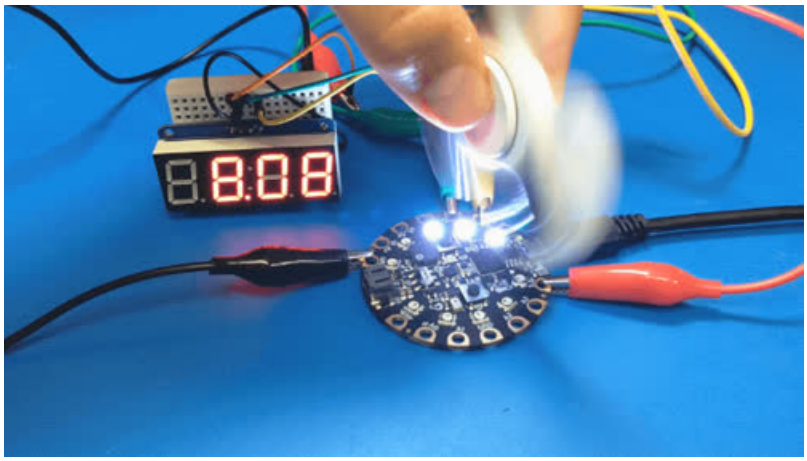
adafruit learning system

Fidget Spinner Tachometer

Created by Tony DiCola

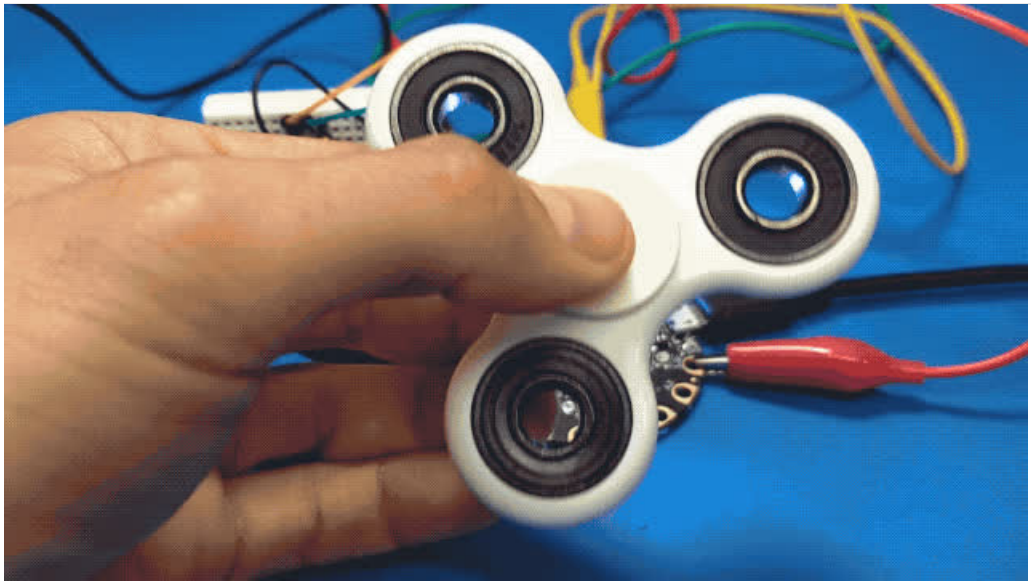


Last updated on 2018-08-22 04:01:15 PM UTC

Guide Contents

Guide Contents 2
Overview 3
Hardware 4
Optional LED Backpack 4
Arduino 6
CircuitPython 9

Overview



Fidget spinners are fun little toys that spin around on a bearing and make surprisingly addictive stress relievers. With a flick of your finger you can set a spinner into a rapid spin and watch as it slowly comes to a stop from the forces of friction. You can even make spinning a competition with friends--see who can flick their spinner the hardest to get it spinning the fastest! But wouldn't it be cool to measure how fast a spinner is spinning? It turns out with an Adafruit Circuit Playground board you can build a simple tachometer that measures the speed of a spinning fidget spinner!

