Choreographed Light Shows

By

Jake Tuohy

Joshua Schultheiss

Miguel Cabrera Del Moral

Group #37

Teaching Assistant

Vignesh Sridhar

Table of Contents

1. Introduction

1.1 Objective

LED arrays controlled by an algorithm are nothing in comparison to a professionally choreographed light show at concert. But the software used to design light shows is akin to that of professional photography; it relies on expensive hardware and has difficult-to-use software. With the resurgence of the "internet of things," we wish to bring the life of a professional light show to the living room.

We are going to make a consumer-facing light show recording ecosystem. Our goal is to design an array of lights using LEDs that can illuminate a room. Those lights will be a playable instrument by an interface similar to a soundboard as shown in figure 1. A web application will emulate a software version of this interface called a "lightboard." Light show recordings can then be mixed through our software and shared with friends.

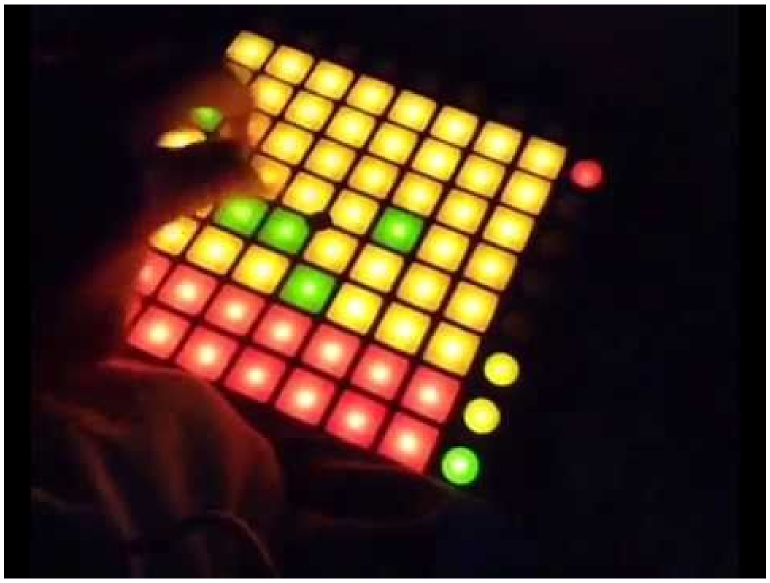


Figure 1 [9]

1.2 Background

The market for this project stems off the popular "Philips Hue" light bulbs, these bulbs have been a part of the growth in the lighting market along with the rise of cheap and bright LEDs. While controllable monocolor LED bulbs now sell for around $15, controllable multi-colored LED bulbs are still sold around $45[1,2]. This product could expand from consumers up to professionals. While an apartment is a good starting place, this could be easily scaled up to houses and small concert venues. Future renditions of the product could include hardware that would compete with large concert venues and arenas.

Our product will be competing with the controllable multi-colored bulbs, but instead of just a single bulb being placed in an existing lighting socket, the user will be able to have a greater amount of lights in any area of their home. Our design has a base of 6 controllable multi-colored LED blocks, where each one is as bright as a single multi-colored LED bulb. This will mean that the user will have a greater amount different effects to use for a more professional looking personal light show.

1.3 High-level Requirements List

- The light array is powered by a standard wall outlet.
- A web application will send data to the hardware unit over WiFi and can perform effects 9-20 listed below.
- Lights are able to update at a minimum rate of 100 Hz.

off	Turn the group to zero brightness
dim	Turn the group to a small brightness
bri	Turn the group to a large brightness
pop	Quickly brighten the group and quickly dim to original brightness
fade	Quickly brighten the group and slowly dim to original brightness
str	Multicolor strobe
col	Make group sudden color change
cyc	Cycle the lights with pop effects
slo	Slow strobe [intense white light only]
fas	Fast strobe [intense white light only]
fla	Long white flash [intense white light only]
lig	Simulates a lightning strike [intense white light only]

2. Design

The core of our project includes two key components: the software application and the hardware peripheral. Each of these include a technology stack as explained in the following sections.
The core of the application will be a MEAN full-stack web application[3]. This will allow for light show recordings that consist only of high-level transition effects to be played on the LED peripherals. Effects on the LEDs will be manually inputted and as the colors will also be inputted by the user, they will automated in the sense of a monitor that will ensure a smooth transition between colors.

2.1 Block Diagrams

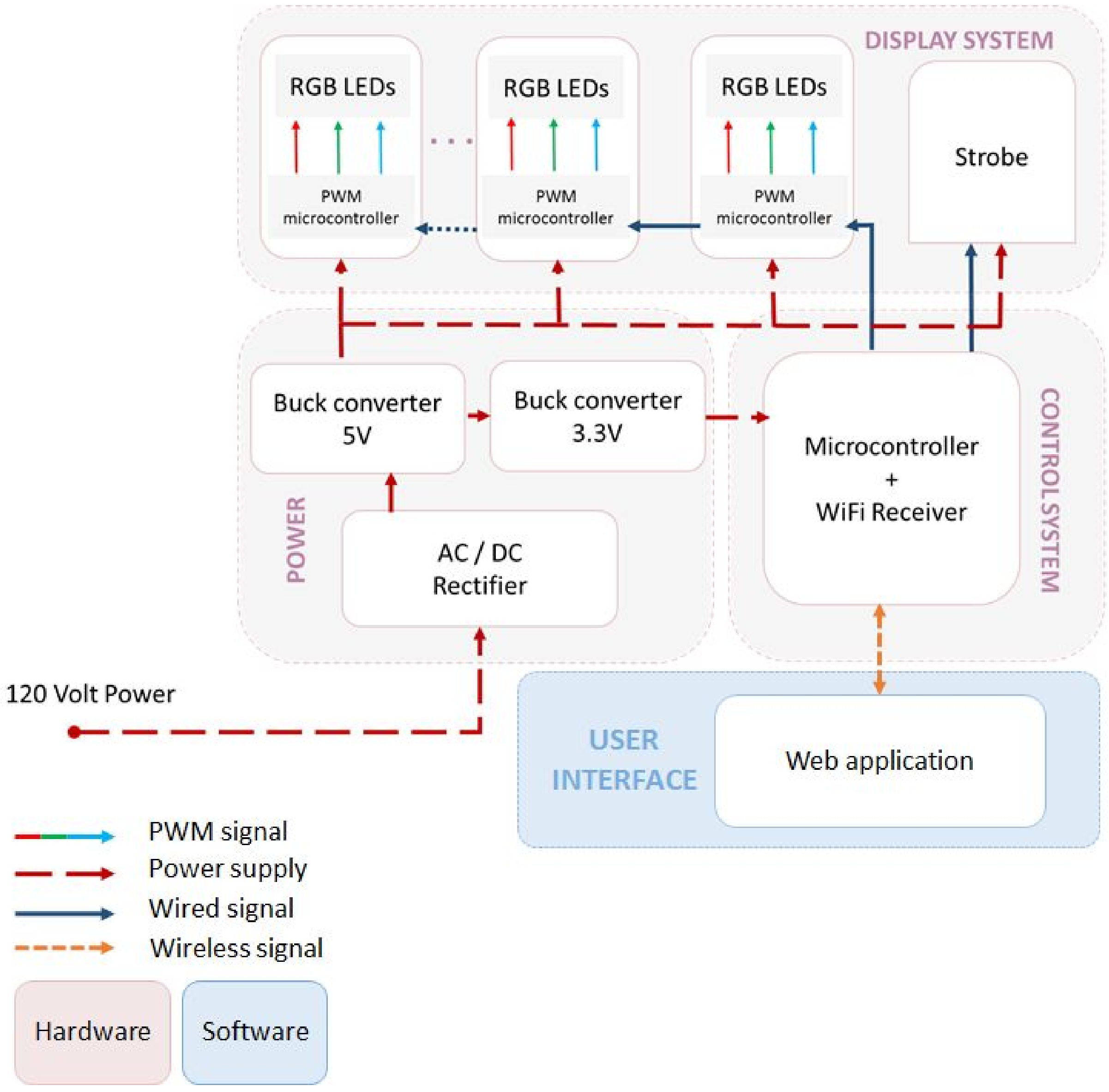


Figure 2: Block Diagram of the Hardware and Software agents of the project

There will be 6 LED blocks in the final design even though only 3 are shown in figure 2. The small dotted lines in between the two LED blocks on the left represent the blocks not shown.

