Textile Potentiometer Hoodie

Created by Becky Stern

https://learn.adafruit.com/textile-potentiometer-hoodie

Last updated on 2024-06-03 01:40:13 PM EDT

Table of Contents

Overview

This guide was written for the 'original' Gemma board, but can be done with either the original or M0 Gemma. We recommend the Gemma M0 as it is easier to use and is more compatible with modern computers!

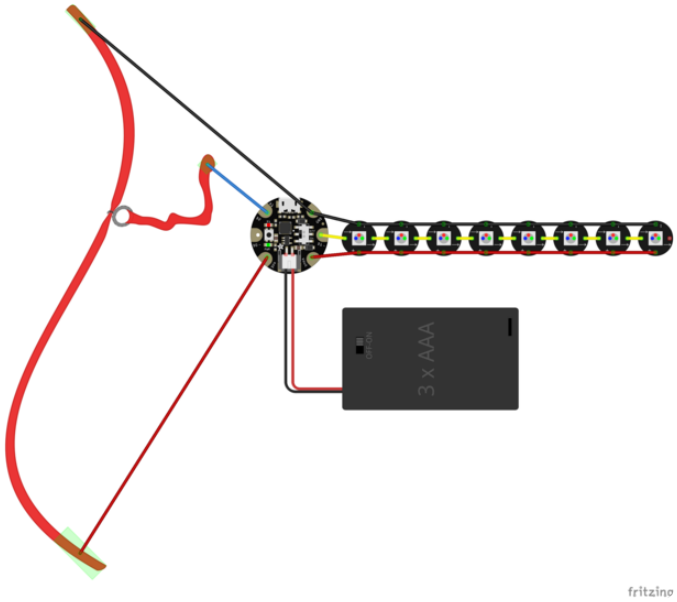
In this all-sewing (no soldering) project, you'll learn to stitch up a textile slide actuator to control color changing LEDs using the GEMMA sewable microcontroller. You will need:

- Soft potentiometer by Plug&Wear (http://adafru.it/2273)
- Adafruit Gemma M0 (http://adafru.it/3501) or Adafruit GEMMA (http://adafru.it/1222)
- Sewable NeoPixels (http://adafru.it/1260)
- 3-ply stainless conductive thread (http://adafru.it/641)
- 3xAAA battery pack (http://adafru.it/727) and JST extension (http://adafru.it/1131)
- Sewing needles (http://adafru.it/615) and plain thread
- Scissors
- Clear nail polish
- Multimeter (optional but useful for troubleshooting)

Before you begin please read these prerequisite guides:

- Conductive Thread (https://adafru.it/dn3)
- Introducing Gemma M0 (https://adafru.it/yeq)or Introducing GEMMA (https://adafru.it/cHH)
- Flora RGB Smart NeoPixels (https://adafru.it/Cf3)
- Washing Wearable Electronics (https://adafru.it/tfM)
- Battery Powering your Wearable Electronics (https://adafru.it/e4c)

Circuit Diagram



This diagram uses the original Gemma but you can also use the Gemma M0 with the exact same wiring!

Click to enlarge.

- One end of soft potentiometer is stitched to GEMMA GND
- Other end of soft potentiometer is stitched to GEMMA 3V
- Slide charm on soft pot is stitched to GEMMA A1 (also marked D2)
- NeoPixels - connected to GEMMA GND
- NeoPixels + connected to GEMMA Vout
- First NeoPixel data in (arrow pointing inward) connected to GEMMA D1
- Battery pack connected to JST port

Prepare and Sew Sensor

We chose to attach the sensor parallel to the zipper on the front of the hoodie and use the drawstring to perform the sliding action. Yes, you can install the sensor along a zipper and use the zipper pull as the slider, which would couple the temperature control action of zipping and unzipping the hoodie with the LED color change effect. For simplicity of demonstration, we've kept them separate.

We placed GEMMA just inside the front lapel of the hoodie. This hoodie has a front facing along the inside edge, and you could easily hide GEMMA completely inside. Again for simplicity of showing you, we've left it visible on the outside of the facing.

Cut the sensor's ribbon into two pieces. We'll use one as the pull tab and one to slide along.

