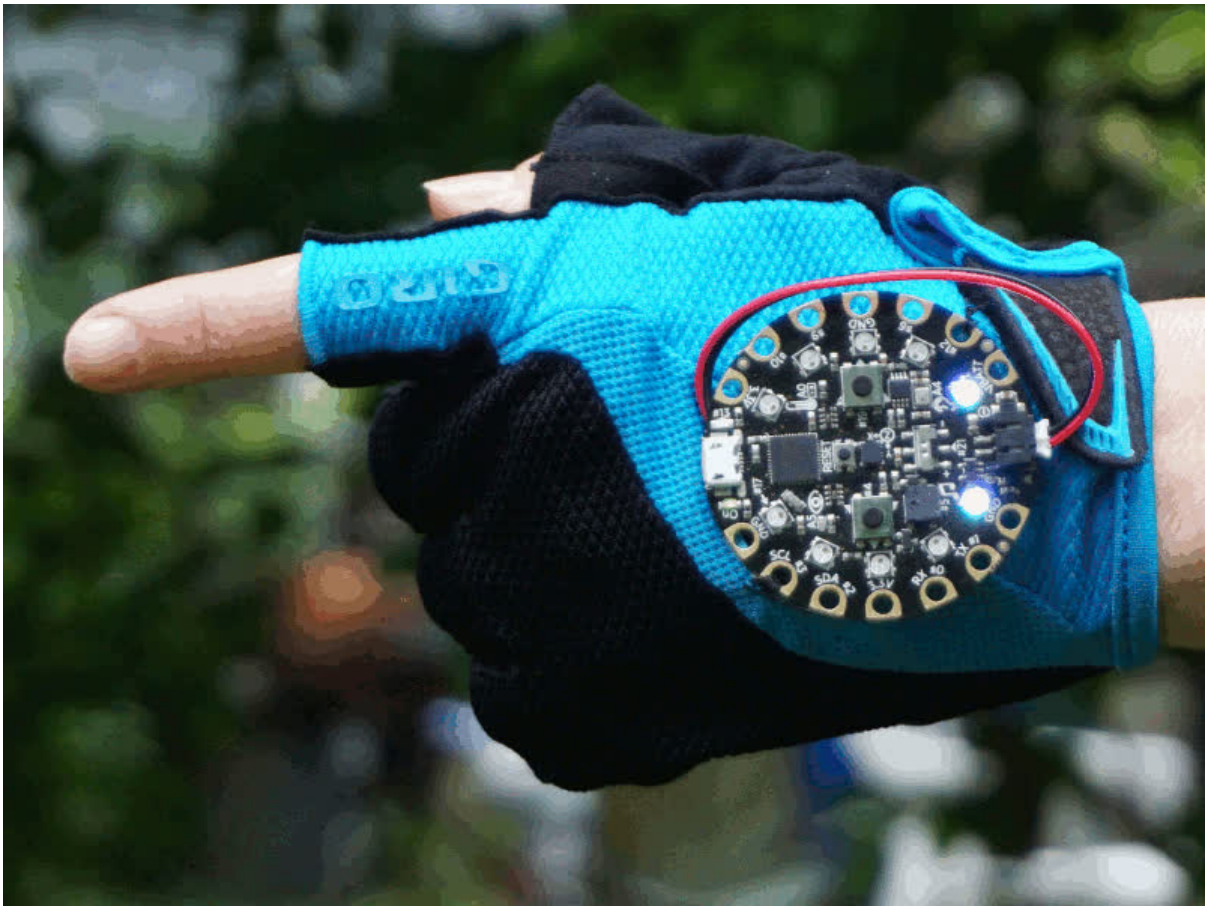
Circuit Playground Bike Glove

Created by Carter Nelson



https://learn.adafruit.com/circuit-playground-bike-glove

Last updated on 2024-03-08 02:36:39 PM EST

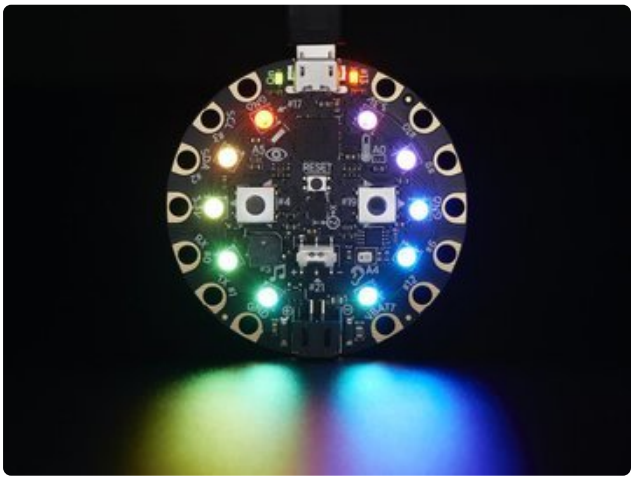
Table of Contents

Overview

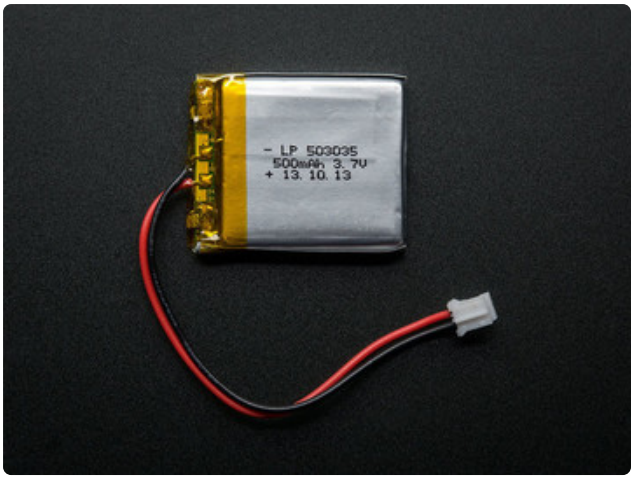
In this guide we will use the Circuit Playground to create a bike turn signal indicator. By sewing the Circuit Playground to a bike glove and using the accelerometer to detect hand position the standard hand turn signals are enhanced with some nice NeoPixel animations.

Required Parts

In addition to a Circuit Playground, you will need some form of battery power. The 500mAh battery is a nice low profile option and is recommended for this build. However, you may be able to also use the AAA battery pack if you can incorporate it into your bike gloves. Oh yeah, you'll also need bike gloves and some needle and thread to sew everything together.



Circuit Playground
Classic (http://adafru.it/3000)
Express (http://adafru.it/3333)



Lithium Ion Polymer Battery - 3.7v 500mAh (http://adafru.it/1578)

Bike Gloves

Also check out these options (https://adafru.it/vof) for charging the LiPo battery.

