adafruit learning system

NeoPixel GoPro Lens Light

Created by Liz Clark

Last updated on 2018-08-22 04:05:36 PM UTC

Guide Contents

Overview and Materials

GoPros and other small action cams are great for project documentation because they can capture interesting angles without getting in the way of your build. You may not always have optimal lighting in your workspace though, which is where a lens mounted light can come in handy.

All of the code and hardware is running off of a Trinket M0 board using the Arduino IDE or CircuitPython depending on your preference. A NeoPixel 16 x RGBW ring is used for the light since it fits perfectly around the lens with some breathing room. This particular temperature of white (~4500K) looks very natural on camera and doesn't give off a blue hue. A potentiometer is also in the circuit to adjust the brightness of the light. All of the components fit snuggly into a 3D printed case that fits over the GoPro's lens.

This light's housing is designed to fit with the GoPro Hero 3+ which is a slightly older model, but the file can be edited to fit other models or even different types of action cams.

For this project you'll need:

1 x NeoPixel Ring - 16 x 5050 RGBW LEDs w/ Integrated Drivers - Natural White - ~4500K

NeoPixel RGBW Ring

ADD TO CART

1 x Adafruit Trinket M0 - for use with CircuitPython / Arduino IDE

Trinket M0 Board

ADD TO CART

1 x Breadboard trim potentiometer - 10K

Small potentiometer to fit in the housing

ADD TO CART

1 x PLA Filament for 3D Printers - 1.75mm Natural Translucent - 1KG

Clear PLA Filament for light diffusion

ADD TO CART

1 x Silicone Cover Stranded-Core Wire - 30AWG in Various Colors

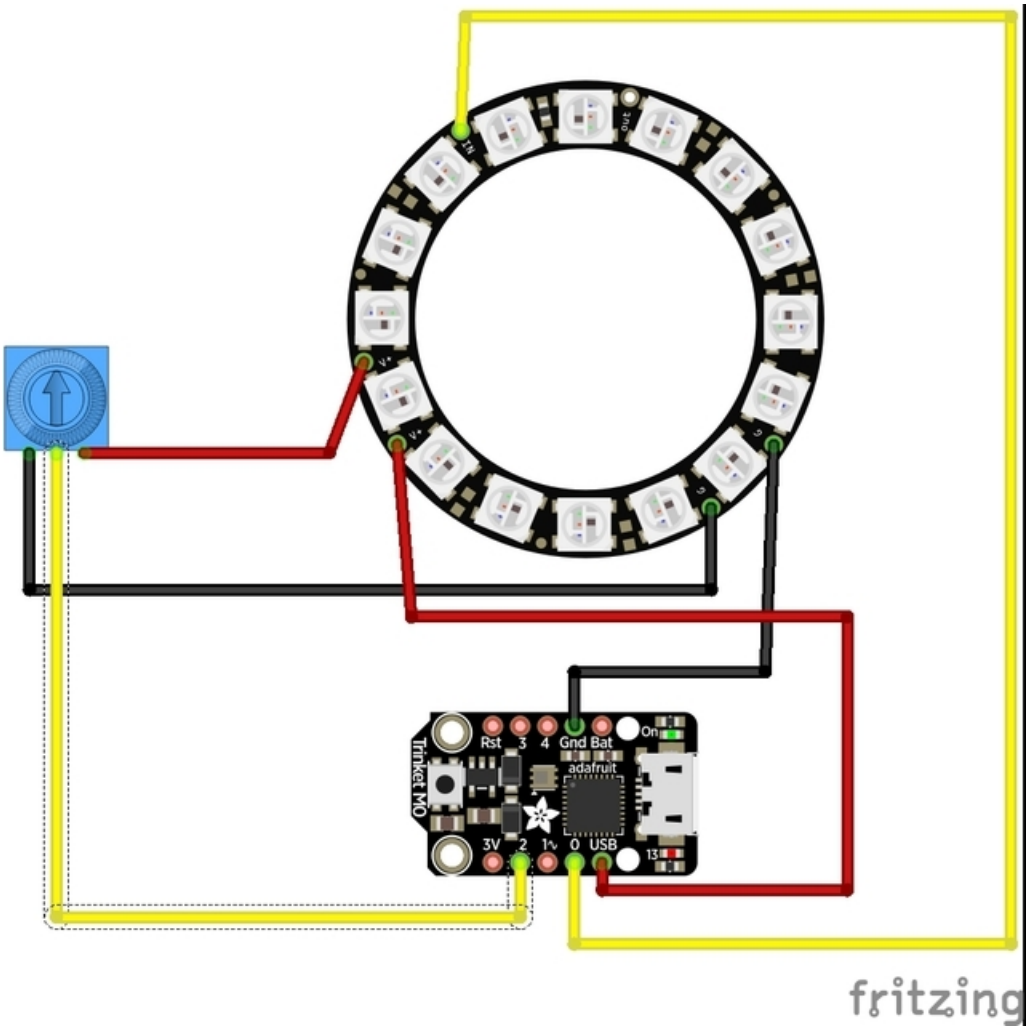
Thin wire for fitting comfortably into the 3D printed housing

