6-15-2017

Waterless Light Fountain

Lucas Villalba Eirea
Santa Clara University, lvillalbaeirea@scu.edu

SANTA CLARA UNIVERSITY

Department of Electrical Engineering

I HEREBY RECOMMEND THAT THE THESIS PREPARED
UNDER MY SUPERVISION BY

Lucas Villalba Eirea

ENTITLED

Waterless Light Fountain

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

BACHELOR OF SCIENCE
IN
ELECTRICAL ENGINEERING

[illegible] 6/15/17

Thesis Advisor(s) date

[illegible] 6/15/17

Department Chair(s) date

WATERLESS LIGHT FOUNTAIN

By

Lucas Villalba Eirea

SENIOR DESIGN PROJECT REPORT

Submitted to
the Department of Electrical Engineering

of

SANTA CLARA UNIVERSITY

in Partial Fulfillment of the Requirements
for the degree of
Bachelor of Science in Electrical Engineering

Santa Clara, California

2017

Waterless Light Fountain

Lucas Villalba Eirea

Department of Electrical Engineering
Santa Clara University
2017

ABSTRACT

The Waterless light fountain is born from the need for art projects for Franklin street in Santa Clara University than is going to be remodeled into an arts and history paseo, one of the ideas discussed was to determine if there may be a way to develop a waterless light fountain that could be displayed either on a temporary or permanent basis. In this paper, we will explain the design process and build of the fountain, using Neopixels (RGB LEDs) to generate the light used as the main art element. An Arduino is used as the control unit, controlling the Neopixels and receiving input data from the user interface, composed by arcade buttons and a LCD screen with a mounted touchscreen, where users can interact with the fountain by choosing between four different modes of operation, including light shows where LEDs light up in preprogramed sequences, some games as Simon-says or choosing the color of each LED with the help of the touchscreen and light shows combined with music through the performance of a spectrum analyzer controlled by a raspberry pi where the LEDs dance with the music. In the end, the fountain is going to be placed in the engineering building as the design is not minted to endurance the outdoors conditions.

Keywords: Waterless light fountain**,** Light show, Spectrum Analyzer

TABLE OF CONTENTS

Chapter 1- Introduction

1.1 Motivation

The Waterless Light Fountain comes from the necessity for arts projects for Franklin Street at Santa Clara University, that is being remodeled into an Art and History paseo, so one of the ideas discussed was to determine if there may be a way to develop a waterless light fountain that could be displayed either on a temporary or permanent basis. The waterless light fountain uses light as its main element as opposed to conventional fountains, that use water, which has some advantages such as saving water and making it easier to install as there is no need for pipes and pumps. It could be also be used as an alternative for water fountains in places where there is a lack of water, or the conditions are either too hot so it is necessary to add water to the fountain all the time due to high evaporation of it, or is too cold and the water freezes. Nowadays more and more water fountains are adding more light elements especially using LED technology, although in this project we are not using water it would be interesting to use the techniques and technology applied in this project by mixing both elements into a more complex light water fountain.

1.2 Project Objectives

The Waterless Light Fountain was thought to be an electronic art project with light as the expression element of art, so it has to be able to perform different light shows, including a mode were the light would move accordingly to the music. As it is an electronic project placed in a University another objective was to be user interactive where users not only students but also visitors and staff can play with it and select different modes, visualizing want can be done with electronic engineering. Since it is going to be placed on Franklin street it must be compact and easy to install. The person who installs it should not need to understand anything about how it works besides plugging it to the wall plug and turning the switch on, so it should be able to work in stand-alone operation and able to endurance the outdoor conditions.

1.3 Prior Art

There are various projects where the fountain has inspired and taken ideas from especially the “Dazzling light fountain” made by an electronic class from Stanford University [1], the Dazzling Light Fountain **Figure 1.1** stands nearly six-foot-high, with blue and red LEDs flowing along seven semi-transparent cylinders, loosely arranged in a conical shape. Sensors on either side of the base respond to passersby by signaling the LEDs to begin a cycle of programmed light sequences. Another major inspiration is the lighting in some structures as the bay bridge or the light decorations of the houses in Christmas where the lights change and light up with the Christmas carols.

Figure 1.1 Dazzling light fountain build in Stanford

Chapter 2

2.1 Customer Needs

Attending to the requirements and to fit within the budget I have had to give priority to some of the requirements vs the other. My initial idea was to make a fountain of approximately 4 by 4 feet of base and 5 feet tall. But because of budget and time limitations, I had to took and smaller approach, giving priority to functionality and having more modes of operation based on more complex light shows backed in electronics. Another issue was the placement of the final design, at the end for some reasons that I will explain later in the paper, the system will be placed the inside engineering building probably in one of the aisle, so the fountain physical design should be compact and thin, not too bright, but enough bright so the light can be seen even with the lights on, or during the day with sunlight hitting it.

Th final design counts with four different modes of operation:

-Light shows: consist in preprogramed colorful light sequences.

-Play Music: music and light works together, the lights will "dance" to the music that is being played.

-Paint the LEDs: The users with the help of a touchscreen can be paint each pixel with the desire color.

-Simon Says: Users can play the classic game Simon Says were the LEDs will light up in a random sequence and the user must memorize the sequence and repeat it, increasing the difficulty at each level.

For accomplish all the functions above different blocks are necessary including the control of the lights, their color and brightness. The audio processing so the light knows what to do depending on the music that is being played, a user interface so the different modes can be selected and a way to give feedback to the user.

2.2 Layout of System Level Design

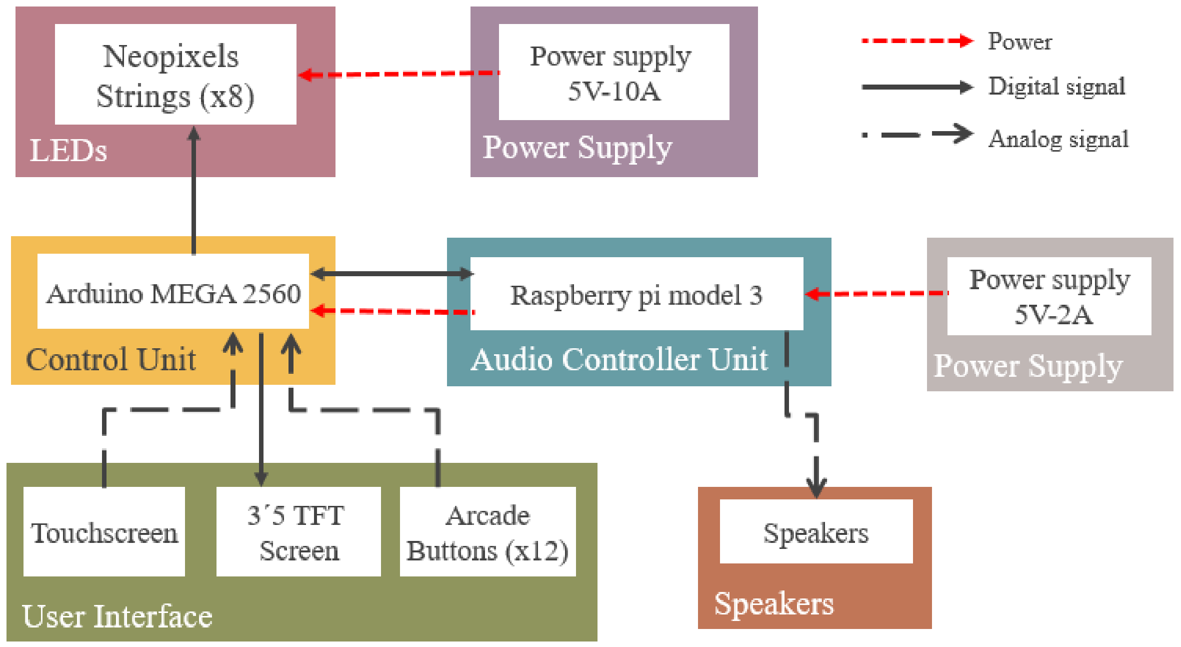


Figure 2.1 Block Diagram

Figure 2.1 shows the high-level design of the waterless light fountain, In the following section we will explain the distinct parts that compound the fountain one by one explaining their functionalities and discussing the design choices of why each component have been selected versus other alternatives, and mentioning those.