Before Starting

If you are new to the Circuit Playground, you may want to first read these overview guides.

Circuit Playground Classic

- Overview (https://adafru.it/ncG)
- Lesson #0 (https://adafru.it/rb4)

Circuit Playground Express

- Overview (https://adafru.it/AgP)

Bike Hand Signals

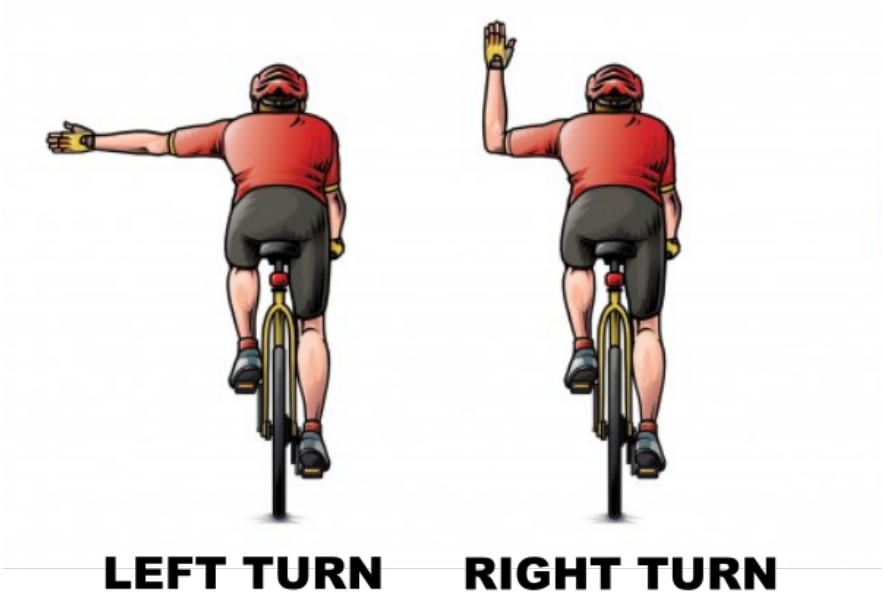
When riding a bicycle various hand signals are used to communicate your intentions to others sharing the road with you. This can be other bike riders, but also, and more importantly, to cars and trucks.

There are all kinds of bicycle hand signals, and they may vary from location to location. For a good overview, read the discussion on this web page:

Rules of the Road: When and How to Use Hand Signals When Riding

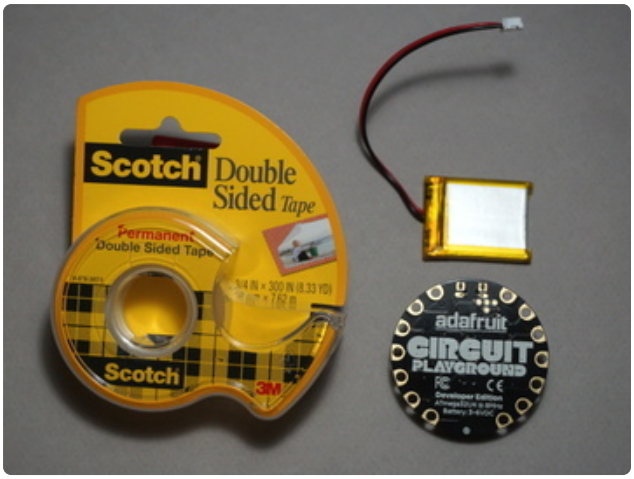
https://adafru.it/wJd

For this project, we will focus on only two hand signals - the ones used to indicate turning direction. The ones we will use are shown below.

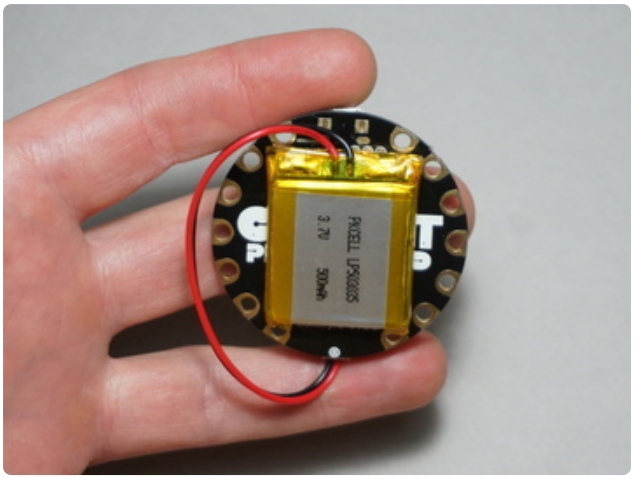


Bike Glove Assembly

Let's start by actually making the bike glove. This way you can upload programs to it as you work through the rest of the guide.

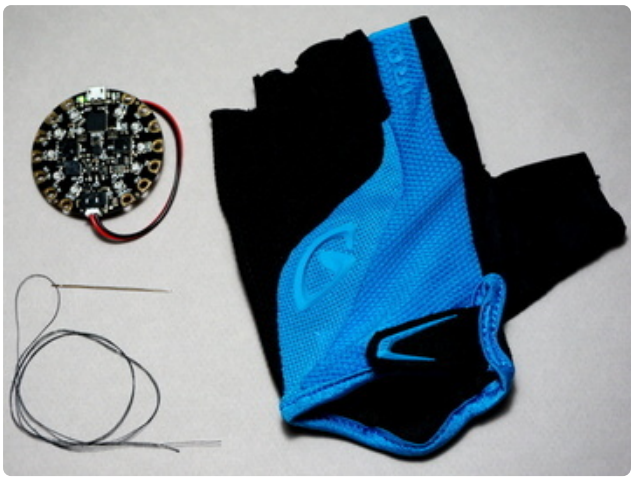


Use some double sided tape to attach the battery to the back of the Circuit Playground.

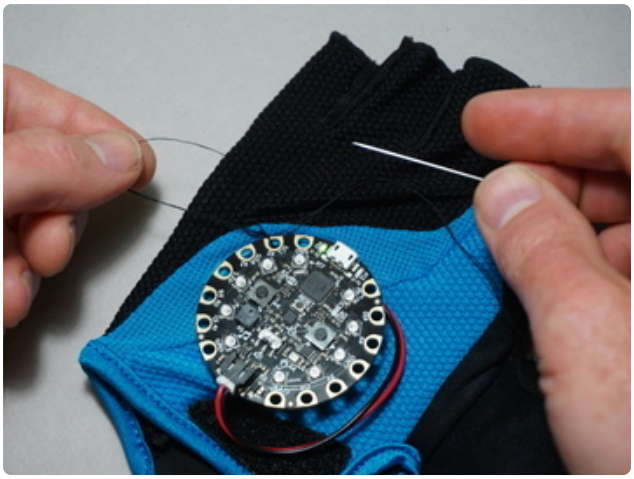


Attach the battery to the back of the Circuit Playground as shown.

