adafruit learning system

Gemma-Powered NeoPixel LED Sound Reactive Drums

Created by Ruiz Brothers

Last updated on 2018-08-22 03:38:49 PM UTC

Guide Contents

Overview

Light up your drums with sound. This guide will help you upgrade your drums to have sound reactive LEDs. This build uses a mic amp sensor and Gemma to light up NeoPixels to the beat of your drums. The cost of this build is considerably lower than other kits. Its also compact, rechargable and mobile!

We made a circuit for a snare, mid-tom, hi-tom and a drum kick. Each drum is independent from one another but can also trigger other pieces if stricken loud enough. Our project cost one third of the price of other led drum kits on the market (https://adafru.it/tEz)! There are other tutorials (https://adafru.it/cYH) out there that use a Piezo element and several components (capacitors, resistors, timers, etc) but our tutorial is a lot easier to achieve with Adafruit's low-cost and powerful micro-controllers, sensors and LEDs.

Prerequisite guides

- NeoPixel Überguide (https://adafru.it/cEz) (https://adafru.it/cEz)
- Getting Started with G (https://adafru.it/cHH)emma (https://adafru.it/cHH)

Tools & Supplies

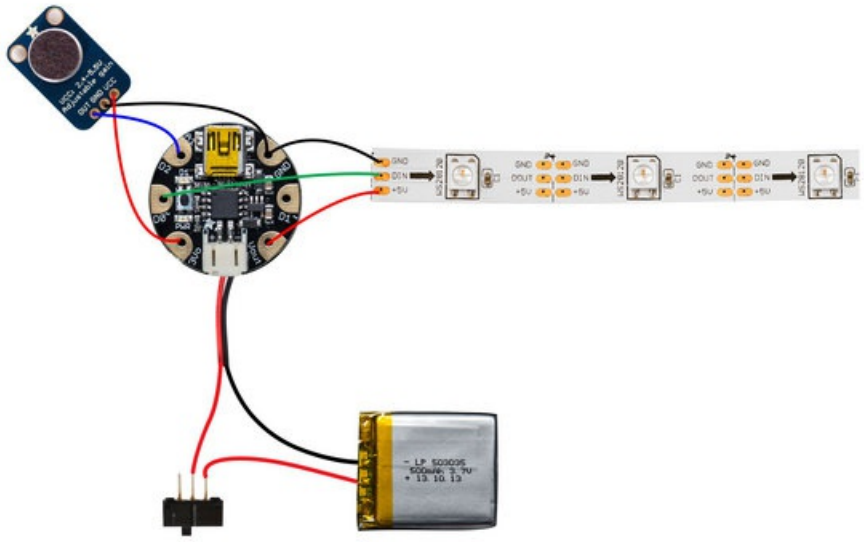
- Soldering iron (https://adafru.it/caH)
- Solder
- Scissors (http://adafru.it/1599)
- wire stripers (http://adafru.it/527)
- wire spool set (http://adafru.it/1311)

- Double sided foam tape
- Paper tape
- 3D Printer (http://adafru.it/1292) (optional)

Parts

- Drum Set
- Gemma (http://adafru.it/1222)
- Mic Amp (http://adafru.it/1063)
- Slide Switch (http://adafru.it/805)
- Lithium Polymer Battery (http://adafru.it/1578)
- NeoPixel Strips (https://adafru.it/caH)

