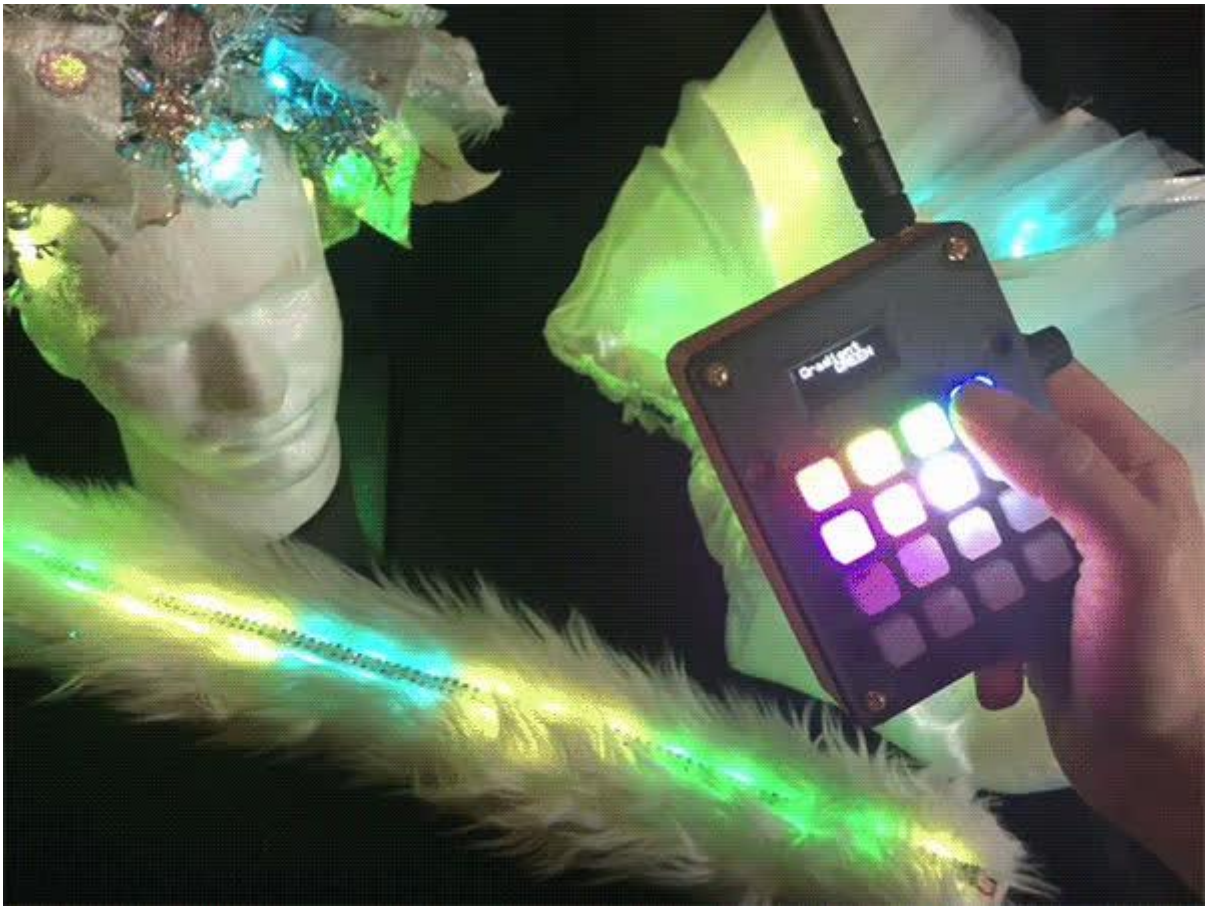
Wireless NeoPixel Controller

Created by Erin St Blaine



https://learn.adafruit.com/neotrellis-neopixel-controller

Last updated on 2023-08-29 03:57:28 PM EDT

Table of Contents

Overview

Build a control box with a NeoTrellis and Feather M0 Radio and use it to wirelessly control multiple strands of NeoPixels. Use the FastLED library to add beautiful NeoPixel color effects. With no line-of-sight requirement (like with infrared remotes) and no pairing requirement (like with Bluetooth or WiFi), the LoRa radio control works over long distances and in crowded environments. It's hard to pair to a Bluetooth device when there are 200 phones in the vicinity cluttering up the bandwidth! This controller box will solve that problem.

This project was created for onstage or performance environments, where a portable and easy-to-setup pixel controller can come in very handy. It can also be used to control your holiday lights display, to coordinate costumes at a festival, or for a zillion other applications where you want ultimate wireless NeoPixel control with no interference.

Parts and Materials

The essential parts for your controller are:

- Packet radio Feather microcontroller and antenna as the brains of the transmitter. Your choice of 900MHz for US, 433MHz for ITU Europe, in either M0 or 32u4 processor variant. The RFM69HCW radios are the easiest to set up and a bit less expensive, while the LoRa boards are slightly more involved to set up, but longer range

- NeoTrellis Button Pad & Elastomer buttons
- OLED display FeatherWing
- Rotary encoder for menu selection

For the receivers you'll need:

- Another Feather M0 microcontroller that exactly matches the one inside the transmitter
- Neopixels () -- any format will work
- A battery to power the NeoPixels

4 x Feather M0 Packet Radio Adafruit Feather M0 RFM69HCW Packet Radio - 868 or 915 MHz	https://www.adafruit.com/product/3176
1 x NeoTrellis NeoTrellis RGB Driver PCB for 4x4 Keypad	https://www.adafruit.com/product/3954
1 x Silicone Keypad Silicone Elastomer 4x4 Button Keypad - for 3mm LEDs	https://www.adafruit.com/product/1611
1 x Antenna 900Mhz Antenna Kit - For LoPy, LoRa, etc	https://www.adafruit.com/product/3340
1 x Antenna Connector uFL SMT Antenna Connector	https://www.adafruit.com/product/1661
5 x Battery One for the Control box and one for each NeoPixel receiver	https://www.adafruit.com/product/328
1 x FeatherWing OLED Adafruit FeatherWing OLED - 128x32 OLED Add-on For Feather	https://www.adafruit.com/product/2900
1 x Short Female Headers Short Headers Kit for Feather - 12-pin + 16-pin Female Headers	https://www.adafruit.com/product/2940
1 x Short Male Headers Short Feather Male Headers - 12-pin and 16-pin Male Header Set	https://www.adafruit.com/product/2940

Rotary Encoder + Extras
Rotary Encoder + Extras

1 x Rotary Encoder — https://www.adafruit.com/product/377
Rotary Encoder + Extras

---

1 x Slide Switch — https://www.adafruit.com/product/805
Breadboard-friendly SPDT Slide Switch

---

1 x Nylon Screws — https://www.adafruit.com/product/3299
Black Nylon Screw and Stand-off Set – M2.5 Thread

---

1 x Neopixel Strand — https://www.adafruit.com/product/3630
Adafruit NeoPixel LED Dots Strand - 20 LEDs at 2" Pitch

---

1 x NeoPixels 60/m — https://www.adafruit.com/product/1138
Adafruit NeoPixel Digital RGB LED Strip - White 60 LED - WHITE

