LED Matrix Online Billboard

Dionne Pasion

April 16 2018

200344266

CS 207 – Interactive Systems

Contents

[Figures 2](#_Toc511662374)

[Introduction 3](#_Toc511662375)

[Inspirations 3](#_Toc511662376)

[Design 4](#_Toc511662377)

[LED Design 4](#_Toc511662378)

[Web Page Design 4](#_Toc511662379)

[Web Server Design 5](#_Toc511662380)

[Raspberry Pi Communication with Arduino 5](#_Toc511662381)

[Arduino Design 5](#_Toc511662382)

[Building Process 6](#_Toc511662383)

[Building the LED Matrix 6](#_Toc511662384)

[Programming the Arduino 7](#_Toc511662385)

[Programming Serial Communication between Arduino and Raspberry Pi using Python 8](#_Toc511662386)

[Webpage and Utilizing Serial Communication Function 9](#_Toc511662387)

[References 10](#_Toc511662388)

[Appendix 11](#_Toc511662389)

Figures

[Figure 1 - 100 LED circuit diagram 11](#_Toc511661786)

[Figure 2- Mock up v1 12](#_Toc511661787)

[Figure 3- Mock up v2 12](#_Toc511661788)

[Figure 4 - FSM Arduino 13](#_Toc511661789)

[Figure 5 - LED Matrix 14](#_Toc511661790)

# Introduction

The LED Matrix Online Billboard is a combinations of a couple ideas that would ultimately take a lengthy amount of time to complete if it were to be done as proposed. Therefore, the project’s direction changed to take the essence of the original idea comprised of utilizing an LED Matrix, an Arduino, Raspberry PI. Put together to produce a simple API driven LED Matrix utilizing the Arduino to actuate the information sent to it and display it onto the LED matrix with the Raspberry Pi 3 to act as the webserver and middle man between the hardware and the user.

The LED matrix is comprised of two-dimensional arrangement of individually addressable LEDs. The individual addressable LEDs make it possible to specifically target a single LED without interacting with the others. As such the LEDs serve a purpose of producing a display based on the information given to it at anytime.

Originally the project was intended to be a smaller build but more robust scale, in other words the possibilities to expand the project to be more capable was the goal of the original idea. Multiple iterations were made to produce a product that was within a fixed budget and the constraint of time with other priorities. Changes include, decreasing the number of LEDs, the scale of which the LED matrix mounted on, from two sensors such as an accelerometer and gyroscope to none. However, building the project proved to be challenging and provided insight on future builds in interactive systems.

# Inspirations

The project was initially inspired by an idea provided by friend from a Hackathon in Alberta, which was a wearable led matrix driven by an Arduino and a Bluetooth component to connect to the phone via Bluetooth. The Bluetooth device in this case any phone with access to the internet was suffice. Using the Google Maps API, it accessed current location of the user and destination, using the direction it could tell the Arduino running the led matrix to display the left or right signal if approaching a turn from the Google Map API and as well as using an accelerometer to enable brake sign to be displayed by LED matrix if the speed of the user started to slow down drastically. The uses of the product were for bicycle users that were using their bike on the road. Another inspiration came from a reddit user /u/zzzaruak who designed a 6-sided 64x64 LED cube that incorporated several sensors to be able to gather more information from the device which initially produced an LED display akin to fluid dynamic.

# Design

## LED Design

Designing was fairly straight forward as it was an array of 100 W2812B LEDs that were constructed, circuit diagram Figure 1 provides a basis of what was to be built. This design is similar to the to GreatScott’s Youtube video demonstrating building a 2 dimensional led matrix (GreatScott!, 2016).

## Web Page Design

There are two (2) version of the website mock up, one using paint to get preliminary thoughts on the design Figure 2 and then second version can be seen Figure 3 and it was to use a draw.io to construct a much better mock up with lined up elements and easier to design interface when it comes to implementing. The design for these are in help from a fully developed program by TylerTimoJ which to replicate but instead of an executable built on Unity to a website that can accomplish close to the similar functionality. As the source code was not available, an interpretation of the design and the underlying back end had to be engineered.