2.2 Physical Designs

Software Interface

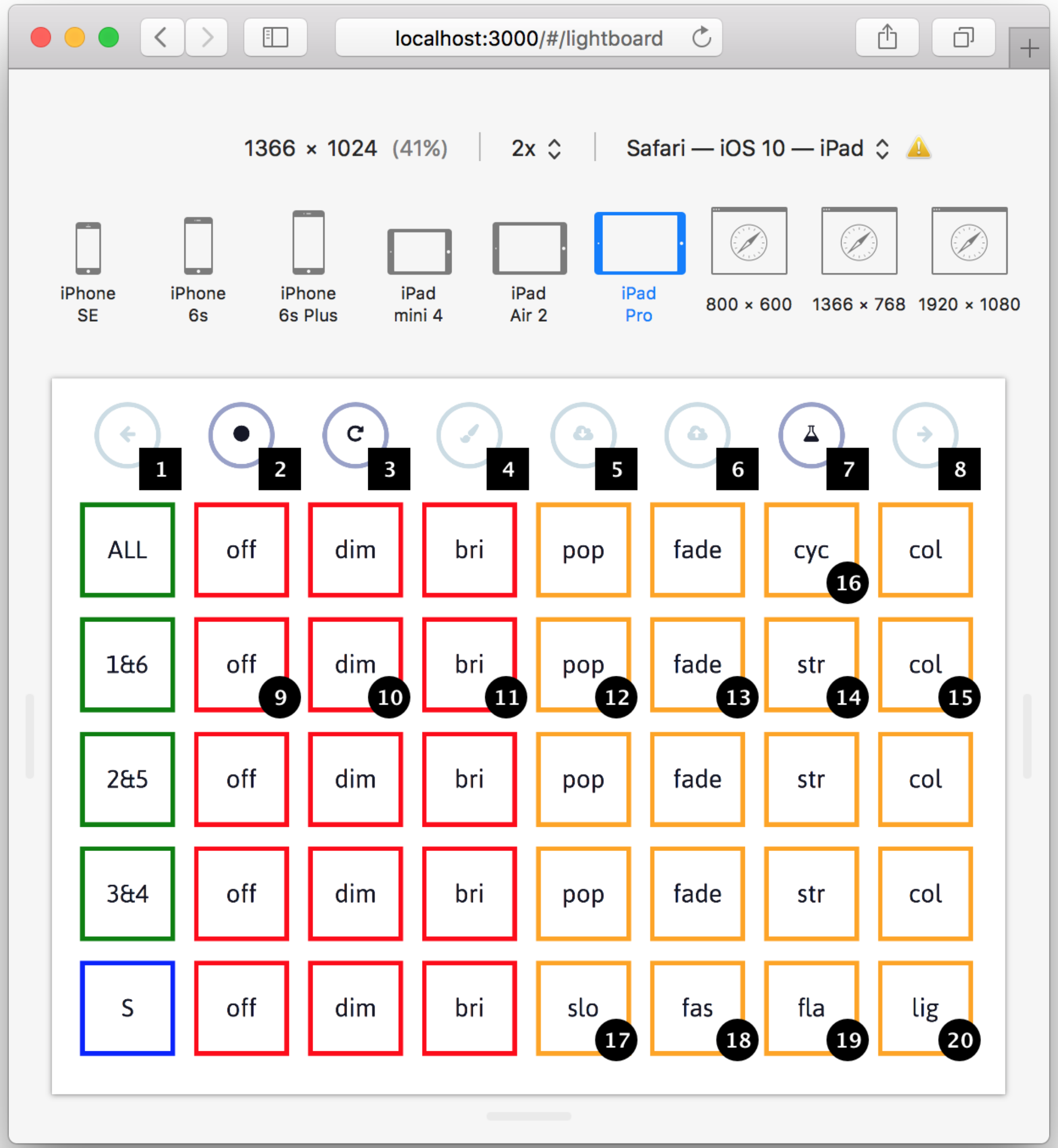
The design is centered around two main concepts: usability and scalability. The primary interface is a software emulation of a MIDI soundboard controller. By using different colors and shapes, the user is able to visualize the functionality of each button. This also makes the application easier to use for musicians that are already familiar with MIDI soundboard controllers.

Since the hardware will be scalable to include any number of lights, the application will be scalable as well. Each row of the board corresponds to a light or a group of lights. For example, if a user is using 6 LED blocks, each row could control 2 blocks. If the user wants to create more rows than can fit on the screen, the view will paginate instead of scrolling.

Figure 2 is a functional mockup of the main view, the lightboard. Descriptions of the annotations found in figure 2 can be found in table 2. A full list of all the views that will be implemented in the project can be found in table 1.

Table 1: List of views included in the application

View	**Purpose**
/#/signup	Create an account
/#/login	Log in to existing account
/#/configure	Configure the lightboard
/#/edit	Edit the current recording
/#/loops	Manage loops
/#/groups	Manage groups
/#/palettes	Manage palettes
/#/lightboard	See figure 2
/#/upload	Upload the current recording to the database
/#/download	Download a recording from the database
/#/settings	Adjust settings


localhost:3000/#/lightboard

1366 × 1024 (41%) | 2x | Safari — iOS 10 — iPad

Figure 3: Annotated interface rendered on an iPad

Table 2: Descriptions for the annotations in figure 3

Annotation	Name	Description
Orange	effect	This button will signal a light effect
Red	brightness	This button will signal the brightness level
Green	group	This row identifier corresponds to LED light(s)
Blue	strobe	This row identifier corresponds to a strobe light
1	arrow-left	Paginate left
2	record	Begin show recording
3	loop	Begin loop recording
4	assign	Assign loop recording
5	download	Download a previous recording
6	upload	Upload the current show
7	configure	Configure the lightboard
8	arrow-right	Paginate left
9	off	Turn the group to zero brightness
10	dim	Turn the group to a small brightness
11	bri	Turn the group to a large brightness
12	pop	Quickly brighten the group and quickly dim to original brightness
13	fade	Quickly brighten the group and slowly dim to original brightness
14	str	Multicolor strobe
15	col	Make group sudden color change
16	cyc	Cycle the lights with pop effects
17	slo	Slow strobe [intense white light only]
18	fas	Fast strobe [intense white light only]
19	fla	Long white flash [intense white light only]
20	lig	Simulates a lightning strike [intense white light only]

Hardware Interface

For the physical design of the LED modules, we have thought of individual blocks that can be separated or connected together into an array of lights. This design allows the user to place each light in any configuration they desire. All of the individual blocks would be wired to the main block that includes the power converters, the microcontroller, the WiFi receiver and the strobe light.

The individual LED blocks are composed of two parts as you can see in figure 3, the base and the LED light enclosure. The front part of the enclosure would be transparent and a curved shaped in order to reflect as much light as possible. Inside the light enclosure we will have the smaller microcontroller that each LED block uses along with 6 RGB LEDs. It will be two separate components connected on an axis in order to allow for movement in the direction that the light shines. The size of each block will be 120 mm in length, 120 mm in width and 50 mm in depth. The diameter of the light component will be 80 mm.

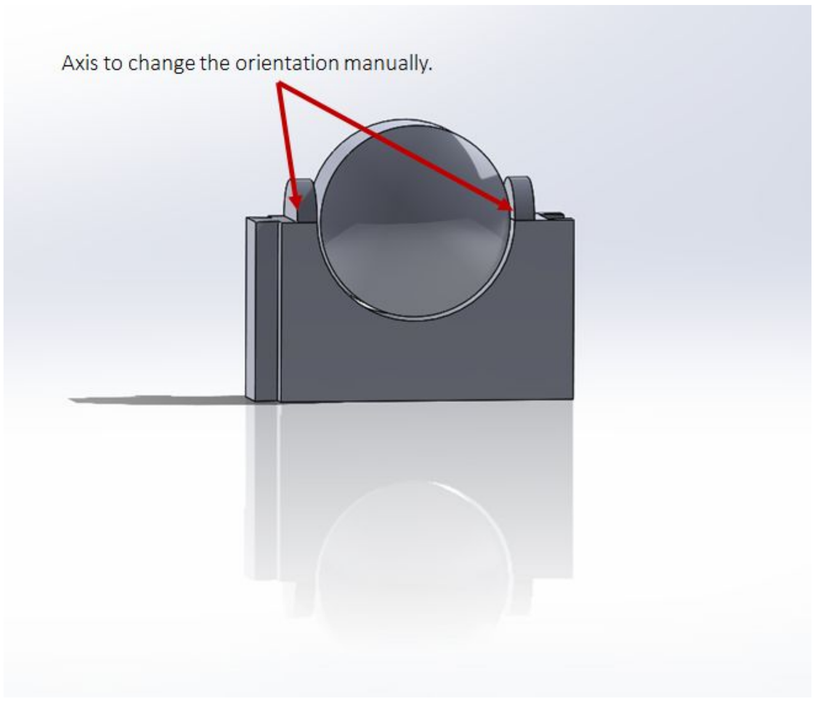


Figure 4: Front view of the main LED block.

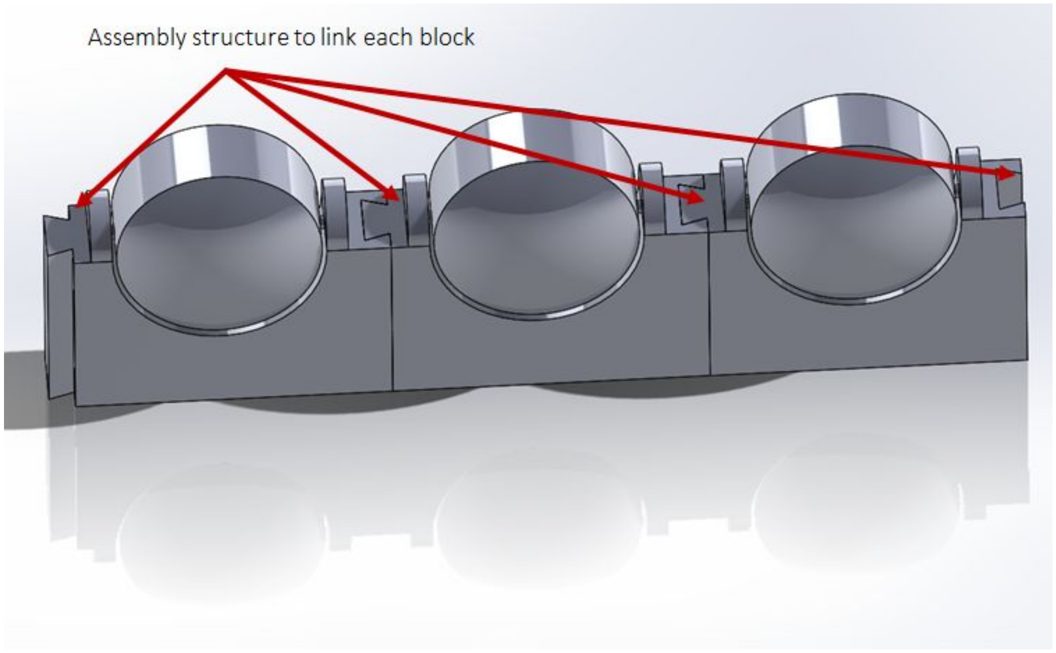


Figure 5: Front view of the blocks join together.