(battery chargers (https://adafru.it/wJe))



Now get some needle and thread ready for sewing the Circuit Playground to the glove.

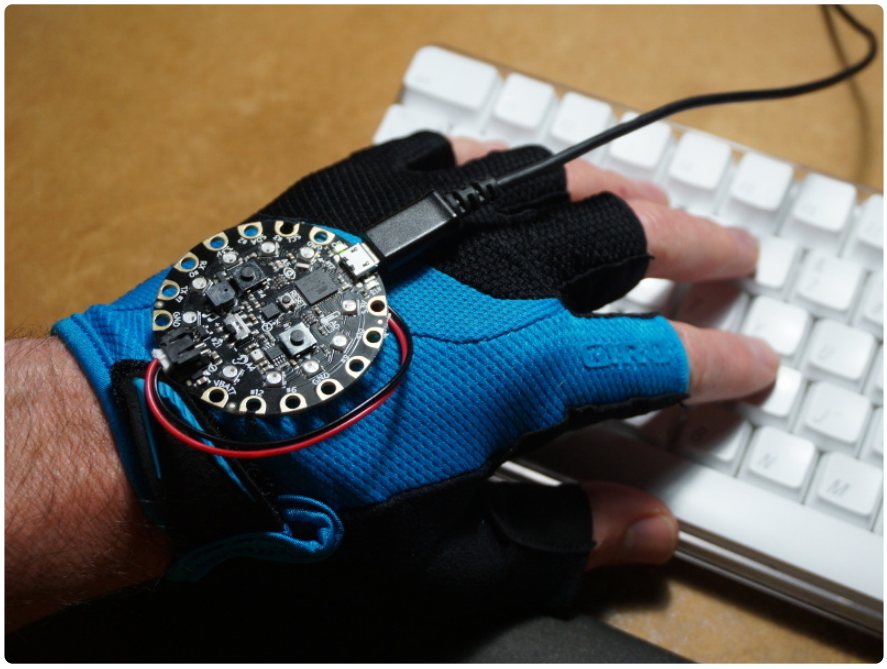


Sew the thread through the pad holes. Make several loops and then tie it off in a knot.

Two locations should be enough. I used the **SCL #3** and **#12** pad holes.

Done!

If you want, you can continue to wear the glove as you work through the guide. Just jack in to upload the sketches. You'll look totally cyberpunk!

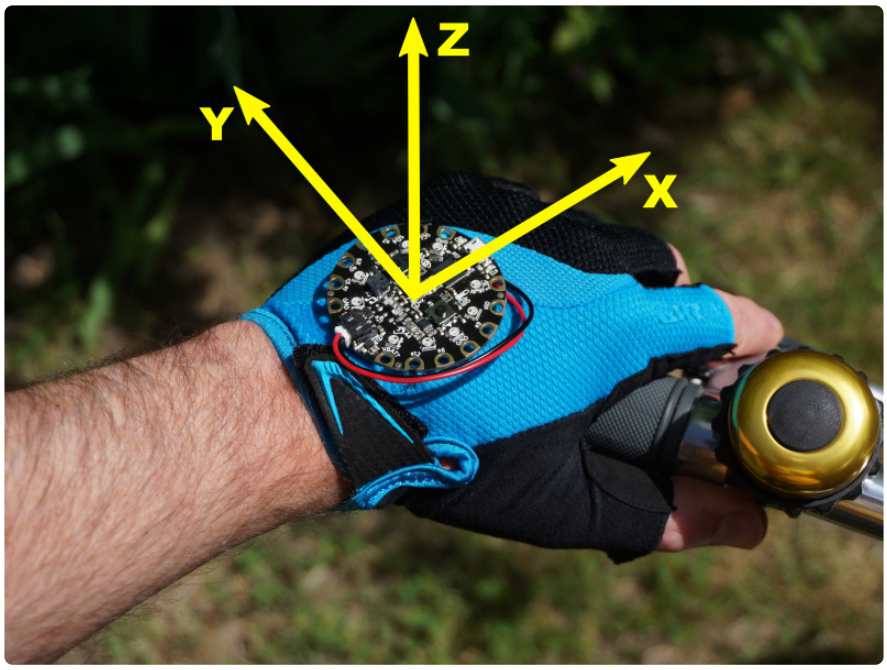


Hand Position and Accelerometer

We will use the accelerometer to determine the hand position. For a good overview of the basics of how an accelerometer works, check out the How Tall Is It? (https://adafru.it/t9f) guide. You can also read some technical details in the Lesson #0 Guide (https://adafru.it/s5A).

With the Circuit Playground sewn on the bike glove as in the previous section, the three main axis line up as shown in the image below.

The coordinate system for the accelerometer on the Circuit Playground Express is slightly different - it's rotated 90 degrees clockwise, we'll account for it in our code later!



Let's start by looking at the accelerometer output in the various hand positions. You can use the simple sketch below which will send the three accelerometer values to the serial port.

```
#include &lt;Adafruit_CircuitPlayground.h&gt;

float X, Y, Z;

void setup() {
  Serial.begin(9600);
  CircuitPlayground.begin();
}

void loop() {
  X = CircuitPlayground.motionX();
  Y = CircuitPlayground.motionY();
  Z = CircuitPlayground.motionZ();
```