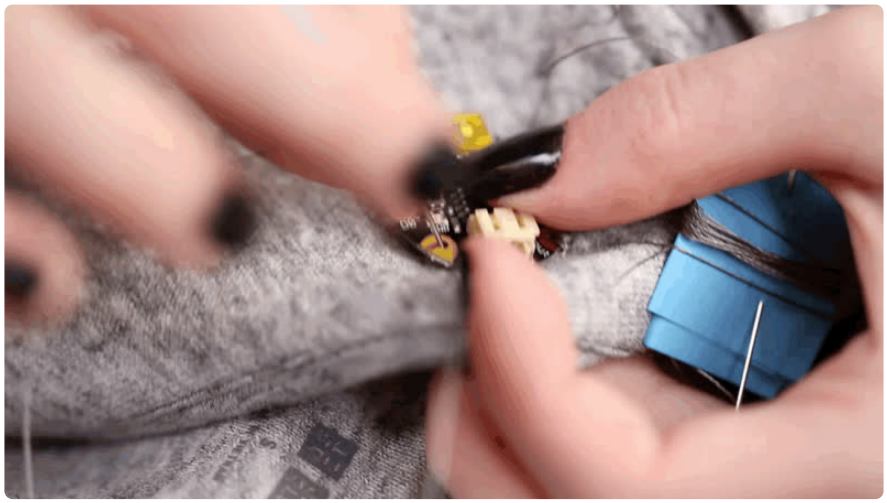
Use a needle and thread (knotted at the end) to gather up the stitches at one end of the slide-along piece and thread it through the small end of the charm that comes with the soft potentiometer. It's a snug fit, so you may need to try a few times.

To attach the pull tab, stitch the other piece of ribbon to the large loop on the charm. You can use regular thread for this if you wish, but conductive thread won't hurt. Cinch the ribbon to the charm tightly for a stable electrical signal.

Stitch the other end of the pull tab ribbon to the outlet of the hoodie's drawstring, then back to the pad marked A1 (also D2) on GEMMA. Refer to the circuit diagram.

Next stitch down both ends of the slide-along ribbon using conductive thread and connect them to GEMMA 3V and GND according to the circuit diagram using a running stitch.

For the far-away end, you can stitch along the inside of the facing to insulate your stitches.



When you reach GEMMA, loop around the pad many times and then use the needle to knot the thread at the back or away from the GEMMA pad. Your stitches should be nice and tight to ensure a secure connection and prevent short circuits.

Seal knots by pulling tight and dabbing on a small amount of clear nail polish. A little goes a long way! Try not to get it on the garment itself. Periodically pull the thread tight while it dries, applying more nail polish if necessary to prevent the knots from springing open.

Do not cut your thread tails until the nail polish is dry and you've tugged them to ensure the knots are secure.

Plug GEMMA into your compute with a USB cable and load up the following code:

```
//very slightly modified version of Arduino's "AnalogInput" example sketch for
testing the textile potentiometer with GEMMA

int sensorPin = 1;    // select the input pin for the potentiometer, analog 1 on
GEMMA is digital 2
int ledPin = 1;      // select the pin for the LED, GEMMA has one attached to D1
int sensorValue = 0;  // variable to store the value coming from the sensor

void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  // turn the ledPin on
  digitalWrite(ledPin, HIGH);
  // stop the program for &lt;sensorValue&gt; milliseconds:
  delay(sensorValue);
  // turn the ledPin off:
  digitalWrite(ledPin, LOW);
  // stop the program for for &lt;sensorValue&gt; milliseconds:
  delay(sensorValue);
}
```

This code uses the changing value of the slide sensor to adjust the blinking speed of GEMMA's onboard LED. Slide the sensor and watch the LED blink faster or slower.

LED not changing blink speed? Unplug and try checking your circuit for shorts, using a multimeter if possible.

Is the blinking speed changing? Great! Now let's add some color changing NeoPixels.

Stitch Pixels

Add a chain of NeoPixels to GEMMA's GND, D1, and Vout pads. Any NeoPixels will work just fine (strips, sticks, rings, etc.) but in keeping with the textile theme of this project, we'll be using individual sewable pixels.

Start by stitching between GEMMA D1 and the data input of your first pixel. Knot the thread at each and and seal/snip.

Stitch very long strands of conductive thread to GEMMA GND and Vout and stitch each up to - and + on the first pixel. Knot the threads at the pixel but do not seal or cut the leads short. Instead wrap the excess threads around bobbins or scraps of paper to keep them out of the way.

Stop stitching after each pixel to test your stitched connections.

Eight pixels stitched around the inside of the hood:

Arduino Code

The Arduino code presented below works equally well on all versions of GEMMA: v1, v2 and M0. But if you have an M0 board, consider using the CircuitPython code on the next page of this guide, no Arduino IDE required!