Chapter 3- Design

3.1 LED Display

LEDs were chosen as the source of light due to the fact of their efficiency being up to 80% more efficient than traditional lighting as fluorescent or incandescent lights, they also have other sustainable advantages as being nontoxic and that their life span is longer than other types of lights lasting up to six times longer. Another great advantage especially if we are going to be exposed to the light for longer periods is that the LED spectrum is closer to natural daylight than incandescent, having a positive impact in human cognition and mood feeling, making us more alert and helping fighting symptoms of depression [2].

3.1.1 Neopixels

The selection of the LEDs wasn't easy, there are plenty in the market with varying prices and specs. Once decided that it would be much more interesting to purchase strings of LEDs individually addressable and with enough colors to be able to create different and many colorful light shows, the search narrowed. I finally decided to purchase the strings from Adafruit that I know makes high quality products and always adds useful guides for installation and use, which offered many options of LEDs with different technologies, sizes and formats. The ones chosen for prices and because the technology used were suitable for the project at least at the time of purchasing them are the strings of Neopixels, you can choose how many LEDs for string between 30/60/144 Neopixels per meter. Looking at the price being for 4 meters 68/100/240 dollars correspondingly. Taking in account that the more LEDs the bigger visual effect. But also that the consumption of energy is greater, the decision was to take something in the middle and go for the 60 LEDs per meter of string fitting within budget and creating enough visual impact for our purpose. The LEDs used are the Neopixels Digital RGB strings of 60 LEDs per meter from Adafruit Industries, which are strings of individually addressable LEDs, that combining the Neopixel technology with the use of the WS2812 driver have created a scalable and affordable full-color LED.

The power voltage necessary to power the Neopixels is 5V, consuming a maximum of 60mA per LED at maximum brightness, we are using a total of 8 strings of 30 RGB LEDs each, that is a total of 240 Neopixels, so we will need a maximum of 14.4 A, the brightness of the LEDs is set up to a maximum of 70% the full brightness, needing a maximum of 10.08A in the worst case were all LEDs are turned full white, a power supply of 5V and 10A has been the one selected to power the Neopixels [3].

3.1.2 Theory of Color

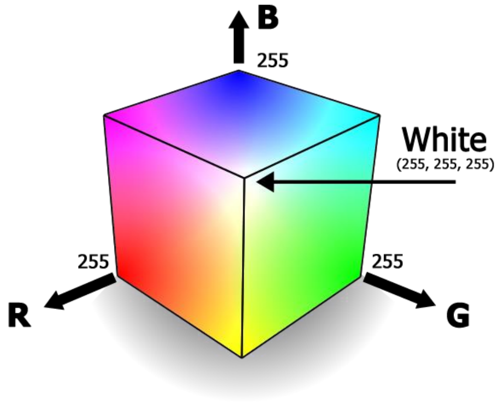


Figure 3.1: RGB Color Cube

In visual arts, theory of color is a body of practical guidance to color mixing and the visual effects of a specific color combination, it explains how to get all the colors using the three primary colors. With RGB LEDS intensifying the different Red, Green or Blue colors (being close to the primary colors) we can move through the RGB color cube **Figure 2.1** from black to white, passing through all the colors. Going each from an intensity from 0 that is the equivalent to being off to 255 that it is its maximum brightness having a total of 16.777.216 possibilities of colors. Colors are also use to transmit feelings and emotions, as we associated different colors to emotions, this is highly used by designers.

3.1.3 Alternatives

Many alternatives of light sources were investigated as to be used instead of the Neopixels or to be combined with them. With a special interest in fiber optic materials lighting whereas instead of the usual fiber optic cables, this are optimized for transmitting light, not high-speed signals. Side-emitting fibers have a rough interface between the core and the cladding to scatter some of the light out of the core along the length of the fiber to create a consistent lighted look similar to neon light tubes [4]. I also looked at lasers and strobe lights to maybe move them with servos.

3.1.4 Schematic

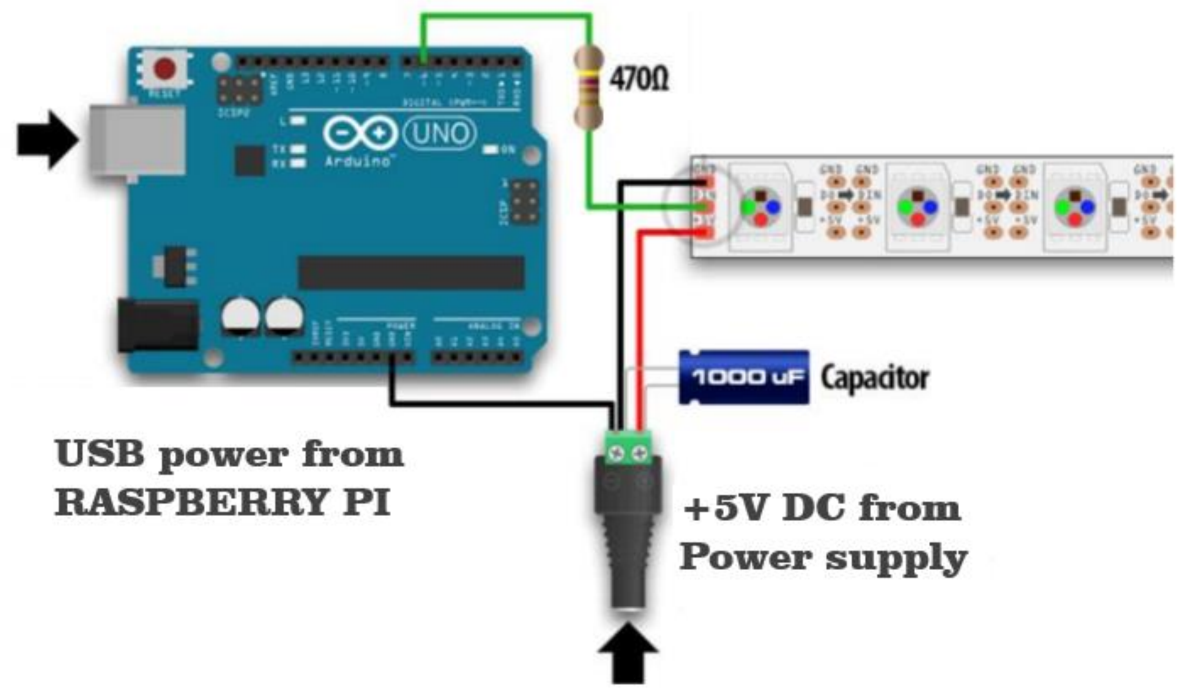


Figure 3.2: Schematics of how to connect each string of Neopixels

When using a DC power supply, the use of a large capacitor across the + and – terminals is recommended **Figure 3.2**, this prevents the initial onrush of current from damaging the pixels. In our case we are using a capacitor of 1000µF ,12V.