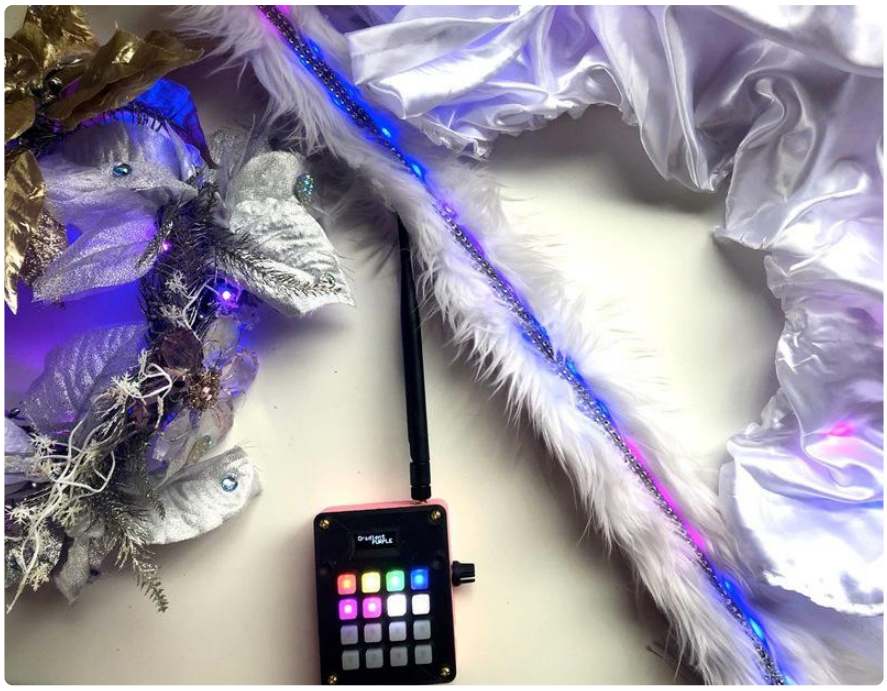
In addition to the parts listed above, you'll also need six screws to fasten the enclosure:

- 4ea. #6-32 1-1/4" hex drive rounded head screws ()
- 2ea. #6-32 1-1/8" hex drive rounded head screws ()

And, if you're 3D printing the enclosure, you'll want some PLA filament for that, such as:

- 1.75mm PLA white ()
- 1.75mm PLA black ()
- 1.75mm PLA red ()
- 1.75mm PLA/PHA blue ()

I found that black PLA works the best for the face of the controller. White PLA lets a lot of light bleed through and the NeoTrellis looks better against a dark background.



Code with Arduino

There are two different code files here. One will go onto the Feather M0 Lora inside the controller box and broadcast signals. The other goes onto a Feather M0 LoRa connected to your wearable project, and will receive the signals and run the FastLED code.

Before You Start

You'll need at least two of the same (identical!) packet radio Feather boards (e.g., RFM69HCW to RFM69HCW or LoRa RFM95 to LoRa RFM95, no mixing and matching) set to the same frequency in order to send and receive messages -- one board on its own is very lonely, and semi-useless :(

If this is your first foray into the world of arduino-based microcontrollers, you'll need to install some software first. Head over to the Feather M0 LoRa guide () for a detailed explanation of what you need to do.

Once you've gone through the guide and have the "blink" sketch working, come back here and continue.

Install Libraries

In Arduino, choose Sketch > Include Library > Manage Libraries. If you are new to using libraries, see this guide for details ().

You can search for the various libraries in this window. Find each one and click "Install". When you have them all installed, restart Arduino.

- Adafruit_NeoTrellis
- Adafruit_SSD1306
- Adafruit_GFX
- Adafruit_BusIO
- FastLED
- Encoder Quad Linear (Encoder Library by Paul Stroffgren)

Install the RadioHead library as shown here () and then upload the RadioHead -> feather -> RadioHead69_RawDemo_TX example on one board and RadioHead -> feather -> RadioHead69_RawDemo_RX on the other. Open your serial port on one board and make sure that they're talking to each other properly.