This project will show you how to use a Circuit Playground Classic (https://adafru.it/ncE) or Circuit Playground Express (https://adafru.it/wpF) board to build a fidget spinner tachometer that measures the speed of a spinner. Circuit Playground is Adafruit's all-in-one electronics learning board that has all the necessary components for this project, specifically a light sensor, NeoPixel LEDs, and the brains of an Arduino, built in to it so you can get started immediately with no soldering or other hardware setup. Even better you can build this project using either Arduino or CircuitPython code. If you're new to Python this is a good example of how a project can be converted from Arduino to Python code.

Before you get started you'll want to read the Adafruit Circuit Playground guide (https://adafru.it/pAP) to learn more about the board.

Also you might find the Circuit Playground digital fidget spinner (https://adafru.it/xAS) another fun spinner project to build with Circuit Playground. The digital fidget spinner uses the accelerometer built in to Circuit Playground to make a virtual fidget spinner that animates LEDs from the flick of your finger.

Hardware

To build this project you only need a Circuit Playground Classic (https://adafru.it/ncE) or Circuit Playground Express (https://adafru.it/wpF) board. Everything for the project is built into the Circuit Playground board--no assembly necessary!

The Arduino version of this project can work with either the original Circuit Playground Classic (https://adafru.it/ncE) board and its ATMega32u4 processor, or the newer Circuit Playground Express (https://adafru.it/wpF) board and its fancier SAMD21 processor. However the CircuitPython version of this project **only** works with the Circuit Playground Express (https://adafru.it/wpF) board. If you don't have a Circuit Playground board yet grab the Circuit Playground Express (https://adafru.it/wpF) board as it includes the same sensors and outputs but with more memory and processor speed compared to the classic board.

In addition to the Circuit Playground board you'll need a fidget spinner of some sort. The fidget spinner craze is in full force so check local convenience stores, department stores (Walmart, etc.) and you should be able to find one locally. Or check Amazon (https://adafru.it/xAT) for many, many types of fidget spinners.

Optional LED Backpack

For the Arduino version of this project you can optionally connect a little LED backpack to display the speed of a spinner instead of using Arduino's built in serial monitor. To add the LED backpack display you'll need these parts:

- One 7 segment LED backpack (https://adafru.it/xAU) (pick any color, but be sure to use the 0.56" 4-digit 7-segment display like featured on the right column).
- Small breadboard (https://adafru.it/kft) and hookup wires (https://adafru.it/fE2) to connect to the LED backpack.
- Alligator clip wires (https://adafru.it/dWJ) to connect the backpack to the Circuit Playground board.
- Alternatively, our nice Alligator -> Jumper wires (https://adafru.it/xAV)combine both of the above for nice and tidy wiring.
- Soldering tools (https://adafru.it/fE3) to solder a header and the LED display to the LED backpack board. If you're new to soldering be sure to check out the guide to excellent soldering (https://adafru.it/dxy) too!

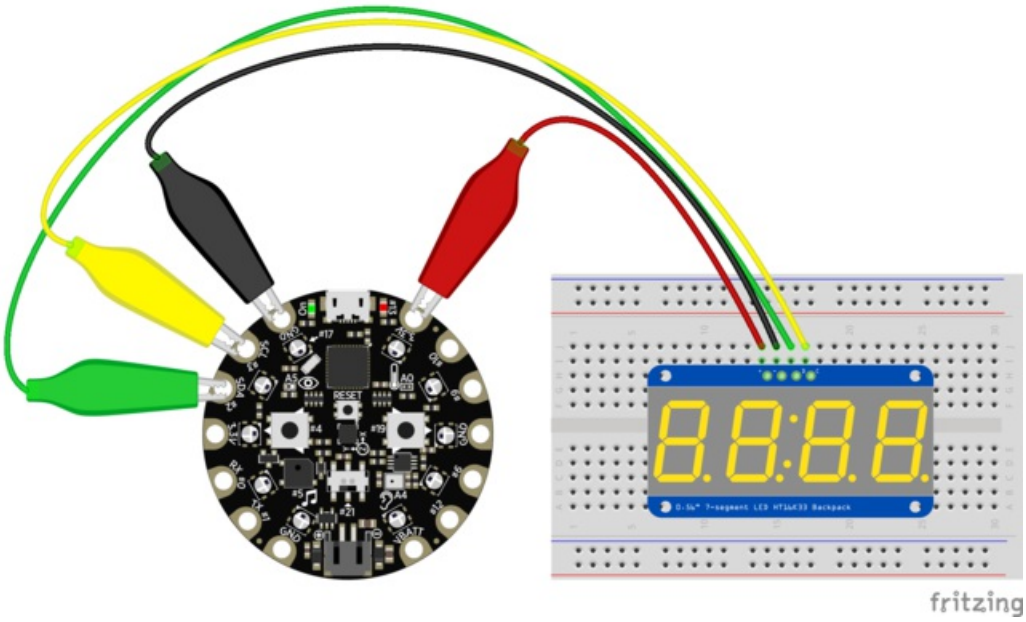
Once you have your Circuit Playground board be sure to follow its guide to setup the Arduino IDE to program the board:

