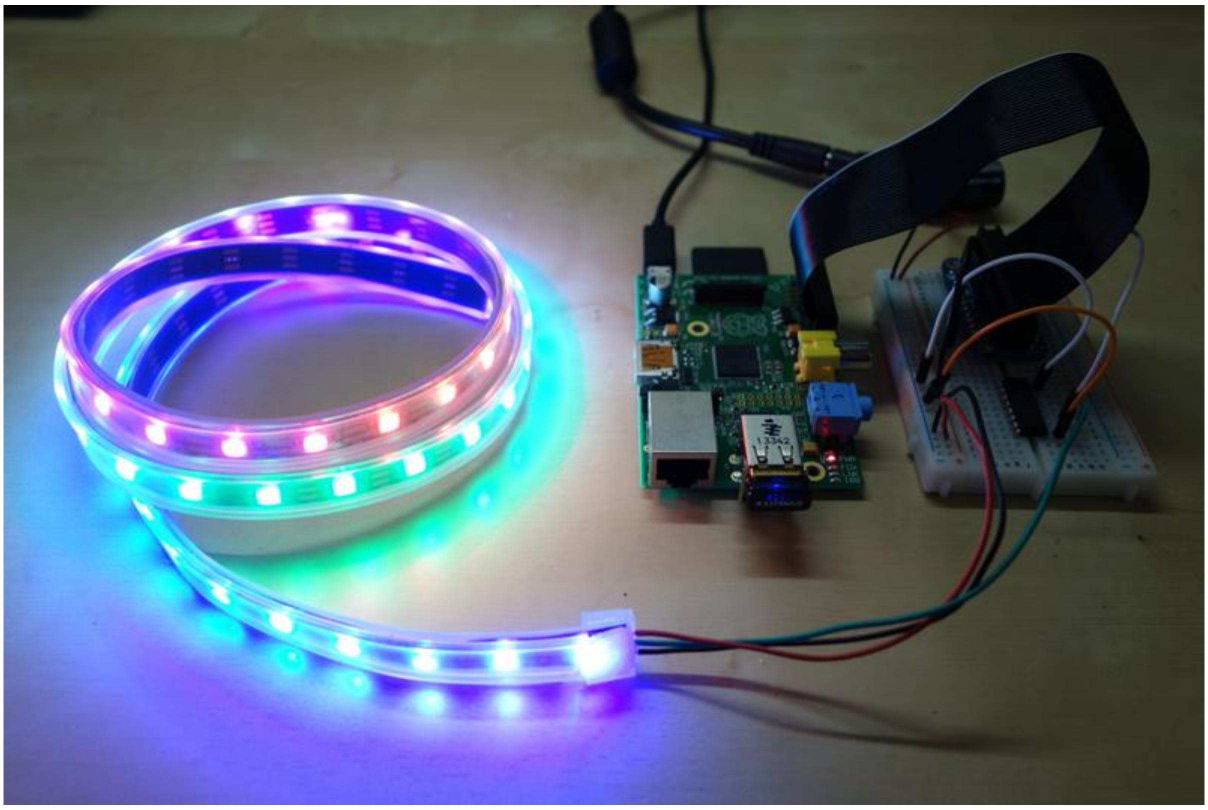
NeoPixels on Raspberry Pi

Created by Abigail Torres



https://learn.adafruit.com/neopixels-on-raspberry-pi

Last updated on 2023-08-29 02:38:21 PM EDT

Table of Contents

Overview

Wouldn't it be fun to add bright, beautiful NeoPixels to your Raspberry Pi project? NeoPixels, and the WS2811/2812 LEDs that make them up, require a data signal with very specific timing to work correctly. Because the Raspberry Pi runs a multi-tasking Linux operating system it doesn't have real-time control over its GPIO pins and can't easily drive NeoPixels. Typically a small microcontroller like a Trinket or Teensy can be used to communicate with the Raspberry Pi and generate the NeoPixel data signal. But you're in luck! Thanks to the Adafruit CircuitPython NeoPixel () library, you can now control NeoPixels or WS2811/WS2812 LEDs directly from your Raspberry Pi!