As it is shown in figure 5 the shape of the basement component, allows us to connect all of them together making a bigger array of lights. Joining them together with the same mechanism of a puzzle, make connecting and separating the pieces easy but at the same time is a robust way of sticking all the blocks together.

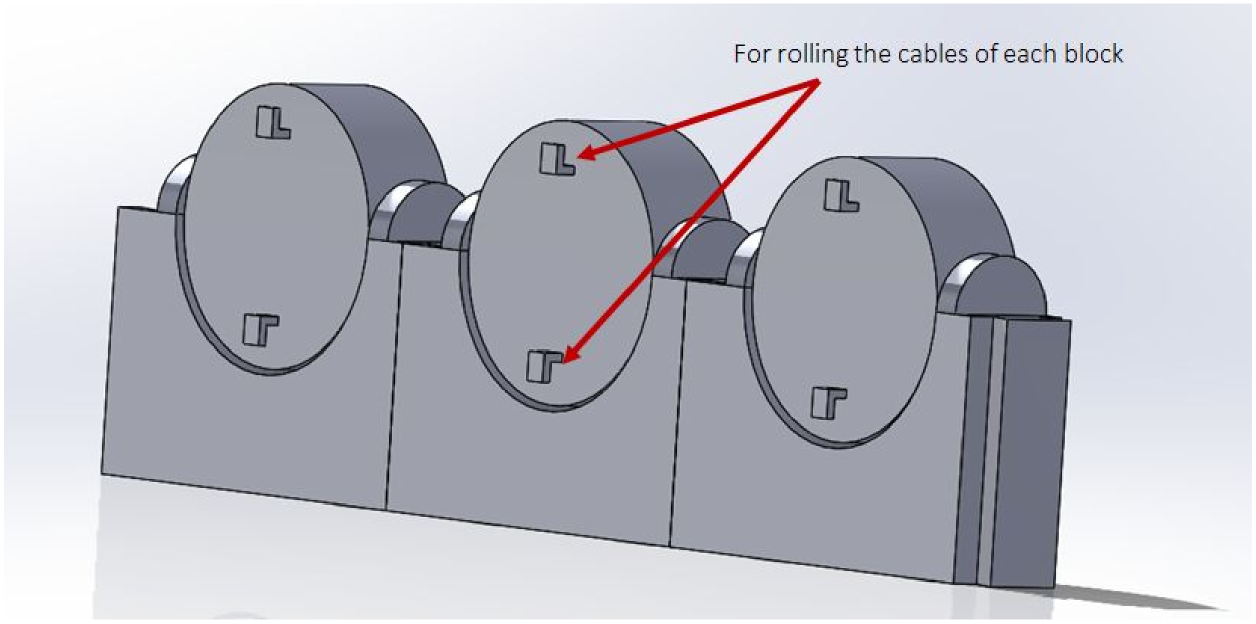


Figure 6: Back view of the blocks join together.

At the back of the block, in the light enclosure we have added two hooks in order to roll the excess of cable, when the physical enclosures are not connected. This will allow the user to have the cables in order, but at the same time have a cable long enough to place the light wherever he wants in the room.

For the speaker block it would be ideal if it has a configuration pretty similar to the LED lights blocks. But maybe in this case more space for the speaker is needed.

Finally figure 7 shows two examples of the possible configurations that this design allows. In the room of the left we see how each light is placed separate, having lights coming from each corner of the room, while in the right room all lights are together some of them pointing up and others pointing down.

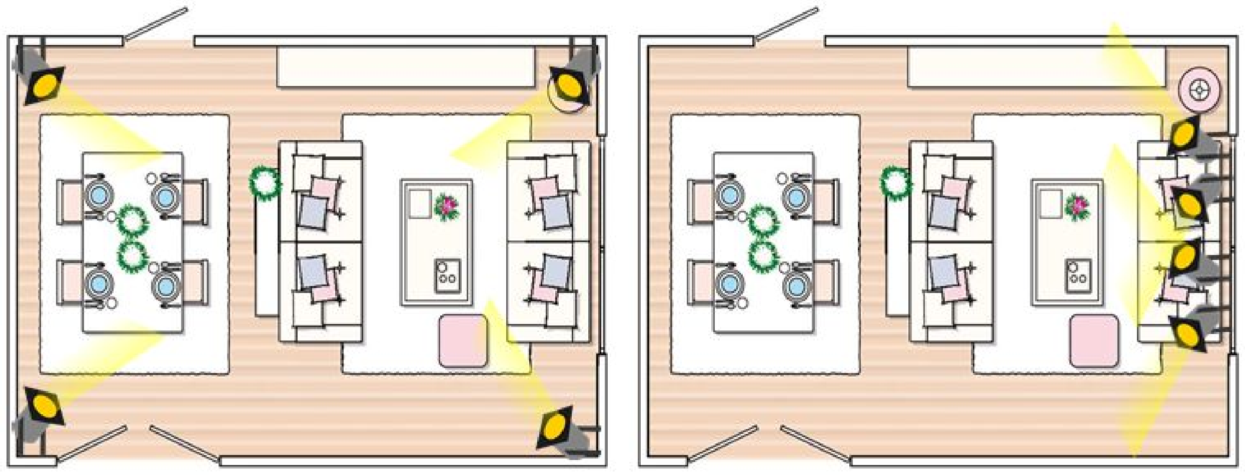


Figure 7: Possible configurations of the lights in a room

2.3 Functional Overviews

Web Application

A remote server will be the back-end of the service. This will interface with the database and web application by creating a RESTful API. The languages used for this will be Node.js and Express.js. This will be hosted on a cloud droplet.

The database will store secure user data and light shows. The light show file will be a JSON object with a list of effects and metadata. This way, shows can be edited and shared with other users. MongoDB will be database schema used.

The front-end application will communicate with the remote server (back-end) in order to save locally recorded shows and load previously recorded shows. It will be preprocessed with SCSS and Angular.js. It will also communicate with a local receiver, which will be our hardware unit that updates the colors of the lights to create effects.

Requirements	Verifications
1. Be able to add user 2. Create lightboard 3. Be able to save recordings	1. Log in user from client/console 2. Load and render original lightboard 3. Loads recordings in client/console