The addition of a resistor of 470-ohm resistor between the microcontroller's data pin and the data input on the NeoPixels **Figure 3.2** can help prevent spikes on the data line that can damage the first pixel.

3.1.5 Controlling the Neopixel

For controlling the Neopixels we use the library Adafruit_Neopixel made by Adafruit that makes the control of the LEDs much easier and easy user friendly. It is as easy as initializing the library telling them the number of pixels we have connected and to which pin that has to be PWM, and what type of bitstream and type of NeoPixels we have. Then there are different ways to change the color of each of the pixels, being the setPixelColor( ) the ne I used the most for being really clear. It´s important to know that all the functions for selecting the brightness and colors on the NeoPixels don´t have an immediate effect, the values are stored into the RAM and it´s necessary to call the function show( ) to send the information to the pixels.

Arduino code for controlling Neopixels with Adafruit_NeoPixel library:

```
// We Initialize the library
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
  #include <avr/power.h>
#endif
// Which pin on the Arduino is connected to the NeoPixels?
#define PINs1           10
// How many NeoPixels are attached to the Arduino?
#define NUMPIXELS      30

// Parameter 1 = number of pixels in strip
// Parameter 2 = pin number (most are valid)
// Parameter 3 = pixel type flags, add together as needed:
//   NEO_KHZ800  800 KHz bitstream (most NeoPixel products w/WS2812 LEDs)
//   NEO_GRB     Pixels are wired for GRB bitstream (most NeoPixel products)

Adafruit_NeoPixel strip1 = Adafruit_NeoPixel(NUMPIXELS, PINs1, NEO_GRB +
NEO_KHZ800);

void setup(void) {
strip1.begin(); // This initializes the NeoPixel library.
strip1.show();  // Initialize all pixels to ‘off’
}

for(int i=0;i<=NUMPIXELS;i++){
    // pixels.Color takes RGB values, from 0,0,0 up to 255,255,255
    strip1.setPixelColor(i, strip1.Color(0,150,0));
}
strip1.show();  // setPixelColor() does have immediate effect, it saves the data
on RAM. To send that information is necessary call the function show().
```

We also use the library NeoPixelPainter [5] to help us create more advance light shows, using it in the light shows corresponding to cases six , seven and eight. Allowing to create a “canvas” and a “brush” associated to each canvas to light the NeoPixels. Selecting three different parameters H, S and V for each brush based on the HSV wheel **Figure 3.3**, being H the color, S the saturation and V the brightness. And then translating/painting that brush into the string.

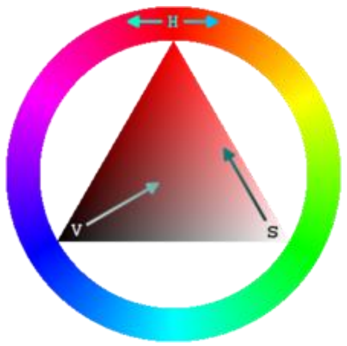


Figure 2.3 HSV Wheel Color

3.2 Control unit

3.2.1 Functionality

The control unit controls the NeoPixels, collects the user´s data input through the user interface and communicates with the audio controller unit.

3.2.2 Design choices and alternatives

We use and Arduino Mega [6] as the control unit. Arduino is a cheap reliable microcontroller that is programmed in its own language very similar to C. The main reasons why Arduino was chosen is that it works at 5V, so there is no need to use voltage level shifters to control the Neopixels. It has a CPU speed clock of 16 MHz necessary to control the Neopixels as they have specific timing requisites needing a speed clock of 8MHz or 16MHz. It also has 54 digital pins, 14 of them PWM, using 8 of them to control each of the Neopixels strings. To control the Neopixels it is the necessary to buffered the entire strip of LEDs in the RAM, having the MEGA up to 8KB, taking the pixels almost 1,5KB of the RAM and taken the variables in the Arduino code another 1,1 KB. Arduino also allows us to communicate with other microcontrollers in various modes of communication including ISP bus, I2C or serial communication.

As an alternative for the Arduino the use of Raspberry Pi was one of the main options as we could have been able to integrate the audio processing in the control unit. But due to specific timing requisites of the Neopixels, the raspberry is unable to control them in a reliable way. It also has a lack of enough pins to control all the Neopixels strings and to receive and give feedback to the user interface. Other alternative was to use an Arduino UNO but due to the lack of enough PWM pins, it has only 6 PWM pins and that is has not enough memory to store the volatile variables and to buffered the strings in the RAM at the same time, the Arduino MEGA was selected.

3.3 Audio processing microcontroller unit

3.3.1 Functionality

The audio processing unit reads audio files from the library and performs a Fast Fourier Transform separating the different frequencies of the audio signal and then divides the different frequencies in a logarithmic scale onto 8 channels. Sending the information to the control unit while playing the music through the speakers.

3.3.2 Design choices

The initial idea was to use either the Raspberry Pi as the control unit and audio processing, or to use the Arduino MEGA to work also as the audio processing unit, performing the spectrum analyzer and or working as Volume meter(VU). The VU does a measurement of the amplitude of the Audio signal no matter what frequencies are. Even though was designed and built. The results were not as spectacular as expected, so I decided to go for something that looked better and were the light really moved to the sound of the music, one of the ideas was to get the rhythm of the song but this idea was discarded for being too complex to read the rhythm of intricated songs, where there is more than one beat at a time and they are all mixed making really hard to read them, so finally the idea to make and spectrum analyzer was taken. However, the Arduino is not able to read audio files by itself either to play quality music, you can do it combining signal with PWM pins. The idea was to add different shields/modules to the Arduino to add these capabilities, as the Adafruit "Music Maker" MP3 Shield for Arduino and also a Bluetooth module, looking forward to making a mobile app to control the fountain. Also, the FFT requires a lot of CPU power, so the use of a Raspberry Pi 3 model B was selected as the audio processing unit. The Pi can perform a faster FFT than the Arduino computing more values per second and giving the impression of a more consistent spectrum, it also counts with a Jack output to play audio signals and it is able to read a vast variety of audio files, having also Bluetooth and WIFI connectivity very useful for a future mobile app [6].

3.3.2.1 VU meter

Before deciding to make the spectrum analyzer, I decided first to make a Volume meter, consisting of a microphone and an amplifier connected to one of the analog input to the Arduino to “read” the amplitude of the sounds, to measure the acoustic noise, the design was very simple it consisted in a microphone with an amplifier connected to one of the Analog inputs of the Arduino where the Arduino mapped the Amplitude of the signal onto one LED string, the VU meter was taking out of the final design as we were unable to incorporated with the rest of the code to work simultaneously, although we could have add it as a mode of operation, because of the final location of the light fountain that is going to be in the engineering building of Santa Clara University, I thought it would be a bad idea to add this function as I didn´t want people making noises to light up the LEDs and distracting the other students and generating sound contamination.

