ILLUMINATIONS SEMINAR - Week 3

Sept 20, 2022

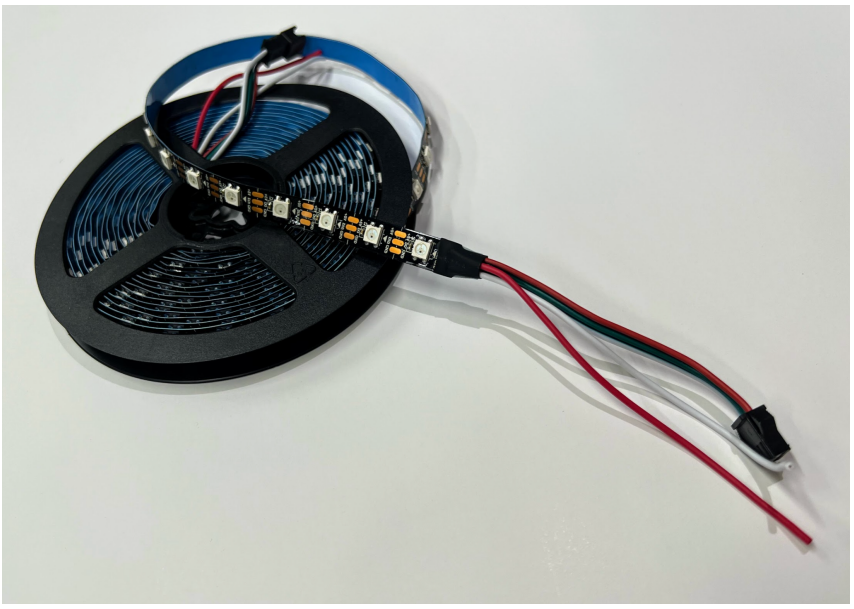
Overview

Through this exercise, we're going to learn how to interface with the individually addressable LEDs. Specifically, as the WS2812B LEDs. While there are many different flavors of individually addressable LEDs (also sometimes called Node LEDs or NeoPixels®), the WS2812B is one of the more common protocols due to it's high reliability, low driving power required (can run on 5v), good color mixing, and simple protocol. They're also reasonably priced at about $30-40 for 16 feet.

Get some lights blinking!

Identification

The first step we're going to take is just to get the LEDs connected to the Arduino and run some sample code to get it to blink.



Take a closer look at the LED strip and you should see 5V, Din/Do, and GND. You'll likely see **5 wires** coming out of each end (2 x Red, 2 x White, and Green). You can ignore the lone red and white strand of wire, as they're actually soldered onto the same connection as the other red and white. If you need more power, or you're

connecting to an external power source, these wires are here to make it easier to do so. **We won't be using these today.**

Next, you'll notice one end of the strip has 3 exposed metal contacts (the male end). You'll also notice the other (more likely the end that's unfurling right now) accepts the contacts (the female end).

One important electrical engineering skill is to be able to find what connector this is. Take a quick look at the packaging and/or online to see if you can determine the name of this connector.

The connector is:
(fill in the blanks below - each with one number of letter)

_ _ _ _ _ _ _ _ _ _ _ _ _ _

Wiring it Up

There are many different ways to wire these LEDs up, including using the proper connectors. For now, while we're still in the 'prototyping' phase, we're going to use these jumper cables. Grab 3 of them (ideally matching the colors, but no worries if not) and "plug" them into the female end of the LED strip.

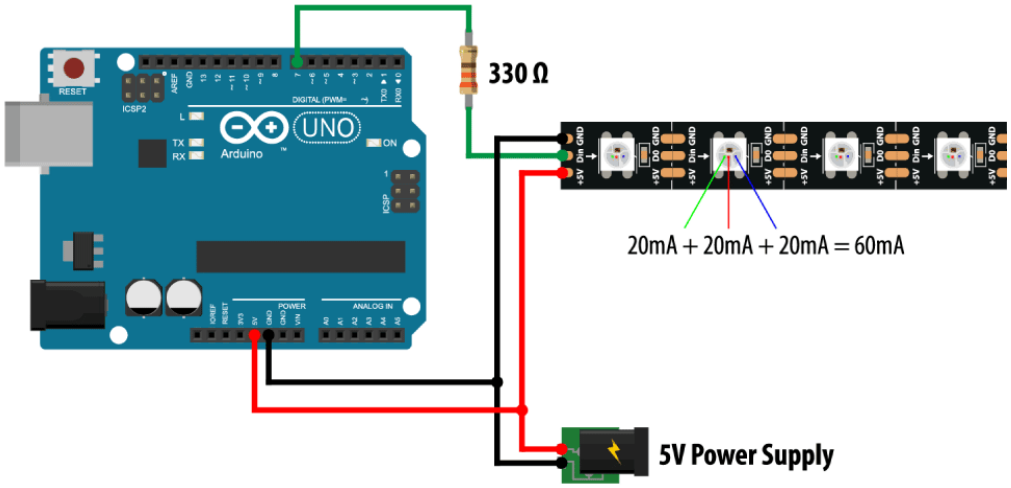
You'll notice this is a very flimsy way of attaching it, but it'll work for now!

On the other end, connect it to your Arduino as follows:

Red: 5V Arduino Pin
Green: Arduino Pin 2
White: Ground on the Arduino (any ground)

If you follow the instructions above, you should be all set.

However, take a look at the diagram below



This shows 2 things.

- A resistor is sometimes placed between the data (Din) and the Arduino (in this case, it's **330Ω**. This extra bit of resistance will help protect the data pin especially if you have a long data line, which can cause unwanted noise/reflections/ringing. Because our jumper cables are so short, it's not something we'll need
- If you ever need **more power**, note how the 5v power supply is wired up in this diagram. The extra red and white wires are essentially the two leads coming out for your 5V power supply. To select a 5V power supply, as long as it's 5V and has an operating current of **greater than** the amount of current you're drawing, that's all you need to worry about.