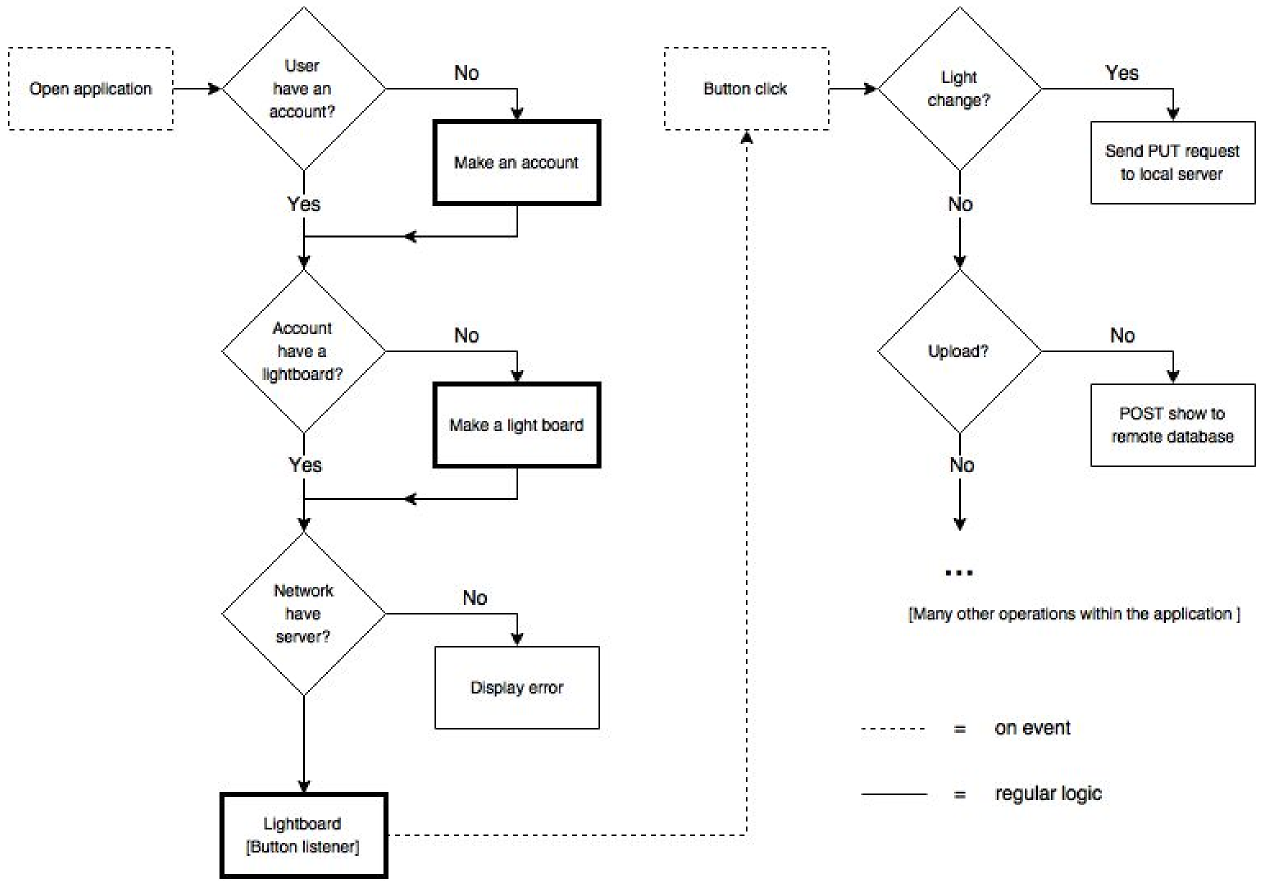


Figure 8: Software Flow Chart

Control System

ESP8266 Microcontroller

This is the main microcontroller that will receive the input signals through the WiFi. The WIFI Receiver will be used by receiving the signals stored on the server that were input by the user, and communicating to the microcontroller. We are utilizing the use of WIFI because of its larger broadcast range so the device can be controlled from a larger area in a user's house. The fallback of using WIFI instead of Bluetooth is its larger power consumption, but because we are not using a battery to power the device this issue is easily avoidable.

The data used to update the state of the LEDs will be sent from a web client. The data sent will include: brightness, color, transition time, and type of effect. An effect is a function that executes on the processor over a transition time while changing the brightness and color of the lights. That data will output to each of the PIC16(L)F1825 Microcontrollers for each LED block using SPI communication. The signal will be 24-bits: 8-bits per red, green, and blue brightness levels.

Requirements	Verifications
1. Pair to WiFi network 2. Run a web server 3. Generate digital signals at a minimum rate of 100 Hz 4. Output data to slave controllers with error rate of less than 1%	1. Connected to the internet 2. Send a HTTP request from client 3. Speed Check a. Connect microcontroller to computer b. Send 1000 commands c. Measure total time to be under 10 seconds 4. Error Rate a. Connect microcontroller to computer b. Send 1000 commands c. At least 990 correct commands are received

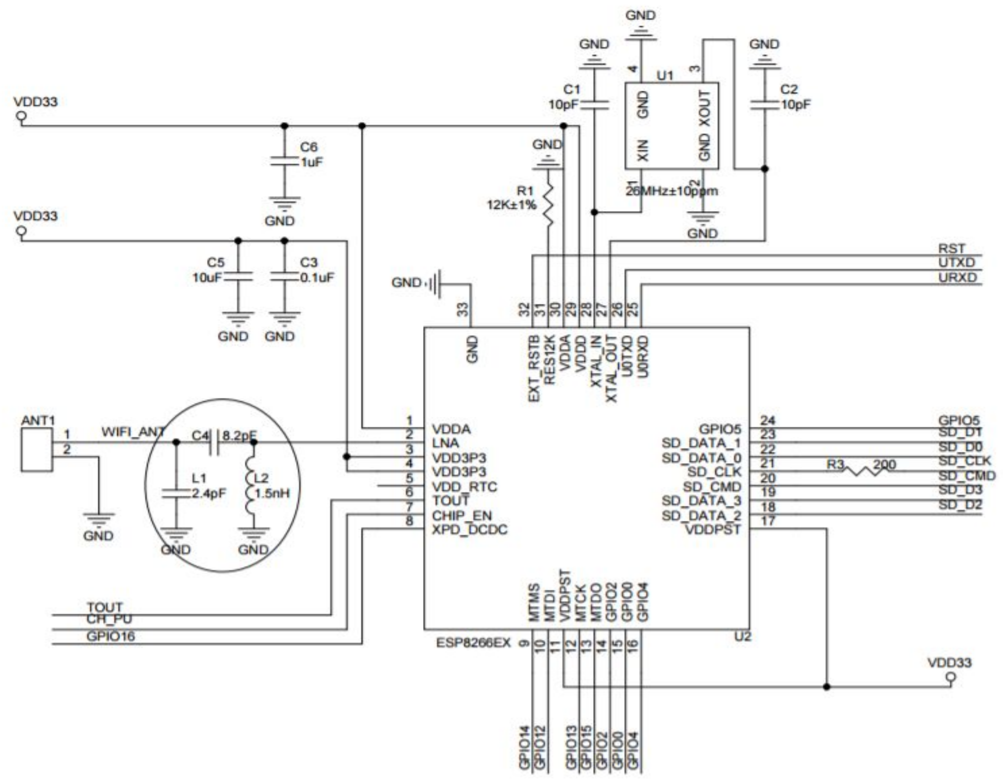


Figure 9: ESP2866 Circuit Diagram [6]

AC/DC Rectifier

This rectifier would transform the alternating current coming from the AC wall socket, to a direct current. Then it would pass this direct current through a buck converter. This component will not be designed by us, as a purchased wall wart will be a successful solution

Requirements	Verifications
1. Transform 120V to 9V ± 5%	1. Monitor voltage output over period of time to make sure voltage varies within 5% of 9 volts

5V Buck converter

The DC-DC buck converter will take the 9V DC voltage and convert it to a 5V ± 5% DC voltage source. All components used in our design except the ESP8266 microchip, are rated for a 5V input, so this will be the main voltage level used. In order to drive the PWM signal shown in figure 11 we will be using a NJM555 timing chip, which will be powered via a linear voltage regulator off of the 9V source.

Requirements	Verifications
1. Transform 9V to 5V ± 5% 2. Output current has ripple of less than 5% 3. Handle current draw of up to 6 amps	1. Monitor voltage output over standard load shown in figure 11 to make sure voltage varies within 5% of 5 volts 2. Monitor voltage output over standard load shown in figure 11 to make sure current varies within 5% 3. Measure current at maximum load of all LEDs outputting white light

Typical array size at maximum power

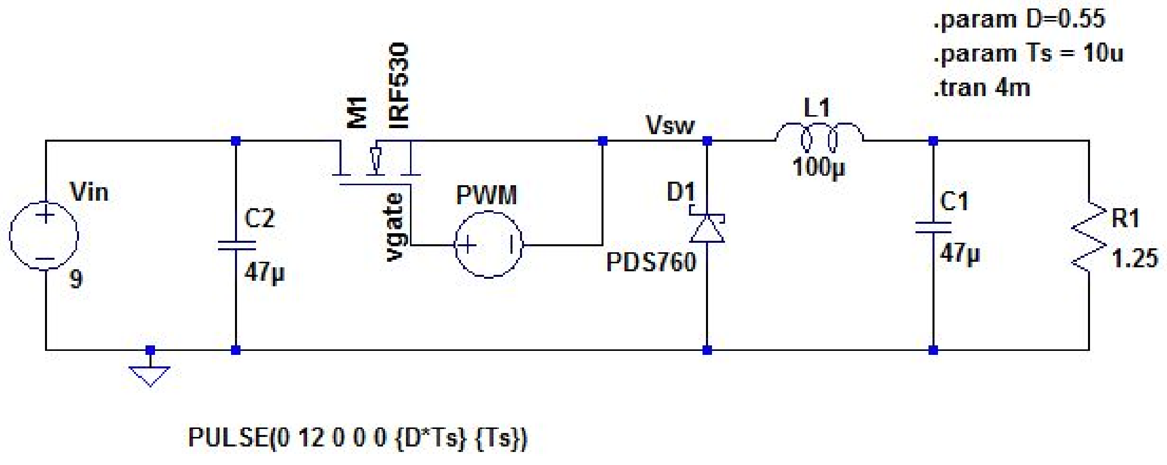


Figure 10: 9V to 5V buck converter schematics working at typical array at maximum power

In figure 11 we can see the final output voltage, and in figure 12 we have the output current, for figure 13 we have augmented the time scale in order to see the small ripple in the current, which is within our specifications.