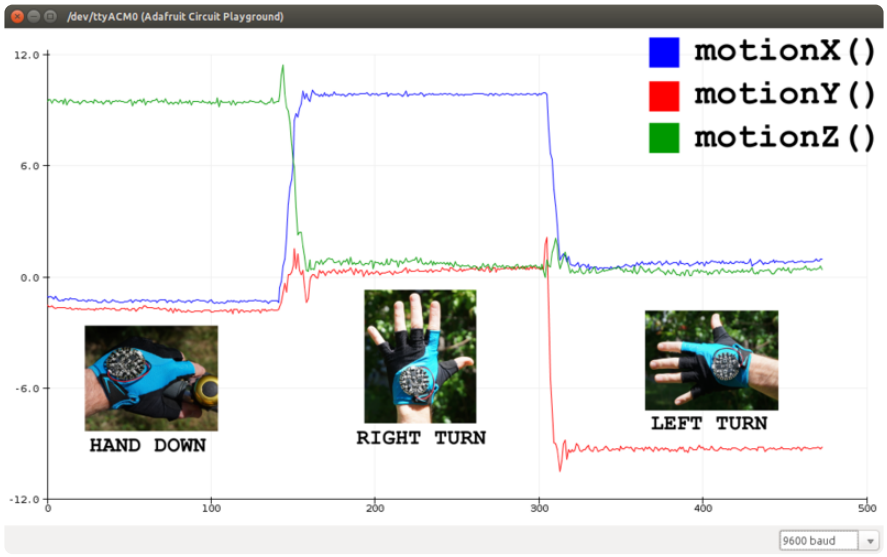
```
  Serial.print(X);
  Serial.print(",");
  Serial.print(Y);
  Serial.print(",");
  Serial.println(Z);

  delay(100);
}
```

With this sketch loaded and running on the Circuit Playground, open the Serial Plotter:

Tools -> Serial Plotter

You should see three lines being drawn, one for each axis of the accelerometer. Now move the glove into the three main positions:

- **HAND DOWN** (on bike grip)
- **RIGHT TURN**
- **LEFT TURN**



Note how in each position there is one reading that stands out. For the hand down position it is the **Z axis** as reported by `motionZ()`. For the right turn position it is the **X axis** as reported by `motionX()`. For the left turn position it is the **Y axis** as reported by `motionY()`. So we can see a strong correlation between the hand positions of interest and the main axis of the accelerometer as:

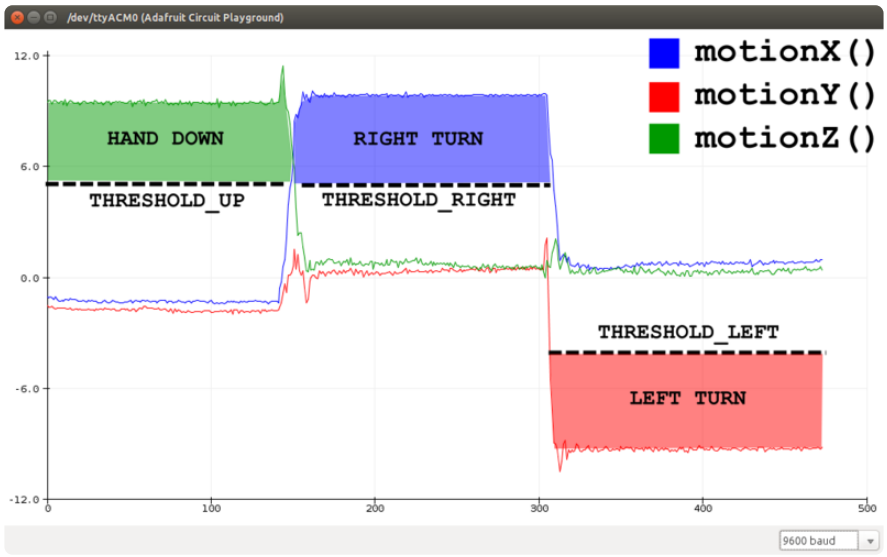
- **HAND DOWN** --> `motionZ()`
- **RIGHT TURN** --> `motionX()`
- **LEFT TURN** --> `motionY()`

So by simply checking the values of the various accelerometer readings, we can determine hand position. Let's see how next.

Hand Position Detection

The basic idea for detecting hand position is to check if the accelerometer values of each of the three axis exceed a preset level. This level is called a threshold. We will use three separate threshold values for the three separate axis values. This idea is shown in the image below.

Note that for the left turn, the value is negative.



Here is a program to demonstrates this. The threshold checks are done using `if` statements. Note that the `if` statements for the left turn/right turn checks are nested within the `if` statement for the hand down check. This is because the only time it makes sense to check for left/right is if the hand is up. We don't want the turn signal lights to turn on while the hand is down on the handle bars.

```
////////////////////////////////////////////////////////////////////////////
// Circuit Playground Bike Glove - Hand Position Detection
//
// Author: Carter Nelson
// MIT License (https://opensource.org/licenses/MIT)

#include &lt;Adafruit_CircuitPlayground.h&gt;

#define THRESHOLD_UP        5   // threshold for hand up test
#define THRESHOLD_RIGHT     5   // threshold for right turn
#define THRESHOLD_LEFT     -5   // threshold for left turn

////////////////////////////////////////////////////////////////////////////
void setup() {
  Serial.begin(9600);

  CircuitPlayground.begin();
}
```

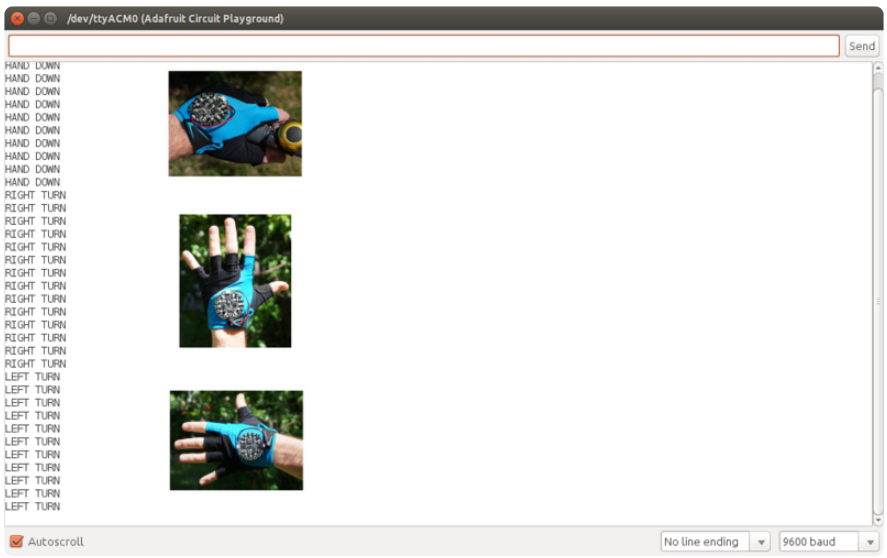
```
////////////////////////////////////////////////////////////////////////////
void loop() {
  // check for hand up or down
  if (CircuitPlayground.motionZ() &gt; THRESHOLD_UP) {

    Serial.println("HAND DOWN");

  } else {

    // check for right turn
    if (CircuitPlayground.motionX() &gt; THRESHOLD_RIGHT) {
      Serial.println("RIGHT TURN");

    // check for left turn
    } else if (CircuitPlayground.motionY() &lt; THRESHOLD_LEFT) {
      Serial.println("LEFT TURN");

    }
  }

  delay(500);
}
```

With this sketch loaded and running on the Circuit Playground, open up the Serial Monitor.