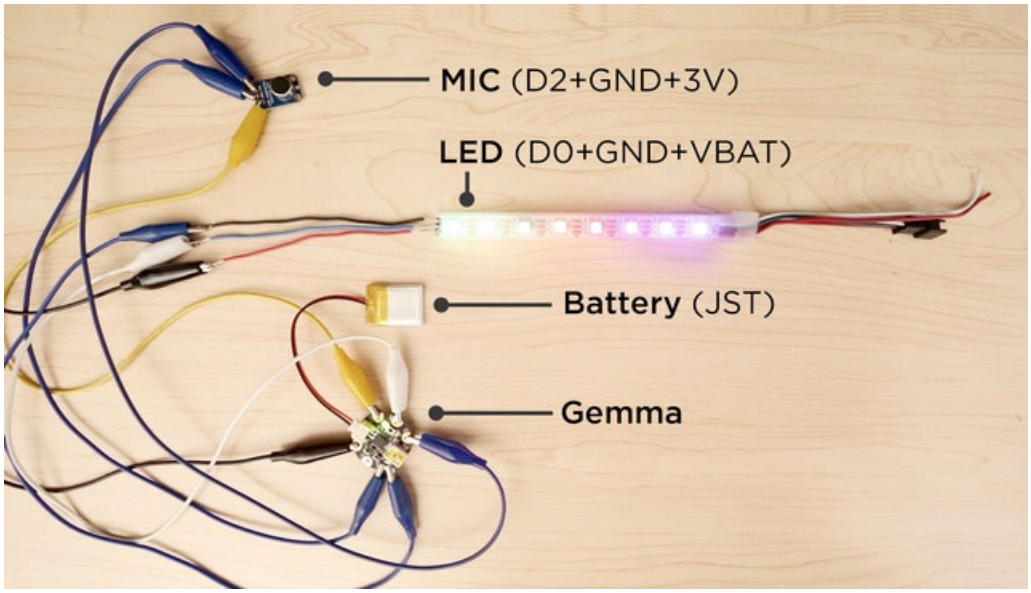
Circuit Diagram



Prototype Circuit

It's good protocol (& wise engineering!) to prototype your circuits, you can use small alligator clips to connect the components together. The pins on the mic sensor are small, so be sure to double check your connections if your having trouble getting the circuit to work. It might be easier to solder wires to the mic and then alligator clip to those. Since drumsets are so loud, the code is set to have a low sensitivity for the mic, so make sure to give a loud sound when testing the NeoPixels audio response. Rubbing the microphone with your finger is a good way to get a reaction.

The NeoPixel strips digital input connects to pin **D0** of the Gemma. The negative connection of the NeoPixel strip goes to the ground pin on the Gemma. The positive power wire of the NeoPixel LED strip connects to the **VBat** pin of the Gemma (not 3.3V!) The out pin on the mic amp goes to pin **A1/D2** of the Gemma - this is an anolog input pin. The positive power breakout pin on the mic amp will connect to the 3.3v pin of the Gemma since it is a clear power source than VBat, the voltage goes through the onboard regulator. The negative ground pin of the mic amp can share the same ground connection on the Gemma (together with the NeoPixel strip).



The Code

The color organ code was originally created for the Trinket but also works great with the Gemma. The Gemma is programmed via USB with the Arduino IDE. You can modify and customize the code to fit your setup. For starters, we can easy change the pin outs and number of neopixels. In our setup, each drum used 60 NeoPixels for each drum piece.