-Microphone and Amplifier:

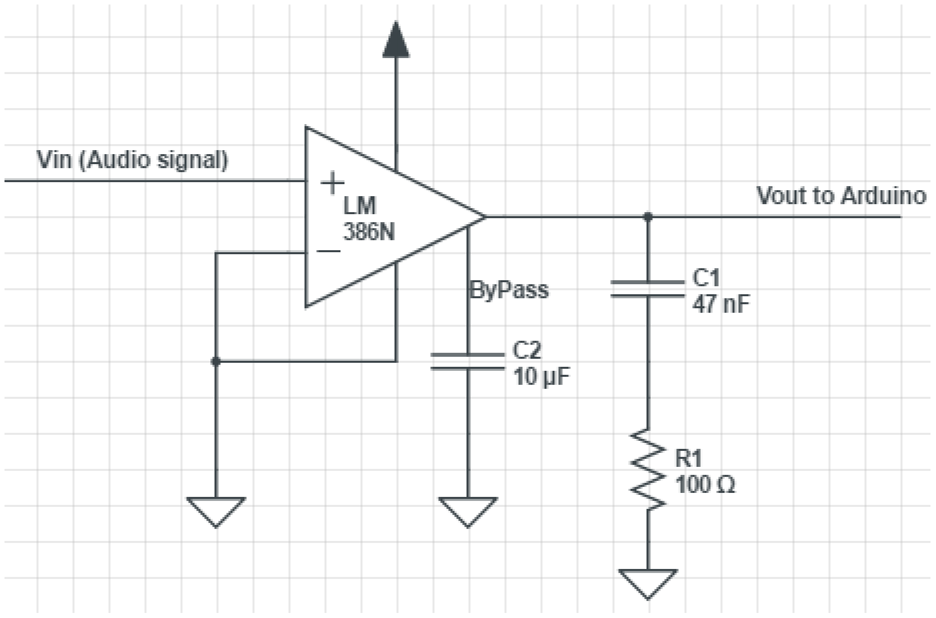


Figure 3.4: amplifier schematics

Figure 3.4 shows the Amplifier schematics that takes the signal coming from the microphone, and amplifies it having an offset of 2.5 V. For the microphone as it was only for demo purpose we tried using some that we already had as they were not very good quality the signal had a lot of noise, although for the VU, noise isn´t as important as in other applications, the add of a Bypass between pin 7 and ground with a 10µF (C2) is indicated to prevent supply noise getting back into the earlier amplifier stages.

3.3.4 Audio processing

The objective is that the LEDs should move accordingly to the music, "dance to the music that is being played". The code is written in python version 2.7, it works the following way, first a song is selected through the user interface controlled by the Arduino. Then the Arduino tells the raspberry which song to play, the raspberry decodes the song using a decoder library and stores it into memory. After that taking one chunk at a time of 2048 bits. The chunk has to be a power of 2 for how the algorithm of the FFT works, the number 2048 was selected by trial and error, if we put too big of a chunk the time the pixels change per second is lower but if we put it to low 1024 bits, then the Pi isn´t able to process them fast enough, so there are cuts in the music. The Pi makes a Fast Fourier Transform **Figure 3.5** getting the frequencies of every chunk. Dividing the ones that are on the hearing range, that describes the range of frequencies that can be heard by humans, were in a healthy human the range is from 20Hz to 20kHz. Into 8 sections in an logarithm scale being the first one from 20 Hz to

156 Hz and the last one from 10 kHz to 20 kHz, then does a calculation of the amplitude of this frequencies and maps each of them from 0 to 30 that are the number of pixels in each string and sends this information to the Arduino, at the same time that does this, is also playing the song through the speakers.

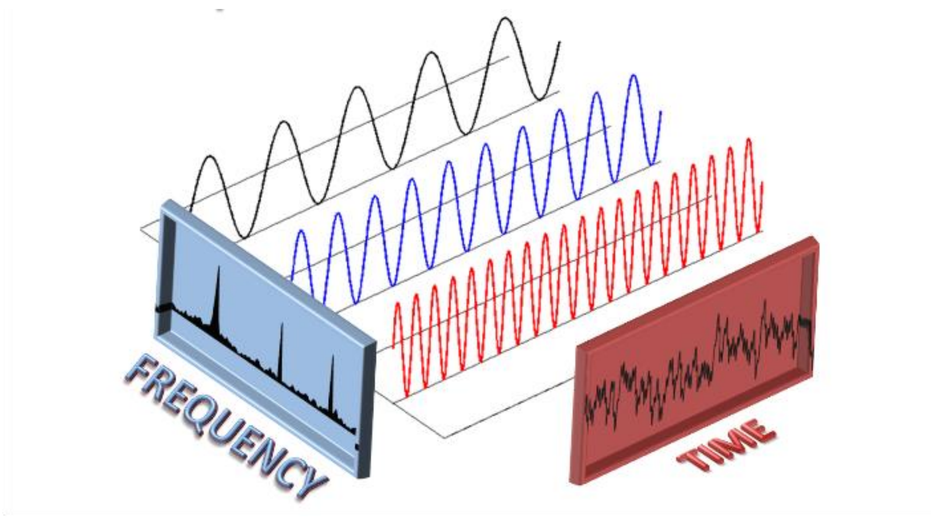


Figure 3.5 FFT

Python code in the Raspberry Pi for the audio processing:

```
import alsaaudio as aa  #use to play the audio
import wave             #open and read wave files
from struct import unpack  #convert the audio data for numpy
import numpy as np   #to do calculations as the fft
import decoder       #decode various tipe of audio files formats as mp3
import os, random
matrix=[0,0,0,0,0,0,0,0] #number of strips
power=[]
weighting=[2,8,8,16,16,32,32,64]#scales to the display

def calculate_levels(data, chunk, sr):
    # Use FFT to calculate volume for each frequency
    global matrix

    # Convert raw sound data to Numpy array
    data = unpack("%dh"%(len(data)/2),data)
    data = np.array(data, dtype='h')

    # Apply FFT-real data
    fourier = np.fft.rfft(data)
    #Remove last element in array to make it the same size as chunk
    fourier=np.delete(fourier,len(fourier)-1)
    #Find average 'amplitude' for specific frequency ranges in Hz
    power=np.abs(fourier)
    matrix[0]=int(np.mean(power[piff(20,sr)         :piff(156,sr):1]))
              .                                              .
              .                                              .
    matrix[7]=int(np.mean(power[piff(10000,sr)     :piff(20000,sr):1]))

    #tidy up column values for the LED matrix
    matrix=np.divide(np.multiply(matrix,weighting),1000000)
    # Set floor at 0 and celing at 30
    matrix=matrix.clip(0,30)
    return matrix

def piff(val,sample_rate):
    CHUNK_SIZE= 1024 # multiple of 8
    #Return the power array index corresponding to a particular frequency.'''
    return int(CHUNK_SIZE * val / sample_rate)

musicfile=decoder.open("/home/pi/Music/"+playlist[i])

                #read first chunk of musicfile
                data = musicfile.readframes(CHUNK_SIZE)
                stop=0
                while stop != 1 or data != '':

                    output.write(data)#play song
                    #calculate value matrix
                    matrix=calculate_levels(data,CHUNK_SIZE,sample_rate
                    print matrix
                    c.send("data_ping",matrix[0],matrix[1],matrix[2],matrix[3],ma
                    trix[4],matrix[5],matrix[6],matrix[7])
                    ack=c.receive()
                    ackc=ack[1]
                    print(ackc)
                    if ackc == [999]:
                        stop=1
                    #do the same with the next chunk
                    data=musicfile.readframes(CHUNK_SIZE)
```

3.4 Communication between the Control and the audio processing units

For communicating the Arduino and the raspberry I have different alternatives as I2C communication, serial GPIO communication or the one that we have choose communication via USB cable. That does not need a logic level converter as the other two and it is also easier to implement with no hardware more than a cable [8], it also allows to communicate both of them as “equals” instead of having a master-slave communication. I have use the CmdMessenger library to communicated both, we had to modify a little bit the library to work in Python version 2.7 as the one available only worked in python3.