Controller Code

Download the Ada-remoteFXTrigger_NeoTrellis_FastLED_TX code below and upload it to your controller box.

```
// SPDX-FileCopyrightText: 2019 Erin St. Blaine for Adafruit Industries
// SPDX-FileCopyrightText: 2019 John Edgar Park for Adafruit Industries
//
// SPDX-License-Identifier: MIT
//
//Ada_remoteFXTrigger_NeoTrellis_TX
//Remote Effects Trigger Box Transmitter
//by John Park
// & Erin St. Blaine
//for Adafruit Industries

// General purpose button box
// for triggering remote effects
// using packet radio Feather boards
//
//
//MIT License

#include <SPI.h>
#include <RH_RF69.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <Adafruit_NeoTrellis.h>

#include <Encoder.h>
int m = 0; //variable to increment through menu list
int x = 17; //variable for referencing buttons -- setting X to a non-existent button
/********* Encoder Setup ***************/
#define PIN_ENCODER_SWITCH 11
Encoder knob(10, 12);
uint8_t activeRow = 0;
long pos = -999;
long newpos;
```

```
int prevButtonState = HIGH;
bool needsRefresh = true;
bool advanced = false;
unsigned long startTime;


/********* NeoTrellis Setup ***************/
Adafruit_NeoTrellis trellis;

#define MOMENTARY 0
#define LATCHING 1
#define MODE LATCHING //all Trellis buttons in latching mode
#define NUMTRELLIS 1
#define numKeys (NUMTRELLIS * 16)
#define INTPIN A2
int NUM_SCREENS = 3;  // change this number to add more screens


//define a callback for key presses
TrellisCallback blink(keyEvent evt){
  // Check is the pad pressed?
  if (evt.bit.EDGE == SEESAW_KEYPAD_EDGE_RISING) {
    //trellis.pixels.setPixelColor(evt.bit.NUM, 0xFFFFFF); //on rising
  } else if (evt.bit.EDGE == SEESAW_KEYPAD_EDGE_FALLING) {
  // or is the pad released?
  //trellis.pixels.setPixelColor(m, Wheel(map(m*2, 0, trellis.pixels.numPixels(),
0, 255))); //off falling
   x = evt.bit.NUM;
  }

  // Turn on/off the neopixels!
  trellis.pixels.show();

  return 0;
}



/************ OLED Setup ***************/
Adafruit_SSD1306 oled = Adafruit_SSD1306();
#if defined(ESP8266)
  #define BUTTON_A 0
  #define BUTTON_B 16
  #define BUTTON_C 2
  #define LED      0
#elif defined(ESP32)
  #define BUTTON_A 15
  #define BUTTON_B 32
  #define BUTTON_C 14
  #define LED      13
#elif defined(ARDUINO_STM32F2_FEATHER)
  #define BUTTON_A PA15
  #define BUTTON_B PC7
  #define BUTTON_C PC5
  #define LED PB5
#elif defined(TEENSYDUINO)
  #define BUTTON_A 4
  #define BUTTON_B 3
  #define BUTTON_C 8
  #define LED 13
#elif defined(ARDUINO_FEATHER52)
  #define BUTTON_A 31
  #define BUTTON_B 30
  #define BUTTON_C 27
  #define LED 17
#else // 32u4, M0, and 328p
  #define BUTTON_A 9
  #define BUTTON_B 6
  #define BUTTON_C 5
```

```
  #define LED      13
#endif


/************ Radio Setup ***************/
// Can be changed to 434.0 or other frequency, must match RX's freq!
#define RF69_FREQ 915.0

#if defined (__AVR_ATmega32U4__) // Feather 32u4 w/Radio
  #define RFM69_CS      8
  #define RFM69_INT     7
  #define RFM69_RST     4
#endif

#if defined(ARDUINO_SAMD_FEATHER_M0) // Feather M0 w/Radio
  #define RFM69_CS      8
  #define RFM69_INT     3
  #define RFM69_RST     4
#endif

#if defined (__AVR_ATmega328P__)  // Feather 328P w/wing
  #define RFM69_INT     3  //
  #define RFM69_CS      4  //
  #define RFM69_RST     2  // "A"
#endif

#if defined(ESP32)    // ESP32 feather w/wing
  #define RFM69_RST     13   // same as LED
  #define RFM69_CS      33   // "B"
  #define RFM69_INT     27   // "A"
#endif


// Singleton instance of the radio driver
RH_RF69 rf69(RFM69_CS, RFM69_INT);

int lastButton=17; //last button pressed for Trellis logic

int menuList[8]={1,2,3,4,5,6,7,8}; //for rotary encoder choices
//int m = 0; //variable to increment through menu list
int lastTB[8] = {16, 16, 16, 16, 16, 16, 16, 16}; //array to store per-menu Trellis
button


/*******************SETUP************/
void setup() {
  delay(500);
  Serial.begin(115200);
  //while (!Serial) { delay(1); } // wait until serial console is open,
  //remove if not tethered to computer

pinMode(PIN_ENCODER_SWITCH, INPUT_PULLUP);//set encoder push switch pin to input
pullup
pinMode(INTPIN, INPUT);

 // digitalPinToInterrupt(10); //on M0, Encoder library doesn't auto set these as
interrupts
 // digitalPinToInterrupt(12);

  // Initialize OLED display
  oled.begin(SSD1306_SWITCHCAPVCC, 0x3C);  // initialize with the I2C addr 0x3C
(for the 128x32)
  oled.setTextWrap(false);
  oled.display();
  delay(500);
  oled.clearDisplay();
  oled.display();
  oled.setTextSize(2);
  oled.setTextColor(WHITE);
```

```
  pinMode(BUTTON_A, INPUT_PULLUP);
  pinMode(BUTTON_B, INPUT_PULLUP);
  pinMode(BUTTON_C, INPUT_PULLUP);
  pinMode(LED, OUTPUT);
  pinMode(RFM69_RST, OUTPUT);
  digitalWrite(RFM69_RST, LOW);

  // manual reset
  digitalWrite(RFM69_RST, HIGH);
  delay(10);
  digitalWrite(RFM69_RST, LOW);
  delay(10);

  if (!rf69.init()) {
    Serial.println("RFM69 radio init failed");
    while (1);
  }
  Serial.println("RFM69 radio init OK!");

  // Defaults after init are 434.0MHz, modulation GFSK_Rb250Fd250, +13dbM (for low
power module)
  // No encryption
  if (!rf69.setFrequency(RF69_FREQ)) {
    Serial.println("setFrequency failed");
  }

  // If you are using a high power RF69 eg RFM69HW, you *must* set a Tx power with
the
  // ishighpowermodule flag set like this:
  rf69.setTxPower(14, true);

  // The encryption key has to be the same as the one in the server
  uint8_t key[] = { 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08,
                    0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08};
  rf69.setEncryptionKey(key);

  pinMode(LED, OUTPUT);

  Serial.print("RFM69 radio @");  Serial.print((int)RF69_FREQ);  Serial.println("
MHz");

  oled.setCursor(0,0);
  oled.println("RFM69 @ ");
  oled.print((int)RF69_FREQ);
  oled.println(" MHz");
  oled.display();
  delay(1200); //pause to let freq message be read by a human

  oled.clearDisplay();
  oled.setCursor(0,0);
  oled.println("REMOTE FX");
  oled.setCursor(0,16);
  oled.println("TRIGGER");
  oled.display();

   if (!trellis.begin()) {
    Serial.println("Could not start trellis, check wiring?");
    while(1);
  } else {
    Serial.println("NeoPixel Trellis started");
  }

  //activate all NeoTrellis keys and set callbacks
  for(int i=0; i<NEO_TRELLIS_NUM_KEYS; i++){
    trellis.activateKey(i, SEESAW_KEYPAD_EDGE_RISING);
    trellis.activateKey(i, SEESAW_KEYPAD_EDGE_FALLING);
    trellis.registerCallback(i, blink);
  }
```

```
  //do a little animation to show we're on
  for (uint16_t i=0; i<trellis.pixels.numPixels(); i++) {
    trellis.pixels.setPixelColor(i, Wheel(map(i, 0, trellis.pixels.numPixels(), 0,
255)));
    trellis.pixels.show();
    delay(50);
  }
  for (uint16_t i=0; i<trellis.pixels.numPixels(); i++) {
    trellis.pixels.setPixelColor(i, 0x000000);
    trellis.pixels.show();
    delay(50);
  }

}

////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////

void loop() {


/*************Rotary Encoder Menu***********/

    //check the encoder knob, set the current position as origin
    long newpos = knob.read() / 4;//divide for encoder detents

    /* // for debugging
     Serial.print("pos=");
     Serial.print(pos);
     Serial.print(", newpos=");
     Serial.println(newpos);
    */

    if(newpos != pos){
      int diff = newpos - pos;//check the different between old and new position
      if(diff>=1){
        m++;
        m = (m+NUM_SCREENS) % NUM_SCREENS;//modulo to roll over the m variable
through the list size
       }

      if(diff==-1){ //rotating backwards
         m--;
         m = (m+NUM_SCREENS) % NUM_SCREENS;
       }
      /* //uncomment for debugging or general curiosity
      Serial.print("Diff = ");
      Serial.print(diff);
      Serial.print("  pos= ");
      Serial.print(pos);
      Serial.print(", newpos=");
      Serial.println(newpos);
      Serial.println(menuList[m]);
      */

      pos = newpos;

      // Serial.print("m is: ");
      //Serial.println(m);
  //write to the display
      oled.setCursor(0,3);
      oled.clearDisplay();

      int p; //for drawing bullet point menu location pixels
      int q;

      if (m==0){
        for(p=4;p<8;p++){
          for(q=0;q<4;q++){
```

```
            oled.drawPixel(q,p,WHITE);
          }
        }
        oled.print(" Solids");
        clearPixels();
        trellis.pixels.setPixelColor(0, 50,0,0);
        trellis.pixels.setPixelColor(1, 50,50,0);
        trellis.pixels.setPixelColor(2, 0,50,0);
        trellis.pixels.setPixelColor(3, 0,0,50);
        trellis.pixels.setPixelColor(4, 20,0,50);
        trellis.pixels.setPixelColor(5, 50,0,20);
        trellis.pixels.setPixelColor(6, 100,100,100);
        trellis.pixels.setPixelColor(7, 10,10,10);
        trellis.pixels.show();
      }
      if (m==1){
        for(p=8;p<12;p++){
          for(q=0;q<4;q++){
            oled.drawPixel(q,p,WHITE);
          }
        }
        oled.print(" Gradients");
        clearPixels();
        trellis.pixels.setPixelColor(0, 50,0,0);
        trellis.pixels.setPixelColor(1, 50,50,0);