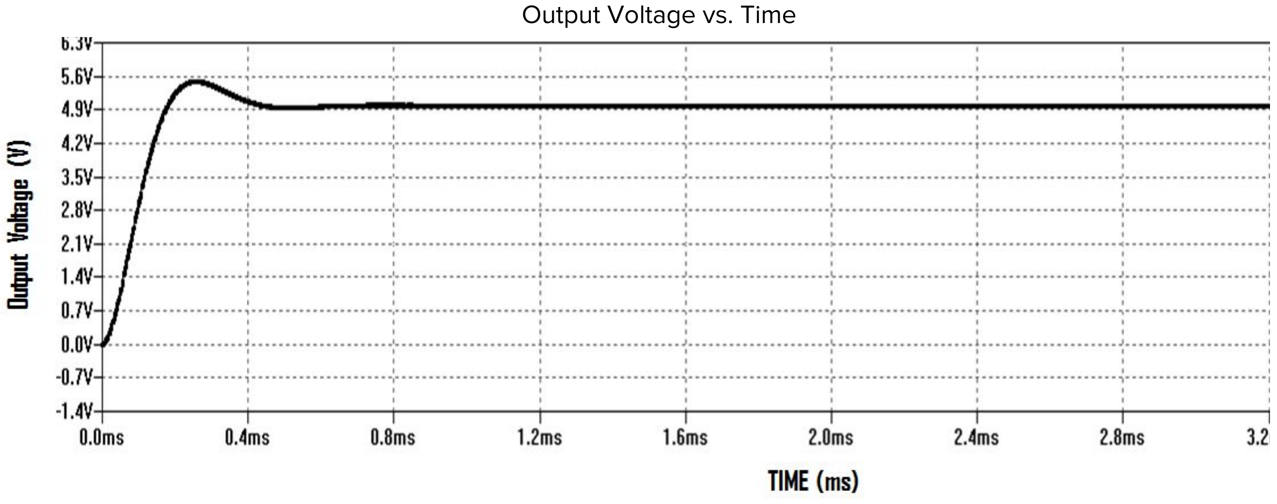


Figure 11: Output voltage of the buck converter at typical array at maximum power

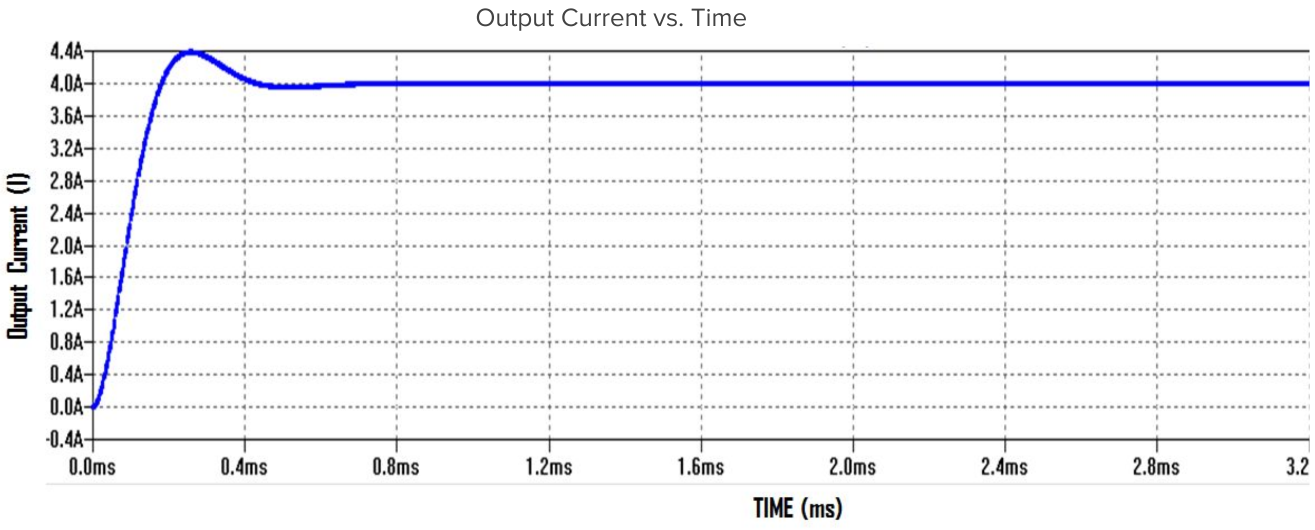


Figure 12: Output current of the buck converter at typical array maximum power

3.3V Buck converter

The DC-DC buck converter will take the 5V DC voltage and convert it to a 3V ± 10% DC voltage source. This voltage level will only be used to power the ESP8266 microchip. In order to drive the PWM signal shown in figure 13 we will be using a NJM555 timing chip, which will be powered via a linear voltage regulator off of the 9V source.

Requirements	Verifications
1. Transform 5V 3V ± 10% 2. Handle current draw of 200 amp	1. Monitor output voltage make sure voltage varies within 10% of 3 volts 2. Measure current output with a load of the microcontroller

Typical array size at maximum power

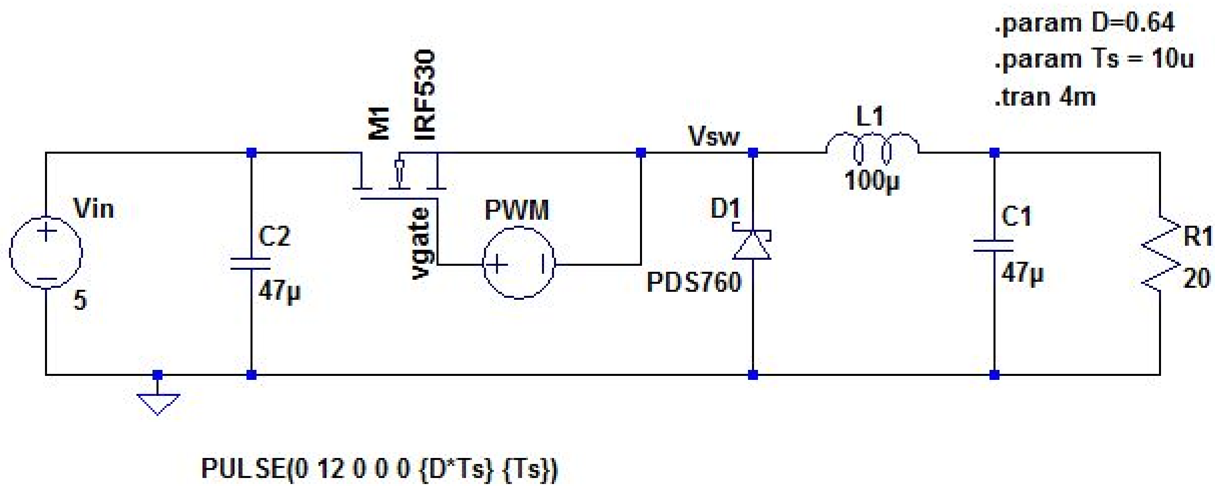


Figure 13: 5V to 3.3V buck converter schematics working at typical array at maximum power

In figure 14 we can see the final output voltage, and in figure 15 we have the output current

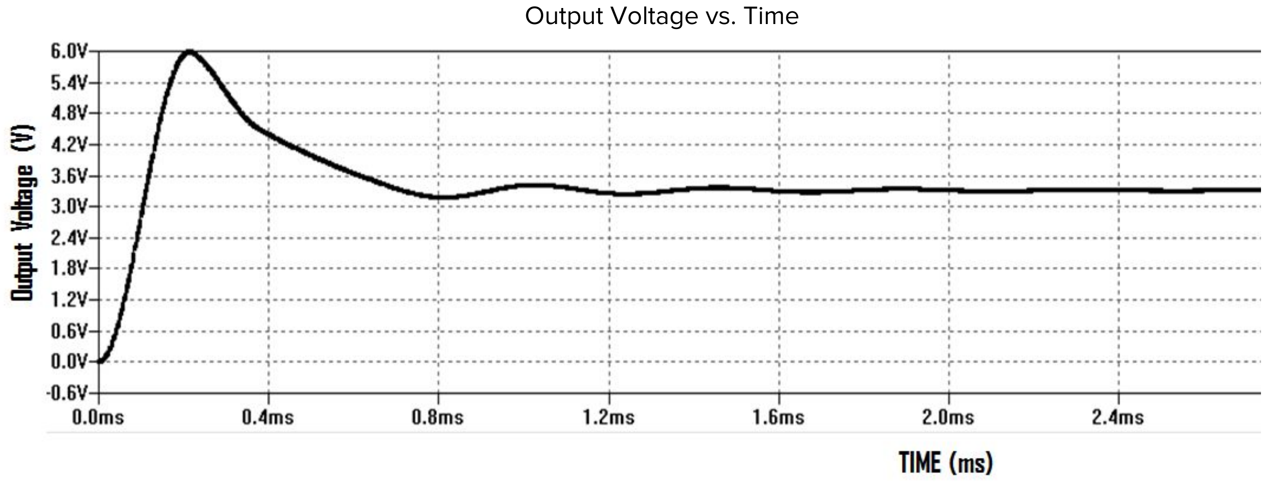


Figure 14: Output voltage of the buck converter at typical array at maximum power

