Activity 6.1 – learning about NeoPixels – a Hand HEX system

This is going to be the biggest project so far in this book. In this activity, we are going to get complex and build a system that will be useful to visual designers in many different fields. As we've seen, HEX codes and RBG values are important when we are programming NeoPixels. Using a HEX code is how we choose the colors to display. We use three values – red, green, and blue – which make up the output color.

In this circuit, we want to use a color sensor to detect any color around us, even if it's a color on a drink can nearby! Then, we want to use that color to light our NeoPixels. We aren't going to stop there, though! We will also hook up an OLED screen so that we can see the HEX values as their numerical values on a screen for additional output. This is so we can use that exact color in any other sketches we are writing, or for any designers so that they can use it in their work. Pretty cool, hey?

This activity will take us through mapping, hooking up, and programming our circuit board to get it to function. If you want to sew it together, you can do that here or you can wait until a little further into this book. In *Chapter 10*, *Soldering and Sewing to Complete Your Project*, we will learn more about techniques to complete wearables.

> **A Note for Future Circuits You Design**
>
> It's always good practice to *connect the power pin last*. Sometimes, especially if we didn't disconnect our board from the computer, we can get a surge on the power connection. This can damage our components!

You can use a glove that you already have if you think it is suitable or get creative and make your own. It's great practice and then you'll have an additional skill for creating your wearables. Remember: it only needs to be a prototype.

The parts list for a Hand HEX system is shown in *Figure 6.10*. They are as follows:

- Sewable or another color sensor
- NeoPixels, in your chosen form – strip, strand, or shapes
- A glove (buy one, make your own, or upcycle an existing glove)
- An OLED screen. Choose your preferred color and size – I'm using 128x128 yellow/blue
- An Adafruit Flora board
- A rechargeable LiPoly battery

Now that we have the materials, my first step is to map out the circuit in my notebook. I will assemble all my parts and then work through the pinouts:

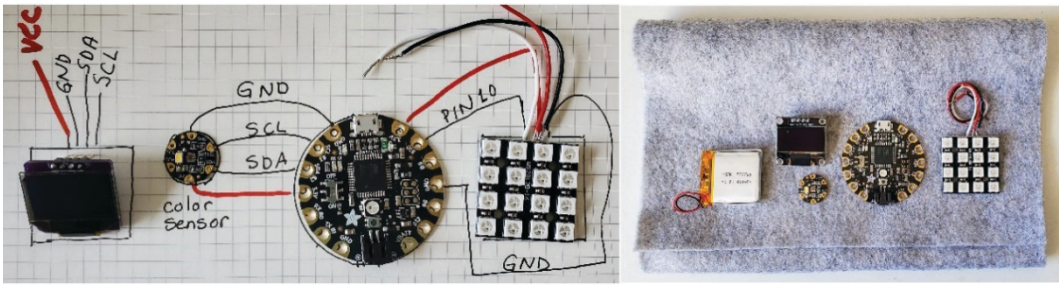

Figure 6.10 – Mapping the complete circuit and the parts for the Hand HEX system

Figure 6.10 also shows my map of the circuit connections that I'll follow. We'll build this project in stages. This is a great way to work because then, as we proceed through the circuit in complexity, we can fix errors as they happen. We'll know that our circuit works as we build it. There will be *three parts*: the *color sensor*, then the *OLED*, and then the *NeoPixels*.

First, let's add the color sensor.

Part 1 – color sensor

I'll start by hooking up the Flora sewable color sensor to the Flora board. It uses the I^2C protocol that we learned about in the previous chapter. To connect the sensor and Flora, you must set the following:

- Sensor 3V to Flora 3.3V
- Sensor GND to Flora GND
- Sensor SCL to I^2C Clock on Flora (SCL pin 3)
- Sensor SDA to I^2C Data on Flora (SDA pin 2)

Once you've connected them (*Figure 6.11*), plug the board into your computer so that we can try out the sensor:

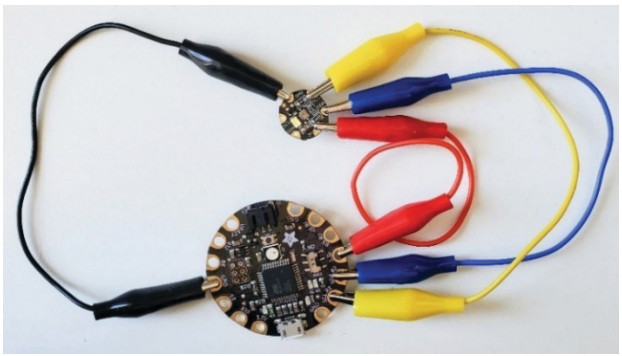

Figure 6.11 – Color sensor connected to Flora

Let's open the Arduino IDE and install a library to use this sensor. Open the **Library Manager** area (*Figure 6.12*) and search for `Adafruit TCS34725`:

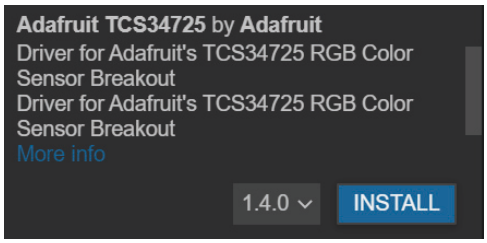

Figure 6.12 – Installing the library for the color sensor

The color sensing library didn't come up when I searched the exact name, so if you don't see it in the list, try searching for `color sensor` – that worked for me. When you do click **Install**, it will ask about dependencies. As shown in the following screenshot you'll want to choose **Install all**:

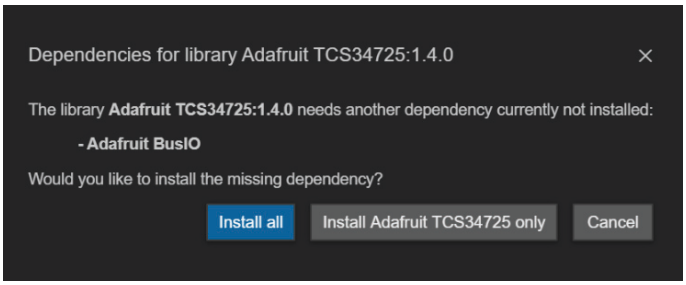

Figure 6.13 – Installing the dependencies

Be sure to wait for the notification stating that you have successfully installed the library. Sometimes, there is a glitch with the Arduino IDE 2.0 where you need to repeat this process and try the install process more than once as it may not install the library the first time. If you have the installed messages in the output window, then you're good to go!

Because this library is used for a slightly different color sensor, we'll use my modified code at `https://github.com/cmoz/Ultimate/tree/main/C6/C6_ColorSensor` for this example:

```
#include <Wire.h>
#include "Adafruit_TCS34725.h"
Adafruit_TCS34725 tcs = Adafruit_TCS34725(TCS34725_
INTEGRATIONTIME_50MS, TCS34725_GAIN_4X);

void setup() {
  Serial.begin(9600);
```

```
  Serial.println("Color View Test!");
  if (tcs.begin()) {
  Serial.println("Found sensor");
  } else {
  Serial.println("No TCS34725 found");
  while (1);
  }
}

void loop() {
  uint16_t clear, red, green, blue;
  tcs.setInterrupt(false);
  delay(60);
  tcs.getRawData(&red, &green, &blue, &clear);
  tcs.setInterrupt(true);
  uint32_t sum = clear;
  float r, g, b;
  r = red;
  r /= sum;
  g = green;
  g /= sum;
  b = blue;
  b /= sum;
  r *= 256; g *= 256; b *= 256;

  Serial.print("HEX: \t");
  Serial.print((int)r, HEX); Serial.print((int)g, HEX);
  Serial.print((int)b, HEX); Serial.print("\t \t RGB: \t");
  Serial.print((int)r ); Serial.print(" ");
  Serial.print((int)g); Serial.print(" ");
  Serial.println((int)b); Serial.println();
}
```

Upload the sketch to the Flora board and open Serial Monitor. You should see a display of the color values in the output, as shown in the following screenshot:

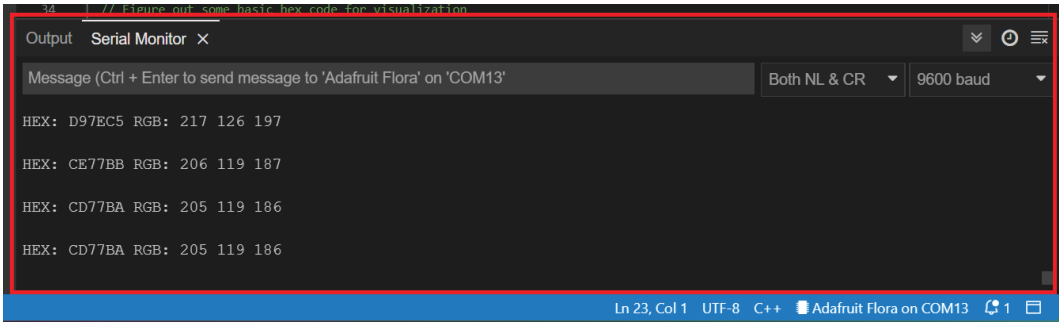

Figure 6.14 – Serial Monitor output from the color sensor

Your color sensor blinks as it's sensing. The output to the monitor depends on the color of the item you hold over the sensor. I held a pink item, which is why I was given that readout.

Now that we have one part of our circuit working, we can connect the NeoPixels, which we'll use to display the color of the item we are sensing. Remember, though, that this will only be as accurate as the LEDs can create, so it won't be an exact match.

Part 2 – OLED

Unplug your Flora board from your computer and hook up the OLED display. Connecting the OLED should be familiar to you now because it follows the same I^2C protocol that we covered in *Chapter 5*, *Working with Sensors: All about Inputs!*. The OLED also has four pins, two of which are for the I^2C protocol. As mentioned previously, connect SDA on the screen to SDA on the Flora, and connect the SPI on the screen to the SPI on the Flora. Then hook up the ground pin and finish with the power pin.

We are using the same SDA and SCL pins because each of the components we are putting on these pins has a unique address. If you are curious about the address for your components, then you can run the sketch that outputs the address of your I^2C devices.

To easily program the OLED, let's install a library. Open the Library Manager area and search for `Adafruit_SSD1306`. This library will also ask you to install **Adafruit_GFX** at the same time, so install both (see *Chapter 4*, *Implementing Arduino Code Using Gemma M0 and Circuit Playground*, if you need more guidance on installing libraries):

```
#include <Wire.h>
#include "Adafruit_GFX.h"
#include "Adafruit_SSD1306.h"

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
```

```
Adafruit_SSD1306 oled(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

void setup() {
  Serial.begin(9600);
  if (!oled.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
  Serial.println(F("SSD1306 allocation failed"));
  while (true);
}
  delay(2000);
  oled.clearDisplay();
  oled.setTextSize(3);
  oled.setTextColor(WHITE);
  oled.setCursor(0, 10);
  oled.println("Hello World!");
  oled.display();
}
```

This time, there isn't anything in our `loop()` because this code is just checking everything is working. The OLED should now be displaying text that says *Hello World*. Play around with this code to have it display something else, such as *Ultimate Wearables*! This is a good way to get to know the code. When you've finished trying different effects and code, we can put the color sensor and OLED code together.

Open your code for the color sensor and save it as a new file. Then, add the libraries we are using, add the initialization, and then the rest of the code.