## Web Server Design

The webserver was design to take POST information from the front-end and package it up into a JSON object then to be decoded and sent to the Arduino. The language to be used was decided to be Python using Django as the framework. The design pattern utilized was MVC (Model-View-Controller) due to its simplicity and that it was something that was used in the past for other projects so it was familiar territory. Python was a language I wanted to use because it was something I was learning and it provided a good space to use it in a web development project when previous I would have chosen C#, PHP or Node.JS to handle the back end.

## Raspberry Pi Communication with Arduino

The design for the website was finished, the backend programming was taking the data from the webpage in this case a JSON object which holds the array of values, the led index, the RGB values which are converted from HSL to RGB utilizing a HSL to RGB algorithm(mjijackson, 2008) and pushing that data to the Arduino to be displayed was the communication between the two devices.

## Arduino Design

The Arduino serves as the actuator between the information from the Raspberry Pi and the LED matrix display. Couple iterations of the design, the first design was simply sending an array of the LED information and displaying it. Second iteration was inspired by adding a queue system on the Arduino to hold information so the Raspberry Pi can keep sending information which proved to helpful but memory capabilities was obviously lacking if I held information for too long without removing it. Improving the polling function from serial communication was my second attempt at creating a better way to handle information coming in and that can fit on the serial connection as the communication link only allowed 16 bytes to be sent at a time and I did not have a way to preserve information if any was corrupted on the way Robin2 from the Arduino Forums provided a simple by eloquent solution to my problem (Robin2, 2014). Robin2’s solution was able to poll the serial connection until there was new data available and then run my function to utilize the LEDs which means I needed 4 integers to enable the LED function. The forth design was to create more efficient solution where I waited until there was enough information to run all the LEDs so I needed to look for max number of LEDs to be reached when receiving data so it will not display the array if insufficient information was sent

# Building Process

## Building the LED Matrix

The LED’s chosen are the WS2812B LEDs (WS2812B intelligent control LED integrated light source.) which are individually addressable LEDs. The LEDs when cut is 1cm in length and approximately 5 cm in width. The LED matrix is 10x10 LEDs which equates to a 100 LEDs required. The specifications require the plywood which would house these LEDs to be at least 50cm in width and 10 cm in length and requiring about 1 cm between each LED increases the minimum specification to 70cm for width and 30 cm width. The plywood was cut into those minimum specification and marked where each LED would be housed so a grid of 100 rectangles marked and the middle of each rectangle was marked as well. The LED are then cut to expose the copper padding so that a wire can be soldered on. Each LED was hot glued to the center of each unique rectangle which can be seen on Figure 5.

Using a drill, a hole was drilled through the plywood using a 1/8” drill bit for each of the copper pads of the LEDs which is 600 holes. The holes provide a way to hide wiring on the underside of the plywood and keep away from the front. Then each pad had a 3cm copper wire solder on and threaded through the hole. 20 25mm in diameter copper wires were cut to 70cm length and hot glued on the underside of the plywood with one above each row of LEDs and one below the row of LEDs. The copper wire served as power rails to deliver 5v and GND to each row of LEDs in parallel instead of utilizing the built in serial circuit that the LEDs provided by connecting each associated copper pad, 5v to 5v, GND to GND and data out to data in to each other in a linear construct. Each GND wire was soldered to the closest designated ground rail and the same was done for the 5v wire and the 5v rail. The data out of each LED was solder to the next LED’s data in to the right, and after each row the last LED data in is connected to the first LED of the next row. Then utilizing insulated copper wires each was cut to be able to connect the rails together so all the ground rails were connected and the 5v rails were also connected. The first most rails were connected to a micro B USB breakout board to their associated 5v and GND paddings. The micro B USB served as a way to utilize an electricity source such as the Mains electricity instead of a battery to power the LEDs. The first LED of the matrix Data in was connected to the Arduino’s pin 4 as this was specified when the program was implemented. The entire build process mimicked the process done by GreatScott!(GreatScott!, 2016)

Testing the LED Matrix utilized example was uploaded to the Arduino where each LED was lit up individually in cascading effect this made sure the LEDs were working as well as providing a demonstration of being individually addressable as the example utilized a for loop that treated the LED matrix as an 1D array.