Output Voltage vs. Time

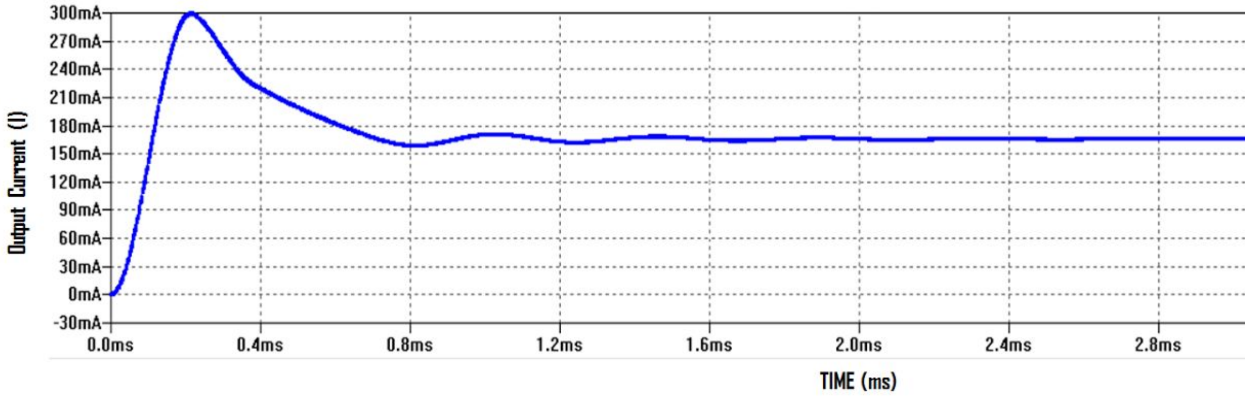


Figure 15: Output current of the buck converter at typical array maximum power

PIC16(L)F1825 Microcontroller

The PIC16(L)F1825 microcontroller has 14 pins but, we would only be using 9 of them, this is the smallest microcontroller which has up to 4 CCP modules, we would only use 3 of them, one for each RGB color[12]. One of the main reasons this microcontroller fits our project is its compatibility with the SPI communication system between ICs. We would use the SPI with a daisy chain configuration. This cascading communication between blocks allows the project to be scaled independently of the number of ports of the master microcontroller. One concern with the cascading communication use would be a time delay in the light signals reaching every block, as it can be seen from figure 17 and equation 1 this timing delay will not be an issue.

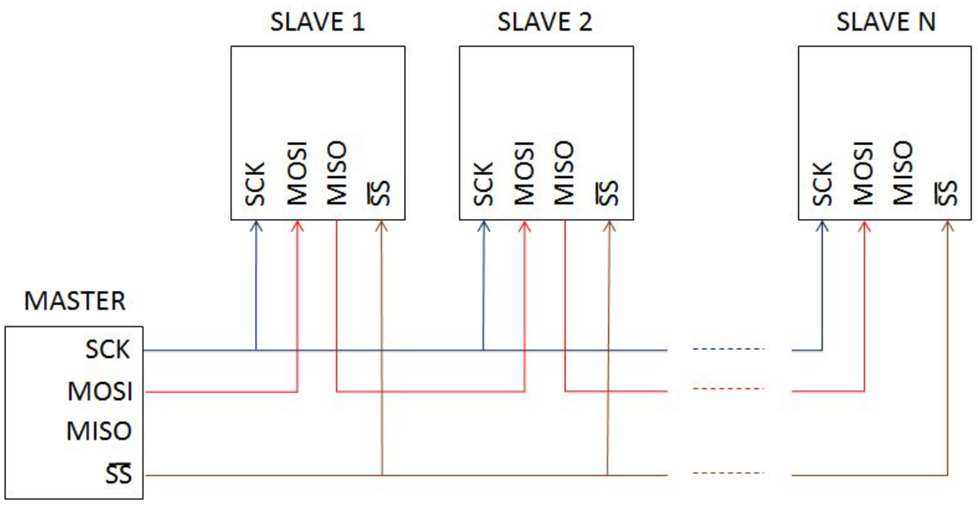


Figure 16 Daisy chain connection with a SPI bus.

In daisy-chaining what our microcontroller would do is register one command 24 bits (8 bits each RGB color) through DIN (Data In) during a given command-cycle (defined by the number of clock pulses required to clock in one command), then the microcontroller would output the same command it has register through DOUT (Data Out) during the subsequent command-cycle. So there is a DIN-to-DOUT delay of one command-cycle, our total delay would be equal to time of a command cycle multiplied by the number of blocks in the array. The slave, would execute the new command written to it on the rising edge of active-low SS (Slave Select) signal. This means that as long as active-low SS remains low, the slave would ignore the command and would output it to DOUT on the following command-cycle. If active-low SS goes high after a given command-cycle, all slaves execute the commands just written to their respective DIN inputs. If active-low SS goes high, data is not output at DOUT. This process makes it possible for every slave in the chain to execute a different command. The only problem with this protocol is that if you want to change only the value of one of the blocks you would have to update all the values in the array.

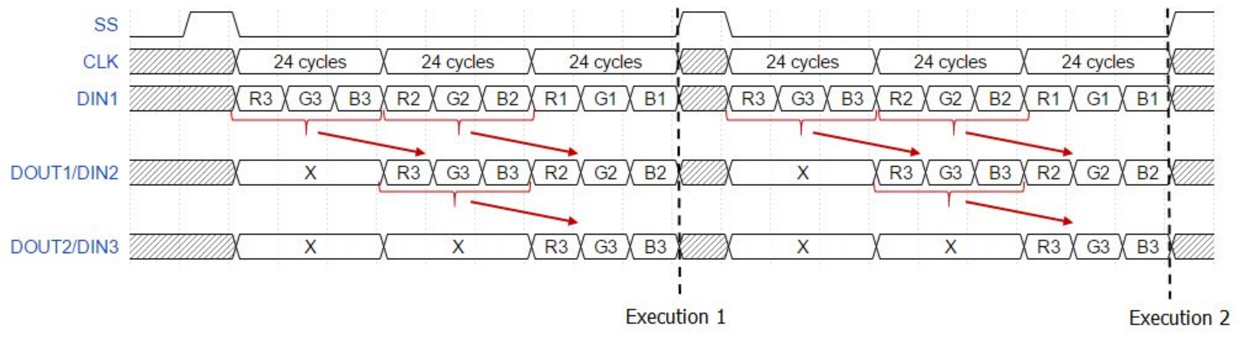


Figure 17: Time diagram daisy chain protocol

In the following equation we would calculate the total delay of our daisy chain communication protocol:

$T_{delay} = Total\ delay \qquad n_{bits} = number\ of\ bits\ in\ one\ command \qquad N_{blocks} = number\ of\ blocks\ in\ the\ array$

$$T_{delay} = (Clk_{frequency})^{-1} \cdot n_{bits} \cdot N_{blocks} = 8 \cdot 10^{-9} \cdot 24 \cdot 12 = 36\ \mu s \qquad (1)$$

As this project is meant to be used in synchronization with music, we have to make sure the delay of the lights will not affect the user's experience. The Advanced Television and Systems Committee says that the light delay should be no more than 45 milliseconds[10]. As shown in equation 1, the delay of our lights will be below this threshold.

In order to determine the duty cycle corresponding to each 8 bit value the microcontroller would perform the following equation:

$$D = \frac{L[7]\cdot 2^7 + L\,[6]\cdot 2^6 + L[5]\cdot 2^5 + L[4]\cdot 2^4 + L\,[3]\cdot 2^3 + L[2]\cdot 2^2 + L[1]\cdot 2^1 + L\,[0]}{255} \quad (2)$$

$Where\ L[x] = \text{“0”}\ or\ \text{“1”}\ and\ x = 1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7$

Requirements	Verifications
1. Receive, process and output signals to LEDs at a rate of 100 Hz 2. 255 PWM different duty ratio configurations from 0% to a 100% in each of the pins with a switching frequency of 1 kHz	1. Timing a. Send 1000 repetitive signals b. Measure total output time of LEDs being on c. Divide total time by 1000 to calculate rate 2. Use a multimeter to monitor the PWM signal at the pins, check the frequency and duty ratio for a sample of 25, randomly chosen colors. Acceptable error of 5%

RGB LEDs

Each block would be composed of 6- 259RGBM5C-013 LEDs. The LEDs will be distributed in our block as shown in figure 19.

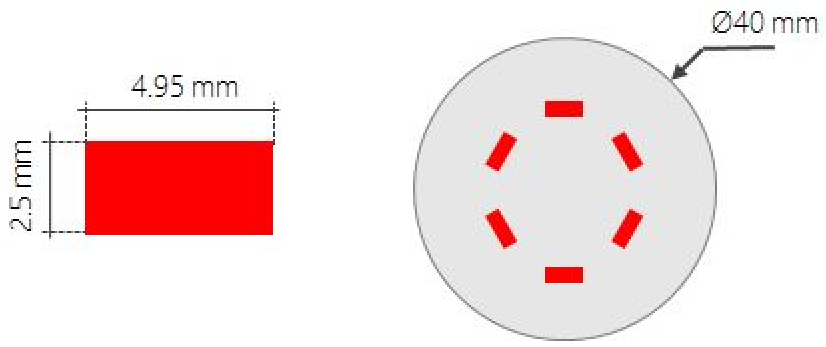


Figure 18: Size and Distribution of LEDs

We put the LEDs pointing slightly outward as the the inside of the enclosure will be reflective and a conical shape in order maximize the amount of light projected from the LEDs.

The 6 LEDs would be sourced by the 5V obtained from the buck converter, all of them would be connected in parallel each of the color pin to its color pin mates, then all of the color pin mates would be attached to a single resistor as shown in figure 19, this figure is a partial view of the complete LED block schematic, that we would show down below.