The Adafruit CircuitPython NeoPixel library solves the real-time control problem by using the PWM and DMA hardware on the Raspberry Pi's processor. The PWM (pulse-width modulation) module can generate a signal with a specific duty cycle (), for example to drive a servo or dim an LED. The DMA (direct memory access) module can transfer bytes of memory between parts of the processor without using the CPU. By using DMA to send a specific sequence of bytes to the PWM module, the NeoPixel data signal () can be generated without being interrupted by the Raspberry Pi's operating system.

The great thing about this library is that it does all the hard work of setting up PWM and DMA to drive NeoPixels. You can use these LEDs with a single-board computer (like Raspberry Pi!) that has GPIO and Python thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library ().

Before you get started you will want to be familiar with how to connect to a Raspberry Pi's terminal using SSH (). It will also be helpful to check out the NeoPixel Uberguide ( ) for more information on using NeoPixels.

Raspberry Pi Wiring

Wiring NeoPixels to work with a Raspberry Pi is quite simple. The only issue to deal with is converting the Pi's GPIO from 3.3V up to about 5V for the NeoPixel to read. There are two ways you can do this level conversion, either with a simple 1N4001 power diode or with a level converter chip like the 74AHCT125.

Note that you might be able to get your NeoPixels to work without any level conversion, but it's not really guaranteed because the data line needs to be at least 0.7 * VDD (5 volts), or about 3.5 volts. Try one of the level conversion options below if you can't directly drive the pixels from your Raspberry Pi.

The diode method is a quick way to reduce the power supply voltage slightly so the NeoPixels can read the Pi's 3.3V output. However you need to be careful to use a diode that can handle all the current drawn by the NeoPixels. The diodes Adafruit sells only handle 1 Amp of continuous current so they're good for driving up to about 16 NeoPixels at full 100% bright white - and about 50 NeoPixels if they're all lit with various colors. Also because the NeoPixels aren't running at 5 volts they might be a little dimmer than normal.

A level converter chip like the 74AHCT125 is a better method because it will convert the Pi's 3.3V output up to 5V without limiting the power drawn by the NeoPixels. You'll get full NeoPixel brightness that's only limited by the current capability of the power supply.

Sound must be disabled to use GPIO18. This can be done in /boot/config.txt by changing "dtparam=audio=on" to "dtparam=audio=off" and rebooting.

Note that the pixel will not light up automatically simply by applying power. You need to program the Pi to send the commands to turn them on.

NeoPixels must be connected to GPIO10, GPIO12, GPIO18 or GPIO21 to work! GPIO18 is the standard pin.

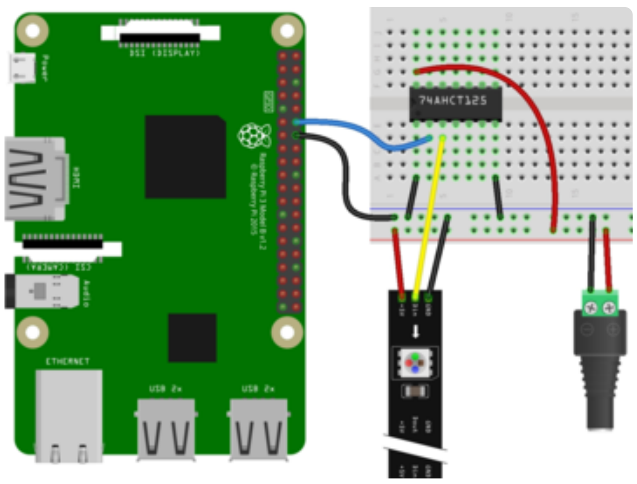
Be aware, you can only create one strip at a time! If you have more than one, connect them together and then wire them to your Raspberry Pi using a single connection.

You can use the following wiring diagrams to connect your NeoPixels to your Raspberry Pi.

