Ean Vandergraaf

Dr. Branton – CSCI 342

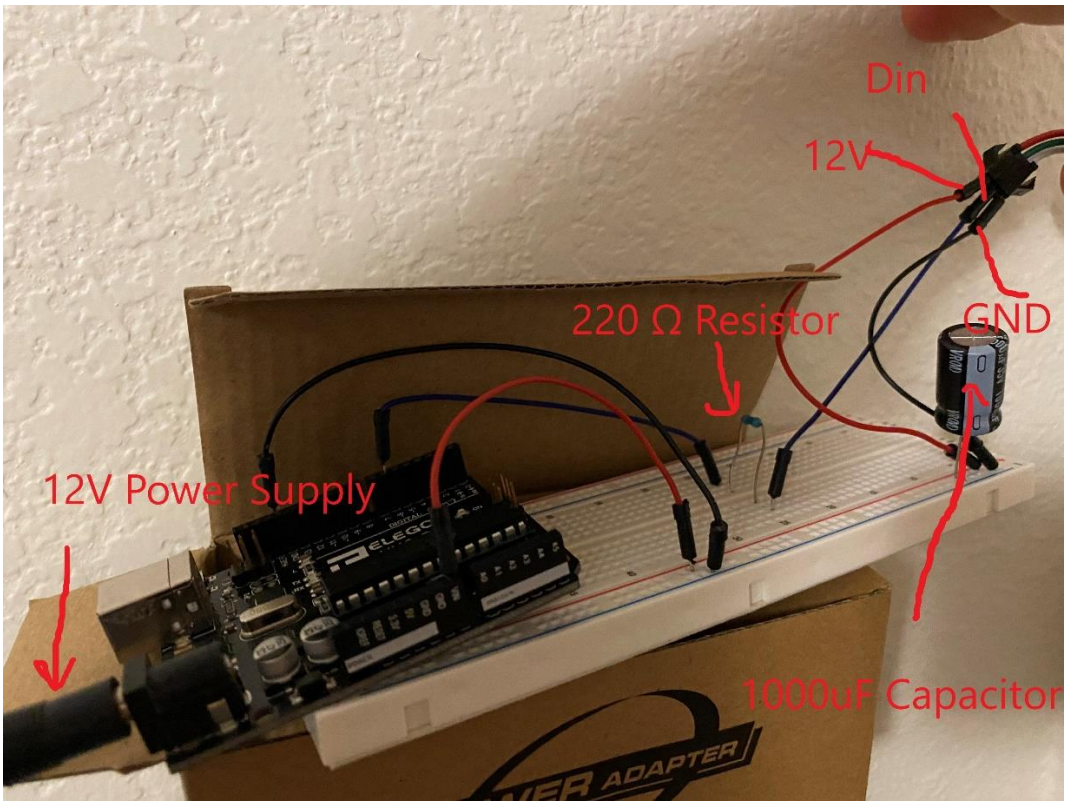
Stretch Assignment – Arduino & LED Strip

**Option 1. Electronics: B. "setting up a connection between your Arduino and another device using one or more of the Arduino pins"**