        trellis.pixels.setPixelColor(2, 0,50,0);
        trellis.pixels.setPixelColor(3, 0,0,50);
        trellis.pixels.setPixelColor(4, 20,0,50);
        trellis.pixels.setPixelColor(5, 50,0,20);
        trellis.pixels.setPixelColor(6, 100,100,100);
        trellis.pixels.setPixelColor(7, 10,10,10);
        trellis.pixels.show();
      }
      if (m==2){
        for(p=12;p<16;p++){
          for(q=0;q<4;q++){
            oled.drawPixel(q,p,WHITE);
          }
        }
        oled.print(" Rainbows");
        clearPixels();
        trellis.pixels.setPixelColor(0, 50,0,0);
        trellis.pixels.setPixelColor(1, 50,50,0);
        trellis.pixels.setPixelColor(2, 0,50,0);
        trellis.pixels.setPixelColor(7, 10,10,10);
        trellis.pixels.show();
      }



      oled.display();
    }

    char radiopacket[20];


/**************Solids**************/

        if(m==0){ //next menu item

            if (x==0){ //button 1 sends button A command
              radiopacket[0] = 'A';
              oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Solid");
              oled.setCursor(50,16);
              oled.print("RED");
              oled.display();
            }
```

```
            if (x==1){ //button 2 sends button B command
              radiopacket[0] = 'B';
             oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Solid");
              oled.setCursor(50,16);
              oled.print("GOLD");
              oled.display();
            }
            if (x==2){ //button 3 sends button C command
              radiopacket[0] = 'C';
             oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Solid");
              oled.setCursor(50,16);
              oled.print("GREEN");
              oled.display();
            }

            if (x==3){ //button 4 sends button D command
              radiopacket[0] = 'D';
             oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Solid");
              oled.setCursor(50,16);
              oled.print("BLUE");
              oled.display();
            }

            if (x==4){ //button 5 sends button E command
              radiopacket[0] = 'E';
              oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Solid");
              oled.setCursor(50,16);
              oled.print("PURPLE");
              oled.display();
            }
            if (x==5){ //button 6 sends button F command
              radiopacket[0] = 'F';
             oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Solid");
              oled.setCursor(50,16);
              oled.print("PINK");
              oled.display();
            }
            if (x==6){ //button 7 sends button G command
              radiopacket[0] = 'G';
             oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Solid");
              oled.setCursor(50,16);
              oled.print("WHITE");
              oled.display();
            }

            if (x==7){ //button 8 sends button H command
              radiopacket[0] = 'H';
             oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Solid");
              oled.setCursor(50,16);
              oled.print("OFF");
              oled.display();
            }


trellis.pixels.show(); // tell the trellis to set the LEDs we requested
```

```
}

/**************Gradients**************/

        if(m==1){ //next menu item

            if (x==0){ //button 1 sends button A command
              radiopacket[0] = 'I';
              oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Gradient");
              oled.setCursor(50,16);
              oled.print("RED");
              oled.display();
            }
            if (x==1){ //button 2 sends button B command
              radiopacket[0] = 'J';
             oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Gradient");
              oled.setCursor(50,16);
              oled.print("GOLD");
              oled.display();
            }
            if (x==2){ //button 3 sends button C command
              radiopacket[0] = 'K';
             oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Gradient");
              oled.setCursor(50,16);
              oled.print("GREEN");
              oled.display();
            }

            if (x==3){ //button 4 sends button D command
              radiopacket[0] = 'L';
             oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Gradient");
              oled.setCursor(50,16);
              oled.print("BLUE");
              oled.display();
            }

            if (x==4){ //button 5 sends button E command
              radiopacket[0] = 'M';
              oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Gradient");
              oled.setCursor(50,16);
              oled.print("PURPLE");
              oled.display();
            }
            if (x==5){ //button 6 sends button F command
              radiopacket[0] = 'N';
             oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Gradient");
              oled.setCursor(50,16);
              oled.print("PINK");
              oled.display();
            }
            if (x==6){ //button 7 sends button G command
              radiopacket[0] = 'O';
             oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Gradient");
              oled.setCursor(50,16);
              oled.print("WHITE");
```

```
              oled.display();
             }

             if (x==7){ //button 8 sends button H command
               radiopacket[0] = 'H';
              oled.clearDisplay();
               oled.setCursor(0,0);
               oled.print("Gradient");
               oled.setCursor(50,16);
               oled.print("OFF");
               oled.display();
             }
 trellis.pixels.show(); // tell the trellis to set the LEDs we requested
}


/**************Rainbows**************/

         if(m==2){ //next menu item

             if (x==0){ //button 1 sends button A command
               radiopacket[0] = 'P';
               oled.clearDisplay();
               oled.setCursor(0,0);
               oled.print("Rainbow");
               oled.setCursor(50,16);
               oled.print("FADE");
               oled.display();
             }
             if (x==1){ //button 2 sends button B command
               radiopacket[0] = 'Q';
              oled.clearDisplay();
               oled.setCursor(0,0);
               oled.print("Rainbow");
               oled.setCursor(50,16);
               oled.print("SOFT");
               oled.display();
             }
             if (x==2){ //button 3 sends button C command
               radiopacket[0] = 'R';
              oled.clearDisplay();
               oled.setCursor(0,0);
               oled.print("Rainbow");
               oled.setCursor(50,16);
               oled.print("TWINKLE");
               oled.display();
             }

             if (x==7){ //button 8 sends button H command
               radiopacket[0] = 'H';
              oled.clearDisplay();
               oled.setCursor(0,0);
               oled.print("Rainbow");
               oled.setCursor(50,16);
               oled.print("OFF");
               oled.display();
             }
 trellis.pixels.show(); // tell the trellis to set the LEDs we requested
}



Serial.print("Sending ");
           Serial.println(radiopacket[0]);

           rf69.send((uint8_t *)radiopacket, strlen(radiopacket));
           rf69.waitPacketSent();
           //reset packet so unassigned buttons don't send last command
           radiopacket[0]='Z'; //also being used to turn off NeoPixels
```

```
          //from any unused button

          if (rf69.waitAvailableTimeout(1000)) {
            // Should be a message for us now
            uint8_t buf[RH_RF69_MAX_MESSAGE_LEN];
            uint8_t len = sizeof(buf);

            if (! rf69.recv(buf, &len)) {
              Serial.println("Receive failed");
              return;
            }
            digitalWrite(LED, HIGH);
            rf69.printBuffer("Received: ", buf, len);
            buf[len] = 0;

            //Serial.print("TX Got: ");
            //Serial.println((char*)buf);
            Serial.print("RSSI: ");
            Serial.println(rf69.lastRssi(), DEC);

            //delay(1000);//chill for a moment before returning the message to RX
unit

            /*************Reply message from RX unit***********/
            //oled.clearDisplay();
            //oled.print((char*)buf[0]);
            //oled.print("RSSI: "); oled.print(rf69.lastRssi());
            //oled.display();


            digitalWrite(LED, LOW);
          }

          //lastButton=i;//set for next pass through to turn this one off


       trellis.read();
     delay(100); //the NeoTrellis has a resolution of around 60hz

 }


/****************************************/

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

void clearPixels() {
 for (uint16_t i=0; i<trellis.pixels.numPixels(); i++) {
    trellis.pixels.setPixelColor(i, 0x000000);
    trellis.pixels.show();
    delay(2);
  }
  }
```