## Programming the Arduino

Programming the Arduino based on multiple iteration of designs where each problem encountered was thought to be a design flaw so it made it back to the design stage to be redone or improved as the previous implementation’s design gave insight of the issues. So, this will be done with the final version as the focus as that design was given the most thought and time. Initially the program just needed to read integers sent via serial and broken down to the four parts to actuate the Adafruit NeoPixel library(). The programming for this part required a lot of debugging to get the input being read properly by the Arduino using Serial monitor to manually test then using Python to develop a test script to test serial input as well as send as many arrays as possible to determine the processing time and amount of data that can be processed.

The test resulted in after 48 integers were sent the Arduino required approximately 500ms to process the next 48 integers without corrupting them due to serial communication bottleneck and due to trying to make an efficient polling function that was as closed to a nonblocking polling function. Required the script to be multi threaded and used a timer to sleep the script for 500ms to enable brief gap for the Arduino to process the current array of information. The threaded functions made it so the program required a simple lock to stop additional threads from accessing the COM when one thread was already sending data. This was removed as the number of users for this program would be one at most and because properly creating thread pool and managing them all was not part of the scope of the project.

## Programming Serial Communication between Arduino and Raspberry Pi using Python

Utilizing the PySerial library to handle the back-end of the communication between the Raspberry Pi and the Arduino. A windows computer was substituted for the Raspberry Pi for debugging. The program was simply taking a simple list and converting it to fit the design specification of the Arduino. The design specified a start and end marker surrounding each element of the list in this case were integers and send them to the Arduino as a set of four(4) as this represented the needed information to start up utilize the NeoPixel’s LED set function. The same principles using the test data when developing the python script to test the Arduino where required a window of 500ms to provide enough processing time for the Arduino to process 48 elements before continuing the sending process.

## Webpage and Utilizing Serial Communication Function

The webserver was simply built to host the single webpage built using the MVC(Model-View-Controller) software pattern. The view was built to replicated Figure 3 and it uses JavaScript to

# Other Considerations

The development was rushed as there was a lot of designing and testing of the multiple devices as to how they work and why they work. The exploratory phase lasted longer than expected and thus

# Conclusion

# References

References

GreatScott!. (2016). How to easily create animations for your LED matrix. Retrieved from <https://www.youtube.com/watch?v=o6_UYb6I4x4>

mjijackson. (2008). *RGB to HSL and RGB to HSV color model conversion algorithms in JavaScript&nbsp;*. Github: Retrieved from <https://gist.github.com/mjackson/5311256>

Robin2. (2014). *Serial input basics*. Arduino Forum: Retrieved from <https://forum.arduino.cc/index.php?topic=288234.0>

WS2812B intelligent control LED integrated light source. Retrieved from <http://www.seeedstudio.com/document/pdf/WS2812B%20Datasheet.pdf>