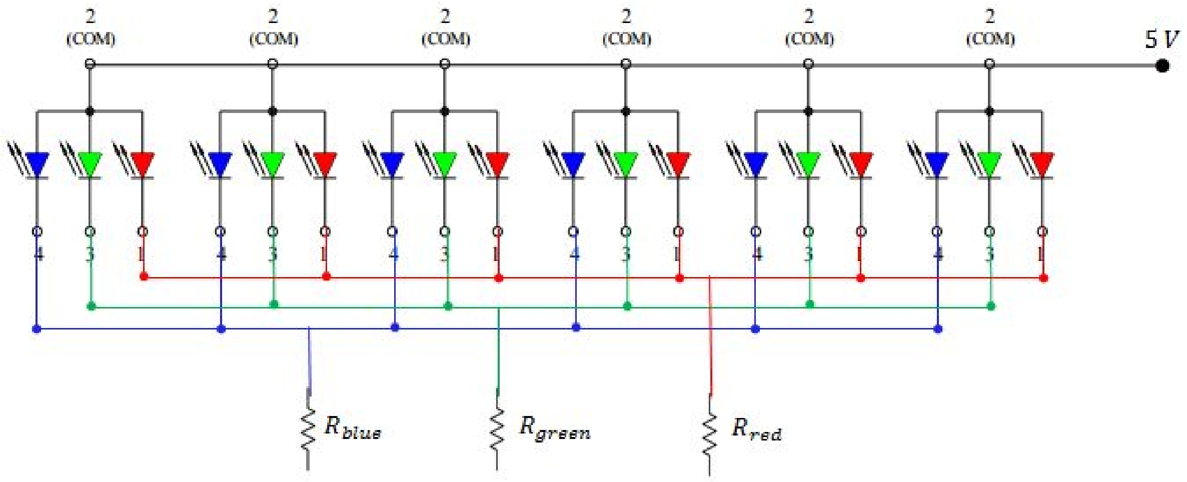


Figure 19: Partial schematic view of the connection of the LEDs

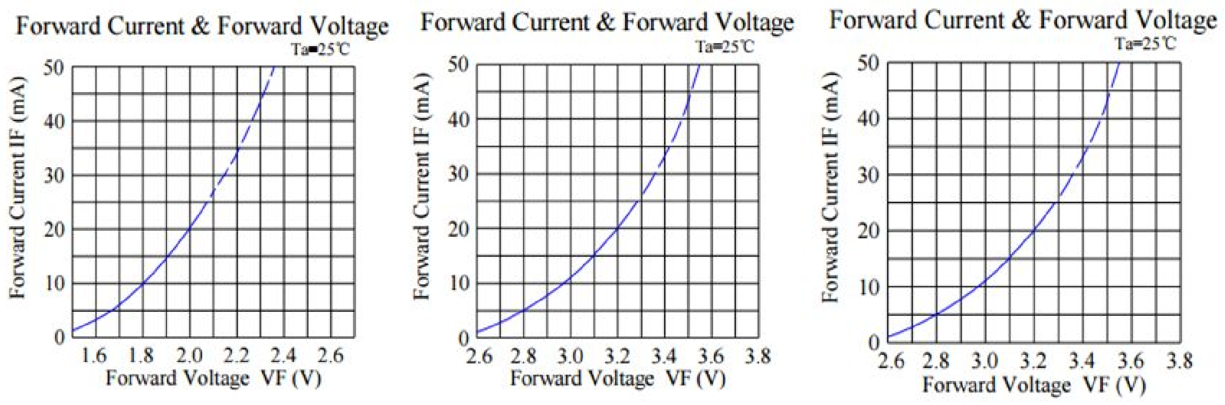


Figure 20: Forward current , forward voltage graph 259RGBM5C-013 data sheet, from left to right, red, green, blue[11].

To calculate the value needed for each color resistor, we make use of the datasheet of the 259RGBM5C-013 LEDs. As we want to run this LEDs controlling the brightness with PWM, we should determine the value of the resistors at full brightness when the duty ratio of the PWM is equal to 1, this means maximum forward current in each LED for a continuous mode, no pulse modulation. Looking at figure 20 we can see where the output is maximized for each color, where the line on the graph switches from solid to dotted, from that and equations 3,4, and 5 we can deduce the following calculations:

$$V_{color} = V_{input} - V_{forward} \quad (3)$$

$$I_{color} = (\#\ of\ LEDs) * I_{colormax} \quad (4)$$

$$R_{color} = V_{color}/I_{color} \quad (5)$$

Calculations for the red resistor:

$$V_{Rred} = 5\ V - 2.1\ V = 2.9\ V \quad (6)$$

$$I_{Rred} = 6 \cdot I_{LED\ red\ max} = 6 \cdot 25\ mA = 150\ mA \quad (7)$$

$$R_{red} = V_{Rred} / I_{Rred} = 2.9\ V\ /\ 150\ mA = 19.33\ \Omega \simeq 20\ \Omega \quad (8)$$

Calculations for green resistor:

$$V_{Rgreen} = 5\ V - 3.3\ V = 1.7\ V \quad (9)$$

$$I_{Rgreen} = 6 \cdot I_{LED\ green\ max} = 6 \cdot 25\ mA = 150\ mA \quad (10)$$

$$R_{green} = V_{Rgreen} / I_{Rgreen} = 1.7\ V\ /\ 150\ mA = 11.33\ \Omega \simeq 12\ \Omega \quad (11)$$

Calculations for blue resistor:

$$V_{Rblue} = 5\ V - 3.3\ V = 1.7\ V \tag{12}$$

$$I_{Rblue} = 6 \cdot I_{LED\ blue\ max} = 6 \cdot 25\ mA = 150\ mA \tag{13}$$

$$R_{blue} = V_{Rblue} / I_{Rblue} = 1.7\ V / 150\ mA = 11.33\ \Omega \simeq 12\ \Omega \tag{14}$$

Values are rounded up to give more available resistor values. Also since these are max values by increasing resistance slightly we go to a safer level below max.

Figure 21 shows the complete schematic circuit of 1 LED block. To control the sinking of the current with our microcontroller we would use a BS170G small signal MOSFET, this MOSFET has a maximum drain current of 500mA and a maximum drain to source voltage of 60 V, this value are enough for our project as we would be working with 5V and sinking a maximum current of 450 mA. We attach to each mosfet a pulldown resistor of 10KΩ between the gate and the source, in order to create a more true on-off switch. We have also added a current limiting resistor in between the pin and the MOSFET to ensure a maximum current of 25mA.

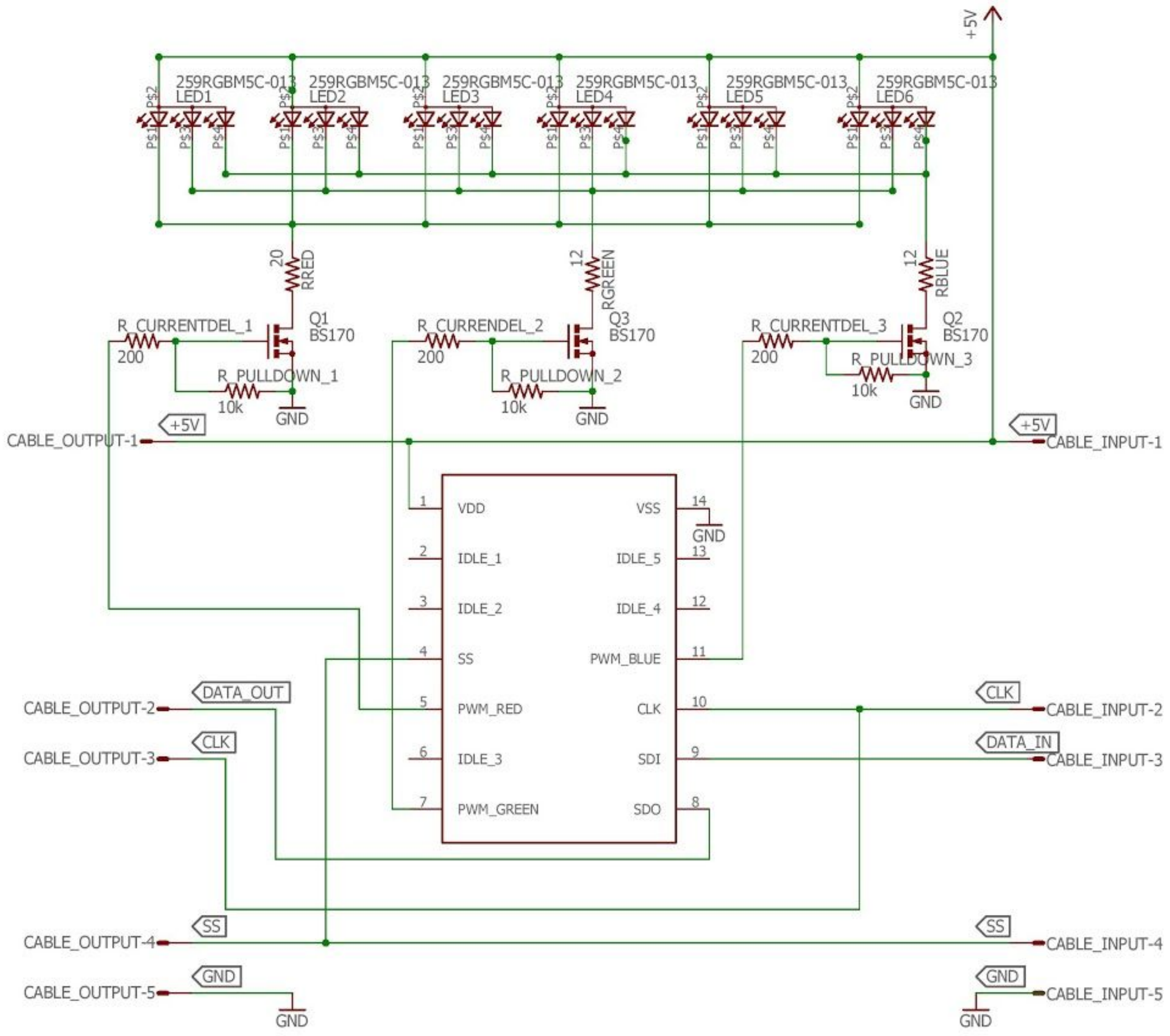


Figure 21: Complete schematic circuit of each LED block.

