Audio Visualization Over Wi-Fi Using ESP8266 and WS2812B LEDs

David Cain

Department of Electrical Engineering

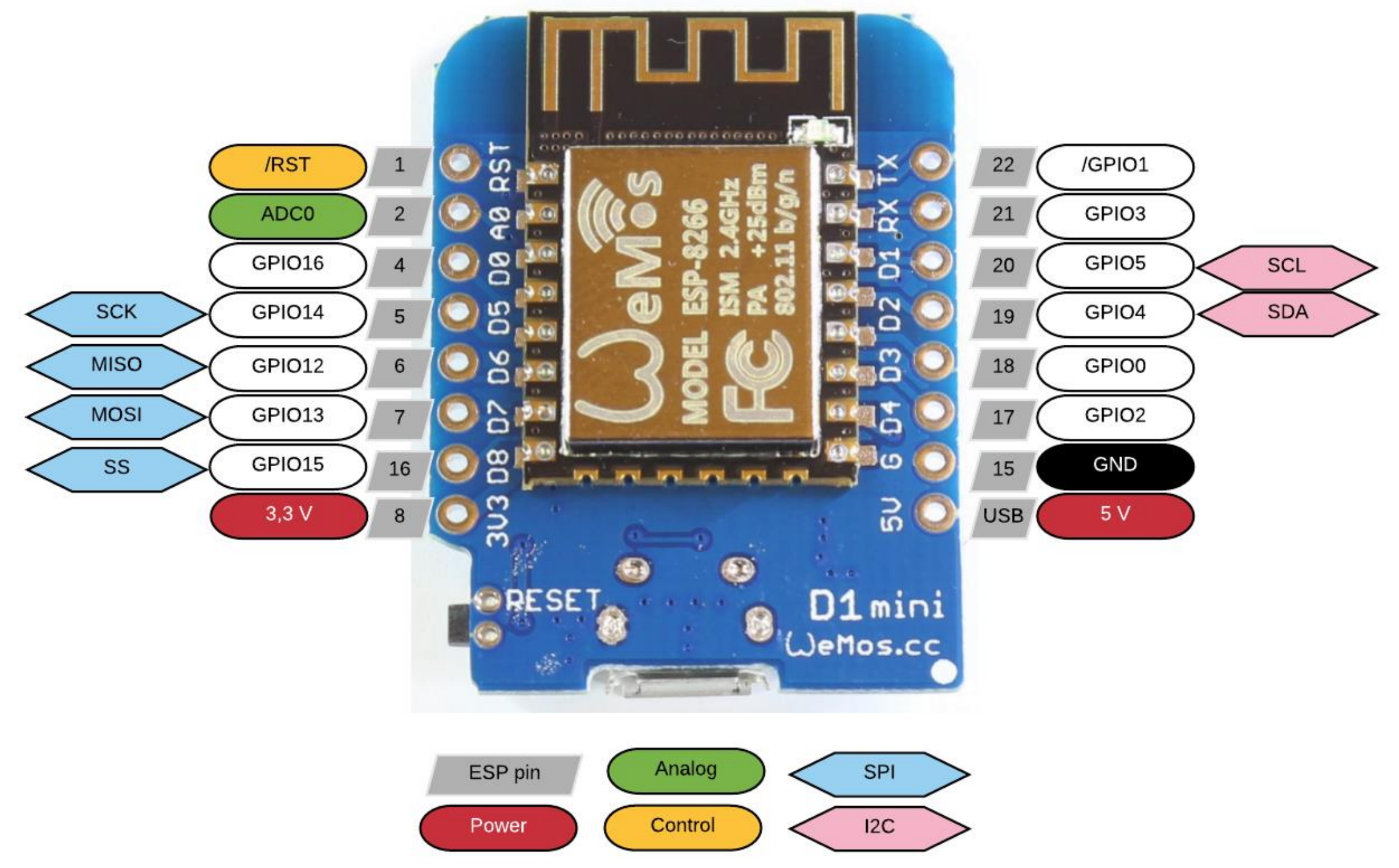
UNIVERSITY of LOUISIANA LAFAYETTE

What is an ESP8266?

A low cost low power microcontroller developed by Espressif Systems. The version used in this project is the Wemos D1 mini ESP8266.

It is an SoC developed for IoT projects but has become very common the microcontroller community for simple projects that need to implement Wi-Fi.

Bonus Fact - Using proper coding and hardware it is possible to reliably run these controllers off of AA batteries for months at a time.



What are WS2812B LEDs?