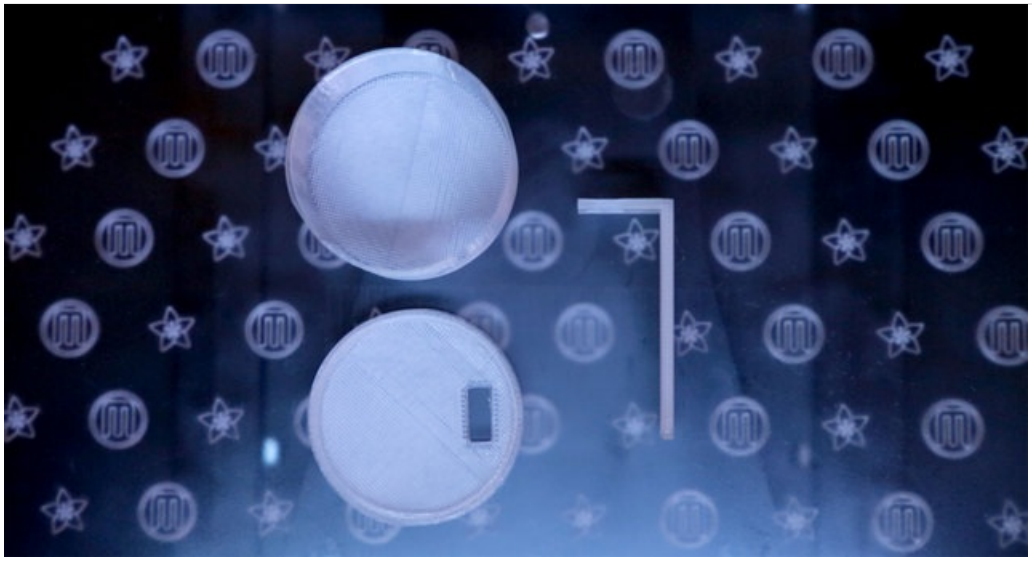
If your new to using the Gemma and the Arduino IDE, check out this guide (https://adafru.it/cYI) for setting up. Have some extra time on your hands? Learn how to customize (https://adafru.it/cYJ) the code to change the color!

```
/* LED "Color Organ" for Adafruit Trinket and NeoPixel LEDs.

Hardware requirements:
 - Adafruit Trinket or Gemma mini microcontroller (ATTiny85).
 - Adafruit Electret Microphone Amplifier (ID: 1063)
 - Several Neopixels, you can mix and match
   o Adafruit Flora RGB Smart Pixels (ID: 1260)
   o Adafruit NeoPixel Digital LED strip (ID: 1138)
   o Adafruit Neopixel Ring (ID: 1463)

Software requirements:
 - Adafruit NeoPixel library

Connections:
 - 5 V to mic amp +
 - GND to mic amp -
 - Analog pinto microphone output (configurable below)
 - Digital pin to LED data input (configurable below)

Written by Adafruit Industries.  Distributed under the BSD license.
This paragraph must be included in any redistribution.
*/
#include <Adafruit_NeoPixel.h>

#define N_PIXELS  60  // Number of pixels you are using
#define MIC_PIN   A1  // Microphone is attached to Trinket GPIO #2/Gemma D2 (A1)
#define LED_PIN    0  // NeoPixel LED strand is connected to GPIO #0 / D0
#define DC_OFFSET  0  // DC offset in mic signal - if unsure, leave 0
```

```
#define DC_OFFSET  0  // DC offset in mic signal - if unsure, leave 0
#define NOISE     100  // Noise/hum/interference in mic signal
#define SAMPLES   60  // Length of buffer for dynamic level adjustment
#define TOP       (N_PIXELS +1) // Allow dot to go slightly off scale
// Comment out the next line if you do not want brightness control or have a Gemma
#define POT_PIN    3  // if defined, a potentiometer is on GPIO #3 (A3, Trinket only)

byte
  peak      = 0,      // Used for falling dot
  dotCount  = 0,      // Frame counter for delaying dot-falling speed
  volCount  = 0;      // Frame counter for storing past volume data

int
  vol[SAMPLES],       // Collection of prior volume samples
  lvl       = 10,     // Current "dampened" audio level
  minLvlAvg = 0,      // For dynamic adjustment of graph low & high
  maxLvlAvg = 512;

Adafruit_NeoPixel  strip = Adafruit_NeoPixel(N_PIXELS, LED_PIN, NEO_GRB + NEO_KHZ800);

void setup() {
  //memset(vol, 0, sizeof(vol));
  memset(vol,0,sizeof(int)*SAMPLES);//Thanks Neil!
  strip.begin();
}
void loop() {
  uint8_t  i;
  uint16_t minLvl, maxLvl;