# Appendix

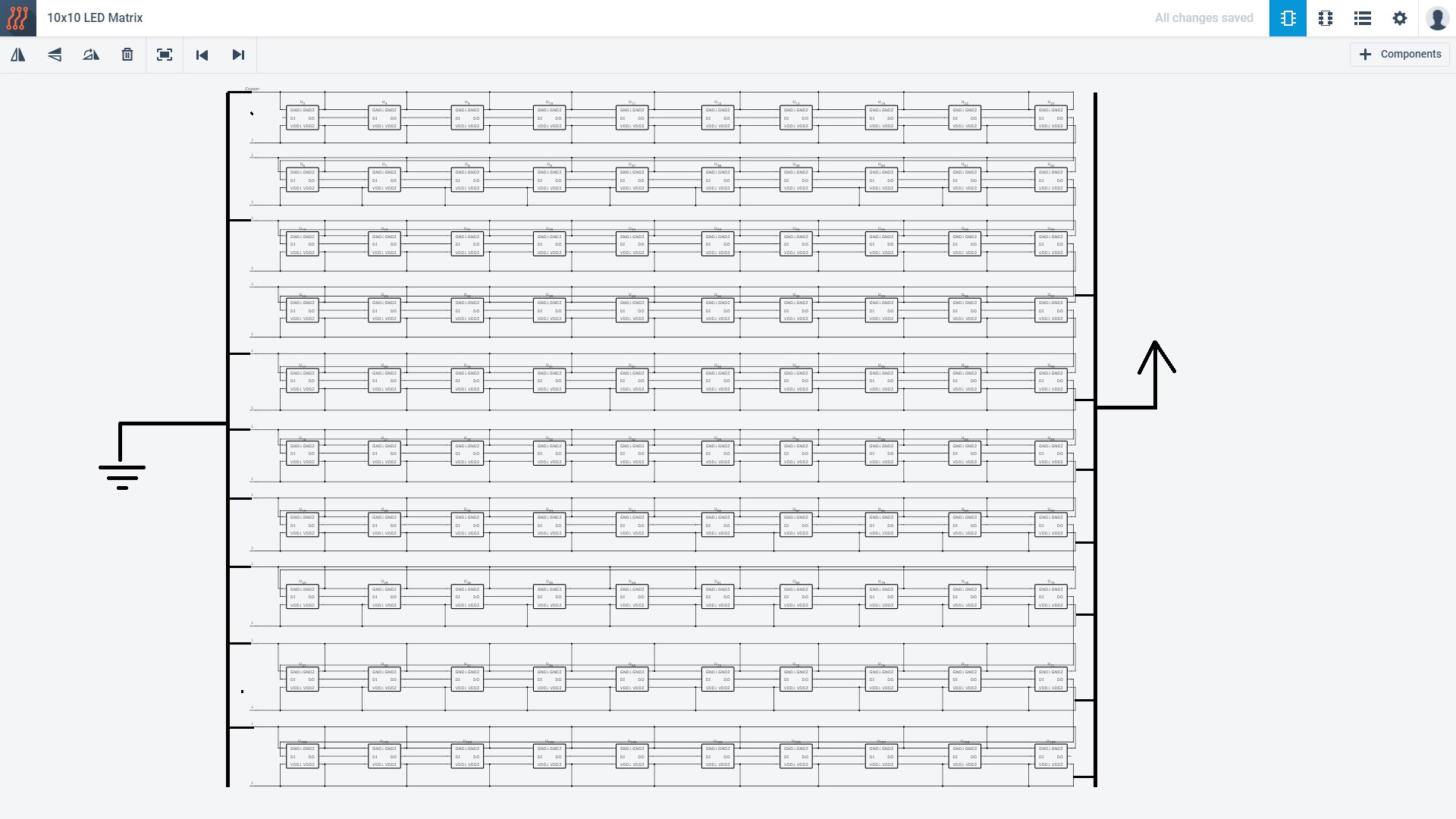


Figure 1 - 100 LED circuit diagram

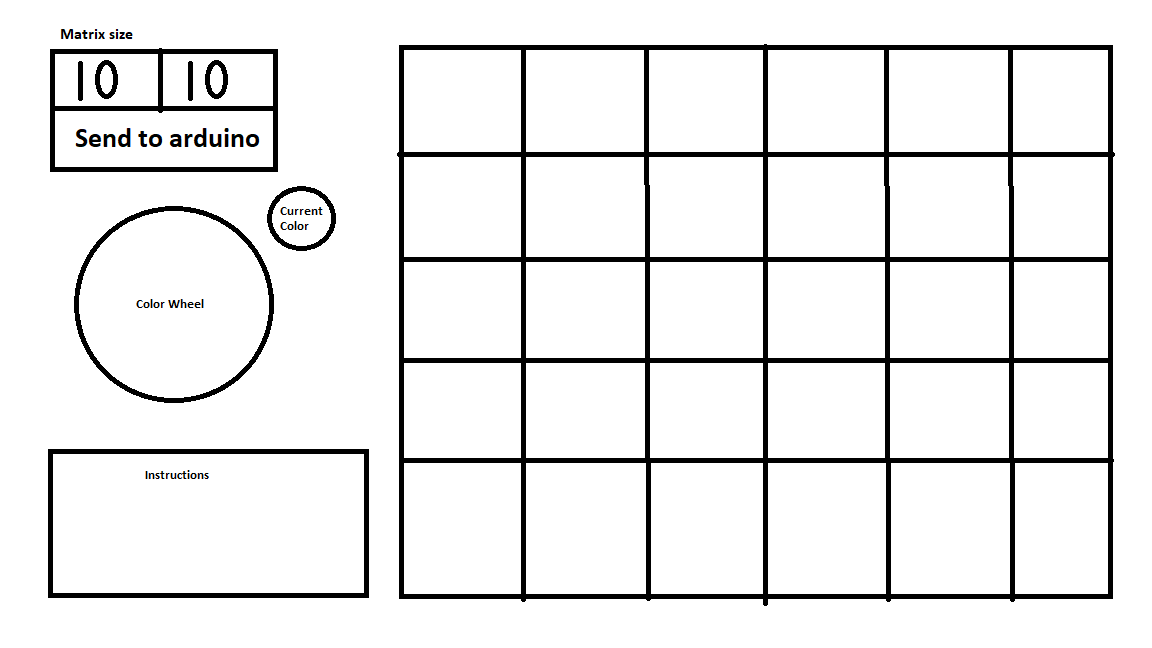


Figure 2- Mock up v1

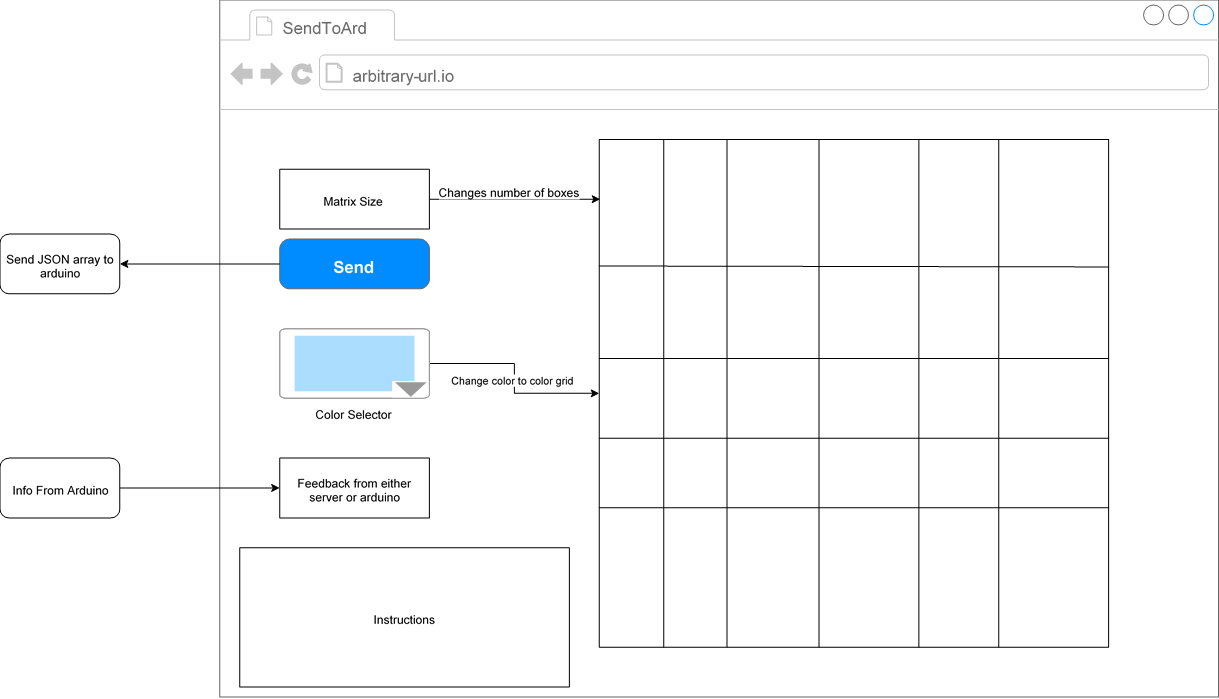


Figure 3- Mock up v2

Check for any

Serial Data

Check

no starter marker

Check for

marker

character

Start

Get character

Not end marker

is End marker

Check for end

marker

Insert current digit to

current index of array

Current index < 4

Current index >= 4

Increment

current index

Convert to digit

Set up LED

reset current index to

0

Show LED

Figure 4 - FSM Arduino



Figure 5 - LED Matrix