3.4.1 Serial Communication with CmdMessenger

It sends and receives messages, automatically converting python data types to arduino types and vice versa [9]. To ensure stable communication with PyCmdMessenger, instances must be given a list of command names in the same order as those commands are specified in the arduino sketch, instances should be given a list of the data types for each command in the same order as those commands are specified in the arduino sketch. separators must match between the PyCmdMessenger instance and the arduino sketch being the field separator (default ",") command separator (default ";") escape separator, so field and command separators can be sent within strings (default "/"). Also, the baud rate, a number related to the speed of data transmission in a system indicating the number of electrical oscillations per second that occurs within a data transmission, must match between PyCmdMessenger class and arduino sketch choosing 115200 for being the maximum in the arduino.

Python example code on Raspberry Pi for Serial Communication:

```
import PyCmdMessenger # communicated by serial with Arduino
arduino=PyCmdMessenger.ArduinoBoard("/dev/ttyACM0", baud_rate=115200,
timeout=6000.0)
    # List of commands and their associated argument formats
    # These must be in the same order as in the sketch.
    commands = [["data_ping","iiiiiiii"],
                ["ack_pong","i"],
                ["Demo_play_music",""],
                ["Demo_yesorno","i"],
                ["error","s"]]
# Initialize the messenger
    c = PyCmdMessenger.CmdMessenger(arduino,commands)
    while True:
        c.send("Demo_play_music")
        yesorno=c.receive()
```

Arduino example code for Serial Communication with Raspberry Pi:

```
/* Initialize CmdMessenger -- this should match PyCmdMessenger instance*/
const double BAUD_RATE = 115200;
CmdMessenger c = CmdMessenger(Serial,',',';','/');

/*Initiaziled serial communication*/
#include "CmdMessenger.h"

/* Define available CmdMessenger commands */
enum {
  data_ping,
  ack_pong,
  Demo_play_music,
  Demo_yesorno,
  error,
};
void on_Demo_play_music(void){
  c.sendBinCmd(Demo_yesorno,answer);
}
         .
         .
void attach_callbacks(void) {

    c.attach(data_ping,on_data_ping);
    c.attach(on_unknown_command);
    c.attach(Demo_play_music,on_Demo_play_music);
}
```

3.5 User Interface

One of the requirements of the fountain is that it should be user interactive, for that reason making it user friendly and with different modes of operations was one of the main priorities, we achieve that by creating a simple and easy to use interface, using buttons and a touchscreen to receive input from the user and the use of a TFT to give feedback to the user.

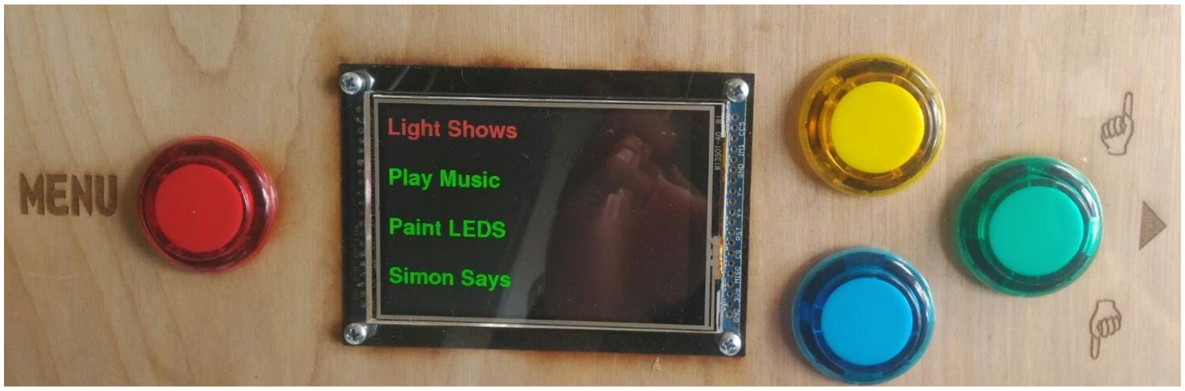


Figure 3.6: User Interface, TFT +Touchscreen and arcade buttons

3.5.1 Screen and Touchscreen

For the input and to give feedback to the user the Adafruit 3.5" 320x480 Color TFT Touchscreen Breakout plus touchscreen has been selected for being suitable in our design and within budget, for the touchscreen two different types of connections are possible 8-Bit or SPI [10], the 8-Bit communication has been selected for being the fastest one of the two and that we had more than enough available pins in the Arduino MEGA, needing 8 digital pins and 4 analog pins for TFT plus the use of another 4 analog pins for the touchscreen.

3.5.2 Arcade buttons

12 arcade buttons of 30mm have used to receive inputs from the user, 8 whites one positioned in the lower part of the display and four more of different to help them navigate through the different modes of the fountain and interact with it, to avoid the noise of the buttons a smith trigger[11] that is a comparator with hysteresis has been add to each button to eliminated the noise **Figures 3.4 and 3.5**.

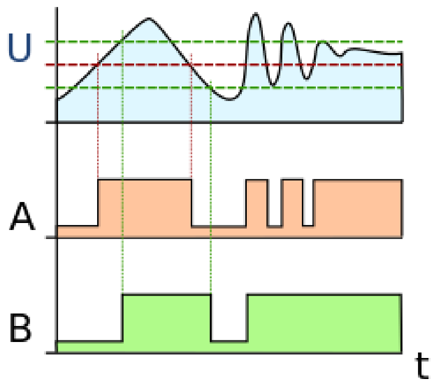


Figure 3.4: Effect of a Smith Trigger

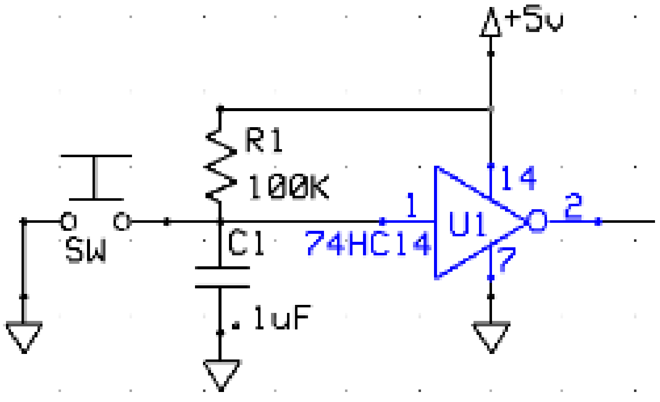


Figure 3.5: Schematic button with smith trigger

Chapter 4-Physical Design

The structure has been divided in two parts, 8 columns to support the Neopixel strings and a wooden case to hold the rest of the hardware and to function as support for the fountain, it has a square profile with one of the faces with an angle of 30 degrees to support the TFT screen and the buttons, so the users can look at the pixels while using the interface.

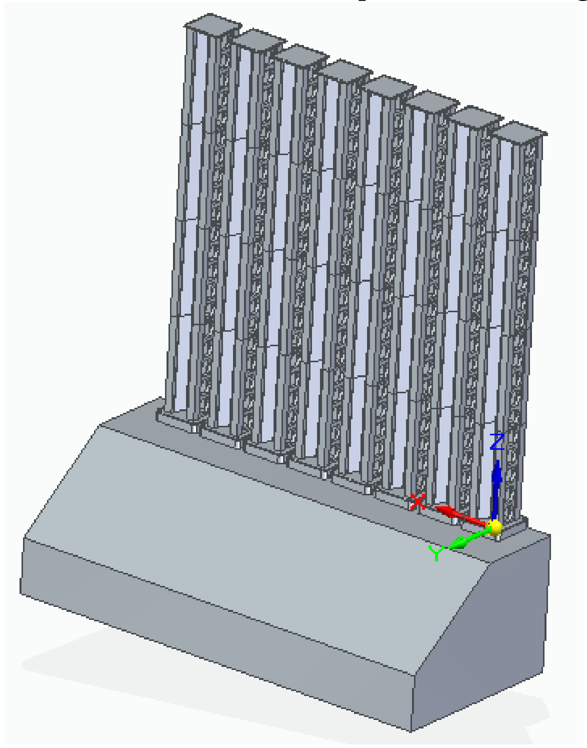


Figure 4.1 Physical Design in Solid Edge

4..1 Design Choices

The first idea was to make the waterless light fountain able to endurance outdoors, using acrylic tubes sealed with silicone for placing the NeoPixels strings and make a water-resistant box made of plastic or metal, but due to the dimensions of the fountain and the safety issues as the box is connected to the main, we decided to make it for being inside, to be place in one of the engineering buildings of Santa Clara University.