Upload the code below to GEMMA to see the interactive color changing effect using the soft potentiometer. This sketch doesn't do any smoothing of the sensor value before directly mapping it to a color, so the LEDs will flash different colors as you slide, then stay on a single color when you let go of the slide charm. We wanted to keep the code as simple as possible so you can learn exactly how the sensor works, so feel free to upgrade it to suit your project's needs!

```
// SPDX-FileCopyrightText: 2017 Mikey Sklar for Adafruit Industries
//
// SPDX-License-Identifier: MIT

#include <Adafruit_NeoPixel.h>

#define PIN 1
// 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)
Adafruit_NeoPixel strip = Adafruit_NeoPixel(8, PIN, NEO_GRB + NEO_KHZ800);

int sensorPin = 1;    // select the input pin for the potentiometer (analog 1 is
digital 2)
int sensorValue = 0;  // variable to store the value coming from the sensor
int colorValue = 0;

void setup() {
  // Set internal pullup resistor for sensor pin (analog 1 is digital 2)
  pinMode(2, INPUT_PULLUP);
  strip.begin();
  strip.setBrightness(40); //adjust brightness here
  strip.show(); // Initialize all pixels to 'off'
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  colorValue = map(sensorValue, 0, 1024, 0, 255); //map sensor values from 0-124 to
0-255
  for (int i = 0; i<strip.numPixels(); i++){
    strip.setPixelColor(i, Wheel(colorValue)); //use Wheel function to set color
  }
  strip.show();
}

// Input a value 0 to 255 to get a color value.
// The colours are a transition r - g - b - back to r.
uint32_t Wheel(byte WheelPos) {
  if(WheelPos < 85) {
   return strip.Color(WheelPos * 3, 255 - WheelPos * 3, 0);
```

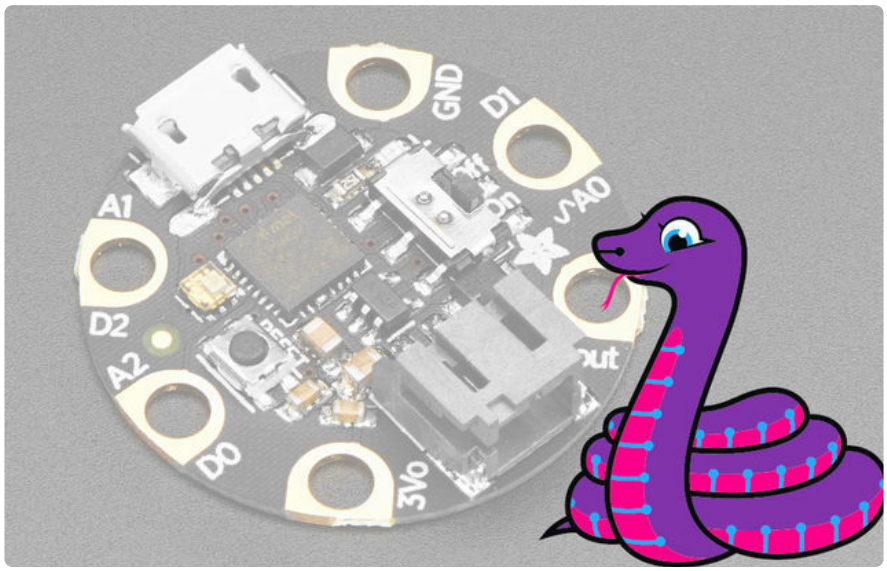
```
  } else if(WheelPos < 170) {
   WheelPos -= 85;
   return strip.Color(255 - WheelPos * 3, 0, WheelPos * 3);
  } else {
   WheelPos -= 170;
   return strip.Color(0, WheelPos * 3, 255 - WheelPos * 3);
  }
}
```

If this is your first time using GEMMA, work through the Introducing G (https://adafru.it/cHH)EMMA (https://adafru.it/cHH) guide first; you need to customize some settings in the Arduino IDE. Once you have it up and running (test the 'blink' sketch), then follow the instructions on the following page for installing the NeoPixel library:

NeoPixel Überguide: Arduino Library Installation (https://adafru.it/nBF)

Plug in your circuit and load up the sketch below:

CircuitPython Code



GEMMA M0 boards can run **CircuitPython** — a different approach to programming compared to Arduino sketches. In fact, **CircuitPython comes factory pre-loaded on GEMMA M0**. If you've overwritten it with an Arduino sketch, or just want to learn the basics of setting up and using CircuitPython, this is explained in the **Adafruit GEMMA M0 guide** (https://adafru.it/z1B).