- Adafruit Circuit Playground Classic guide (https://adafru.it/pAP)
- Adafruit Circuit Playground Express guide (https://adafru.it/xAW) (note there isn't a dedicated Circuit Playground Express with Arduino guide yet, but follow the same steps as the Metro M0 guide linked above and select the 'Adafruit Circuit Playground Express' board from the Adafruit SAMD Boards option in the board list. In addition install the Adafruit Circuit Playground library (https://adafru.it/pAQ) to gain access to the board's features in Arduino sketches).

If you're connecting a display to your Circuit Playground be sure to follow its guide too to learn how to solder the display to its backpack and configure Arduino to use its software library:

- Adafruit 7-segment Backpack guide (https://adafru.it/xvB)

Then connect the display to your Circuit Playground as follows:



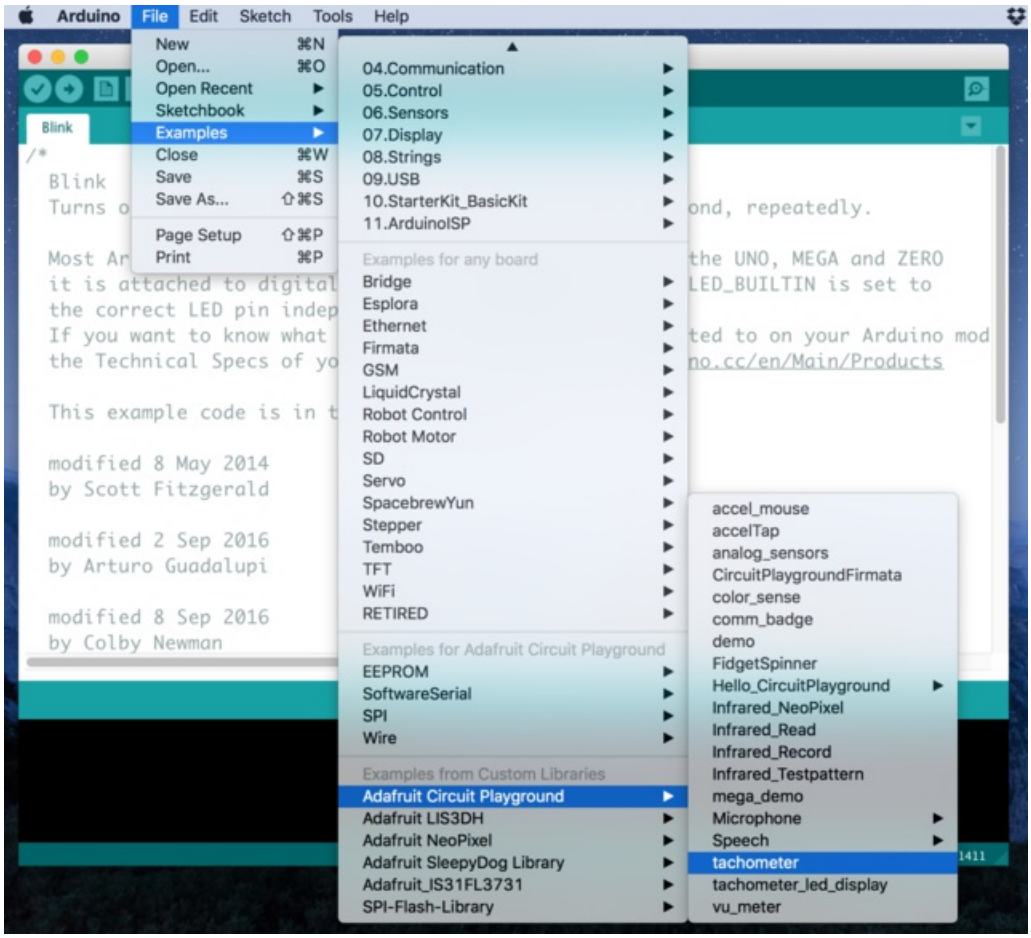
- **Circuit Playground GND / ground** to **LED backpack - / ground** (black wire).
- **Circuit Playground 3.3V** to **LED backpack + / voltage input** (red wire).
- **Circuit Playground SCL** to **LED backpack C / clock** (yellow wire).
- **Circuit Playground SDA** to **LED backpack D / data** (green wire).