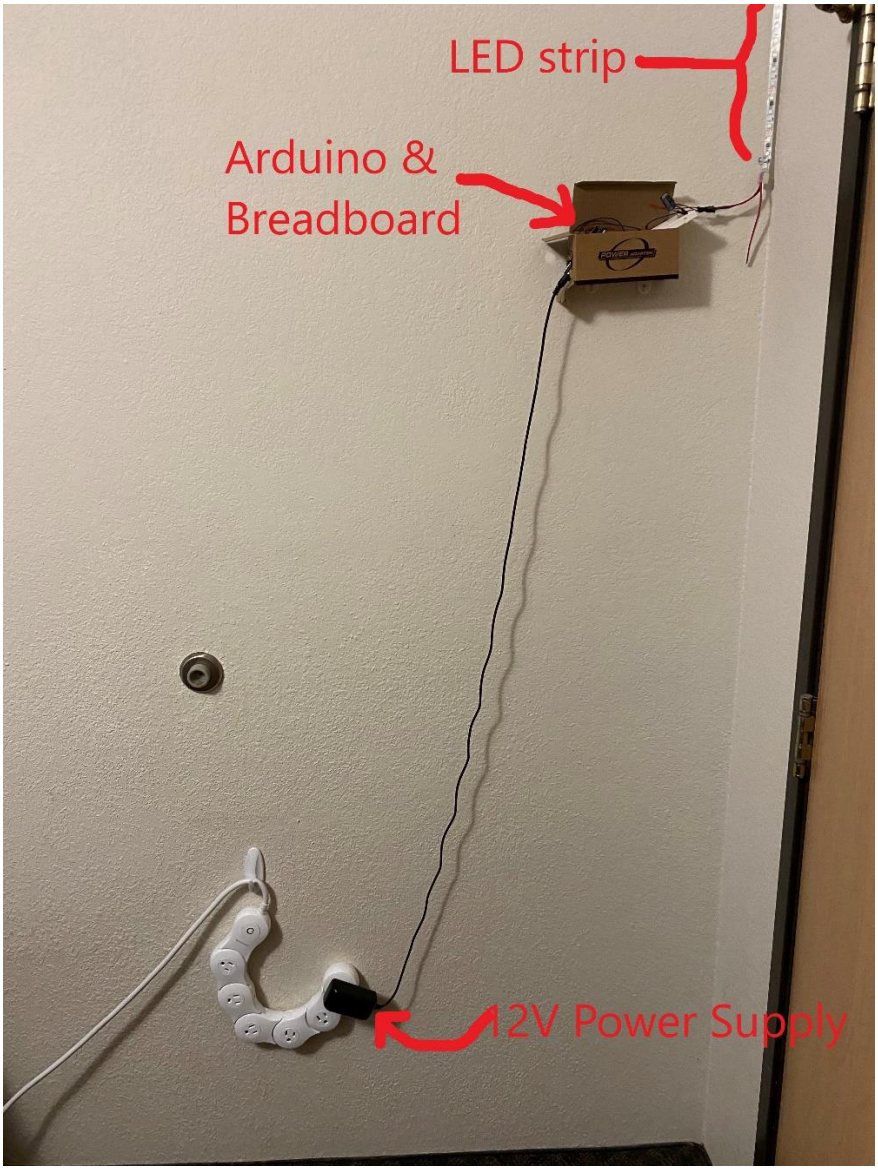
For my stretch assignment I connected my Arduino to a strip of LED lights and a controllable joystick.

## I. Getting Started and Wiring the Arduino to the LEDs

The type of LED strip that I used was the WS2811. These lights allow for individual assignment, meaning that each individual LED on the strip can be assigned to a different color. This strip also requires 12V in order to show the full color spectrum, so extra power had to be supplied. In order to supply power, I ran a 12V power adapter from the wall outlet to the Arduino. This enabled the lights to get enough power. Additionally, I used a 1000 µF capacitor to make sure the LEDs received a consistent power supply. Here is what the wiring looks like up close:



And from a distance:




## II. Exploring LED Libraries (NeoPixel and FastLED)

After wiring the Arduino to the LEDs, I started to code the lights. The two most used libraries for programming LED lights are NeoPixel and FastLED. After working with them, both libraries have their own advantages. I found that FastLED has a lot of built-in functions and is more powerful, while NeoPixel is straight forward and easier to learn. Both libraries provided example programs which I used to learn how each one was used. FastLED provided a program named “DemoReel100”, which would change between different patterns every couple seconds. Additionally, the NeoPixel library provided a program named “strandtest” which changes to a different pattern when the previous one finishes.

## III. Programming the Lights

While programming the lights, I decided to use both the NeoPixel and FastLED libraries due to each of their advantages. Here are some examples from both libraries:

Have the lights turn all red (NeoPixel):

```
void loop(){
    //go through every pixel in the strip
    for(int i = 0; i < NUMPIXELS; i++){
      //setPixelColor(pixel, red, green, blue);
      pixels.setPixelColor(i, 255, 0, 0);
    }
    //displays the colors
    pixels.show();
}
```

Have the lights turn all green (FastLED):

```
void loop(){
  //built-in function to fill in every LED
  fill_solid(leds, NUMPIXELS, CRGB::Green);
  //update the LEDs
  FastLED.show();
}
```

Rainbow Effect (FastLED):