OUT OF STOCK

Electronics

The Trinket M0 is the brain of this operation. The NeoPixel ring can be attached to any of the pins, but since the potentiometer is an analog input, it needs to be attached to pin 2, which is Analog 1 on the Trinket.

To reduce wire bulk, 5V and ground will be supplied to the potentiometer from the NeoPixel ring since the ring has two 5V and ground pins for daisy chaining.



Arduino IDE Code

The code is written in the Arduino IDE and uses the NeoPixel library. After the pins are defined, the analog values produced by the potentiometer are mapped to digital values. This is then plugged in as the value for the amount of white light being produced by the RGBW NeoPixel ring. The RGB values are left at zero to avoid any blue tinted light. This allows you to control the “brightness” of the NeoPixels without messing with the brightness parameter in the NeoPixel library which is not meant to be changed after the initial definition in the code setup.

Be sure to update your Arduino libraries!

```
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
  #include <avr/power.h>
#endif

#define PIN 0 //neopixel pin

int POT = A1; //analog pin 1 = Trinket m0 Pin 2
int val = 0;

// Parameter 1 = number of pixels in strip
// Parameter 2 = Arduino pin number (most are valid)
// Parameter 3 = pixel type flags, add together as needed:
//   NEO_KHZ800  800 KHz bitstream (most NeoPixel products w/WS2812 LEDs)
//   NEO_KHZ400  400 KHz (classic 'v1' (not v2) FLORA pixels, WS2811 drivers)
//   NEO_GRB     Pixels are wired for GRB bitstream (most NeoPixel products)
//   NEO_RGB     Pixels are wired for RGB bitstream (v1 FLORA pixels, not v2)
//   NEO_RGBW    Pixels are wired for RGBW bitstream (NeoPixel RGBW products)
Adafruit_NeoPixel ring = Adafruit_NeoPixel(16, PIN, NEO_RGBW + NEO_KHZ800);

void setup() {
  ring.begin();
  ring.setBrightness(255); //max brightness available for the pot
  ring.show(); // Initialize all pixels to 'off'
}

void loop() {
  val = analogRead(POT); //will hold analog value sent by pot
  val = map(val, 0, 1023, 0, 255); //maps analog values to digital so that it can be used by the neopixel

  uint16_t i;

  //sets all 16 pixels to on and only show level of white light defined by analog value of the pot
  for (i = 0; i < ring.numPixels(); i++) {
    ring.setPixelColor(i, 0, 0, 0, val);
  }
  ring.show();
}
```

CircuitPython Code

Since we're using a Trinket M0 board, we can also write the code with CircuitPython!

https://adafru.it/Bid

https://adafru.it/Bid

```
import time

import board
import neopixel
from analogio import AnalogIn

pot = AnalogIn(board.A1)  # what pin the pot is on
pixpin = board.D0  # what pin the LEDs are on
numpix = 16  # number of LEDs in ring!
BPP = 4  # required for RGBW ring

ring = neopixel.NeoPixel(pixpin, numpix, bpp=BPP, brightness=0.9)


def val(pin):
    # divides voltage (65535) to get a value between 0 and 255
    return pin.value / 257


while True:
    # Two lines for troubleshooting to see analog value in REPL
    # print("A0: %f" % (pot.value / 65535 * 3.3))
    # time.sleep(1)

    # change floating point value to an int
    ring.fill((0, 0, 0, int(val(pot))))
    time.sleep(0.01)
```

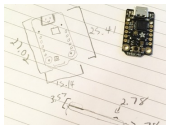
As you can see, the code is very similar to the Arduino IDE code. We're still reading the analog value of **Pin 2** (**A1**), converting it to a value between 0 and 255 and then having that be the value of the white portion of the NeoPixel.