You can use ANY Raspberry Pi computer (Zero, A+, Pi 4, etc!) but this guide is not for 'Raspberry Pi shaped' boards that are not Raspberry Pi (e.g. Banana Pi, etc)

Raspberry Pi Wiring with Level Shifting Chip

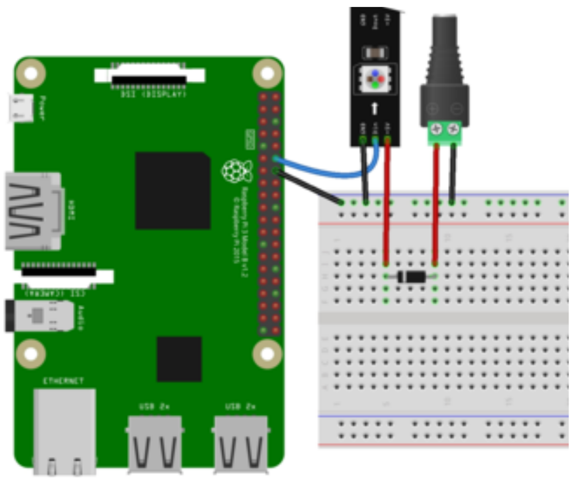
If you're using the 74AHCT125 level converter chip, wire up your Raspberry Pi as follows:



Pi GPIO18 to 74AHCT125 pin 1A
74AHCT125 pin 1Y to NeoPixel DIN
Power supply ground to 74AHCT125 ground
Power supply ground to 74AHCT125 pin 1OE
Power supply ground to Pi GND
Power supply ground to NeoPixel GND
Power supply 5V to 74AHCT125 VCC
Power supply 5V to NeoPixel 5V.

Raspberry Pi Wiring with Diode

If you're using a 1N4001 diode wire up your hardware as follows:



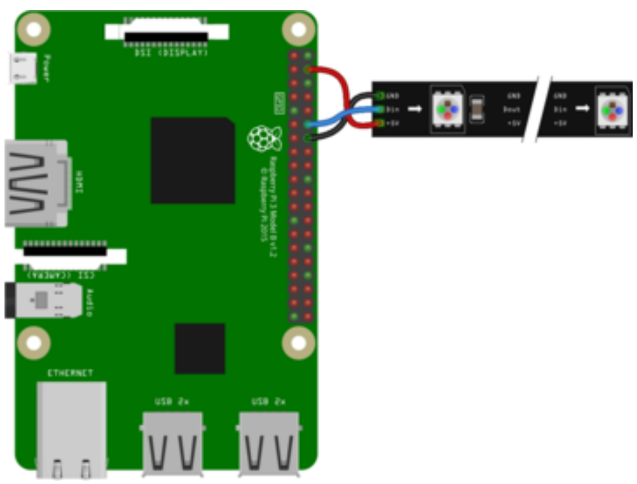
Pi pin 18 to NeoPixel DIN.
1N4001 diode cathode (side with the stripe) to NeoPixel 5V.
Power supply ground to Pi ground.
Power supply ground to NeoPixel GND.
Power supply 5V to 1N4001 diode anode (side without the stripe).
Make sure to get the orientation of the diode correct, with the cathode (side with the stripe) going to the NeoPixel!

Powering NeoPixels from Raspberry Pi Without Level Shifting

Remember, your NeoPixels may not work connected directly to the Raspberry Pi without a level shifter. If you run into issues, try adding a level shifter to your project.

Do not power more than a few NeoPixels from the Raspberry Pi's 5V output! The Pi cannot source enough current to light many pixels and will be damaged. Use a good quality external 5V power supply that can handle the current demands of all the pixels.

If you're only powering a few pixels, you can power them from the Raspberry Pi 5V pin.