  int      n, height;
  n   = analogRead(MIC_PIN);                 // Raw reading from mic
  n   = abs(n - 512 - DC_OFFSET);            // Center on zero
  n   = (n <= NOISE) ? 0 : (n - NOISE);      // Remove noise/hum
  lvl = ((lvl * 7) + n) >> 3;    // "Dampened" reading (else looks twitchy)

  // Calculate bar height based on dynamic min/max levels (fixed point):
  height = TOP * (lvl - minLvlAvg) / (long)(maxLvlAvg - minLvlAvg);

  if(height < 0L)       height = 0;      // Clip output
  else if(height > TOP) height = TOP;
  if(height > peak)     peak   = height; // Keep 'peak' dot at top

// if POT_PIN is defined, we have a potentiometer on GPIO #3 on a Trinket
//    (Gemma doesn't have this pin)
  uint8_t bright = 255;
#ifdef POT_PIN
   bright = analogRead(POT_PIN);  // Read pin (0-255) (adjust potentiometer
                                  //   to give 0 to Vcc volts
#endif
  strip.setBrightness(bright);    // Set LED brightness (if POT_PIN at top
                                  //  define commented out, will be full)
  // Color pixels based on rainbow gradient
  for(i=0; i<N_PIXELS; i++) {
    if(i >= height)
       strip.setPixelColor(i,   0,   0, 0);
    else
       strip.setPixelColor(i,Wheel(map(i,0,strip.numPixels()-1,30,150)));
    }

   strip.show(); // Update strip
```

```
  vol[volCount] = n;                      // Save sample for dynamic leveling
  if(++volCount >= SAMPLES) volCount = 0; // Advance/rollover sample counter

  // Get volume range of prior frames
  minLvl = maxLvl = vol[0];
  for(i=1; i<SAMPLES; i++) {
    if(vol[i] < minLvl)      minLvl = vol[i];
    else if(vol[i] > maxLvl) maxLvl = vol[i];
  }
  // minLvl and maxLvl indicate the volume range over prior frames, used
  // for vertically scaling the output graph (so it looks interesting
  // regardless of volume level).  If they're too close together though
  // (e.g. at very low volume levels) the graph becomes super coarse
  // and 'jumpy'...so keep some minimum distance between them (this
  // also lets the graph go to zero when no sound is playing):
  if((maxLvl - minLvl) < TOP) maxLvl = minLvl + TOP;
  minLvlAvg = (minLvlAvg * 63 + minLvl) >> 6; // Dampen min/max levels
  maxLvlAvg = (maxLvlAvg * 63 + maxLvl) >> 6; // (fake rolling average)
}

// Input a value 0 to 255 to get a color value.
// The colors 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);
  }
}
```