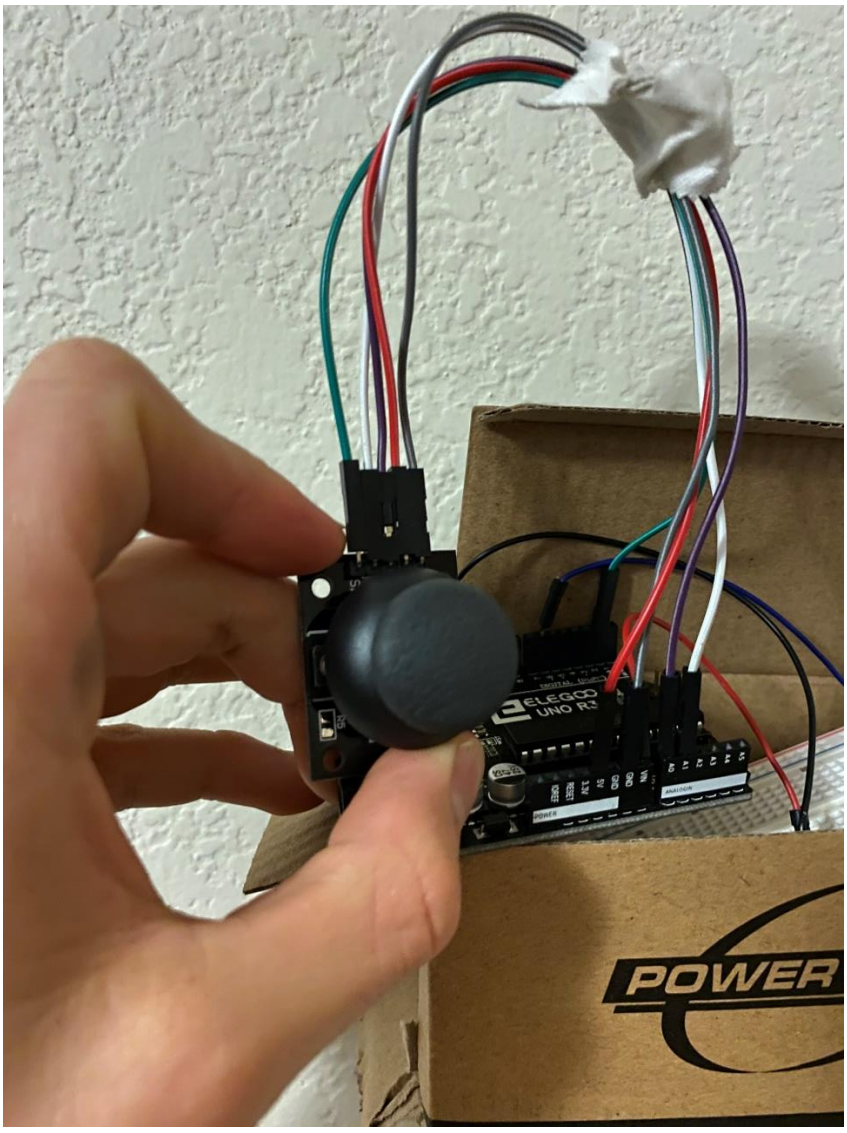
```
void rainbow()
{
  // FastLED's built-in rainbow generator
  //gHue is rotating "base color" used by many of the patterns
  fill_rainbow( leds, NUM_LEDS, gHue, 7);
}
```

Theater Lights (NeoPixel):

```
void loop(){
  //loop through and set every even pixel to red and every odd to blue
  for(int i = 0; i < NUMPIXELS; i = i + 2){
    pixels.setPixelColor(i, 255, 0, 0);
    pixels.setPixelColor(i + 1, 0, 0, 255);
  }
  pixels.show();
  delay(100);
  //loops through and does the opposite
  for(int i = 0; i < NUMPIXELS; i = i + 2){
    pixels.setPixelColor(i, 0, 0, 255);
    pixels.setPixelColor(i + 1, 255, 0, 0);
  }
  pixels.show();
  delay(100);
}
```

## IV. Wiring the Joystick

To add an external component to the system, I decided to wire a joystick to the Arduino that could control the lights. This additionally makes it easier to provide input to the Arduino without having to use my computer. Here is what the wiring looks like:




## V. Controlling Patterns Using the Joystick

Now that I can receive input from the joystick into the Arduino, I programmed the Arduino to change pattern based on how the user controls the joystick. I changed the "DemoReel100" starter code to change pattern, not after a certain time, but after the user moves the joystick to the left or right. Here is what the code for changing the pattern looks like:

```
//if user moves joystick all the way to the right
//move to the next pattern
if(analogRead(X_pin) == 1023) {
  //function that changes to the next pattern
  nextPattern();
  //delays so that the patterns don't change too fast
  delay(1000);
};
//if user moves joystick all the way to the left
//move to the previous pattern
if(analogRead(X_pin) == 0){
  //function that changes to the previous pattern
  prevPattern();
  delay(1000);
}
```

The function “nextPattern()” was originally written in the code. The function “prevPattern()” is an adjustment of nextPattern that will work in the reverse, and will wrap the array. Both these functions return other functions that will display patterns on the strip.