4.2 Wood Case

The material used in the case in utility wood, which is made of a mix of different layers of plywood with some glue, the box was designed using Corel Draw 4 and printed using the Epilog laser cutter, also some square wooden pieces have been used to give robustness to the unions attach with screw of ¼ diameter 1" inch long that can be screw on and off to open the case at any moment, making reparations and modifications easier.

4.3 NeoPixels String Tower Support

For making the LED towers the use of a 3D printer MakerBot replicator X2 was used, because of the size of the printable area has a maximum size of 16 cm, the structures have been divided in 4 parts that fit together through pins, making it possible to add more or less piece depending how long the LEDs strings are, so if we wanted to make strings of 1 meter instead of 50 cm we would just have to add 4 intermediate modules more and in the code change the number of LEDs (NUMOFLEDS) to 60 instead of 30, supposing we were using strings of 60 LEDs per meter.

Chapter 5-System Testing and Results

Due to the nature of the projects constant tests are being perform over and over, as it is a very visual project most of test were easy to validate visually. For the implementation of the system, I worked block by block into the different subsections of the design to integrate them later together.

5.1 Light System Testing

The first step was to connect a short Neopixel string to the arduino and start working from there, understanding how the library works and learning how to control the pixels to do what we wanted to do, change it them one by one to the color we wanted, once I tested the Neopixel could do what the project required, I purchase a bigger string of 4 meters, cut it into 8 strings of 50 cm long, weld them and connected them to the arduino and from there start working on the light shows.

I also tried to connect and control them with the Raspberry Pi but I was unable to make them do want I wanted them to do, so I had to stick with the Arduino to control them.

5.2 Subsystem Testing Of Audio Processing

Fort the Audio processing I based my code in the code made by Scott Driscoll [12]. As I was unable to find all the python libraries he used, I had to made my code simpler, first I tried with a wav file as thy are not compressed and try by calling it directly from the function by his name store in the same directory as the python script, and once it was working I start to test the decoder library and instead of calling the song directly by putting the name of the file. I created a playlist and from there select a random song to be played. I also played with different frequencies to divide the channels, the fountain works from 20 Hz to 20 kHz while most of the time, songs have the majority of the data in between the second and the sixth octave corresponding between 16 Hz to 2KHz, the spectrum looks shifted to the left to the low frequencies corresponding to the first strings starting from the left, I tested different options by making the spectrum range smaller but then all the channels were too close to each other and made it harder to see the effect of the different frequencies.

5.3 Communication Testing Between Arduino and Pi

For communicating the Arduino and the Pi, I tried first using the Pyserial library which allow you to send information from one to another but it is more thought to have one of them as a master and one of them as a slave where only one is telling the other one what to do, in our case both are communicating equally and due to the complexity of are data types, strings of numbers and initially also name of songs, the fact that we also wanted to send this information fast enough so there wouldn´t be any delay between the music that is being played and the pixels, we change to the PyCmdMessenger library that is based in a great part in the Pyserial library, in a start because I didn’t have a user interface, I selected the different demos by logging in the raspberry through my laptop and selecting them from there, once I made the user interface change the functions so that the arduino told the raspberry when to play the music, because of this, the raspberry has to be always listening, waiting for the arduino to signal it when someone has selected the Demo Music and then start all the communication necessary between them. The library has showed to very useful, making it way easier to send complex data and to also make it faster enough so there is no delay between the music and the pixels lighting up to its rhythm.

5.4 User Interface Testing

For the user interface I connected first the twelve arcade buttons that were giving noise problems and were being read twice, so I add a smith trigger to avoid this problem, and for giving feedback to the user of what is happening in the fountain, I then added the TFT so it displays information of where the user is and what the fountain is doing, when the fountain is turn on, it starts in the main menu and with the three top left buttons we can chose between the different modes being the yellow and the blue to go down respectively and the green to select the option, once the option is selected we either use or the lower buttons for the selecting the light show, selecting the song or playing Simon says or we use the touchscreen to paint the different pixels as we want, and for the one button left the red button is to go back to the menu and it is connected to one of the attach interrupt of the Arduino so it can be pressed at any moment to go back to the main menu and select the option we want.

5.5 Stand Alone Operation

The arduino has auto-launch so that if you load any program in the arduino it will run automatically, the raspberry Pi does not have this, so it’s necessary to create a launcher script, with the instructions to tell the Pi what script has to open when turn on, I had some problems doing this, as while trying to do the songs menu, it would not connected with the arduino or it will read data that make no sense, after simplifying the menu to one song for each button and the python code a little bit to make it simpler, it works really good, but once in a while it needs to be power cycle, or in other words turn on and off to restart the raspberry and reset the communication between the arduino and the Pi.

Chapter 6- Costs

ITEM	**COST**
Digital RGB Neopixel strip (x4)	$99.96
5V 10A switching power supply	$24.99
Arduino MEGA	$48.81
Raspberry pi R3 complete started kit (including power supply and case)	$70
Arcade Buttons (x12)	$38
3´5” TFT +Touch Screen	$39.95
Speakers	$30
Constructions materials including wood, screws and varnish.	$50
Breadboards, jumper cables, 74HC14 Schmitt-triggers(x3), capacitors, resistances	$40
Total	$441,71

6.1 Economic

The project cost could be reduced by taking different design approaches, one of the main expenses are the Neopixels which cost could be reduce by selecting cheaper strings of LEDs although the quality may be not as good as with the Neopixels there are definitely cheaper alternatives solutions, also if we get rid of either the raspberry pi or the arduino we will also save some money, either controlling the pixels with the Pi or trying to make a poorer spectrum analyzer that could reproduce wav files from an SD card, wav files could be read by adding a SD card reader module to the arduino, the other alternative could be do it with hardware and although making one PCB is probably even more expensive, if the idea was to make a lot of fountains it will be cheaper and easier to implement in the design, not having to set up the Pi and downloading all the libraries and uploading the songs and the code to it. I don´t think a better version of this product could be sell in copious amounts, but I think it could work very well as a custom product were each custom designs could meet the client desires, space and budget, maybe taking orders and working with the clients through the internet.

The idea of the fountain and some of the functions have the potential to be modify and expanded to big markets. I believe in could be sell really well if make modifications to be implemented in recreational areas such as museums, amusements parks or music clubs.

7. Professional Issues and Constraints

7.1 Health and Safety

As I discussed in the LED display section, the LEDs can have positive impact into people, creating emotions with the light shows and even helping fight depression thanks to having a similar spectrum to natural light, but it could also a have a negative impact into people due to the light pollution generated by the fountain.