Tools -> Serial Monitor

You should see the current hand position printed out twice a second.

Try moving to the various hand positions and see how they are printed out as shown below.



Pretty simple, but it works.

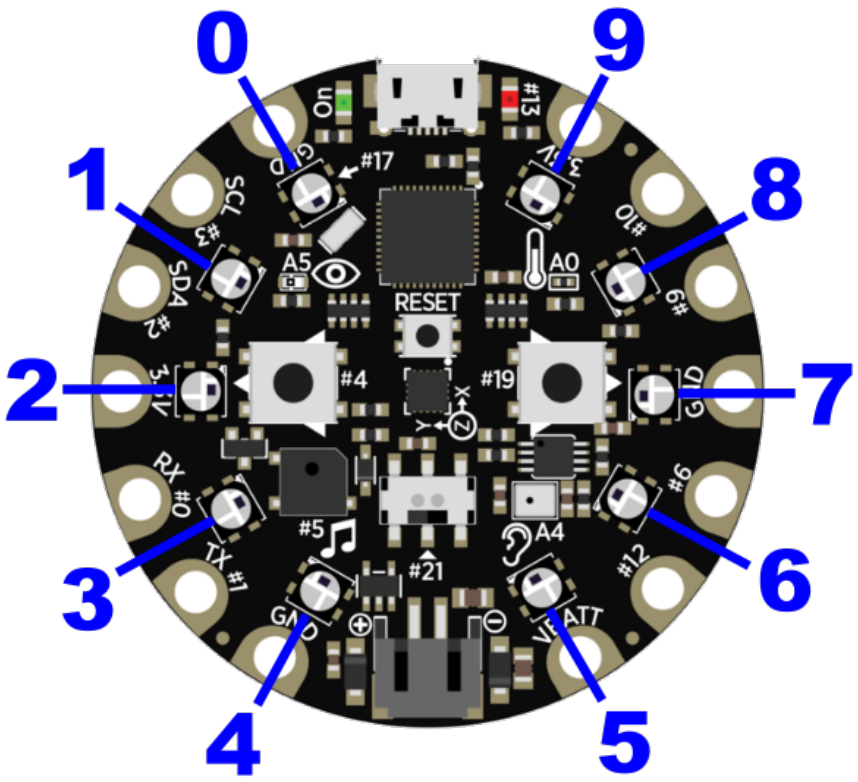
Turn Signal Animations

OK, we got hand position detection working. However, car drivers aren't going to have the Serial Monitor running on their dashboards. So we should do something other than serial output. How about the NeoPixels?

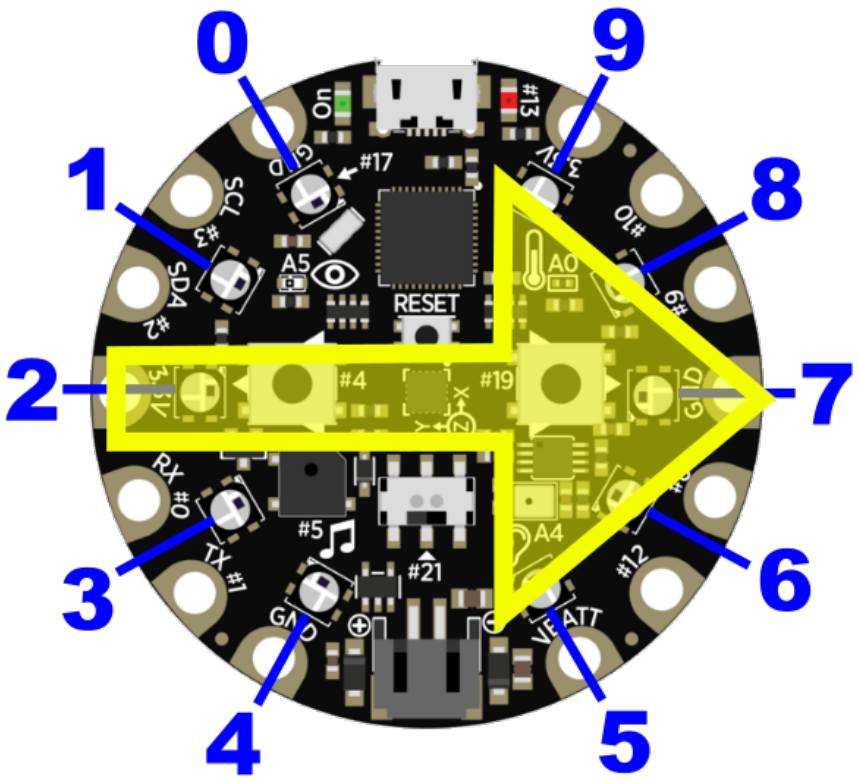
We can replace the `Serial.println()` commands with code that will activate the NeoPixels. Even better, since this code will be in the main `loop()` function, it will be called over and over again. That way we can create some simple animations and let the main `loop()` function drive them.

Right Turn Animation

When the glove is in the 'right turn' orientation, the NeoPixels are arranged as shown in the image below.



What would be neat is if we could make an arrow pointing to the right. Something like this:



It won't be a perfect arrow, but what about using NeoPixels 2, 5, 6, 7, 8, and 9? Here's the code to test this out.

```
////////////////////////////////////////////////////////////////////////////
// Circuit Playground Bike Glove - Right Turn Animation
//
// Author: Carter Nelson
// MIT License (https://opensource.org/licenses/MIT)

#include &lt;Adafruit_CircuitPlayground.h&gt;

#define BRIGHTNESS          255
#define RIGHT_COLOR         0xFFFFFF
#define ANIM_DELAY          200

////////////////////////////////////////////////////////////////////////////
void rightTurnAnimation() {
  // just to be sure, turn off all NeoPixels
  CircuitPlayground.clearPixels();

  // turn on NeoPixels to make an arrow shape
  CircuitPlayground.setPixelColor(2, RIGHT_COLOR);
  CircuitPlayground.setPixelColor(5, RIGHT_COLOR);
  CircuitPlayground.setPixelColor(6, RIGHT_COLOR);
  CircuitPlayground.setPixelColor(7, RIGHT_COLOR);
  CircuitPlayground.setPixelColor(8, RIGHT_COLOR);
  CircuitPlayground.setPixelColor(9, RIGHT_COLOR);

  // wait a little bit
  delay(ANIM_DELAY);

  // turn them all off
  CircuitPlayground.clearPixels();

  // wait again
  delay(ANIM_DELAY);
}
```