Your turn: try to combine the code. It's good practice to combine the code yourself first. You can build it up with each new function you want your wearable to perform. Try it now with the two components:

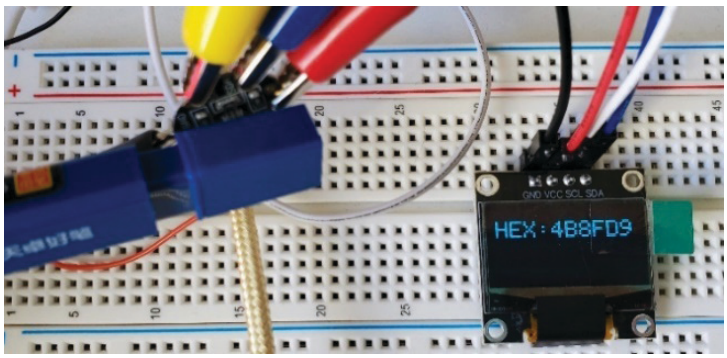

Figure 6.15 – The OLED display showing the HEX reading taken from the color scanner

The code from `https://github.com/cmoz/Ultimate/tree/main/C6/C6_ColorSensor_andOLED` will contain both elements – that is, the color sensor and the OLED. Upload it to your Flora board and then run it. Test it by holding something of color against your color sensor. *Figure 6.15* shows the OLED displaying a HEX value for a blue pencil I scanned by holding the end against the sensor. We now have an input that is receiving data and creating an output based on that data!

Part 3 – NeoPixels

Let's create a second output that will visually echo the color that we are scanning with the color sensor.

Make sure you unplug the Flora board from the computer before you hook up the NeoPixels. Once *unplugged*, connect the NeoPixels to the Flora board. Your configuration of NeoPixels might be different than mine. I'm going to use the strand version. Whatever version you choose, there will be *one data pin* and a *power* and *ground* pin:

1. Connect your NeoPixel *Data In* pin to a pin that is available of your choice. This pin will be added to the code. I'm choosing pin 10 on the Flora board.
2. Then, connect the power and ground of the NeoPixel to the power and ground of the Flora.
3. In the Arduino IDE, add the **Adafruit NeoPixel** library. This will install sample files for us that we will use to test whether our lights are working. Open the `strandtest` file by going to **File** | **Examples** | **Adafruit NeoPixel** | **strandtest**.

Change the `#define LED_PIN 6` line of code so that it contains the same value as your pin; I chose pin 10. The new line of code will look like this: `#define LED_PIN 10`. The line after that one is a count of the number of NeoPixel LEDs being used. Count the lights you are using in your strip, strand, or shape, and put this number in. Mine looks like this: `#define LED_COUNT 8`. Note that if you are using *RGBW* pixels, then you need to change `NEO_GRB` in the following code line:

```
Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRB + NEO_
KHZ800);
```

You must change it to `NEO_RGBW`:

```
Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_RGBW + NEO_
KHZ800);
```

The NeoPixels will still work if this isn't correct, but you will see some *unlit* NeoPixels when they are supposed to be white, for example. Some strange lighting effects will happen, so you will know that you should change the code. **Upload** the sample `strandtest` code to the Flora board.

Once uploaded, you should see a wonderful display of colors and effects.

Do they light up? If not, make sure you're using the **Data In** (**DI**) line and not the **Data Out** (**DO**) line. Have you got power and ground to your NeoPixels? Check those connections. Are you using the right pin? Did you change the pin's number in the code to reflect the pin that the NeoPixels are connected to on the circuit board?

I always find this an exciting part of the process; seeing a NeoPixel in action is always fun!

Understanding how to control a single NeoPixel is good knowledge to have. It will help with your programming and controlling NeoPixels for your future wearables. Let's look at the sample code for seeing the NeoPixels light up one at a time with one color.

You can use the following code to do this. Looking through the code to understand it can be a good way to learn how to control NeoPixels:

```
#include "Adafruit_NeoPixel.h"
#define LED_PIN 10
#define LED_COUNT 8

Adafruit_NeoPixel strip(LED_COUNT, LED_PIN, NEO_GRB + NEO_
KHZ800);

void setup() {
  strip.begin();
  strip.show();
  strip.setBrightness(50);
}

void loop() {
  for(int i=0; i<strip.numPixels(); i++) {
  strip.setPixelColor(i, (0, 0, 220));
  strip.show();
  delay(1000);
  }
}
```

Note that the color values for each of the three colors can be set to a maximum value of 255. Full brightness is also a value of 255. `strip.numPixels()` will take our predefined number of pixels in our strand, and use that number to count up to light each one. In the `strip.setPixelColor(i, (0, 0, 220));` line we are setting the `i` value to the LED number, so it will increment, lighting the first, then the second, and so on. Then, it sets the values of the color we want. Here, it is 0 = no red, 0 = no green, and 220 = almost full blue. If you wanted to only light up the fourth pixel in a string of eight, for example, how do you think we would write the code? If you wrote `strip.setPixelColor(4, (0, 0, 220));`, you'd be correct.