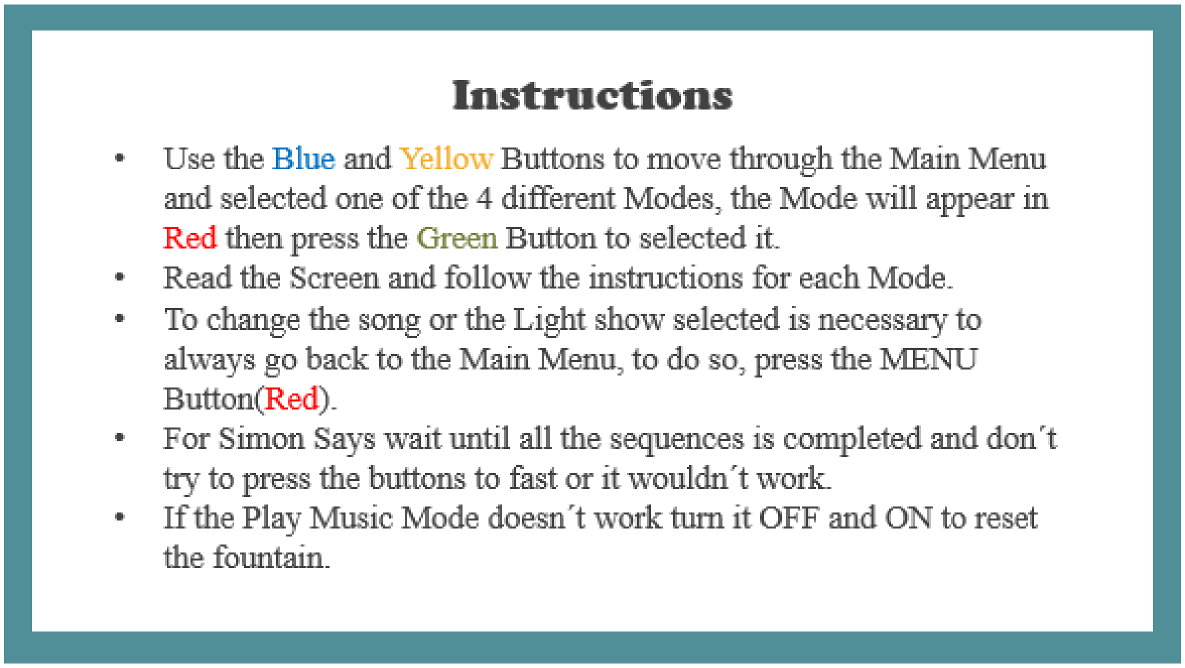
As for the safety one of the main reason that make the fountain unable to endurance outdoors is that I didn´t know how to isolate all the components so it would be water resistance. As it is connected to the main it is a critical issue as I didn´t want to have any shortcuts that can make it catch on fire or electrocute anyone. So, it would be necessary to change the material of the fountain from wood to plastic or metal, and sealed so no water can come in, as for the LEDs acrylic tubes could be used to avoid water touching the Neopixels.

7.2 Manufacturing

I think for the Project to be manufactured it would be necessary to make some changes as to replace the raspberry pi for another alternative probably a PCB that did the TFT with filters. As for the Pi if I wanted to produce in mass production I will have every time set up every Pi. Consisting in changing a lot of configurations on it, including the initial set ups, downloading the libraries, downloading the music, uploading the code and creating a launcher script making it really slow and needing an operator to do so. The solution will be to make a hardware version with a jack input and maybe a usb or SD card that could read the different audio files. It would be also necessary to design another PCB to hold all the components, as right now the hardware and connections are made on a protoboard. Also the LED structure fabrication process should be change as the LED towers take lot of time to get 3D printed, so make each tower in one piece and find the appropriate fabrication process. Or create and simpler structure to display the pixels.

7.3 Usability

The fountain is easy to use, very visual and with the instructions **Figure 6.1** anyone can know how to use it in less than a minute, so overall, I´m really satisfy with usability of the project, is also easy to set up, you only need to plug it to the wall.



Instructions

- Use the Blue and Yellow Buttons to move through the Main Menu and selected one of the 4 different Modes, the Mode will appear in Red then press the Green Button to selected it.
- Read the Screen and follow the instructions for each Mode.
- To change the song or the Light show selected is necessary to always go back to the Main Menu, to do so, press the MENU Button(Red).
- For Simon Says wait until all the sequences is completed and don´t try to press the buttons to fast or it wouldn´t work.
- If the Play Music Mode doesn´t work turn it OFF and ON to reset the fountain.

Figure 6.1: Waterless Light Fountain Instructions

7.4 Sustainability & Environmental Impact

The waterless light fountain as a prototype or even as a product should work for a great amount of time. It shouldn´t failed for software but it may fail by hardware for example if one of the strings stops working but that shouldn´t be hard to repair. It’s also really easy to update the raspberry Pi can be accessed with an ethernet cable from the computer and from the raspberry Pi is possible to access the arduino. Making it really easy to update the code, and it would be also possible to modify the design as the parts are screw or tape to the box so with just a simple screwdriver it is possible to open and take apart all the components making it really easy the change of parts or modify the design. For those reasons, we can say that is a sustainable product in the sense to the degree to which a product that is developed can continue to be viable and useful for a reasonable amount of time.

As it is very easy to open the fountain and take each of the components, it is easy to reutilize the components for other projects in the case that the fountain failed and it couldn´t be repair or that the fountain was no longer in use. Being the most vulnerable component the LEDs that are the most likely to stop working, to replace them it would be necessary to order them from Adafruit and also weld some cables and make the connection so some basic knowledge of welding and electronics will be necessary, as for the other components the arduino and the raspberry should not give any problems and if they do, just need to load the code and replace them and make the appropriate connections. For the disposal of the LEDs there is no need to dispose them in any special place as they don't have mercury as CFLs they are not as harmful and the disposal of this one is not required, but there are places where you can take them to prevent from hazardous substances such as lead and arsenic to enter the waste stream. The arduino and raspberry Pi can be used for other electronic projects and the wood of the structure can be recycle really easy or reused for other projects or even as firewood, for the acrylic and plastic materials there are no natural processes in place that can absorb non-biodegradable plastic back into the biological cycle but they can be recycled and reused for other products.

As for the energy consumption I haven´t estimate the energy of the fabrication of the electronic components and the energy used by the laser cutter and the 3D printer to build the structure of the fountain. I know that each LED consumes a maximum of 60 mA at 5 V when working at full brightness, having a total of 240 LEDs and knowing the average is around 20 mA of power consumption at 5V that makes a total of 24 Watts per hour adding the consummation of the microcontrollers that is minimum and the speakers it would be around 30 Watts per hour as maximum, estimating a use of 8 hours a day that would be a total of 87,6 kW per year, assuming a price for electricity as (11 cents/kWh) that would be a total of 9,64 dollars around three times less consumption of the average TV household.

An environmental problem of the light fountain is the uncertain effects than can produce in the area where is going to be display, creating sound pollution and light contamination that may affect people in the area and may harm or have a negative impact in the animals and insects in the area modifying their environment.

8. Conclusion

We have successfully design and built a waterless light fountain, the fountain satisfies the most important requirements an exception of being able to endurance outdoors conditions such as rain. The different modes are diverse and complete, the user interface is easy to use, the project is compact and is all in a wooden box and although I was unable to 3D print all the LEDs towers it still look pleasant and with a unique style **Figure 8.1**.

During the making of the project I have acquired different skills and valuable knowledge such as how to manage a project from start to end, learned about how to work with arduino and raspberry pi working with C and Python languages respectively and how to communicate both, I also learned about software design in 2D and 3D and the use of the 3D printer and the laser cutter, as well as being able to put in practice knowledge acquired during the bachelors.