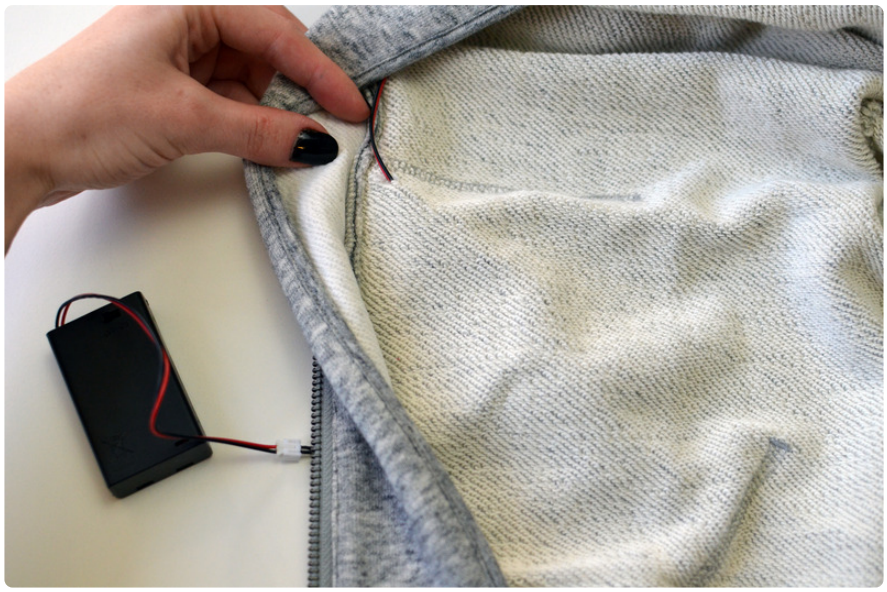
These directions are specific to the "M0" GEMMA board. The original GEMMA with an 8-bit AVR microcontroller doesn't run CircuitPython...for those boards, use the Arduino sketch on the "Arduino code" page of this guide.

Below is CircuitPython code that works similarly (though not exactly the same) as the Arduino sketch shown on a prior page. To use this, plug the GEMMA M0 into USB...it should show up on your computer as a small **flash drive**...then edit the file "**code.py**" with your text editor of choice. Select and copy the code below and paste it into that file, **entirely replacing its contents** (don't mix it in with lingering bits of old code). When you save the file, the code should **start running almost immediately** (if not, see notes at the bottom of this page).

If GEMMA M0 doesn't show up as a drive, follow the GEMMA M0 guide link above to prepare the board for CircuitPython.

```
# SPDX-FileCopyrightText: 2018 Mikey Sklar for Adafruit Industries
#
# SPDX-License-Identifier: MIT

import analogio
import board
from rainbowio import colorwheel
import neopixel

# Initialize input/output pins
sensor_pin = board.A1  # input pin for the potentiometer
sensor = analogio.AnalogIn(sensor_pin)

pix_pin = board.D1  # pin where NeoPixels are connected
num_pix = 8  # number of neopixels
strip = neopixel.NeoPixel(pix_pin, num_pix, brightness=.15, auto_write=False)

color_value = 0
sensor_value = 0


def remap_range(value, left_min, left_max, right_min, right_max):
    # this remaps a value from original (left) range to new (right) range
    # Figure out how 'wide' each range is
    left_span = left_max - left_min
    right_span = right_max - right_min

    # Convert the left range into a 0-1 range (int)
    valueScaled = int(value - left_min) / int(left_span)

    # Convert the 0-1 range into a value in the right range.
    return int(right_min + (valueScaled * right_span))


# Loop forever...
while True:

    # remap the potentiometer analog sensor values from 0-65535 to RGB 0-255
    color_value = remap_range(sensor.value, 0, 65535, 0, 255)

    for i in range(len(strip)):
        strip[i] = colorwheel(color_value)
    strip.write()
```

Wear it!



Create a small hole in the upper inside edge of the hoodie's front pocket, and thread through the JST battery wire. Store the AAA pack in the pocket, and run the JST extension up through the front facing to plug into GEMMA's JST port.

Turn twilight into your own colored light show! You can use this basic project as a jumping-off point for your own e-textile slidey wearables; what will you make?

Do not wear this circuit in the rain! Power down and keep the battery pack dry in case of inclement weather. To wash, remove battery pack and hand/machine wash, line dry.