Individually addressable RGB LEDs that are supported by many platforms, allowing easy code to create complex designs and patterns.

As with any serial data protocol, a clock is necessary to interpret the data being received. For these LEDs, the every bit is a 1 followed by a 0, acting as the 'clock' signal. The value of these two bits is determined by whether the 1 or 0 held its value longer.

The instructions for the LEDs are 24 bits, 8 bits each for R,G, and B. This allows for 2^24 or roughly 16.7 unique color combinations.

Project Goal

Have two microcontrollers in a master/slave configuration to display real-time audio visualization based off of volume intensity. The master will act as a Wi-Fi access point and the slave will connect receiving audio signals to be processed and displayed.

To the right is a simple wiring scheme for the controller's sensors/inputs.

Specifications

This implementation will be battery powered, emphasizing the wireless communication. In this implementation the controllers will be powered from consumer power banks.

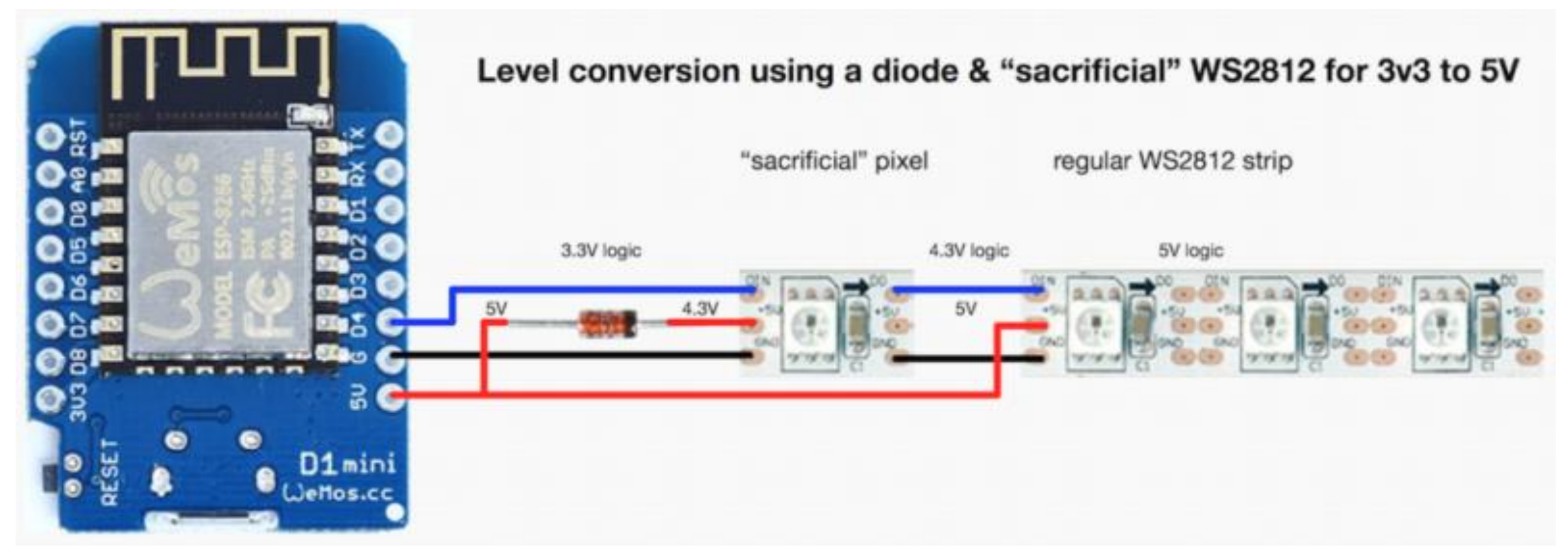
The master controller will do minimal 'processing' allowing for most of its resources to be allocated to access point control and communication with the slave.

The slave controller will have simple biasing function(s) for the audio resulting in a more impressive visual response. This is shown to the right.

This implementation will **NOT** include frequency response, i.e. higher or lower tones dictate specific color/animations.