If you are not familiar with using Arduino to upload code, see this guide ().

Customizing

This code has three different screens you can scroll through: Solids, Gradients, and Rainbows. Each screen lights up a few buttons on the NeoTrellis. The unlighted buttons are currently unassigned.

To add more screens, look for this line in the code under NeoTrellis Setup and change the number to reflect the total number of screens you want.

```
int NUM_SCREENS = 3;  // change this number to add more screens
```

Then scroll down and find the "`Rotary Encoder Menu`" section. Here is where you can set the pixel colors on the NeoTrellis for each screen, and give each screen a title. You can also add or remove screens here (just remember to change `NUM_SCREENS` to match your total).

```
    if (m==0){
        for(p=4;p&lt;8;p++){
          for(q=0;q&lt;4;q++){
            oled.drawPixel(q,p,WHITE);
          }
        }
        oled.print(" Solids");
        clearPixels();
        trellis.pixels.setPixelColor(0, 50,0,0);
        trellis.pixels.setPixelColor(1, 50,50,0);
        trellis.pixels.setPixelColor(2, 0,50,0);
        trellis.pixels.setPixelColor(3, 0,0,50);
        trellis.pixels.setPixelColor(4, 20,0,50);
        trellis.pixels.setPixelColor(5, 50,0,20);
        trellis.pixels.setPixelColor(6, 100,100,100);
        trellis.pixels.setPixelColor(7, 10,10,10);
        trellis.pixels.show();
      }
```

Scrolling some more, find the section titled "`Solids`". This is where you will tell the controller to send a specific command, like "`A`" or "`B`" when each button is pressed. The first button on the controller is 0, and the last one is 15. You can also customize the OLED display messages here.

```
/**************Solids**************/

        if(m==0){ //next menu item

            if (x==0){ //button 1 sends button A command
              radiopacket[0] = 'A';
              oled.clearDisplay();
              oled.setCursor(0,0);
              oled.print("Solid");
              oled.setCursor(50,16);
              oled.print("RED");
```

```
            oled.display();
          }
```

That's all this box does: shows pretty colors and sends a letter when a button is pressed. All the rest is done on the receiver end.

Receiver Code

Download the Ada-remoteFXTrigger_NeoTrellis_FastLED_RX code below and upload it to your controller box.

```
// SPDX-FileCopyrightText: 2019 Erin St. Blaine for Adafruit Industries
// SPDX-FileCopyrightText: 2019 John Edgar Park for Adafruit Industries
//
// SPDX-License-Identifier: MIT
//
//Ada_remoteFXTrigger_RX_NeoPixel
//Remote Effects Trigger Box Receiver
//by John Park & Erin St Blaine
//for Adafruit Industries
//
// Button box receiver with NeoPixels using FastLED
//
//
//MIT License

#include <FastLED.h>

#define LED_PIN  12
#define NUM_LEDS    20
#define LED_TYPE    WS2812B
#define COLOR_ORDER GRB

CRGBArray<NUM_LEDS> leds;

#include <SPI.h>
#include <RH_RF69.h>
#include <Wire.h>

#define LED 13

/********** NeoPixel Setup *************/

#define UPDATES_PER_SECOND 100
CRGBPalette16 currentPalette( CRGB::Black);
CRGBPalette16 targetPalette( PartyColors_p );
TBlendType    currentBlending;

int SPEEDO = 25;
int STEPS = 20;
int HUE = 200;    // starting color
int SATURATION = 255;
int BRIGHTNESS = 200;
int glitter = 0;

/************ Radio Setup ***************/

// Change to 434.0 or other frequency, must match RX's freq!
#define RF69_FREQ 915.0

#if defined (__AVR_ATmega32U4__) // Feather 32u4 w/Radio
  #define RFM69_CS      8
```

```
  #define RFM69_INT     7
  #define RFM69_RST     4
#endif

#if defined(ARDUINO_SAMD_FEATHER_M0) // Feather M0 w/Radio
  #define RFM69_CS      8
  #define RFM69_INT     3
  #define RFM69_RST     4
#endif

#if defined (__AVR_ATmega328P__)  // Feather 328P w/wing
  #define RFM69_INT     3  //
  #define RFM69_CS      4  //
  #define RFM69_RST     2  // "A"
#endif

#if defined(ESP32)    // ESP32 feather w/wing
  #define RFM69_RST     13   // same as LED
  #define RFM69_CS      33   // "B"
  #define RFM69_INT     27   // "A"
#endif


// Singleton instance of the radio driver
RH_RF69 rf69(RFM69_CS, RFM69_INT);


bool oldState = HIGH;


void setup() {
  delay( 3000 ); // power-up safety delay
  FastLED.addLeds<LED_TYPE, LED_PIN, COLOR_ORDER>(leds, NUM_LEDS).setCorrection(
TypicalLEDStrip );
  FastLED.setBrightness(  BRIGHTNESS );
  pinMode(LED, OUTPUT);

  pinMode(RFM69_RST, OUTPUT);
  digitalWrite(RFM69_RST, LOW);

  Serial.println("Feather RFM69 RX/TX Test!");

  // manual reset
  digitalWrite(RFM69_RST, HIGH);
  delay(10);
  digitalWrite(RFM69_RST, LOW);
  delay(10);

  if (!rf69.init()) {
    Serial.println("RFM69 radio init failed");
    while (1);
  }
  Serial.println("RFM69 radio init OK!");

  // Defaults after init are 434.0MHz, modulation GFSK_Rb250Fd250, +13dbM (for low
power module)
  // No encryption
  if (!rf69.setFrequency(RF69_FREQ)) {
    Serial.println("setFrequency failed");
  }

  // If you are using a high power RF69 eg RFM69HW, you *must* set a Tx power with
the
  // ishighpowermodule flag set like this:
  rf69.setTxPower(14, true);

  // The encryption key has to be the same as the one in the server
  uint8_t key[] = { 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08,
                    0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08};
```

```
  rf69.setEncryptionKey(key);

  pinMode(LED, OUTPUT);

  Serial.print("RFM69 radio @");  Serial.print((int)RF69_FREQ);  Serial.println("
MHz");

  delay(500);
  Gradient();  //So the lights come un upon startup, even if the trigger box is off
}

  void loop(){



  if (rf69.waitAvailableTimeout(1000)) {
    // Should be a message for us now
    uint8_t buf[RH_RF69_MAX_MESSAGE_LEN];
    uint8_t len = sizeof(buf);

    if (! rf69.recv(buf, &len)) {
      Serial.println("Receive failed");
      return;
    }

    //digitalWrite(LED, HIGH);

    //rf69.printBuffer("Received: ", buf, len);
    //buf[len] = 0;

    //Serial.print("Got: "); Serial.println((char*)buf);
    //Serial.print("RSSI: "); Serial.println(rf69.lastRssi(), DEC);