Setting these values to all 0s will turn off the lights.

Putting it all together

Now, we can combine the code further. At `https://github.com/cmoz/Ultimate/tree/main/C6/C6_ColorSensor_andOLED_Neo`, I've created a file of the completed code that has all three parts put together. When the code is working, you'll see flashing on the color sensor. This means it is taking a reading:

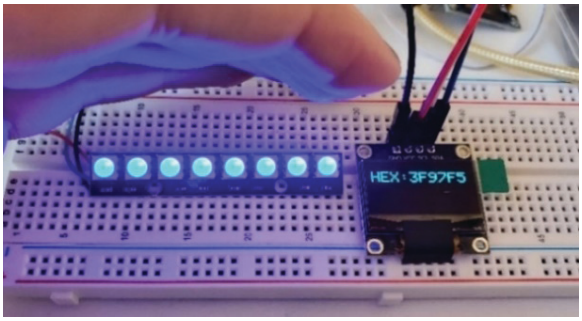

Figure 6.16 – The color sensor and two outputs – an OLED screen and NeoPixels

Then, the HEX code is written to the OLED screen, and the color is sent as RGB values to the NeoPixels (*Figure 6.16*). I've also checked the HEX code that was displayed on the screen and typed the value into a color HEX website, `https://www.color-hex.com/`, to check whether the value is close to what I'm expecting.

To make it into a wearable glove, you can sew the components to a glove that you already have. However, if you have some felt, you can use that to create a quick prototype fingerless glove. We just need a piece of felt big enough for our hand, front and back:

Figure 6.17 – Checking the size of the fabric

Place your hand on top of the felt, and measure across where your knuckles are. This will be the widest part of the glove. The material will need to cover that part of your hand and account for the thickness of your hand too. *Figure 6.17* shows the area we need to measure, and the felt draped over the hand to be sure there is enough of the fabric.

Then, once you've cut the felt, fold it over your hand to be sure there is overlap to it so that it will fit once it is sewn (or glued) together. To start putting it together, it's a good idea to use safety pins to hold the shape in place. Then, put the glove on to feel where the components will fit. Keep in mind that the purpose of this glove is for it to be used by someone typing on a computer because we want to help them with finding HEX code. So, make sure you can hold a mouse easily (*Figure 6.18*) when the components are on it.

Once you are satisfied with the placement, place the components on the felt and glue them into place:

Figure 6.18 – Trying it before sewing the final version

The sensor is just out of view on mine because I want it near my pinky. Once the glue is dry, you can sew the components into place. I map out my connections before I sew to be sure I know where the threads might overlap. This rapid prototype in felt is a good way to test your wearable and find if there are usability issues.

I'm going to transfer my circuit to a glove design made from scuba style fabric *neoprene* for a stretchable fit. This would become a prototype iteration to test a more permanent solution because I like the functionality of this wearable!

Once you build upon your skill set, you may want to look at soldering NeoPixels yourself, which can be a great way to have the freedom to put these lights anywhere into a wearable technology garment (see *Chapter 10*, *Soldering and Sewing to Complete Your Project*).

Now that we've had a look at a complex circuit, let's see how we can attach EL wire to a wearable.

Activity 6.2 – sewing EL wire

Adding EL wire to a wearable is a fun way to create an impact. It's great to put on a backpack, for example, so that it can be seen at night. This could be a good idea for cyclists. You can add EL wire to an existing item or make a garment/bag and plan for the EL wire to go in the seam of it. Either way, it's easy to do and only requires a few steps. To add EL wire to an existing bag, you'll need the EL wire in the length that will work for you. You can cut it down to size, though. You'll also need the inverter battery pack to power it.