Since we're using an RGBW ring, we need to declare this in the NeoPixel object. This is done using `bpp` (bytes per pixel). RGBW NeoPixels have 4 BPP, where as RGB NeoPixels have 3 BPP.

The other important point is that analog values in CircuitPython range from 0 to 65535. If you divide 65535 by 257, then it converts the range to 0 to 255 so that it can be read as a digital value for the NeoPixels.

3D Printing

The housing was designed in Fusion360. First, the parts for all the hardware were designed and printed individually to ensure a good fit. After that, it was back to Fusion360 to join the parts together with channels for the wiring, which is the .STL file that you have here for download.



Prototyping

Print Settings

Be sure to print with the lens part on the print bed. Surface supports will be needed for the Trinket portion of the housing. Printing at .2 resolution and 15% infill should yield good results.

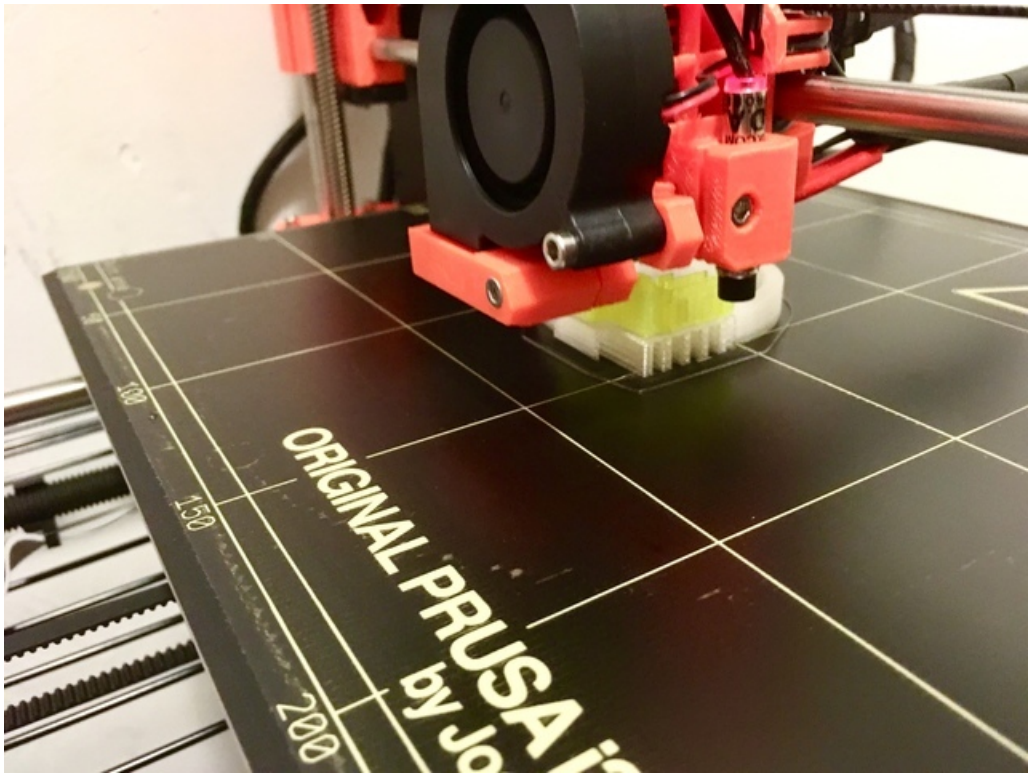