    char radiopacket[20] = "Button #";//prep reply message to send



    if (buf[0]=='A'){ //the letter sent from the button
      ledMode(0);
        radiopacket[8] = 'A';
    }
     else if (buf[0]=='B'){ //the letter sent from the button
      ledMode(1);
      radiopacket[8] = 'B';
    }

     else if (buf[0]=='C'){ //the letter sent from the button
      ledMode(2);
       radiopacket[8] = 'C';
    }

     else if (buf[0]=='D'){ //the letter sent from the button
      ledMode(3);
      radiopacket[8] = 'D';
    }
         else if (buf[0]=='E'){ //the letter sent from the button
      ledMode(4);
      radiopacket[8] = 'E';
    }

     else if (buf[0]=='F'){ //the letter sent from the button
      ledMode(5);
       radiopacket[8] = 'F';
    }

     else if (buf[0]=='G'){ //the letter sent from the button
      ledMode(6);
      radiopacket[8] = 'G';
```

```
  }
   else if (buf[0]=='H'){ //the letter sent from the button
    ledMode(7);
    radiopacket[8] = 'H';
  }

   else if (buf[0]=='I'){ //the letter sent from the button
    ledMode(8);
     radiopacket[8] = 'I';
  }

   else if (buf[0]=='J'){ //the letter sent from the button
    ledMode(9);
    radiopacket[8] = 'J';
  }
   else if (buf[0]=='K'){ //the letter sent from the button
    ledMode(10);
    radiopacket[8] = 'K';
  }

   else if (buf[0]=='L'){ //the letter sent from the button
    ledMode(11);
     radiopacket[8] = 'L';
  }

   else if (buf[0]=='M'){ //the letter sent from the button
    ledMode(12);
    radiopacket[8] = 'M';
  }
   else if (buf[0]=='N'){ //the letter sent from the button
    ledMode(13);
    radiopacket[8] = 'N';
  }
   else if (buf[0]=='O'){ //the letter sent from the button
    ledMode(14);
    radiopacket[8] = 'O';
  }
   else if (buf[0]=='P'){ //the letter sent from the button
    ledMode(15);
    radiopacket[8] = 'P';
  }
       else if (buf[0]=='Q'){ //the letter sent from the button
    ledMode(16);
    radiopacket[8] = 'Q';
  }
       else if (buf[0]=='R'){ //the letter sent from the button
    ledMode(17);
    radiopacket[8] = 'R';
  }
           else if (buf[0]=='S'){ //the letter sent from the button
    ledMode(18);
    radiopacket[8] = 'S';
  }
       else if (buf[0]=='T'){ //the letter sent from the button
    ledMode(19);
    radiopacket[8] = 'T';
  }
       else if (buf[0]=='Z'){ //the letter sent from the button
    ledMode(20);
    radiopacket[8] = 'Z';
  }


/*   radiopacket[9] = 0;

  Serial.print("Sending "); Serial.println(radiopacket);
  rf69.send((uint8_t *)radiopacket, strlen(radiopacket));
  rf69.waitPacketSent();  */
```

```
    digitalWrite(LED, LOW);
  }

}


void ledMode(int i) {
  switch(i){
    case 0: HUE=0; SATURATION=255; BRIGHTNESS=200; Solid();    // red
            break;
    case 1: HUE=40; SATURATION=255; BRIGHTNESS=200; Solid();    // gold
            break;
    case 2: HUE=100; SATURATION=255; BRIGHTNESS=200; Solid();    // green
            break;
    case 3: HUE=140; SATURATION=255; BRIGHTNESS=200; Solid();    // Blue
            break;
    case 4: HUE=180; SATURATION=255; BRIGHTNESS=200; Solid();    // purple
            break;
    case 5: HUE=220; SATURATION=255; BRIGHTNESS=200; Solid();    // pink
            break;
    case 6: HUE=0; SATURATION=0; BRIGHTNESS=200; Solid();    // white
            break;
    case 7: HUE=0; BRIGHTNESS=0; Solid();    // off
            break;
    case 8: HUE=0; SATURATION=255; BRIGHTNESS=200; Gradient();    // red
            break;
    case 9: HUE=40; SATURATION=255; BRIGHTNESS=200; Gradient();    // gold
            break;
    case 10: HUE=100; SATURATION=255; BRIGHTNESS=200; Gradient();    // green
            break;
    case 11: HUE=140; SATURATION=255; BRIGHTNESS=200; Gradient();    // blue
            break;
    case 12:HUE=180; SATURATION=255; BRIGHTNESS=200; Gradient();    // purple
            break;
    case 13:HUE=220; SATURATION=255; BRIGHTNESS=200; Gradient();    // pink
            break;
    case 14:HUE=160; SATURATION=50; BRIGHTNESS=200; Gradient();    // white
            break;
    case 15:SATURATION=255; BRIGHTNESS=200; Rainbow_Fade();    // rainbow fade
            break;
    case 16:STEPS=4; SATURATION=255; BRIGHTNESS=200; Rainbow();   //rainbow 2
            break;
    case 17:STEPS=20; BRIGHTNESS=200; SATURATION=255; Rainbow();    // rainbow 3
            break;
    case 20:BRIGHTNESS=200;
            break;

  }
}

// GRADIENT ------------------------------------------------------------
void Gradient()
{
  SetupGradientPalette();

  static uint8_t startIndex = 0;
  startIndex = startIndex + 1;  // motion speed
  FillLEDsFromPaletteColors( startIndex);
  FastLED.show();
  FastLED.delay(SPEEDO);
}

// SOLID ---------------------------------------------------
void Solid()
{
   fill_solid(leds, NUM_LEDS, CHSV(HUE, SATURATION, BRIGHTNESS));
   FastLED.show();
   delay(20);
```

```
}

// RAINBOW -------------------------------------------------
void Rainbow()
{
  FastLED.setBrightness(  BRIGHTNESS );
  currentPalette = RainbowColors_p;

  static uint8_t startIndex = 0;
  startIndex = startIndex + 1;

  FillLEDsFromPaletteColors( startIndex);

  FastLED.show();
  FastLED.delay(SPEEDO);
}
// RAINBOW FADE --------------------------------------------------
void Rainbow_Fade() {                         //-m2-FADE ALL LEDS THROUGH HSV
RAINBOW
    HUE++;
    if (HUE > 255) {HUE = 0;}
    for(int idex = 0 ; idex < NUM_LEDS; idex++ ) {
      leds[idex] = CHSV(HUE, SATURATION, BRIGHTNESS);
    }
    LEDS.show();
    delay(SPEEDO);
}


void SetupGradientPalette()
{
  CRGB light = CHSV( HUE + 25, SATURATION - 20, BRIGHTNESS);
  CRGB dark  = CHSV( HUE, SATURATION - 15, BRIGHTNESS);
  CRGB medium = CHSV ( HUE - 25, SATURATION, BRIGHTNESS);

  currentPalette = CRGBPalette16(
    light,  light,  light,  light,
    medium, medium, medium,  medium,
    dark,  dark,  dark,  dark,
    medium, medium, medium,  medium );
}

void FillLEDsFromPaletteColors( uint8_t colorIndex)
{
  uint8_t brightness = BRIGHTNESS;

  for( int i = 0; i < NUM_LEDS; i++) {
    leds[i] = ColorFromPalette( currentPalette, colorIndex, brightness,
currentBlending);
    colorIndex += STEPS;
  }
}
```