The Arduino & The Code

For this, we're going to download a library called 'FastLED'. This library will handle all the communication for us so that we don't have to manually send bits across to the LED strip.

To download this library, open your Arduino IDE and go to `Tools › Manage Libraries`

In the search bar, look up `FastLED` (with that capitalization)

Hit `Install`!

Now, if you go into `Files > Examples`, you should see a new section at the bottom called `FastLED`. Go ahead and open `Cyclon`

Once you have it opened, hit `UPLOAD`, and your LED strip should be making a bunch of colors! (Remember to select your port and your board from the TOOLS menu if errors are appearing)

Exciting! Feel free to take a moment to admire this beauty.

A Quick Gut Check

Now let's go back to the basics. Go ahead and navigate to https://bit.ly/ws2812basic and let's download this code. First, see if you can get the code to compile and upload to your Arduino. It should be working when there's a stream of white LEDs that light up and then turn all off.

Let's modify some of the code. Take a quick look through the Arduino Sketch. Note the line that says:

```
#define NUM_LEDS 10
```

This tells the code how many LEDs are in your stip. You have a much longer LED strip, so let's go ahead and change that number to **300** (you actually have 300 LEDs on this strip!

While we're at it, let's make it brighter. Let's change:

```
FastLED.setBrightness(60);
```

to

```
FastLED.setBrightness(255);
```

What does brightness do? According to the FastLED Documentation: ***Brightness is a 0-255 value for how much to scale all leds before writing them out***

Hit `UPLOAD`!

Yikes. Note down some things that happens (what behaviors are you noticing): (Write below)

Why does this happen? (Write below)

Change the parameters again to make it somewhat reasonable. Were there any challenges in resolving this/reuploading the code?

[CHECKOFF 1] Get a checkoff with one of the LAs to make sure you understand this before moving on, and discuss some ways of mitigating this in the future if you wish to have this many LEDs at this brightness!

Make it your own

Now that we have something reasonable, let's try to modify this code. Change some of the values to see if, instead of a bright white light, it's a red light. And then try to get the same for a green light. And then a blue light.

If you find something is off and not what you expect, note it below.

For the first task, let's try to create a nice gradient from Green to Blue to Red over the course of 50 LEDs. Meaning:

LED 1: Green

LED 2: Green with a hint of blue

...

LED 12: Teal

...

LED 25: Blue

LED 26: Blue with a hint of red

...

LED 38: Purplish Pink

...

LED 50: Red

Do not program each LED value manually - use *FOR* loops (you may use two if you'd like.) You can also look up the cHSV (hue, saturation, value) representation online for FastLED instead of cRGB.

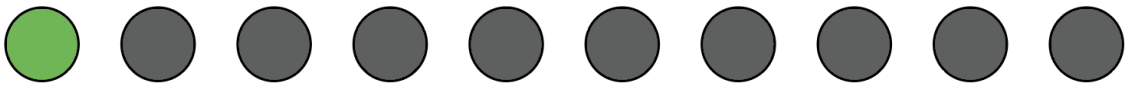
Name this sketch **GreenBlueRedGradient.ino**

[CHECKOFF 2] Get a checkoff with one of the LAs to make sure you understand this before moving on!

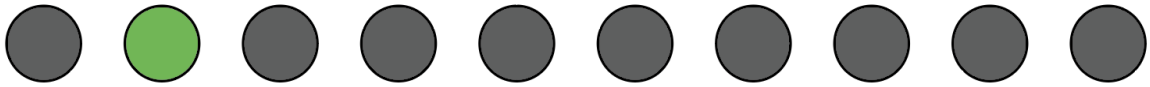
Build a Simple Counter

Now, we're going to implement a button and implement a simple counter with this LED strip! You'll want to start by wiring up a simple button to your circuit (use any of the non-used Arduino digital pins!)

The LED strip is going to start all off. When the button is pressed, you're going to increment the LED that is on by 1.



If you click it again, the LED on will be number 2.



And so on. Once you hit 10, try doing an interesting transition/loop to cycle it back and reset it to the "all off" state.

Name this sketch **ButtonPressCounter.ino**

[CHECKOFF 3] Get a checkoff with one of the LAs before moving on!

Build a Simple Game (Bonus)

Congratulations! You've made it to the bonus part of this handout. Give yourself a pat on the back - this is no easy feat! This is definitely a bonus, so don't feel pressured to complete this within the allotted time/week.

The goal of this challenge is to build a simple version of the 'cyclone/jackpot' arcade machine using a button and an LED strip

For those unfamiliar with this arcade classic, this is how it works:

The game consists of a series of lights. The first one turns on, then off, then the second one turns on, then off, etc. This makes it look like the light is "moving". As the light 'goes around', your goal (as a player!) is to hit the button such that the light lands on the designated blue light.

In the example above, the yellow led is the one that is 'moving', and the blue one in the center is the 'designated' light. Give it a couple of milliseconds and then it transitions to:

... and so on

Once the yellow light is on top of the blue light, your goal as a player is to hit the button and 'stop' the light! If you do get it right, you win!

If the player gets it on the dot, do a short little celebration and restart the game. On the other hand, if you lose (in the example below, the player hits the button after the yellow light has 'moved' off the blue):

Do a tragic light show and reset.

There are plenty of ways to implement this - feel free to have fun with it and experiment with different timings, lighting effects, colors, and more!

[BONUS CHECKOFF 4] Get a checkoff with one of the LAs!