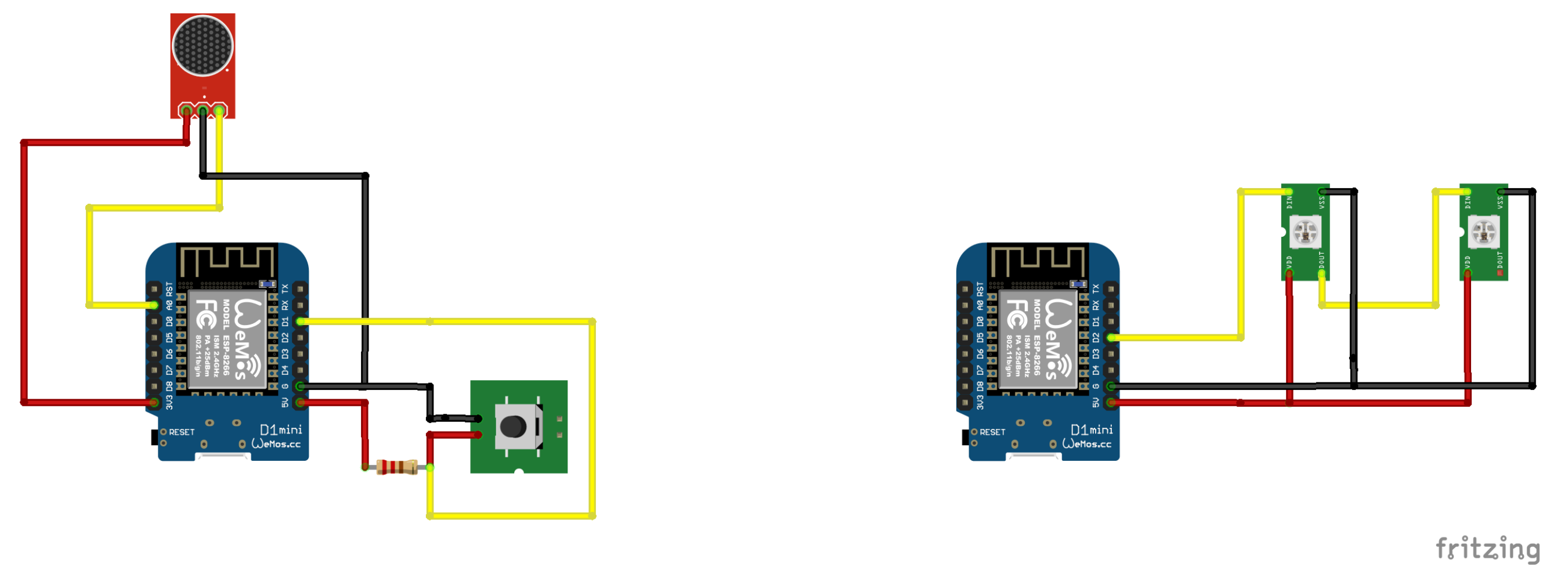
Difficulties Encountered

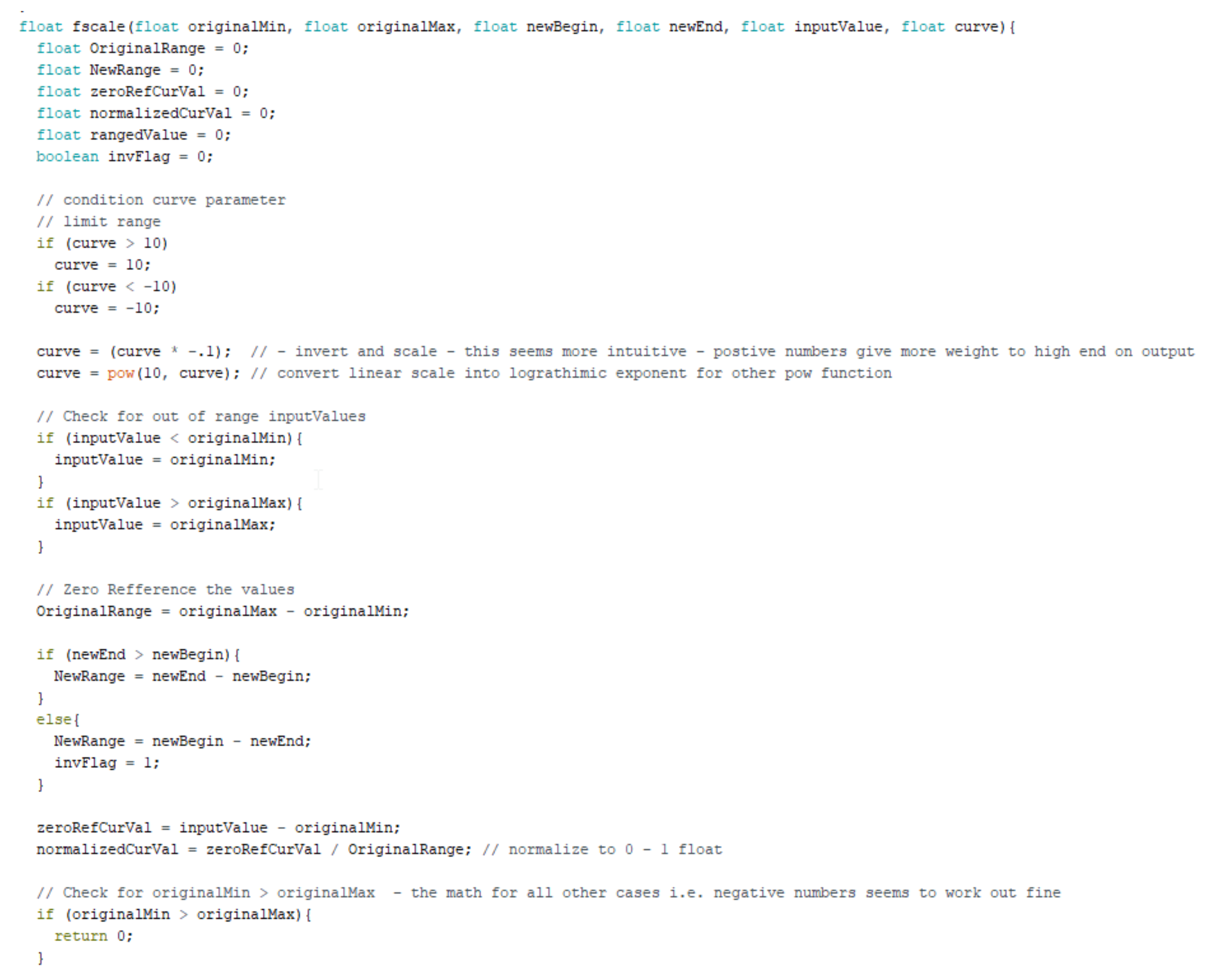
Many consumer microcontrollers operate off of 5V and also use 5V logic, meaning when the controller sends a digital 1, the signal strength is 5V, while a 0 is 0V. Since the D1 mini is designed to be a low power device, the digital high is 3.3V. The diagram below shows the work around.