Customizing

This code allows you to tell the LEDs what to do when the command is received from the control box. This example code matches the TX code in a number of ways. There are 21 different letters set up that can be received and each letter triggers a different LED mode.

A Note about Animations

FastLED is wonderful at producing buttery smooth animations. However, this setup works much better with very little LED motion or solid colors. Every time the signal is sent from the controller box, the animation is interrupted for just a moment. If we didn't interrupt the animation then most of your button presses would be missed while the animation was busy running, making it seem the controller box was not working or that the commands weren't being received. So I've made simple animations and solids the focus of this tutorial. For example, the `Rainbow_Fade()` animation does fade through the rainbow but it does it very slowly due to being interrupted so much.

There are two sections of code to pay attention to. Look for this section:

```
if (buf[0]=='A'){ //the letter sent from the button
     ledMode(0);
       radiopacket[8] = 'A';
   }
    else if (buf[0]=='B'){ //the letter sent from the button
     ledMode(1);
     radiopacket[8] = 'B';
   }
```

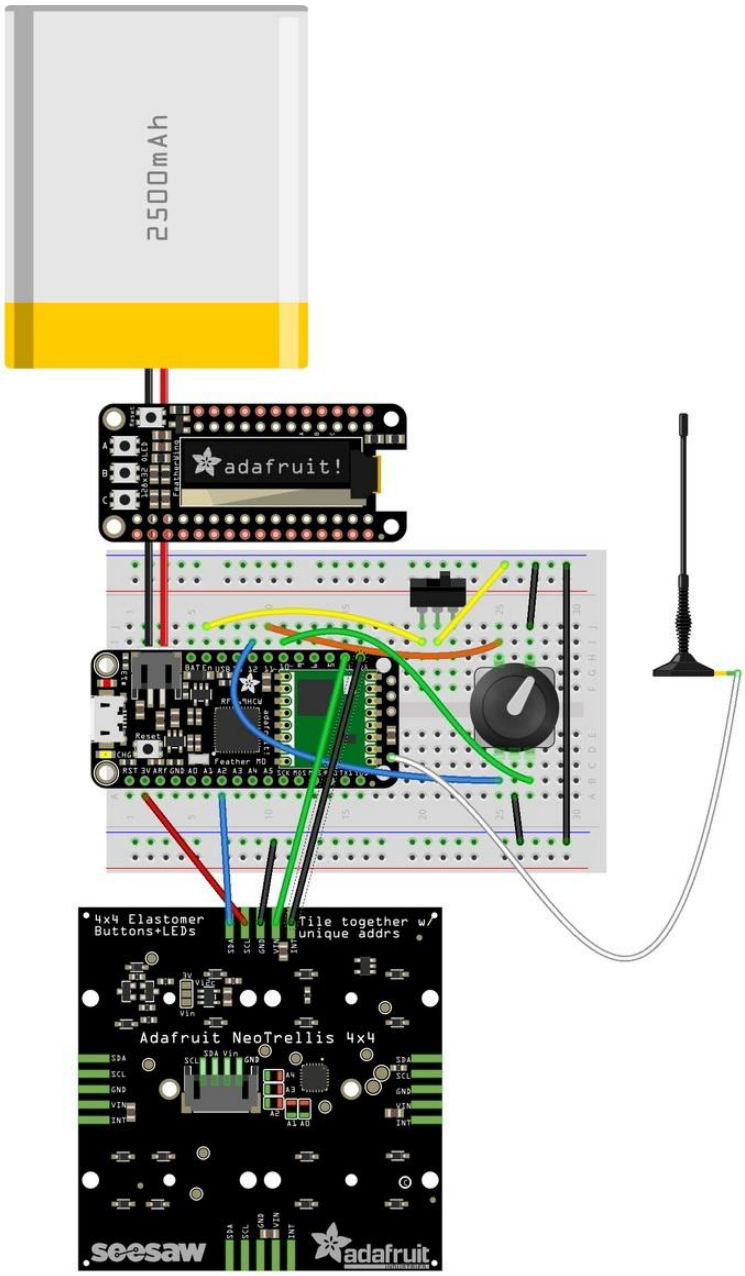
The radio packet being sent (`A` or `B` in this case) will cause the `ledMode` to change.

Further down, you'll find:

```
void ledMode(int i) {
  switch(i){
    case 0: HUE=0; SATURATION=255; BRIGHTNESS=200; Solid();    // red
            break;
    case 1: HUE=40; SATURATION=255; BRIGHTNESS=200; Solid();    // gold
            break;
```

These are the different modes being triggered by each radio packet. Here is where you can change what happens on the LED strip when button 0 is pressed, sending A, and triggering LED mode 0.

Controller Wiring & Build



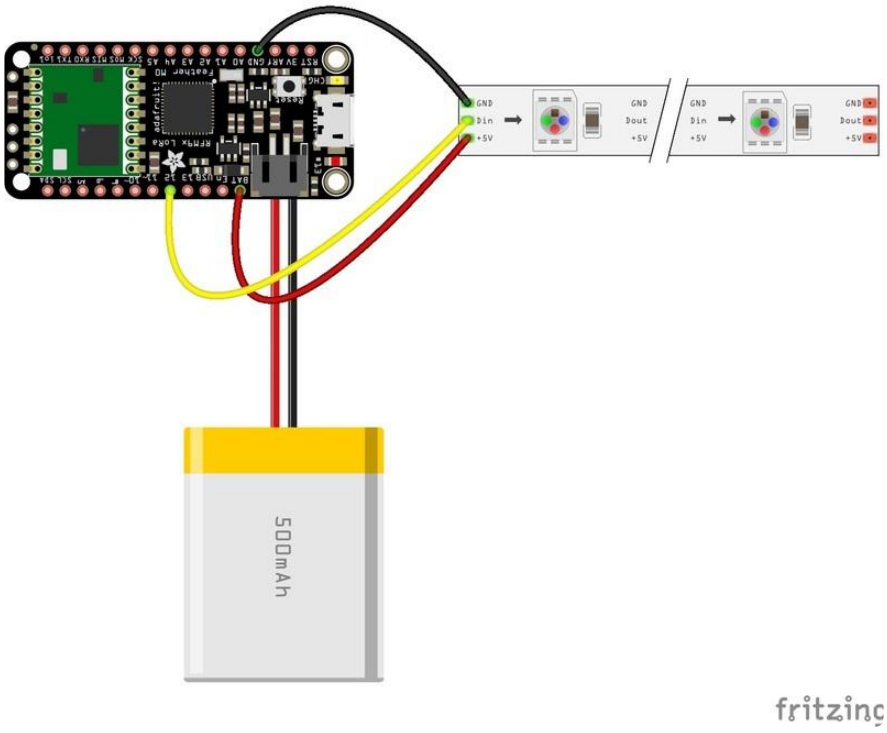
This build is based on John Park's Remote Effects Trigger box (). The only difference is that we'll be using a NeoTrellis () inside, instead of a Trellis. This will give us the option to have multi-colored buttons that react when we press them.

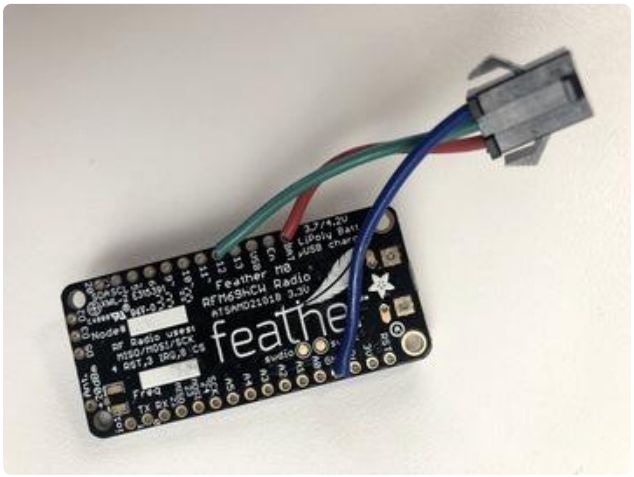
90% of the build is exactly the same. Head over to John Park's guide for more detailed explanations on building the box: Build the Transmitter (), 3d printing (), and Assembly ().

Look at the breadboard diagram to see the connections that will be made. John Park's guide goes into a lot of great detail. The connections from the NeoTrellis to the Feather are SDA, SCL, GND, VIN, and INT.

NeoPixel Wiring & Build

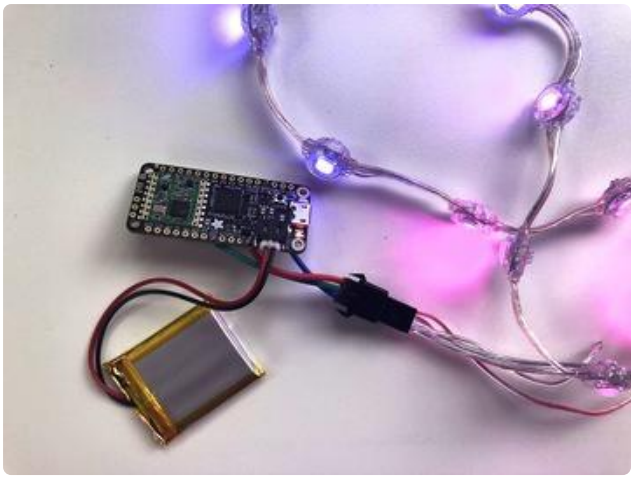
We'll attach a female 3-pin connector to our Feather M0 and then attach the male connector to the IN end of a NeoPixel strand or strip. For this project, I'm routing the power for the NeoPixels right through the Feather M0 board instead of powering directly from a battery. This works great for smaller projects or wearables with up to around 75-100 NeoPixels. If you have significantly more pixels in your project, you'll want to wire the power directly to the NeoPixels as shown in the NeoPixel guide ().