Additionally, I created a new pattern, “colorGlitter”, which will originally start with the color red and have white sparkles randomly show up within the strip. Then whenever the user moves the joystick up or down the color of the strip will change, while keeping the white sparkles. Here is what the code for that pattern looks like:

```
void colorGlitter(){
    fill_solid(leds, NUM_LEDS, colors[gCurrentColorNum]);
    addGlitter(80);
}
```

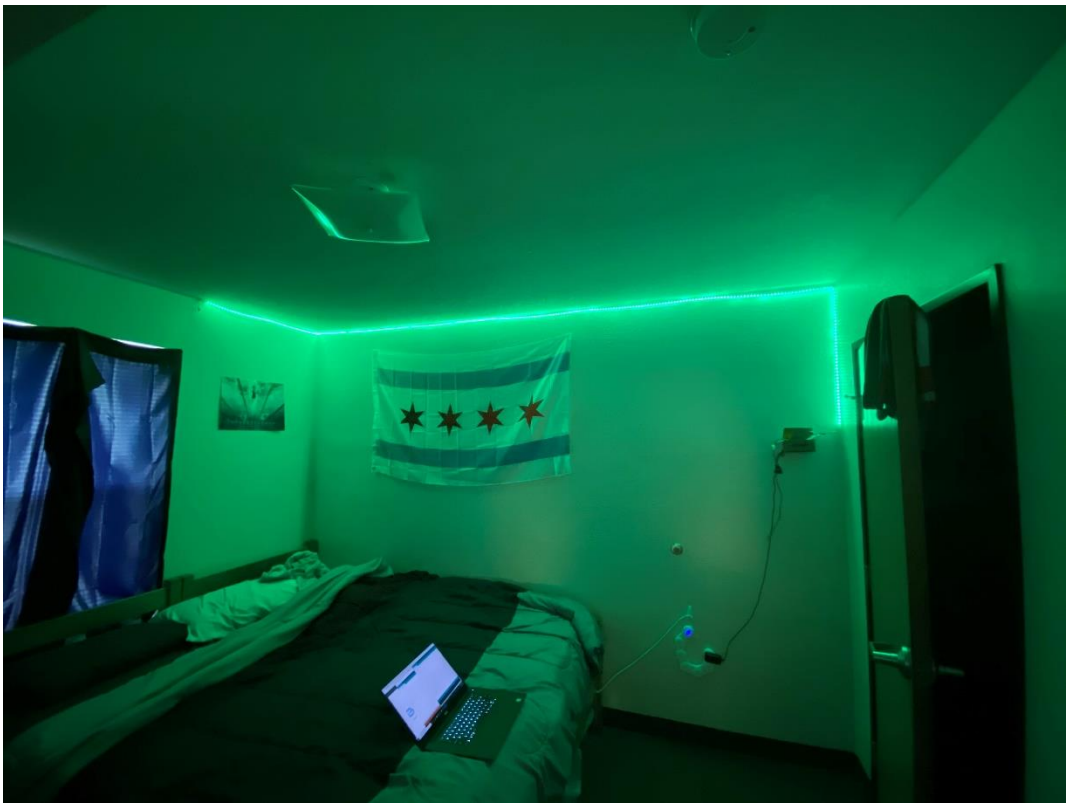
The addGlitter function was already written in “DemoReel100” and I did not change anything from the original code. Also, the background color will be determined by the gCurrentColorNum variable, which changes whenever the joystick is moved. Here is the code for whenever the joystick is moved (this was written within the loop function):

```
  //if on the colorGlitter pattern and the user moves the joy stick up
  if(gCurrentPatternNumber == 6 && analogRead(Y_pin) == 1023){
    //change the background color to the next color
    nextColor();
    //delay so that patterns don't change too fast
    delay(1000);
  }
  //if on the colorGlitter pattern and the user moves the joy stick down
  if(gCurrentPatternNumber == 6 && analogRead(Y_pin) == 0){
    //change the background color to the previous color
    prevColor();
    delay(10);
  }
```

Neither “nextColor()” nor “prevColor()” were included in the original code. Both of these functions will change the color of the background of the strip. The two functions will cycle through an array of color objects that will be displayed on the strip. Here is what that array looks like:

```
CRGB colors[] = {
  CRGB(255, 0, 0), //red
  CRGB(255, 125, 0), //orange
  CRGB(255, 255, 0), //yellow
  CRGB(125, 255, 0), //yellowGreen
  CRGB(0, 255, 0), //green
  CRGB(0, 255, 125), //turquoise
  CRGB(0, 255, 255), //cyan
  CRGB(0, 125, 255), //ocean
  CRGB(0, 0, 255), //blue
  CRGB(125, 0, 255), //violet
  CRGB(255, 0, 255), //magenta
  CRGB(255, 0, 255) //raspberry
};
```

## VI. Pictures (from example code in III)



## VII. Sources

Finally, all the information regarding wiring and setting up the LEDs was found at https://www.makeuseof.com/tag/connect-led-light-strips-arduino/, as well as in the book, “Elegoo Super Starter Kit for UNO” (which was included in the kit). Additionally, all original code for “DemoReel100” is credited to Mark Kriegsman, as I only made adjustments to fit my project.