Continue on to learn about installing and running the software sketch for this project.

Arduino

The software for the fidget spinner tachometer is an Arduino sketch that's included in the Adafruit Circuit Playground library. Make sure you've installed the Adafruit Circuit Playground library (https://adafru.it/pAQ), either using the Arduino library manager or with a manual installation from its source on Github (https://adafru.it/naF). If you already have the library installed be sure to use Arduino's library manager (https://adafru.it/fCN) to update the library to the latest version so you get the most recent example sketch code.

Next look for the **tachometer** or **tachometer_led_display** examples in the Adafruit Circuit Playground library in Arduino:



The **tachometer** example will detect the speed of a fidget spinner and just print it out on the serial monitor, and the **tachometer_led_display** example will do the same but also print the speed on a connected 7-segment LED backpack display. Open the appropriate example for your hardware setup.

If you don't see the tachometer examples be sure to update your Circuit Playground library to the latest version!

At the top of the tachometer sketches are configuration defines which you can optionally change to adjust the behavior of the tachometer:

```
// Configuration values.  You don't have to change these but they're
// interesting to adjust the logic.  If your spinner has more or less
// than three arms be sure to change the SPINNER_ARMS value below:
#define SPINNER_ARMS       3      // Number of arms on the fidget spinner.
                                  // This is used to calculate the true
                                  // revolutions per second of the spinner
                                  // as one full revolution of the spinner
                                  // will actually see this number of cycles
                                  // pass by the light sensor.  Set this to
                                  // the value 1 to ignore this calculation
                                  // and just see the raw cycles / second.

#define SAMPLE_DEPTH       512    // How many samples to take when measuring
                                  // the spinner speed.  The larger this value
                                  // the more memory that will be consumed but
                                  // the slower a spinner speed that can be
                                  // detected (larger sample depths mean longer
                                  // period waves can be detected).  You're
                                  // limited by the amount of memory on the
                                  // board for this value.  On a classic
                                  // Circuit Playground with 2kb of memory
                                  // you can only go up to about 512 or so (each
                                  // sample is 2 bytes, so 1kb of space total).
                                  // On an express board you can go much higher,
                                  // like up to 10240 for 20kb of sample data.

#define SAMPLE_PERIOD_US   1500   // Amount of time in microseconds to delay
                                  // between each light sensor sample.  This is
                                  // a balance between how fast and slow of
                                  // a spinner reading that can be measured.
                                  // The higher this value the slower a spinner
                                  // you can measure, but at the tradeoff of
                                  // not accurately measuring fast spinners.
                                  // Low values (even 0) mean very fast speeds
                                  // can be detected but slow speeds (below 10hz)
                                  // are harder to detect.  You can increase the
                                  // sample depth to help improve the range
                                  // of detection speeds, but there's a limit
                                  // based on the memory available.

#define THRESHOLD          400    // How big the amplitude of a cyclic
                                  // signal has to be before the measurement
                                  // logic kicks in.  This is a value from
                                  // 0 to 1023 and might need to be adjusted
                                  // up or down if the detection is too
                                  // sensitive or not sensitive enough.
                                  // Raising this value will make the detection
                                  // less sensitive and require a very large
                                  // difference in amplitude (i.e. a very close
                                  // or highly reflective spinner), and lowering
                                  // the value will make the detection more
                                  // sensitive and potentially pick up random
                                  // noise from light in the room.

#define MEASURE_PERIOD_MS  1000   // Number of milliseconds to wait
                                  // between measurements.  Default is
                                  // one second (1000 milliseconds).
```

In particular the **SPINNER_ARMS** configuration define might need to be adjusted based on the number of arms on the fidget spinner you're using with the sketch. Set this value to the number of arms or protrusions from the spinner. The sketch uses this value to correctly determine the speed of an entire revolution of the spinner. If you just want to see the speed of a single arm moving by the sensor (or are unsure how many arms are on your spinner) try setting this to a value of 1 to get the raw spinning speed.