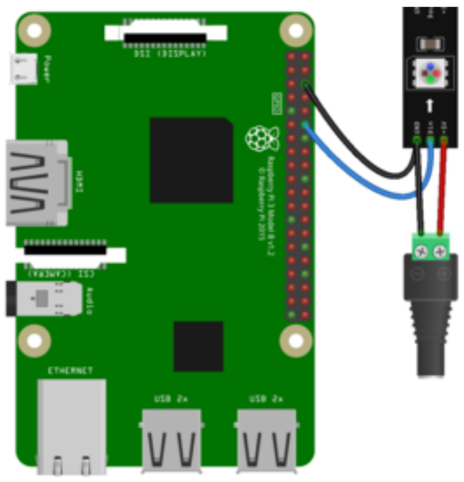


Pi 5V to NeoPixel 5V
Pi GND to NeoPixel GND
Pi GPIO18 to NeoPixel Din

Using External Power Source Without Level Shifting

If you're going to be using more than a few pixels, it's a good idea to connect an external power source. Remember each pixel can draw up to 60mA so don't skimp on the power supply!

Remember, your NeoPixels may not work connected directly to the Raspberry Pi without a level shifter. If you run into issues, try adding a level shifter to your project.



Pi GND to NeoPixel GND
Pi GPIO18 to NeoPixel Din
Power supply ground to NeoPixel GND
Power supply 5V to NeoPixel 5V

Python Usage

Sound must be disabled to use GPIO18. This can be done in /boot/config.txt by changing "dtparam=audio=on" to "dtparam=audio=off" and rebooting. Failing to do so can result in a segmentation fault.

Python Installation of NeoPixel Library

You'll need to install the Adafruit_Blinka library that provides the CircuitPython support in Python. This may also require verifying you are running Python 3. Since each platform is a little different, and Linux changes often, please visit the CircuitPython on Linux guide to get your computer ready ()!

Once that's done, from your command line run the following command:

- `sudo pip3 install rpi_ws281x adafruit-circuitpython-neopixel`
- `sudo python3 -m pip install --force-reinstall adafruit-blinka`

If your default Python is version 3 you may need to run 'pip' instead. Just make sure you aren't trying to use CircuitPython on Python 2.x, it isn't supported!

Python Usage

To demonstrate the usage of this library with NeoPixel LEDs, we'll use the Python REPL.

For NeoPixels to work on Raspberry Pi, you must run the code as root! Root access is required to access the RPi peripherals.

Run the following code to import the necessary modules and initialise a NeoPixel strip with 30 LEDs. Don't forget to change the pin if your NeoPixels are connected to a different pin, and change the number of pixels if you have a different number.

```
import board
import neopixel
pixels = neopixel.NeoPixel(board.D18, 30)
```

Depending on the specific NeoPixels you have connected, you may need to add some additional parameters to the initializer. Here are a couple common ones:

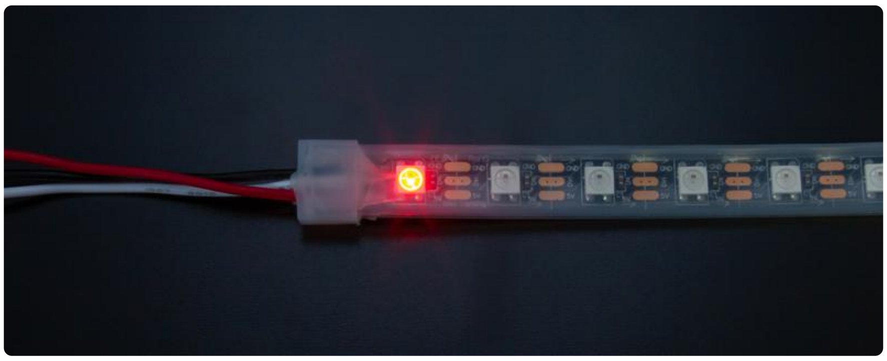
- bpp - Bytes Per Pixel. Defaults to 3. Set this to 4 if you have RGBW NeoPixels. If you try to fill the NeoPixels with a solid color and they light up as different colors, you will probably want to change this.
- pixel_order - If you are seeing the NeoPixels light up as the same colors, but they are a different color than you expect, you may need to change this value.

Now you're ready to light up your NeoPixel LEDs using the following properties:

- brightness - The overall brightness of the LED
- fill - Color all pixels a given color.
- show - Update the LED colors if `auto_write` is set to `False`.