3D Printing



Our three piece design will house the gemma, a recharger battery and the slide switch. The cover is designed to mount a slide switch on top for easy powering access. The cover snaps onto the top of the case. The hanger can be attached to the back of the base. The hanger has a hole that fits into standard sized drum shell pins. The case design is slim and non-intrusive while playing and doesnt look like an eye-sore in transparent PLA material.

https://adafru.it/cYK

https://adafru.it/cYK

Download and print each part with the following settings:

Gemma Drum Enclosure About 30 minutes 4g	PLA @230 No Raft No Support	.20 Layer Hieght 90/150mm/s

Printing Techniques

Build Plate Preparations

There's a great video tutorial (https://adafru.it/cRd) by Dr. Henry Thomas who demonstrations a great technique for preparing acrylic build plates for awesome prints. Wipe down the plate with a paper towel lightly dabbed in acetone. Use another paper towel and apply a tiny dab of olive oil. Wipe down the plate so a small film of oil is applied, this will allow the parts to come off the plate easier.

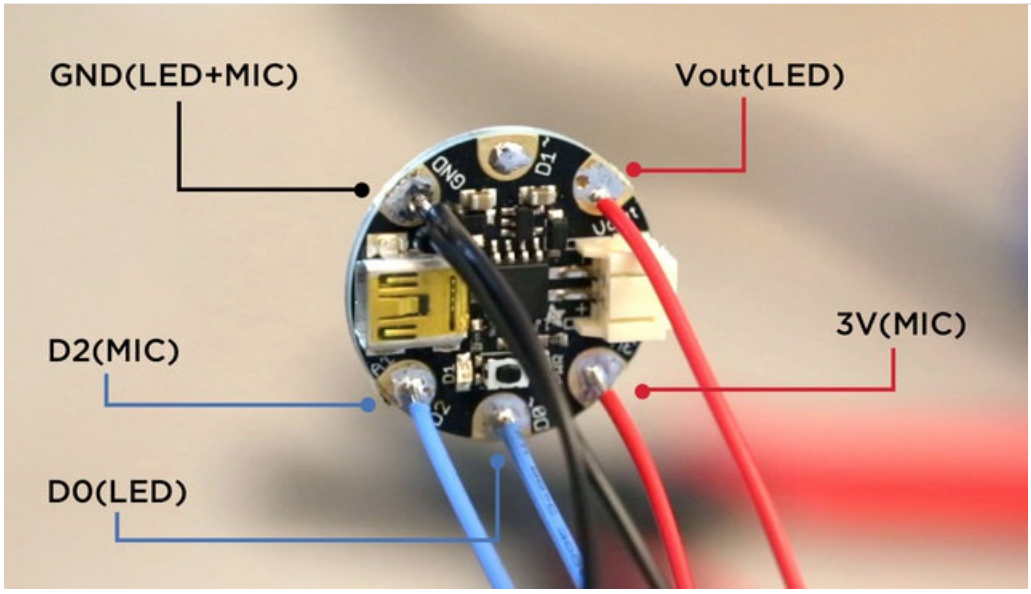
Live Level

We recommend going raft-less for each piece because it will have the best quality result. Each piece will require a well leveled platform. We tend to "live level" our prints, meaning we adjust the build plates thumb screws while the print is laying down filament. This way we can make adjustments directly and improve the leveling by seeing how the extruders are laying down the first layer onto the build plate. We recommend watching the first layer so that you get a more successful print. If you see the layers aren't sticking or getting knocked off, you can always cancel print, peel it off and try again.

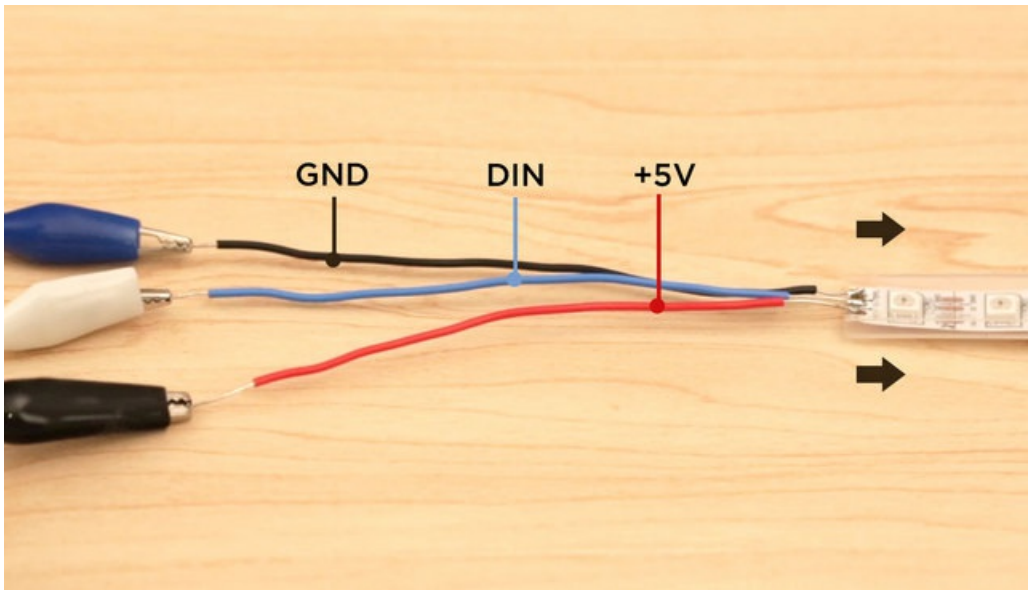
Once your pieces are printed, test out to see if the cover tightly snaps onto the case. Check to see if the slide switch fit through the cutout on the cover. If its too tight, you can use an X-Acto knife to shave off the edges. If its too lose, you

can secure it in place with adhesive. The hanger will be attached to the back of the case later with doubled-sided foam tape. Set these pieces aside for now. It's time to solder some components!