The other configuration defines are optional and adjust the sensitivity and range of speeds the tachometer can detect. Read the comments next to the defines, and watch the video about the tachometer code to see how these values adjust the behavior of the tachometer.

When you're ready to use the tachometer upload it to your Circuit Playground board. Once uploaded you should see the first three NeoPixels turn on to bright white color. Then open the serial monitor at 115200 baud and you're ready to detect the speed of a spinner.

Flick a fidget spinner to get it moving very quickly and then hold it perpendicular (i.e. at a 90 degree angle) to the Circuit Playground board right above the light sensor (the light sensor is right below the 3 lit up NeoPixels and has a little eye symbol printed on the board). Hold the spinner as close to the sensor as you can without it hitting the board. Now look at the serial monitor output and you should see the detect speed printed out every second!

Notice as the spinner slows down the frequency, or number of times the spinner rotates in a second, slows down too. In addition the RPM (revolutions per minute) of the spinner slows down, and the period, or amount of time between a full spin of the spinner, increases.

If you're holding your spinner in front of the sensor and not seeing any output from the serial monitor you might need to adjust the **THRESHOLD** define at the top of the sketch and try again. Try lowering the threshold to smaller value like 300 or 200. The lower you make the threshold the more sensitive the tachometer, however one the threshold is too low it might be too sensitive and detect random noise as speed.

If you're using the **tachometer_led_display** example you should see it print the frequency, or speed of the spinner in revolutions per second, on the display in addition to the serial monitor. If you don't see the display printing the speed but do see it output in the serial monitor double check your LED backpack wiring.

That's all there is to using the fidget spinner tachometer example! Check out the video about this project (https://adafru.it/xAS) to learn how the tachometer works, specifically how it uses reflections of light hitting the light sensor to determine the speed of the spinner. The sketch performs a simple frequency detection by counting how often the light sensor signal changes during a short period of time.

CircuitPython

In addition to Arduino you can also build a fidget spinner tachometer with Circuit Playground Express and CircuitPython. This Python-powered tachometer works just like the Arduino version and can detect the speed of a spinner held in front of the board's light sensor. However instead of using low level Arduino programming code this tachometer is programmed with simpler Python code!

To build the CircuitPython version of the tachometer you **must** use the Circuit Playground Express board. The older Circuit Playground classic board unfortunately can't run CircuitPython so you'll need the latest express board.

Once you have the board you'll need to load it with the latest version of CircuitPython firmware. Follow the steps in this Metro M0 Express guide (https://adafru.it/xAX) to see how to load CircuitPython onto a board--the steps are exactly the same for Circuit Playground Express except you'll download the **Circuit Playground Express** .uf2 firmware from the latest CircuitPython release (https://adafru.it/tBa).

After you've loaded CircuitPython onto the board you should see it appear as a USB drive named **CIRCUITPY** when connected to your computer. This is where you can copy Python code and other files for the board to run.

First you'll need to copy a CircuitPython NeoPixel module that allows code to control the NeoPixels on the board. Go to the releases tab of the CircuitPython NeoPixel module (https://adafru.it/wby) and download the **neopixel.mpy** file from the latest release. Then drag this **neopixel.mpy** file onto the board's **CIRCUITPY** drive.

Next download the code for this project below and save it as a **main.py** file on the board's **CIRCUITPY** drive:

```
# Adafruit Circuit Playground Express Fidget Spinner Tachometer
#
# This code uses the light sensor built in to Circuit Playground Express
# to detect the speed (in revolutions per second) of a fidget spinner.
# Save this code as main.py on a Circuit Playground Express board running
# CircuitPython (see https://github.com/adafruit/circuitpython).  You will
# also need to load neopixel.mpy onto the board's filesystem (from
# https://github.com/adafruit/Adafruit_CircuitPython_NeoPixel/releases).
#
# When the first three NeoPixels light up white you're ready to read the speed # of a spinner. Hold a spi
# the light sensor (look for the eye graphic on the board, it's right
# below the three lit NeoPixels) and look at the serial terminal from the board
# at 115200 baud to see the speed of the spinner printed.  This works best
# holding the spinner perpendicular to the sensor, like:
#       ||
#       || <- Spinner
#       ||
#    ________  <- Circuit Playground
#
# Author: Tony DiCola
# License: MIT License (https://en.wikipedia.org/wiki/MIT_License)
import array

import board
import analogio
import time

import neopixel


# Configuration:
SPINNER_ARMS          = 3       # Number of arms on the fidget spinner.
```