https://adafru.it/Cjq

https://adafru.it/Cjq

Color Change

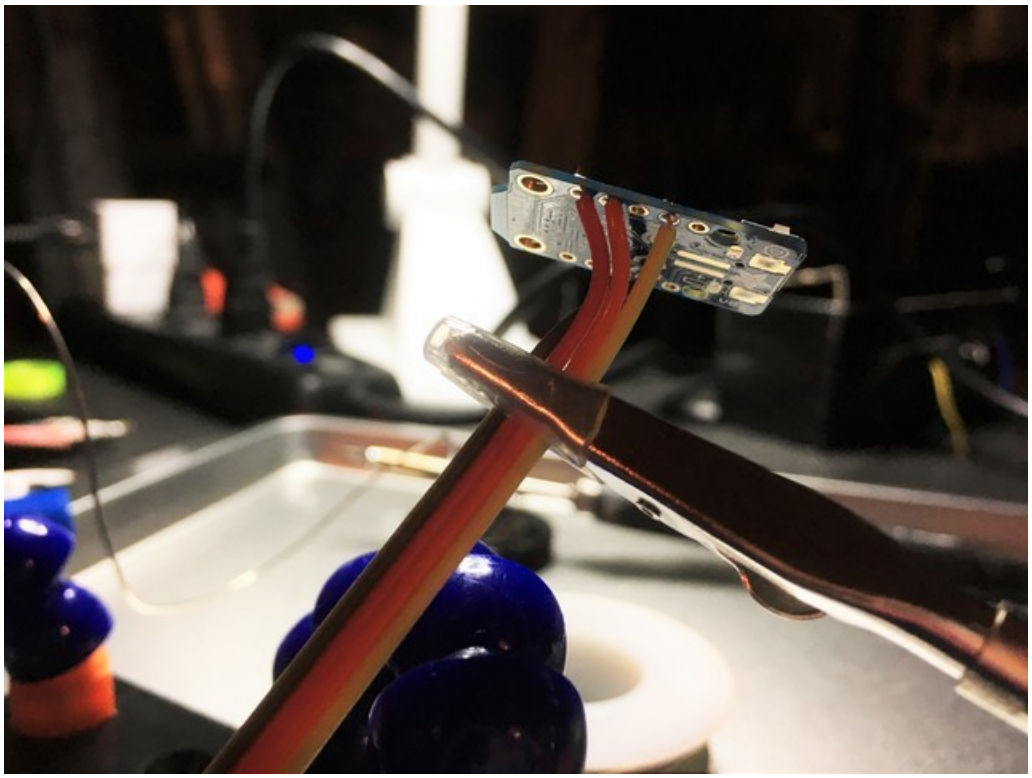
Clear PLA should be used for the lens portion of the housing for proper light diffusion. If you want the rest of the housing to print in a different color though, you can edit your gcode so that the print pauses at a specific layer height, which will then allow you to swap filaments. As long as your printer is running Marlin firmware (aka accepts gcode) then

you can use the Prusa ColorPrint utility (https://www.prusaprinters.org/color-print/ (https://adafru.it/Cjr)). To change colors at the Trinket portion, enter in a color change at layer height 7.80.



Assembly

Now it's time to put everything together. First up is soldering the wires to the Trinket. It's important to solder so that you're soldering on the top of the board allowing for the wires to hang from the bottom. Solder wires to 5V, Pin 0, Pin 2 and Ground. Also be sure to use a thin gauge wire to keep everything neat.