Build

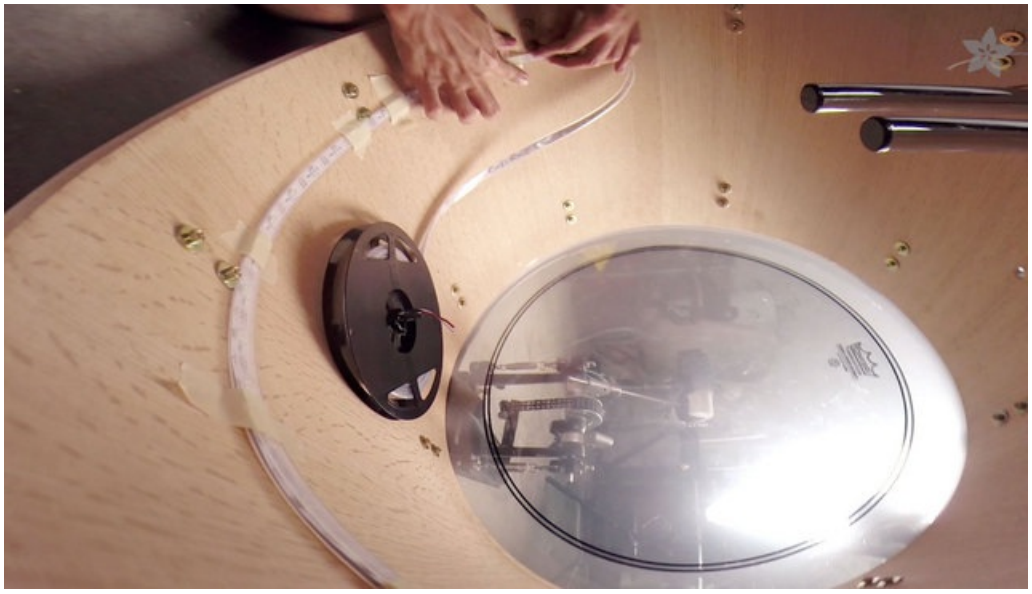


Once you have your circuit prototype tested and working, you will need to solder wires to these components for a solid connection. You should start by measuring lengths of wires needed for connecting the Gemma to the NeoPixels and mic sensor. The wires should be long enough to run through the air hole and inside the drum shell. The main circuit (which contains the gemma, battery and switch) will be fitted inside an enclosure and mounted on the side of the drum shell closest to the air hole. To see if your wires are long enough, place the Gemma into position and see if the wire is long enough to connect the NeoPixel strips inside the drum shell. It's fine to have some extra wire.

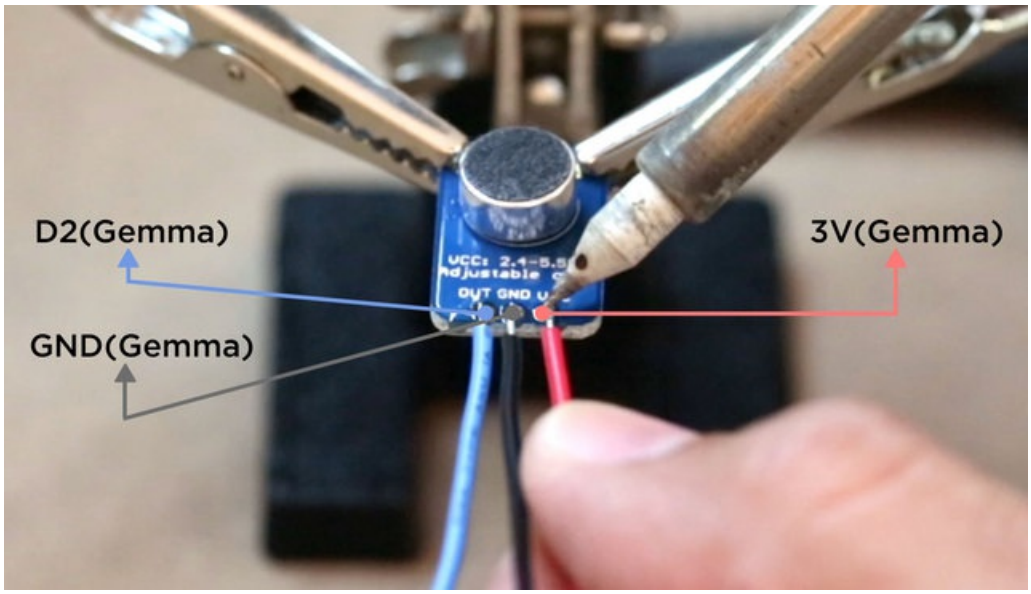


The NeoPixel strip will need three wires soldered to the beginning of the pixel strip. Solder a black wire to the GND pin. A blue wire to the DIN pin. And a red wire to the positive +5V pin. The length of these three wires can be short because we can use alligator clips to connect them to the longer wires that will be soldered to the Gemma.

