

# David Cain

## Changes Made and Current Progress:

Currently I have two lamps as well a central arbiter. I was planning on having a static IP that could be accessed via a browser on a separate device but due to the hardware on hand, it would be a deal of effort to have the central arbiter act as an access point as well host a webserver that runs HTML. Due this, I decided it wasn't worth the effort to get this working for now and plan to upgrade the controller for the central arbiter in the future to the dual core version ESP32. For presentation, Bluetooth will be replacing this wireless input method. The project was supposed to have a mood reactive mode that would alter the colors of the music response based on the beat of the songs, but I've decided too to get the system acting more naturally and debug some things, which won't leave enough time to also implement this.

For the central arbiter there is a microphone (MAX9814) to take in sound signals, a Bluetooth module (HM-10 BLE) for wireless control, and a small battery pack (two 18650s in parallel) being protected by the battery management module (TP4056). I put these into a small wooden box to make it portable enough to place near whatever sound source you're looking to sample.s` Below I attached a couple photos to explain these parts a little better. As of now the mode can be controlled by a small pushbutton soldered to the board or if connected via Bluetooth, you can cycle through the modes by sending strings. A benefit to this is that the module will hold the commands until the microcontroller is ready to receive, so the input is less awkward than the button input method. There are currently 6 modes of operation.

*\*Brightness is a global variable that can be adjusted when the code is uploaded to the lamp controllers\**

In order:

### **Case 0://Soft Sleep**

This mode basically just turns the lights off, but the controllers are still fully functional and do not enter a sleep mode.

### **Case 1://Sound React**

This is the showcase of the project and in this mode the central arbiter is actively sending the sound signals over the Wi-Fi UDP protocol to the two lamps connected to then display on their end.

### **Case 2://All White**

This mode tells the lamps to turn all the LEDs to a white hue.

### **Case 3://Fade**

In this mode, the light will be cycling through the color spectrum

### **Case 4://Larson Scanner**

This is a common animation in most animation libraries and its intention is to mimic knight rider. There are a couple of LEDs that cycle back and forth cycling through the color spectrum at the same time.

#### **Case 5://Rainbow**

This mode has the LEDs cycle through a rainbow effect down the entire strip.

I'll describe one lamp but they're essentially identical in components and code. The lamp has a custom perfboards that is plugged into the wall via a 36Watt 5V adapter. The lamp has a total of 119 LEDs that are individually addressable. The first one is intended to be a status LED and is soldered to the perfboard below the microcontroller, but also doubles for another purpose. The ESP8266 is a 3.3V based microcontroller which causes issues for controlling 5V logic LEDs. I found a good write up to work around this, I'll link it below. The concept behind it has the first LED acting as a buffer LED that will be powered off of 4.3V meaning the ESP8266 will be able to control this one and when this LED sends out commands it will be at a proper voltage for the remainder LEDs to be powered off the 5V rail of the perfboard. The code of the lamps boils down to connecting to the central arbiter and then waiting for commands on what mode to be in. Once the controller receives the mode, all processing happens on the local boards, this hasn't seemed to cause any issues for timing purposes, but I have noticed there is a slight color difference when the lamps are reacting to sound. This comes from the controllers having a local variable to control the hue that is incremented every cycle for the controller.

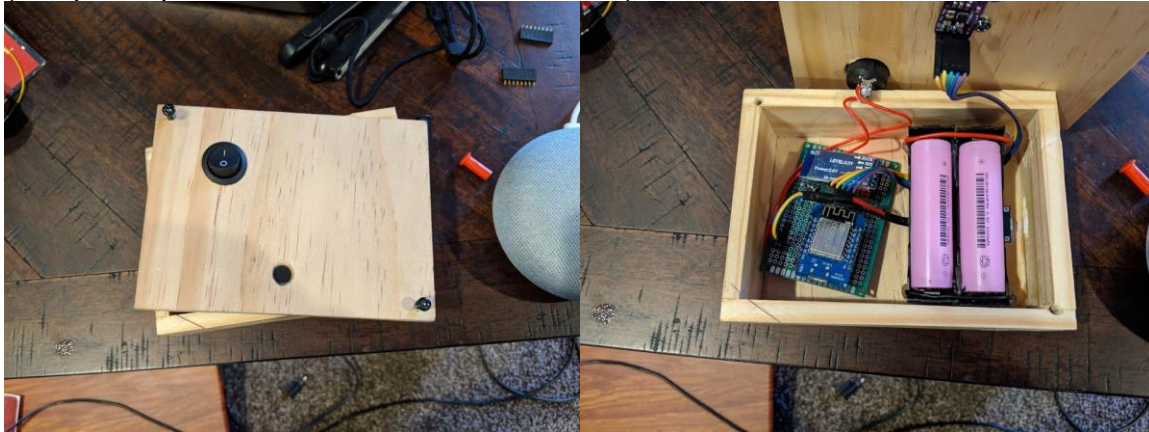
## Pictures and Links:

Write-up about the 5V to 3.3V logic conversion:

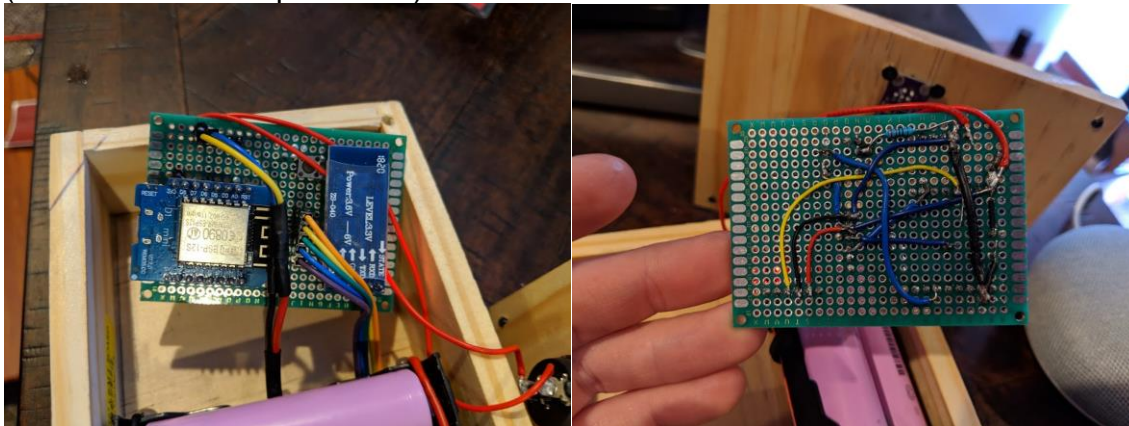
- <https://hackaday.com/2017/01/20/cheating-at-5v-ws2812-control-to-use-a-3-3v-data-line/>

Central Arbiter:

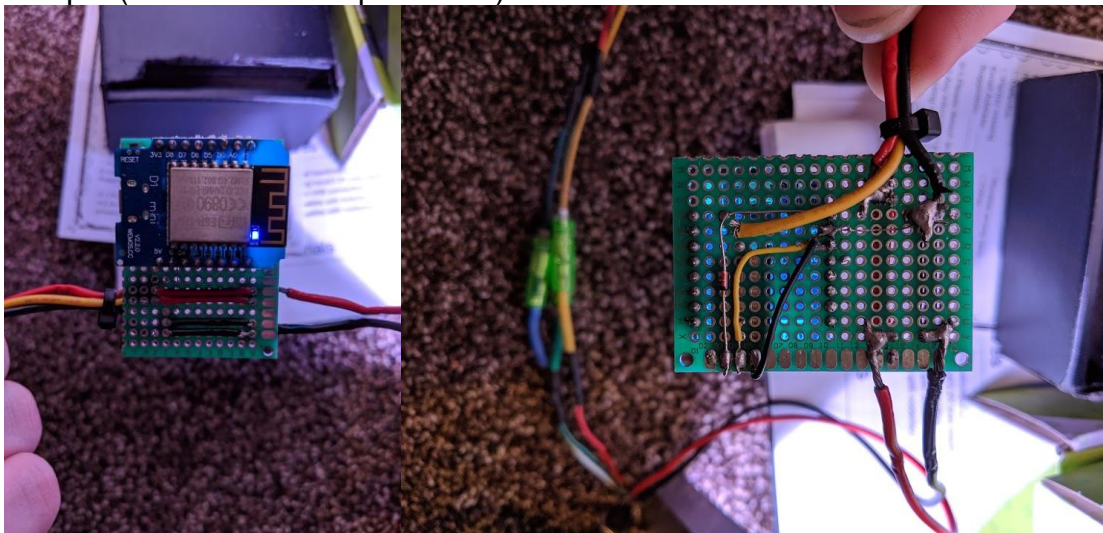
(Couple of pictures of the arbiter's container)



(Front and Back of perfboard)



Lamps: (Front & Back of perfboard)



(Rainbow mode)