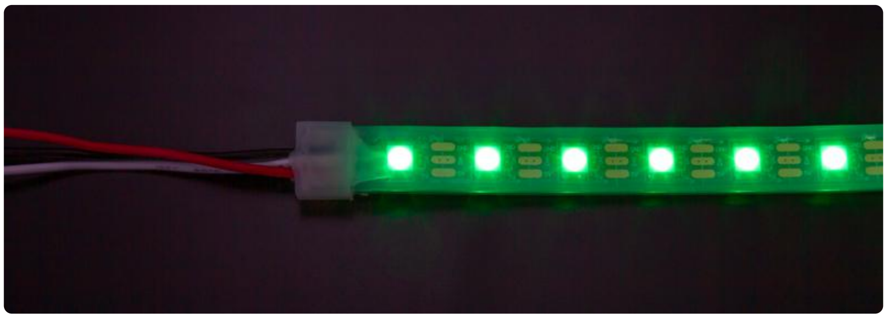
For example, to light up the first NeoPixel red:

```
pixels[0] = (255, 0, 0)
```



To light up all the NeoPixels green:

```
pixels.fill((0, 255, 0))
```



That's all there is to getting started with NeoPixel LEDs on Raspberry Pi!

Below is an example program that repeatedly turns all the LEDs red, then green, then blue, and then goes through a single rainbow cycle. If you chose a pin other than `D18` for your NeoPixels, change the pin to match the pin you chose to use to connect the NeoPixels to your Raspberry Pi. If you're using a different number of NeoPixels, change `num_pixels` to match.

Full Example Code

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

# Simple test for NeoPixels on Raspberry Pi
import time
import board
import neopixel


# Choose an open pin connected to the Data In of the NeoPixel strip, i.e. board.D18
# NeoPixels must be connected to D10, D12, D18 or D21 to work.
pixel_pin = board.D18

# The number of NeoPixels
num_pixels = 30

# The order of the pixel colors - RGB or GRB. Some NeoPixels have red and green
reversed!
# For RGBW NeoPixels, simply change the ORDER to RGBW or GRBW.
ORDER = neopixel.GRB

pixels = neopixel.NeoPixel(
    pixel_pin, num_pixels, brightness=0.2, auto_write=False, pixel_order=ORDER
)


def wheel(pos):
    # Input a value 0 to 255 to get a color value.
    # The colours are a transition r - g - b - back to r.
    if pos < 0 or pos > 255:
        r = g = b = 0
    elif pos < 85:
        r = int(pos * 3)
        g = int(255 - pos * 3)
        b = 0
    elif pos < 170:
        pos -= 85
        r = int(255 - pos * 3)
        g = 0
        b = int(pos * 3)
    else:
        pos -= 170
        r = 0
        g = int(pos * 3)
        b = int(255 - pos * 3)
    return (r, g, b) if ORDER in (neopixel.RGB, neopixel.GRB) else (r, g, b, 0)


def rainbow_cycle(wait):
    for j in range(255):
        for i in range(num_pixels):
            pixel_index = (i * 256 // num_pixels) + j
            pixels[i] = wheel(pixel_index & 255)
        pixels.show()
        time.sleep(wait)


while True:
    # Comment this line out if you have RGBW/GRBW NeoPixels
    pixels.fill((255, 0, 0))
    # Uncomment this line if you have RGBW/GRBW NeoPixels
    # pixels.fill((255, 0, 0, 0))
    pixels.show()
    time.sleep(1)

    # Comment this line out if you have RGBW/GRBW NeoPixels
    pixels.fill((0, 255, 0))
    # Uncomment this line if you have RGBW/GRBW NeoPixels
    # pixels.fill((0, 255, 0, 0))
    pixels.show()
    time.sleep(1)

    # Comment this line out if you have RGBW/GRBW NeoPixels
    pixels.fill((0, 0, 255))
    # Uncomment this line if you have RGBW/GRBW NeoPixels
    # pixels.fill((0, 0, 255, 0))
    pixels.show()
    time.sleep(1)

    rainbow_cycle(0.001)  # rainbow cycle with 1ms delay per step
```

Python Docs

Python Docs ()