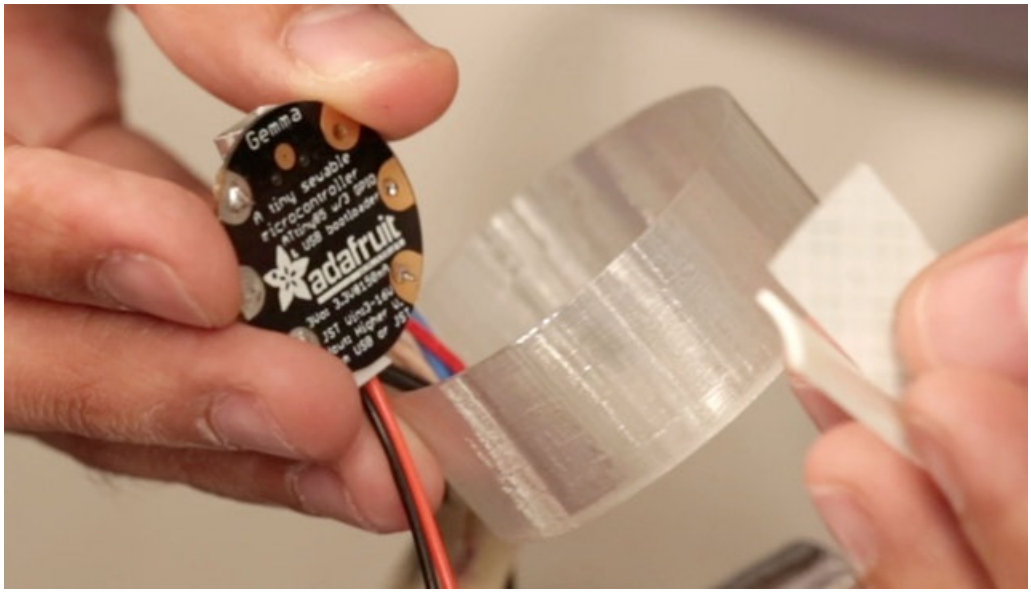
The NeoPixel strips will be mounted to the inner lining of the drum shells. You can measure and cut the length of an LED strip needed to cover the whole shell. Use the indictor on the strip to cut them properly. Position the first LED pixel closest to the air hole of the shell. The wires at the end of the strip should remain inside the shell.



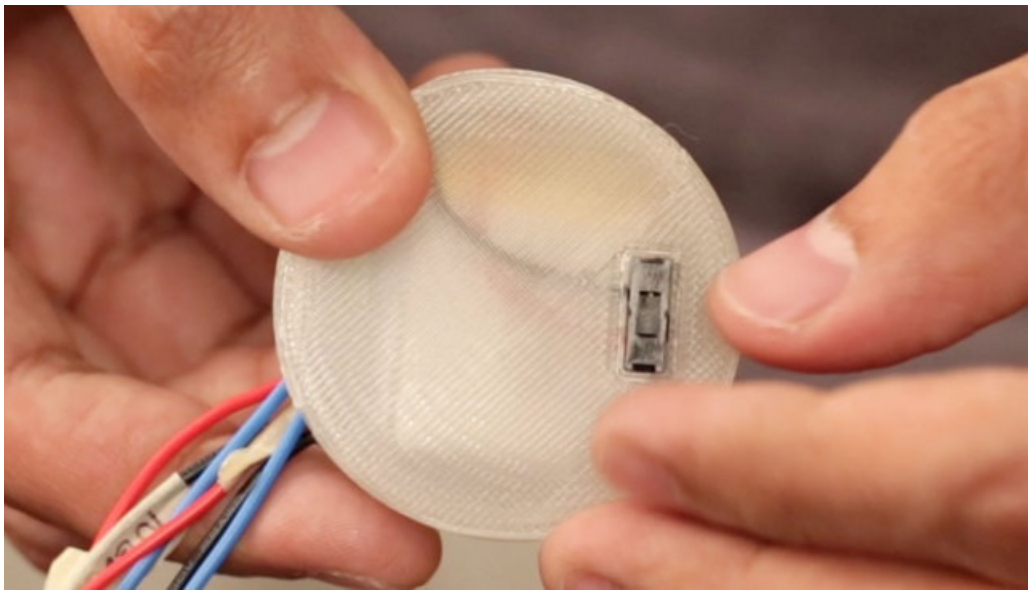
You can choose how deep you want your lights to be in the shells. For the drum kick, we positioned them away from the drum pedal so it's brighter when facing an audience. In the tom toms and the snare, the LEDs are positioned closer to the drummer, but you can mount them closer to the bottom for the audience. You can use paper tape to keep the strips mounted to the inner lining. We can do this after we solder wires to the NeoPixel strips.