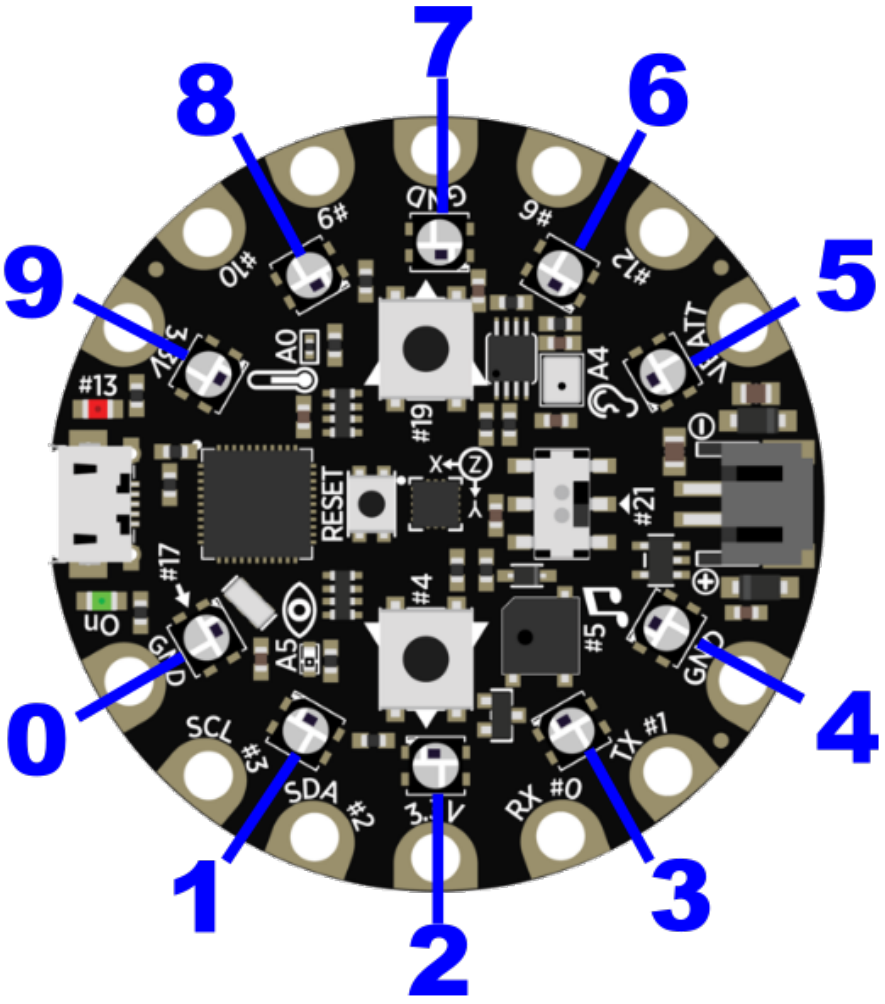
```
////////////////////////////////////////////////////////////////////////////
void setup() {
  CircuitPlayground.begin(BRIGHTNESS);
}

////////////////////////////////////////////////////////////////////////////
void loop() {
  rightTurnAnimation();
}
```

The functions `rightTurnAnimation()` does one 'frame' of the animation. Then, it is called over and over again by being placed in the `loop()` function.

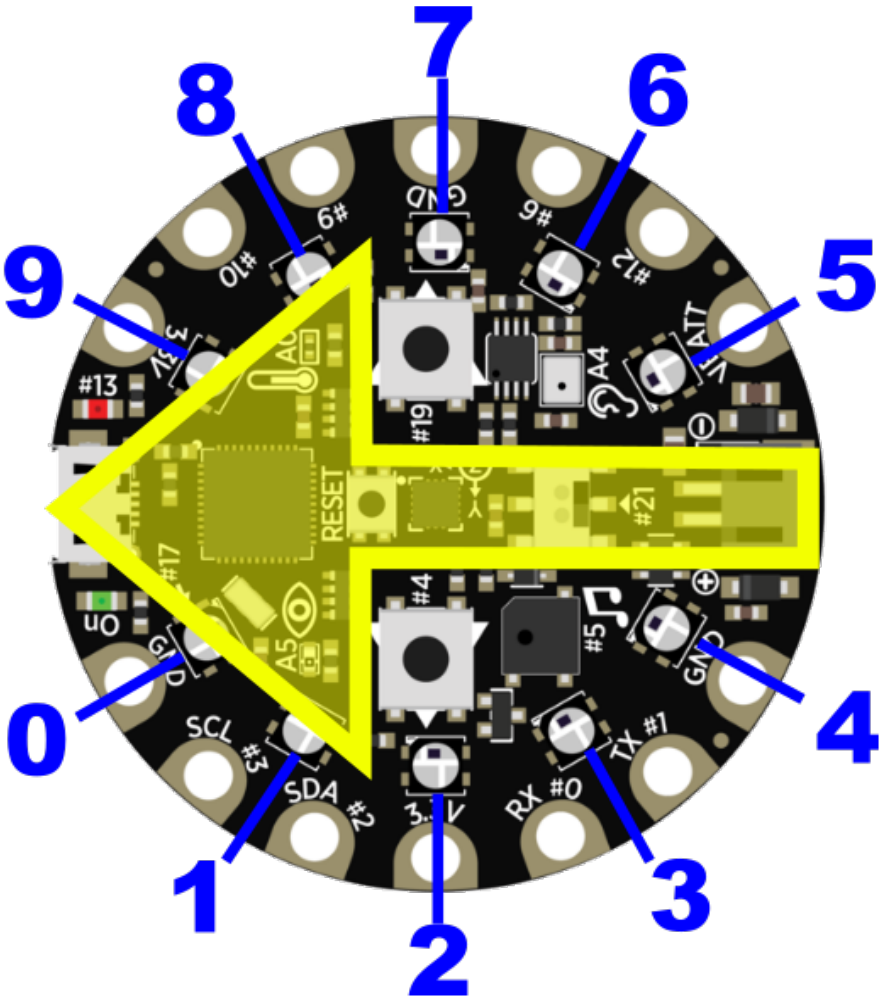
Does this look like an arrow to you? Maybe. Kind of? Well, for now we'll just go ahead and use this as our right turn animation.