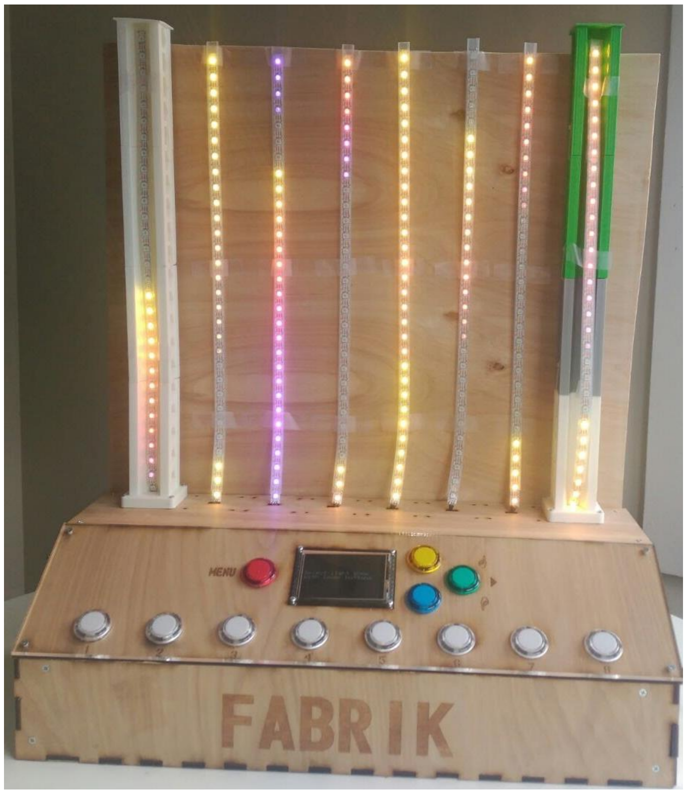


Figure 8.1: Waterless Light Fountain Final Design

8.1 Ethical Considerations

In a world were constant problems and challenges, were people in society has a lot of stress, sometimes people forget how important is to stop and look around, and appreciated the beauty of the world and the small things, the best way to do so is with art that allows people to express themselves and transmit feelings, is also culture, the cave man use to draw in the walls with soil and rocks, now we do it with more advance technology using light. Having fun and apricate beauty are also really important, when people see the fountain I wanted them to stop by, look at it and surprise them as they have seen anything similar to it, I want people to interacting with it surprising themselves what with the fountain can do and for a couple minutes forget about all their problems and enjoy, and with some luck there will be also interested in how it works, they could go to this paper and learn some new staff, or it could motivated to start a similar project combining electronics and art.

As this has been the first big project I have work on, I have had also more time to think how could I can make things better, creating habits of constancy in the work, understanding what I am doing at every moment and what do I want to achieves with the objectives in my mind all the time, as it is an art project and has to transmit feelings as well, asked me questions of how can I possibly make it as esthetically pleasant, how can I spent less amount of resources or how can I impact less the nature in the construction, picking the less harmful components to the environment and trying to make my project as much sustainable as possible making it easier to repair and to reuse for other purposes.

The health effects in the users are unknow, but some studies had proven that art has a great positive impact in people decreasing negative emotions and increasing positive ones, reducing stress and anxiety, increasing spontaneity, positive identity and social network, and as I believe this project is also art, with a high degree of interactivity between the user and the project some of this benefits could make a positive impact in users, it could specially help engineering students as they are sometimes with a lot of pressure that generates stress and negative emotions in them.

My main concern about the fountain are related to electrical hazards and specially to overheating , but this shouldn´t be a problem as the fountain would be inside a building and the power supplies are good quality, as is going to be permanently connected to the main, I tested the temperature of the different components by have been it connected for four days straight and checked if they were getting hotter with time, the components maintained the temperature to normal levels not been even what we can consider as hot, so I think that the problem of catching on fire for overheating shouldn´t be a problem.

8.2 Future Work

While working on the project, specially at the design process many ideas of what could be done have come across my mind, in the final design the only way to interact with the fountain is through buttons and a touch screen but at the start the idea was to put many and different types of sensor, like proximity sensors so it detects people passing by, a temperature sensor so changes the colors of the fountain accordingly to temperature, moving from the cold scales of colors bluish, greenish colors to warm colors reddish orangish, a microphone to get sound inputs using it for creating a VU, showing the noise contamination levels, a light sensor so as it was minted to be outside turn the pixels off when the sun was too bright to be able to see them, also the use of kinet from the Xbox was also in mind so you cold interact with your whole body and exercise as well, and all types of different crazy designs, including the idea of making a 6 by 4 feet San Francisco Bridge for the LEDs display.

8.2.1 Mobile App

One of the main ideas that was to create a mobile app to control the fountain where you could choose the music you wanted or create your own personalized light show from your phone, this was one of the main reasons to also use a raspberry pi as it is relativity easy to control by android apps.

8.2.2 Songs selection and music input

Because of problems in the functions of communication between Raspberry Pi and Arduino the selection of the songs didn´t work very well so at the end I had to just match each of the eight lower buttons to one song, limiting it to eight songs, so one of the main priorities will be to be able to create playlist by gender or artist, and be able to select the songs from there.

Another issue is that for adding music to the library is necessary to log on the Raspberry Pi and download the music files from the internet or usb, so I think it will be really interesting to add a function where you can be able to plug your phone with a jack cable and reproduce the songs you want, as the Raspberry Pi doesn´t have jack input and I´m not sure you could be able to send it to the Arduino and the send this to the raspberry pi, the options will be either make it in hardware or program the arduino to respond to music and change the pixels accordingly to it.

8.2.3 Structure

I would like to finish printing the LED structures and what I think interesting is that the LED display is quite easy to extend, to up to 14 strings or even more if we use logic shifters although I am not sure how this could affect the time requirements of the Neopixels, and easy to expand in terms of having longer strips or strips with more LEDs per meter just by changing the parameter NUM_PIXELS in the arduino code and then ceiling in the python code. And always taking in account the power supply that LEDs require.

The design of a new and bigger fountain that could be visualize from all angles like the one build at Stanford university could be interesting as well specially if was water resistance.

8.2.4 Sustainable

I think it would be interesting to make an innovative design powered by solar energy but that could be an entire project by itself just to figure out how to fit the solar panel on the design to have the best efficiency and provide enough energy to power the fountain, including the installation of a battery to store the energy and voltage shifters for the various parts.

REFERENCES

[1] https://ee.stanford.edu/news/general/03-11-2016/dazzling-light-fountain-rocks-freshman-exploring-ee

[2] Flexfireleds http://blog.flexfireleds.com/superiority-led-color-spectrum/

[3] 2002-16, The Fiber Optic Association, Inc. http://www.thefoa.org/tech/lighting/lighting.html

[4] Adafruit Industries, Neopixels guide https://learn.adafruit.com/adafruit-neopixel-uberguide/

[5] Neopixel Painter Library documentation https://github.com/harmsm/PyCmdMessenger

[6] Arduino MEGA https://www.arduino.cc/en/Main/arduinoBoardMega

[7] Raspberry pi 3 Model B documentation https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[8] Raspberry pi and Arduino serial communication usb https://oscarliang.com/connect-raspberry-pi-and-arduino-usb-cable/

[9] PyCmdMessenger library for Arduino and Python https://github.com/harmsm/PyCmdMessenger

[10] TFT + Touchscreen Adafruit Guide: https://learn.adafruit.com/adafruit-3-5-color-320x480-tft-touchscreen-breakout/overview

[11] Smith Trigger https://en.wikipedia.org/wiki/Schmitt_trigger

[12] http://lightshowpi.org/