Requirements	Verifications
1. Luminosity of 2000 mcd ± 10% for red color at maximum current, binary code value (255,0,0), 4000 mcd ± 10% for green (0,255,0), and 1500 ± 10% mcd for blue (0,0,255). 2. Maximum current through each LED of 25 mA working at full brightness.	1. Brightness a. Send max binary value (255) for each color to microcontroller b. Measure mcd value with lumen meter 2. Current a. Send maximum values for all colors (255,255,255) b. Measure current through LED using ammeter

Strobe

The strobe would be composed of 6-FLR-50T04-HW7 LEDs. In this case the strobe would be controlled by our master microcontroller. The schematic follow to control the strobe would be equal at the one used in the RGB blocks. That means all 6 would be put in parallel and have a single pull down resistor. These LEDs also need to run off a 5V source. In the case of the strobe there is not any complex circuitry behind as it would always have the same brightness and color. It should only turn on and off at the pace of the input signal received.

Calculations for the white resistor using equations 3,4 and 5:

$$V_{White} = 5\ V - 3\ V = 2\ V \tag{15}$$

$$I_{White} = 6 \cdot I_{White} = 6 \cdot 20\ mA = 120\ mA \tag{16}$$

$$R_{red} = V_{Rred} / I_{Rred} = 2\ V\ /\ 120\ mA = 16.66\ \Omega \simeq 17\ \Omega \tag{17}$$

Requirements	Verifications
1. Luminosity of 4000 mcd ± 10% for all colors at maximum current, giving white light 2. Maximum current through each LED of 20 mA working at full brightness.	1. Brightness a. Send high voltage signal to strobe b. Measure mcd value with lumen meter 2. Current a. Send high voltage signal to strobe b. Measure current through LED using ammeter

3. Tolerance Analysis

Delay

The delay of our hardware system is 36 µs. However, a couple of unknown variables will delay our system even more. The WiFi signal will have a certain delay to transfer data to the microcontroller, and the microcontroller will have a delay to process the signal and relay the data to the next microcontroller. Since the human eye can start to notice sound and light out of sync past a 35ms delay, this will be the tolerance for the lifespan of a signal sent from a client to the controller. Other delay factors include the time it takes for the HTTP signal to be sent from the client to the microcontroller and then process the data to output a digital signal. This is difficult to estimate, but should be negligible.

4. Cost and Schedule

3.1 Cost

Total Labor = $45/Hr*10Hrs/Week*16Weeks*3Workers*2.5 = $54,000

Table 3:Materials Cost

Item	Quanity	Cost (Unit)	Cost (Total)
ESP8266 Microcontroller	1	$6.95	$6.95
7-PIC16(L)F1825 Microcontrollers	6	$1.36	$8.16
NJM555 Timing chip	2	$.88	$1.76
36 - 259RGBM5C-013 LEDs	36	$.66	$23.67
6- FLR-50T04-HW7	6	$.28	$1.68
Plastic housing	1	$25.00	$25.00
Assorted resistors, capacitors, transistors, inductors	x	$10.00	$10.00
Total Parts			$78.34

Grand Total = $54,078.34

3.2 Schedule

Table 4: Schedule

Date	Jake	Miguel	Joshua
3/2/17	Start design on 3.3 V buck converter, Complete Physical Design drawing	Design 5 V buck converter	Implement half the views for the front end interface
3/9/17	Finish buck converter design Work on main PCB design	Work on LED block PCB design	Create user endpoints and lightshow schema
3/16/17	Complete and order PCB	Complete and order PCB	Finish views for the front-end
3/23/30	Start programming of LED microcontrollers	Start programming of LED microcontrollers	Get microcontroller to receive signals from web client
4/6/17	Continue programming	Continue programming	Continue programming
4/13/17	Do timing and power anaysis across main PCB	Do timing and power anaysis across main PCB	Continue programming
4/20/17	Finish debug for mock demo	Finish debug for mock demo	Finish debug for mock demo
4/27/17	Finalize project and start final report	Finalize project and start final report	Finalize project and start final report
5/4/17	Complete final report	Complete final report	Complete final report

5. Ethics & Safety

The two largest safety concerns with our design is the power usage and the use of strobe/flashing lights. With any item on the market when using voltage converted from an AC 120 Volt outlet you run the risk overheating and burning up our project. This overheating could be a fire hazard, which is why we must take extra precaution in order to guarantee that none of the parts of our project will become to hot in order to cause a fire hazard. The second safety concern is the flashing lights which only concern a small group of people. Flashing lights can induce seizures from people who suffer from epilepsy. The highest risk frequencies are between 5-30 Hz, but because this is the most prominent range that the lights will be used at, there will be a warning on our project advising that any people that may suffer from seizures avoid the use of product in order to attempt to minimize this risk [5].

From an ethics standpoint the above risk of seizures could be in violation of the IEEE ethics code number 9 stating that we should not injure others[7]. We believe that proper labeling and warning will a way to avoid breaking this ethical code as we will do our best to let people know the risks so they can ultimately make the decision to use the product. Another thing we have to be careful is the rights to the music being played as going along with ACM code 1.6[8]. For this reason in the production of this project, songs would never be loaded into the product when sold as this may infringe on distribution laws. Instead all music would have to be played through a licensed distributor or purchased in a legal manner. This would be a way of complying to code 1.6 in effort to protect others intellectual property.

6. References

[1] "Philips - hue A19 Smart LED Light Bulb - White Only,". [Online]. Available: http://www.bestbuy.com/site/philips-hue-a19-smart-led-light-bulb-white-only/4374300.p?skuId=4374300&ref=212&loc=1&ksid=8ab71c6b-584c-449e-91fe-5aa715945c01&ksprof_id=401&ksaffcode=pg223111&ksdevice=c&lsft=ref:212,loc:2&gclid=CjwKEAiArbrFBRDL4Oiz97GP2nISJAAmJMFaE4BEC9BEsNTJs-axZquFs-1TWUbo_NtutxcwtLLMpRoCsLPw_wcB. Accessed: Feb. 24, 2017.

[2] "WiFi multi-color smart LED bulb, 2-Pack," Walmart.com, 2017. [Online]. Available: https://www.walmart.com/ip/WiFi-Multi-Color-Smart-LED-Bulb-2-Pack/50820255?wmlspartner=wlpa&selectedSellerId=0&adid=22222222227039581096&wl0=&wl1=g&wl2=c&wl3=85303117610&wl4=pla-193197925970&wl5=9022196&wl6=&wl7=&wl8=&wl9=pla&wl10=8175035&wl11=online&wl12=50820255&wl13=&veh=sem. Accessed: Feb. 24, 2017.

[3] "Home - Mongo express angular Node," Mongo Express Angular Node, 2014. [Online]. Available: http://mean.io/. Accessed: Feb. 24, 2017.

[4] IncDigitalOcean™, "DigitalOcean: Cloud computing designed for developers," DigitalOcean, 2017. [Online]. Available: https://www.digitalocean.com/. Accessed: Feb. 24, 2017.

[5] P. Shafer and J. Sirven, "Photosensitivity and seizures," Epilepsy Foundation, 2013. [Online]. Available: http://www.epilepsy.com/learn/triggers-seizures/photosensitivity-and-seizures. Accessed: Feb. 23, 2017.

[6] "File: ESP8266 specifications English.pdf," 2014. [Online]. Available: https://nurdspace.nl/File:ESP8266_Specifications_English.pdf. Accessed: Feb. 23, 2017

[7] "IEEE IEEE code of ethics," 2017. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html. Accessed: Feb. 23, 2017.

[8] "ACM code of ethics and professional conduct," 2017. [Online]. Available: https://www.acm.org/about-acm/acm-code-of-ethics-and-professional-conduct. Accessed: Feb. 23, 2017.

[9] SRC: https://i.ytimg.com/vi/rc7oV5PVxpw/hqdefault.jpg

[10] R. Hoffner, "ATSC recommends AV sync rules," TV Technology, 2015. [Online]. Available: http://www.tvtechnology.com/opinions/0004/atsc-recommends-av-sync-rules/184350. Accessed: Feb. 24, 2017.

[11] Adafruit. [Online]. Available: https://cdn-shop.adafruit.com/product-files/2739/p2739.pdf. Accessed: Feb. 24, 2017.

[12] Microchip. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/41440B.pdf. Accessed: Feb. 24, 2017.