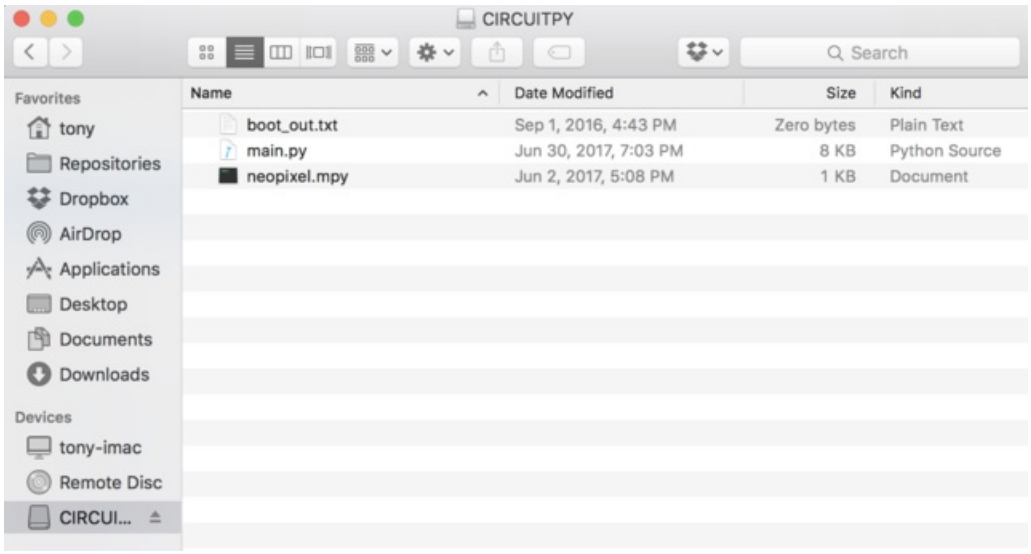
```
                              # This is used to calculate the true
                              # revolutions per second of the spinner
                              # as one full revolution of the spinner
                              # will actually see this number of cycles
                              # pass by the light sensor.  Set this to
                              # the value 1 to ignore this calculation
                              # and just see the raw cycles / second.

SAMPLE_DEPTH           = 256  # How many samples to take when measuring
                              # the spinner speed.  The larger this value
                              # the more memory that will be consumed but
                              # the slower a spinner speed that can be
                              # detected (larger sample depths mean longer
                              # period waves can be detected).  You're
                              # limited by the amount of memory on the
                              # board for this value.

TARGET_SAMPLE_RATE_HZ  = 150  # Target sample rate for sampling the light
                              # sensor.  This in combination with the sample
                              # depth above controls how slow and fast of
                              # a signal you can detect.  Note that the
                              # sample rate can only go so high before it's
                              # too fast for the Python interpreted code
                              # to run.  If that happens the board will
                              # light up LEDs red to indicate the 'underflow'
                              # condition (drop the sample rate down to
                              # a lower value and try again).
                              # A value of 150 times a second means you can
                              # measure a spinner going up to 25 revolutions
                              # per second.

THRESHOLD              = 40000 # How big the magnitude of a cyclic
                              # signal has to be before the measurement
                              # logic kicks in.  This is a value from
                              # 0 to 65535 and might need to be adjusted
                              # up or down if the detection is too
                              # sensitive or not sensitive enough.
                              # Raising this value will make the detection
                              # less sensitive and require a very large
                              # difference in amplitude (i.e. a very close
                              # or highly reflective spinner), and lowering
                              # the value will make the detection more
                              # sensitive and potentially pick up random
                              # noise from light in the room.


# Configure NeoPixels and turn them all off at the start.
pixels = neopixel.NeoPixel(board.NEOPIXEL, 10)
pixels.fill((0,0,0))
pixels.write()

# Configure analog input for light sensor.
light = analogio.AnalogIn(board.LIGHT)

# Take an initial set of readings and measure how long it takes.
# This is used to calculate a delay between readings to hit the desired
# target sample rate.  Use the array module to preallocate an array of 16-bit
# unsigned samples with lower memory overhead vs. a simple python list.
readings = array.array('H', [0]*SAMPLE_DEPTH)
start = time.monotonic()
```

```
start = time.monotonic()
for i in range(SAMPLE_DEPTH):
    readings[i] = light.value
stop = time.monotonic()
print((stop-start)*1000.0)