Left Turn Animation

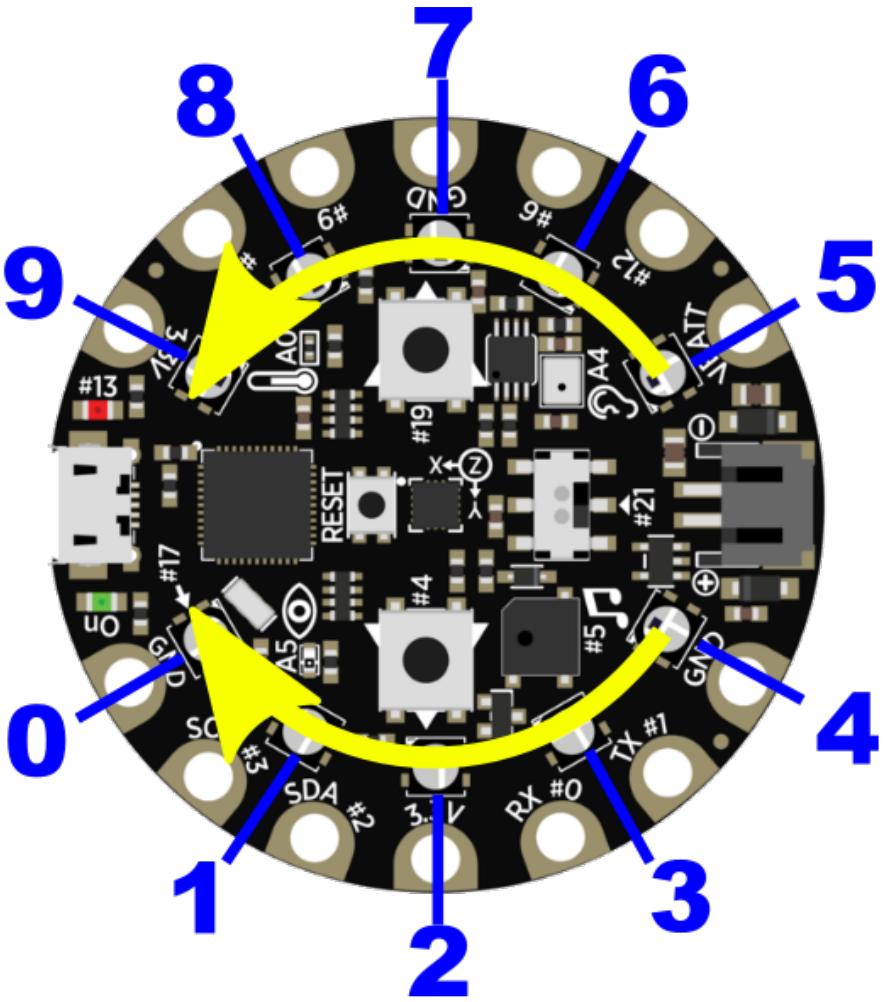
When the glove is in the 'left turn' orientation, the NeoPixels are arranged as shown in the image below.



This is rotated from the right turn orientation, so the NeoPixels are arranged differently. In this arrangement, trying to make a left arrow is a bit of an issue.



Hmmmm. There are only a couple of the NeoPixels that seem to form an arrow. Maybe it would be enough to just use 8, 9, 0, and 1. But let's try something different for the left turn animation. We can make a neat chaser effect and light up the NeoPixels one at a time, moving from right to left. So first 5 & 4, then 6 & 3, then 7 & 2, then 8 & 1, and finally 9 & 0. Something like this:



Here's the code to test this out.

```
////////////////////////////////////////////////////////////////////////////
// Circuit Playground Bike Glove - Left Turn Animation
//
// Author: Carter Nelson
// MIT License (https://opensource.org/licenses/MIT)

#include &lt;Adafruit_CircuitPlayground.h&gt;

#define BRIGHTNESS          255
#define LEFT_COLOR          0xFFFFFF
#define ANIM_DELAY          200

////////////////////////////////////////////////////////////////////////////
void leftTurnAnimation() {
  // just to be sure, turn off all NeoPixels
  CircuitPlayground.clearPixels();

  // Turn on 5 &amp; 4
  CircuitPlayground.setPixelColor(5, LEFT_COLOR);
  CircuitPlayground.setPixelColor(4, LEFT_COLOR);
  delay(ANIM_DELAY);
  CircuitPlayground.clearPixels();

  // Turn on 6 &amp; 3
  CircuitPlayground.setPixelColor(6, LEFT_COLOR);
  CircuitPlayground.setPixelColor(3, LEFT_COLOR);
  delay(ANIM_DELAY);
  CircuitPlayground.clearPixels();
```

```
  // Turn on 7 &amp;amp; 2
  CircuitPlayground.setPixelColor(7, LEFT_COLOR);
  CircuitPlayground.setPixelColor(2, LEFT_COLOR);
  delay(ANIM_DELAY);
  CircuitPlayground.clearPixels();

  // Turn on 8 &amp;amp; 1
  CircuitPlayground.setPixelColor(8, LEFT_COLOR);
  CircuitPlayground.setPixelColor(1, LEFT_COLOR);
  delay(ANIM_DELAY);
  CircuitPlayground.clearPixels();

  // Turn on 9 &amp;amp; 0
  CircuitPlayground.setPixelColor(9, LEFT_COLOR);
  CircuitPlayground.setPixelColor(0, LEFT_COLOR);
  delay(ANIM_DELAY);
  CircuitPlayground.clearPixels();
}

////////////////////////////////////////////////////////////////////////////
void setup() {
  CircuitPlayground.begin(BRIGHTNESS);
}

////////////////////////////////////////////////////////////////////////////
void loop() {
  leftTurnAnimation();
}
```

Just like before, the `leftTurnAnimation()` function does one pass through the animation. Then it is called over and over by the main `loop()`.

This looks pretty cool, so let's use it for our left turns.

Bike Glove Software

Now let's bring it all together into our final code. We just add our animation functions to the Hand Position Detection code and then replace the `Serial.println()` statements with calls to our left/right turn animations.