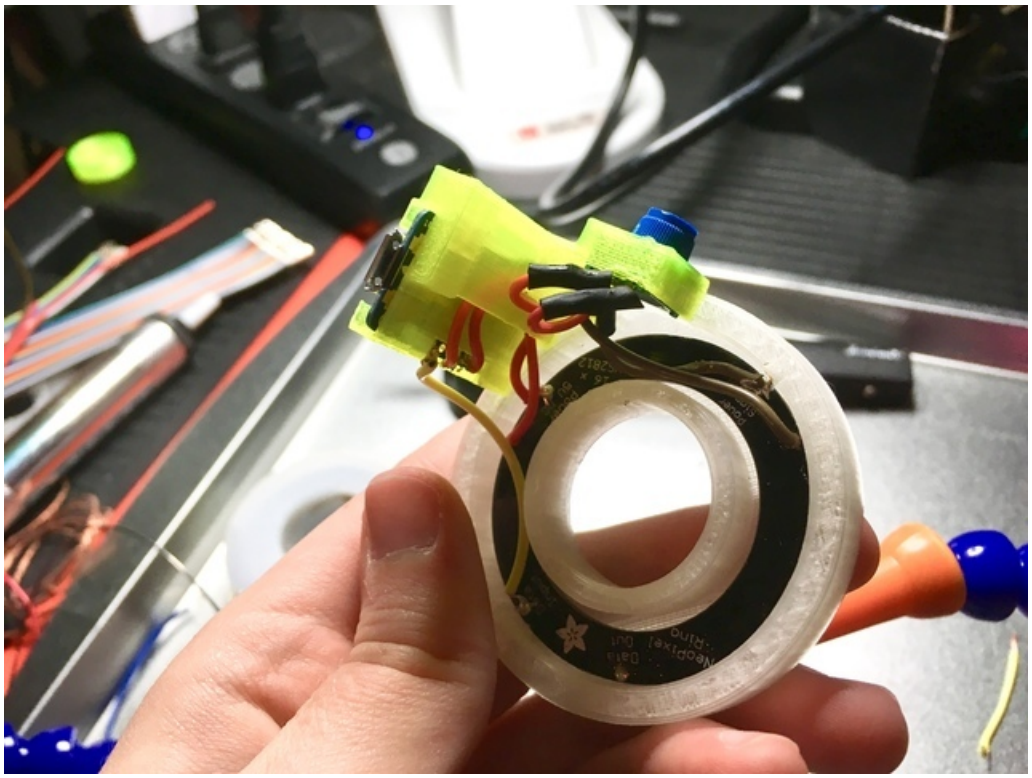


Be sure to tin any stranded wire leads to make soldering easier

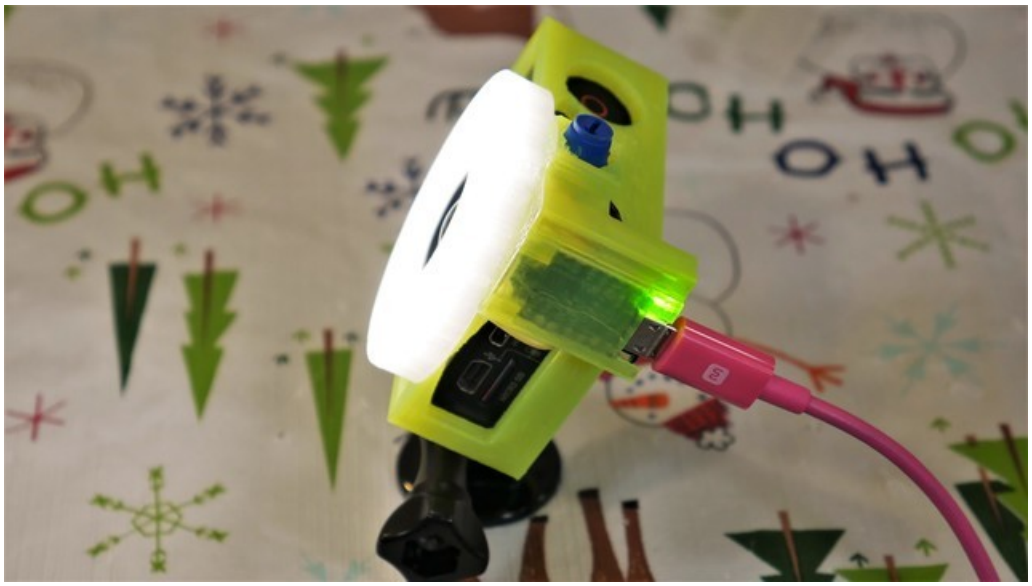
Next you're going to slide the Trinket into the housing, first by gently pulling the wires thru their channels. Ground is going to be led thru the back channel towards the pot section and the data and 5V wires are going to go thru the opening on the Trinket portion of the housing.

After the Trinket is snuggly in the housing with the wires pulled thru, you'll solder the wire from the Trinket's Pin 0 to Data In on the NeoPixel ring and 5V and Ground to the corresponding pads on the ring as well. Similar to the Trinket, be sure to solder the wires to the NeoPixel ring so that they hang down from the back. This eliminates any wire bulk on the sides.

Next up for wiring is the potentiometer. Be sure to use some small heat shrink for the potentiometer's three leads to avoid any shorting. Solder wires to the NeoPixel ring's secondary 5V and Ground pads to connect to the potentiometer. Solder the wire from the Trinket's Pin 2 to the potentiometer's middle pin. Try to gently pull the potentiometer's data and 5V wires thru the cable channels for neat cable management.



Test the circuit by plugging in the Trinket. If all has gone well then you should have a nice bright adjustable light that sits snugly on your camera's lens!




GoPro
ORIGINAL PRUSA i3 MK2