The only microphones on hand when the project was being constructed had limited directional sensitivity. This led to the master controller unreliably picking up sound from the environment. To get around this, a small holder for an old phone was constructed that would direct sound into the microphone allowing for an easy demonstration.

Due to the ESP8266 being less common in projects involving WS2812B LEDs, many libraries do not support control. With the limited support of libraries, creating animation becomes a tedious process since it must be built from the ground up using simple logic commands.





```
float fscale(float originalMin, float originalMax, float newBegin, float newEnd, float inputValue, float curve){
  float OriginalRange = 0;
  float NewRange = 0;
  float zeroRefCurVal = 0;
  float normalizedCurVal = 0;
  float rangedValue = 0;
  boolean invFlag = 0;

  // condition curve parameter
  // limit range
  if (curve > 10)
    curve = 10;
  if (curve < -10)
    curve = -10;

  curve = (curve * -.1);  // - invert and scale - this seems more intuitive - postive numbers give more weight to high end on output
  curve = pow(10, curve); // convert linear scale into lograthimic exponent for other pow function

  // Check for out of range inputValues
  if (inputValue < originalMin){
    inputValue = originalMin;
  }
  if (inputValue > originalMax){
    inputValue = originalMax;
  }

  // Zero Refference the values
  OriginalRange = originalMax - originalMin;

  if (newEnd > newBegin){
    NewRange = newEnd - newBegin;
  }
  else{
    NewRange = newBegin - newEnd;
    invFlag = 1;
  }

  zeroRefCurVal = inputValue - originalMin;
  normalizedCurVal = zeroRefCurVal / OriginalRange; // normalize to 0 - 1 float

  // Check for originalMin > originalMax  - the math for all other cases i.e. negative numbers seems to work out fine
  if (originalMin > originalMax){
    return 0;
  }
```

Future Goals

Implement frequency response

Design more animations for either when the controller is in an idle state or response patterns.

Using 'Computer Aided Design' programs such as EAGLE CAD, create a custom circuit board for components to be mounted to and clean up the internals of the devices

A flaw associated with the WS2812B is that when one LED goes out, all successive LEDs also go out. There has been a revision which resulted in the WS2813 which allows for data to be passed through even though the LED is non-operational.

In Conclusion

In general, the project has reached all of the goals intended. The controllers are quick enough that with minimal code optimization the real-time effect of the visualization is believable even with the assumed latency from Wi-Fi. Both the master and slave are powered from power banks allowing for them to be picked up and freely moved around, again emphasizing this is a purely wireless communication of microcontrollers.