And here it is:

```
////////////////////////////////////////////////////////////////////////////
// Circuit Playground Bike Glove - With "On/Off"
//
// Author: Carter Nelson
// MIT License (https://opensource.org/licenses/MIT)

#include &lt;Adafruit_CircuitPlayground.h&gt;

#define THRESHOLD_UP        5   // threshold for hand up test
#define THRESHOLD_RIGHT     5   // threshold for right turn
#define THRESHOLD_LEFT     -5   // threshold for left turn

#define BRIGHTNESS          255
#define LEFT_COLOR          0xFFFFFF
#define RIGHT_COLOR         0xFFFFFF
#define ANIM_DELAY          200
```

```
////////////////////////////////////////////////////////////////////////
void leftTurnAnimation() {
  // just to be sure, turn off all NeoPixels
  CircuitPlayground.clearPixels();

  // Turn on 5 &amp;amp; 4
  CircuitPlayground.setPixelColor(5, LEFT_COLOR);
  CircuitPlayground.setPixelColor(4, LEFT_COLOR);
  delay(ANIM_DELAY);
  CircuitPlayground.clearPixels();

  // Turn on 6 &amp;amp; 3
  CircuitPlayground.setPixelColor(6, LEFT_COLOR);
  CircuitPlayground.setPixelColor(3, LEFT_COLOR);
  delay(ANIM_DELAY);
  CircuitPlayground.clearPixels();

  // Turn on 7 &amp;amp; 2
  CircuitPlayground.setPixelColor(7, LEFT_COLOR);
  CircuitPlayground.setPixelColor(2, LEFT_COLOR);
  delay(ANIM_DELAY);
  CircuitPlayground.clearPixels();

  // Turn on 8 &amp;amp; 1
  CircuitPlayground.setPixelColor(8, LEFT_COLOR);
  CircuitPlayground.setPixelColor(1, LEFT_COLOR);
  delay(ANIM_DELAY);
  CircuitPlayground.clearPixels();

  // Turn on 9 &amp;amp; 0
  CircuitPlayground.setPixelColor(9, LEFT_COLOR);
  CircuitPlayground.setPixelColor(0, LEFT_COLOR);
  delay(ANIM_DELAY);
  CircuitPlayground.clearPixels();
}

////////////////////////////////////////////////////////////////////////
void rightTurnAnimation() {
  // just to be sure, turn off all NeoPixels
  CircuitPlayground.clearPixels();

  // turn on NeoPixels to make an arrow shape
  CircuitPlayground.setPixelColor(2, RIGHT_COLOR);
  CircuitPlayground.setPixelColor(5, RIGHT_COLOR);
  CircuitPlayground.setPixelColor(6, RIGHT_COLOR);
  CircuitPlayground.setPixelColor(7, RIGHT_COLOR);
  CircuitPlayground.setPixelColor(8, RIGHT_COLOR);
  CircuitPlayground.setPixelColor(9, RIGHT_COLOR);

  // wait a little bit
  delay(ANIM_DELAY);

  // turn them all off
  CircuitPlayground.clearPixels();

  // wait again
  delay(ANIM_DELAY);
}

////////////////////////////////////////////////////////////////////////
void setup() {
  CircuitPlayground.begin(BRIGHTNESS);
}

////////////////////////////////////////////////////////////////////////
void loop() {
  // check slide switch position
  if (CircuitPlayground.slideSwitch()) {
```

```
    // check for hand up or down
    if (CircuitPlayground.motionZ() &gt; THRESHOLD_UP) {

      // do nothing

    } else {

      // check for right turn
      if (CircuitPlayground.motionX() &gt; THRESHOLD_RIGHT) {
        rightTurnAnimation();

      // check for left turn
      } else if (CircuitPlayground.motionY() &lt; THRESHOLD_LEFT) {
        leftTurnAnimation();

      }
    }
  }
}
```

"ON/OFF" Switch

You may have noticed one extra feature added to this final code. Within the `loop()` function, everything is wrapped in an `if` statement that checks the position of the slide switch using `CircuitPlayground.slideSwitch()`. Like this:

```
void loop() {
  // check slide switch position
  if (CircuitPlayground.slideSwitch()) {
    // MAIN CODE HERE
  }
}
```

This was done to provide a simple on/off capability. If the Circuit Playground had an on/off switch for power, then we could just use that and turn it off for real. But since it doesn't we can use the slide switch. The Circuit Playground is still powered, but the code that makes the bike glove work is only run if the slide switch is in the + position.

Therefore:

Make sure the slide switch is in the + position for the bike glove to work.

Maybe you want to stop and pet a cute kitty you see, but don't want to freak it out with flashing lights. So you can just disable the lights real quick by sliding the switch to - position. Slide bike to + position when you're ready to go.

Happy riding!

Bike Glove CircuitPython

Here is a CircuitPython (https://adafru.it/A22) version of the Bike Glove software.

CircuitPython only works on the Circuit Playground Express.

```
# Circuit Playground Express Bike Glove - With "On/Off"
#
# Author: Carter Nelson
# MIT License (https://opensource.org/licenses/MIT)
import time
from adafruit_circuitplayground.express import cpx

THRESHOLD_UP        = 5   # threshold for hand up test
THRESHOLD_RIGHT     = 5   # threshold for right turn
THRESHOLD_LEFT      = -5  # threshold for left turn

LEFT_COLOR          = 0xFFFFFF
RIGHT_COLOR         = 0xFFFFFF
ANIM_DELAY          = 0.200

RIGHT_TURN = (
  (2, 5, 6, 7, 8, 9),
)

LEFT_TURN = (
  (5, 4),
  (6, 3),
  (7, 2),
  (8, 1),
  (9, 0)
)

def animate_glove(animation, color, delay=ANIM_DELAY):
    for frame in animation:
        cpx.pixels.fill(0)
        for pixel in frame:
            cpx.pixels[pixel] = color
        time.sleep(delay)

    cpx.pixels.fill(0)
    time.sleep(ANIM_DELAY)

# Loop forever
while True:
    # Check slide switch position
    if cpx.switch:
        # Get acceleration values
        X, Y, Z = cpx.acceleration
        # Check if glove is down or up
        if Z &lt; THRESHOLD_UP:
            # Determine glove orienation and animate
            if Y &gt; THRESHOLD_RIGHT:
                animate_glove(RIGHT_TURN, RIGHT_COLOR)
            elif X &gt; THRESHOLD_LEFT:
                animate_glove(LEFT_TURN, LEFT_COLOR)
```

Questions and Code Challenges

The following are some questions related to this project along with some suggested code challenges. The idea is to provoke thought, test your understanding, and get you coding!

While the sketches provided in this guide work, there is room for improvement and additional features. Have fun playing with the provided code to see what you can do with it. Or do something new and exciting!

Questions

- Is it possible for multiple hand positions to be reported at the same time?
- Is it possible for hand positions to be falsely reported?
- Why was a value of 5 used for the thresholds?

Code Challenges

- Use the light sensor to automatically adjust the NeoPixel brightness.
- Change the color of the NeoPixels.
- Change the `if/else` block for the hand up/down check into a single `if` statement. (hint: invert logic)
- Use the speaker to add an audio cue that the turn signals are active. This let's the rider know the signals are engaged without needing to look at the glove.
- Add additional hand position detections and animations, for example slow down/stop.