# Calculate how long it took to take all the readings above, then figure out
# the difference from the target period to actual period.  This difference is
# the amount of time to delay between sample readings to hit the desired target
# sample rate.
target_period = 1.0/TARGET_SAMPLE_RATE_HZ
actual_period = (stop-start)/SAMPLE_DEPTH
delay = 0
# Check that we can sample fast enough to hit the target rate.
if actual_period > target_period:
    # Uh oh can't sample fast enough--print a warning and light up pixels red.
    print('Could not hit desired target sample rate!')
    pixels.fill((255,0,0))
    pixels.write()
else:
    # No problem hitting target sample rate so calculate the delay between
    # samples to hit that desired rate.  Then turn on the first three pixels
    # to white full brightness.
    delay = target_period - actual_period
    pixels[0] = (255, 255, 255)
    pixels[1] = (255, 255, 255)
    pixels[2] = (255, 255, 255)
    pixels.write()

# Main loop:
while True:
    # Pause for a second between tachometer readings.
    time.sleep(1.0)
    # Grab a set of samples and measure the time it took to do so.
    start = time.monotonic()
    for i in range(SAMPLE_DEPTH):
        readings[i] = light.value
        time.sleep(delay)  # Sleep for the delay calculated to hit target rate.
    stop = time.monotonic()
    elapsed = stop - start
    # Find the min and max readings from the samples.
    minval = readings[0]
    maxval = readings[0]
    for r in readings:
        minval = min(minval, r)
        maxval = max(maxval, r)
    # Calculate magnitude or size of the signal.  If the magnitude doesn't
    # pass the threshold then start over with a new sample (run the loop again).
    magnitude = maxval - minval
    if magnitude < THRESHOLD:
        continue
    # Calculate the midpoint of the signal, then count how many times the
    # signal crosses the midpoint.
    midpoint = minval + magnitude/2.0
    crossings = 0
    for i in range(1, SAMPLE_DEPTH):
        p0 = readings[i-1]
        p1 = readings[i]
        # Check if a pair of points crossed the midpoint either by hitting it
        # exactly or hitting it going up or down.
```

```
        if p1 == midpoint or p0 < midpoint < p1 or p0 > midpoint > p1:
            crossings += 1
    # Finally use the number of crosssings and the amount of time in the sample
    # window to calculate how many times the spinner arms crossed the light
    # sensor.  Use that period to calculate frequency (rotations per second)
    # and RPM (rotations per minute).
    period = elapsed / (crossings / 2.0 / SPINNER_ARMS)
    frequency = 1.0/period
    rpm = frequency * 60.0
    print('Frequency: {0:0.5} (hz)\t\tRPM: {1:0.5}\t\tPeriod: {2:0.5} (seconds)'.format(frequency, rpm, p
```

You can also grab this code from a gist on GitHub (https://adafru.it/xAY).

Make sure both **neopixel.mpy** and the above code saved as **main.py** are on the board's filesystem, it should look something like this:



Once the main.py file is copied to the board it should restart and run the tachometer code. You'll see the first three NeoPixels light up bright white to indicate the board is ready to detect the speed of a fidget spinner. If you don't see the lights turn on then eject the **CIRCUITPY** drive, disconnect the board, and reconnect it to your computer to reset it.

Once the code is running and the lights are lit up you can open the board's serial terminal at 115200 baud. Hold a spinning fidget spinner in front of the light sensor and you should see the speed printed out every second. You need to hold the spinner very close to the light sensor to make sure it can get a good reading. Watch the video at the top of this page to see an example of using the tachometer.

That's all there is to the CircuitPython version of the fidget spinner tachometer!