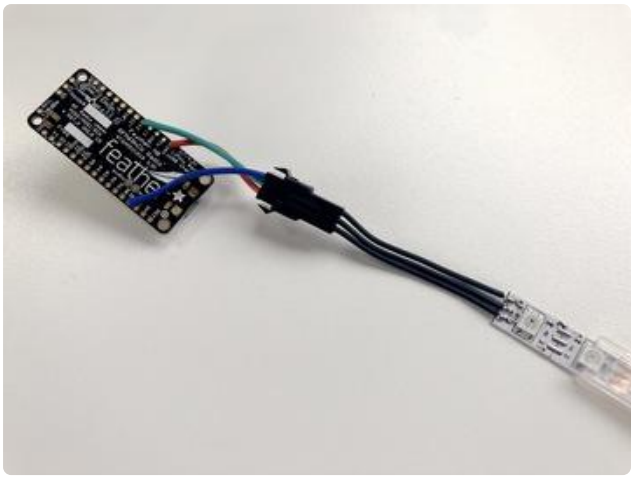


Solder your female 3-pin connector to the Feather as shown:

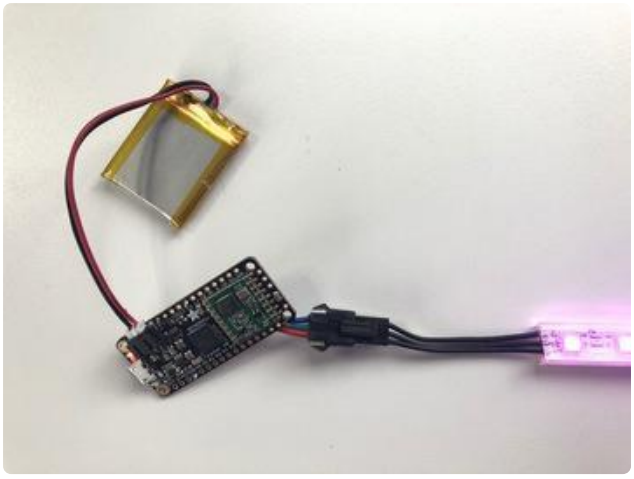
Red wire to BAT
Green wire to 12
Black wire to G



If you're using a NeoPixel strand, all you need to do now is plug your Feather into the male connector that comes with the strand and plug in a battery. Your lights should come on in a blue and purple gradient.



If you're using a NeoPixel strip, cut the 2-pin connector off and expose the bare copper pads at the IN end of the strip. Solder the male side of your 3-pin connector onto the strip. It helps to plug the connector into the female side before you solder, so you can be sure you're getting the wires lined up correctly.



Plug in a battery and the LEDs will come on in a blue and purple gradient, if all goes well.

Troubleshooting

If the lights don't come on, here are a few things to check:

- Did you solder the connector to the IN end of the strip? (arrows on the strip should be pointing away from the Feather M0)
- Did you make sure the connector was lined up correctly? It's easy to solder these on backwards.

- Has the code uploaded correctly? The code for this project is tricky, with lots of libraries to install. If it's just not working, try uploading the strandtest code in Arduino (Examples > NeoPixel Library > Strandtest) making sure to select pin 12. More info about this process can be found here ().
- Are you soldered to pin 12? The labels are small! Be sure you're on the correct pin.

Optional Antenna Connector

I've found that for wearable projects being controlled in the same room or in the vicinity of the controller box, the antenna is not needed on this end. The controller box has an antenna for broadcasting and the Feather picks up the signal with no trouble. It works through windows and up to around 100 yards away.

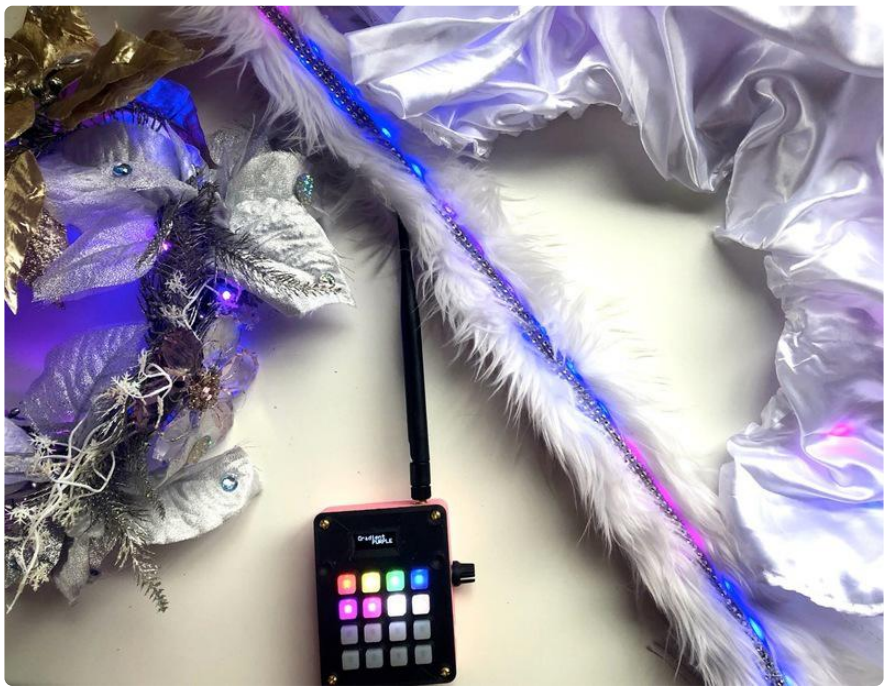
If you want to control your project from further away -- say, change the color of your tent lights from all the way across the Playa -- you'll want to solder the antenna connector onto the back of the Feather M0 and snap on the antenna. The process is the same as what you did when building the NeoTrellis controller box.

Adding Additional NeoPixel Strips

You can add a lot more NeoPixel strips or strands to your project on the other data pins the Feather has available. If you do add more strips, or solder to other pins, note that if you can't use the radio pins (pin 22, 23, and 24) for a NeoPixel data line. Leave those pins alone!

You can see the available pins on the Feather LoRa Pinout here ().

Use it In Your Project



Put a Feather M0 receiver attached to some NeoPixels in all your various projects. The controller box will send a short radio signal (usually one letter, like "A") through the air, and your receiver will listen and obey.

For this example, I've got a NeoPixel crown (), an LED Tutu (), two Raver Bandoliers () and a set of LED Isis Wings, each containing one Feather receiver. I've chosen to upload the same code to all the receivers, so when I press the "red" button on the controller, the command "A" is sent and everything turns red.

It doesn't have to be this way! I could easily upload slightly different code to each receiver, so that the "A" command causes my crown to turn red, my tutu gold and each bandolier could be a blue/purple gradient. It takes a little digging in the code, but it's not too hard to do.

Have fun making everything glow according to your every whim!