Three short wires will need to be soldered to the break out pins. Once soldered, position the mic sensor inside the shell. The mic sensor will be mounted to the inside of the shell with paper tape. Use alligator clips to connect to the mic sensor wires to the Gemma enclosure that is outside of the drum through the air hole.

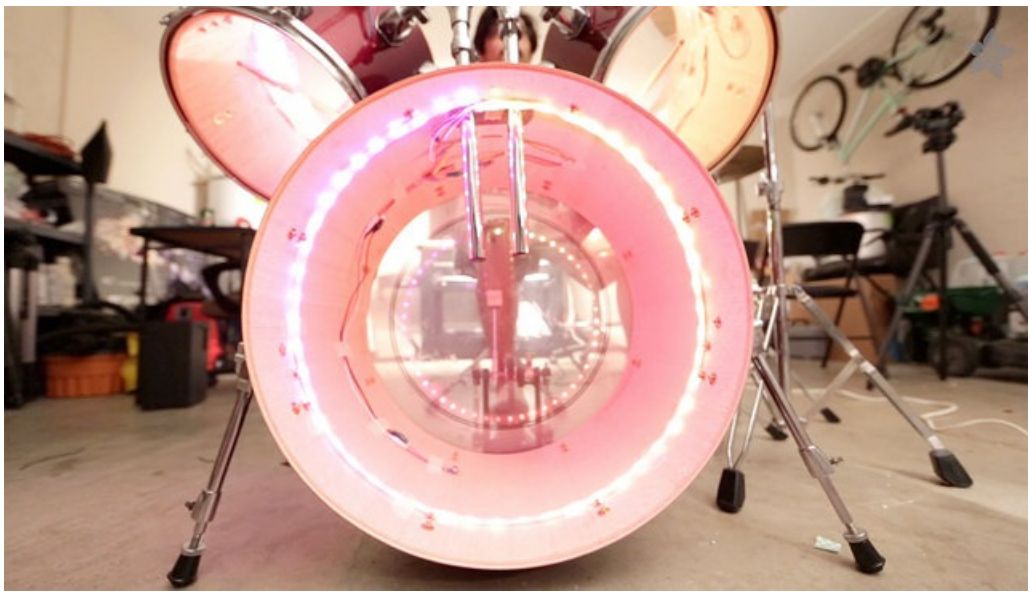


With the wires soldered to the gemma, you can fit each cable through the small slit on the side of the case. Carefully bend the wires down towards the opening of the case for a good fit. You can stick a small piece of doubled-sided foam tape to keep the bottom of the Gemma secured it in place.



The battery should be positioned inside of the case on top of the Gemma. The slide switch can be positioned through the cutout on the enclosure cover. You can use an adhesive to secure the slide switch.

Attach the hanger to the back of the case with a piece of double-sided foam tape. Position the hole on the hanger above the shell pin and press it down to mount it into place.



Once you have all of the parts mounted and secured to the shells, play and test out each piece. If you have issues with the LEDs, make sure your battery is fully charged and check your alligator clip connections to see if any wires are touching. The amount of sensitivity will vary depending on where you position the mic sensor.

We used pieces of cable tube organizers to protect the exposed wired. You can also use shrink tubing for a more permanent finish.

The lithium polymer batteries can be removed and recharged. They should last about an hours worth of play time. You can alternatively use a bigger battery pack for a longer sessions. The cases are non-intrsutive to the look of the drums and the wires and sensor inside the shells don't have an overall change to the sound. Since each drum is independent of each other, you can break down your set without having to worry about disconnecting any wires!

Made an awesome upgrade? Let us know!! Stop by our weekly show-n-tell on youtube (https://adafru.it/cYL), Saturdays at 9:30 PM EST and show off your upgrade to Lady Ada her self (and thousands of people LIVE) also WIN a FREE As Seen on Show-N-Tell Adafruit sticker! Post a comment on our google+ (https://adafru.it/tEA) page in our show related post and we'll invite you!