Water Aliasing

By
Atreyee Roy
Siddharth Sharma

Final Report for ECE 445, Senior Design, Spring 2017

TA: Luke Wendt

Project No. 57

3 May, 2017

Abstract

This report explains the design procedure, details and considerations for the Water Levitation Device. Gravity defying products are very expensive and have very little user interfacing. Our product incorporates an engaging user experience and is cheaper than the products sold on the market. It also allows for further development to take place by incorporating a programmable microcontroller into the design. Note that our device only creates the illusion that water is levitating and defying gravity. It is not actually levitating. This is achieved by having a strobe light illuminate water droplets, dripping at the same frequency as the strobe light is strobing.

Contents

1 Introduction

An optical illusion is an illusion caused by the visual system and characterized by visually perceived images that differ from objective reality. The information gathered by the eye is processed in the brain to give a perception that does not tally with a physical measurement of the stimulus source. In the case of aliasing water, this illusion is achieved by means of a stroboscopic light. The concept involves altering the frequency of the vibrating water with respect to the frequency of the light. Today, these Gravity Defying Water products, at a small scale, are available in the market for as high as $500-$600 [1] and cost more money than we would like to spend for entertainment. The overall aim is to provide an enjoyable experience of "waterbending".

Our project is one born out of the want for aesthetic pleasure. Humans are creatures of habit and hence, our brain makes many assumptions about our surroundings without us actually consciously thinking about them. When we see logic being outright defied without any rational explanation that our brain can come up with, we tend to be intrigued. Water aliasing is something that most people aren't aware of. Hence, the illusion of water rising or levitating instead of falling without the help of any visible device, like a suction or a pump, baffles the mind. So far, the devices that exist on the market have little to no user interaction. We changed this by introducing a software component to our project. With bluetooth interaction and an app to control water movement, we have added some level of control to our system.

2 Outline of Subject Matter

2.1 Introduction

Our design works on a very simple principle: the frequency of the strobe light we build must match the frequency of the vibration of water for us to see floating droplets. Every time the light is on, an image of a different drop at the same position is visible. Similarly, if the frequencies of the light and that of the vibrating stream of water differ, we see the water moving up or down. In the case of water frequency being lower than the frequency of light, the next drop of water is slightly higher than the previous drop when light falls on it. The brain interprets this illusion as water "moving up". If the next drop appears lower, water is "moving down".

Below we have our high level block diagrams. Our system is divided into 2 basic modules, one for the lighting and one for the water. They work independently of each other but perform in sync to create the desired effect.

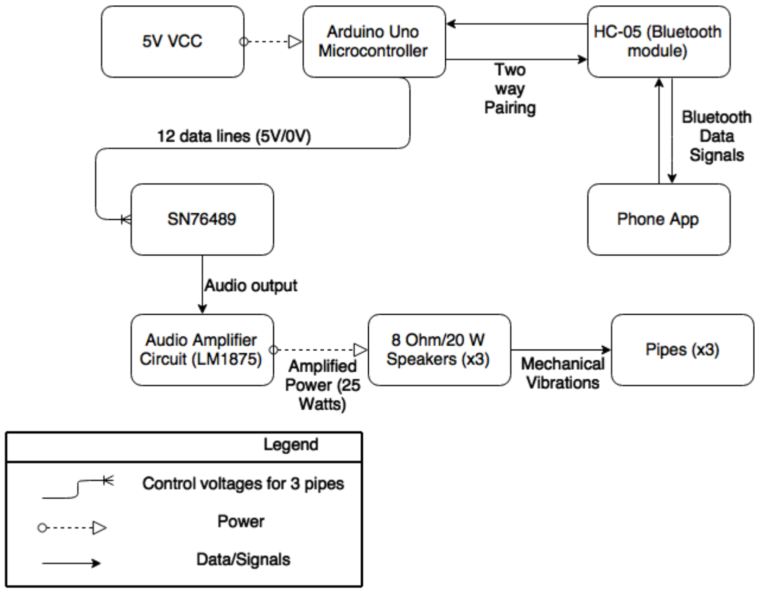


Figure 1: Block Diagram for Lighting Unit

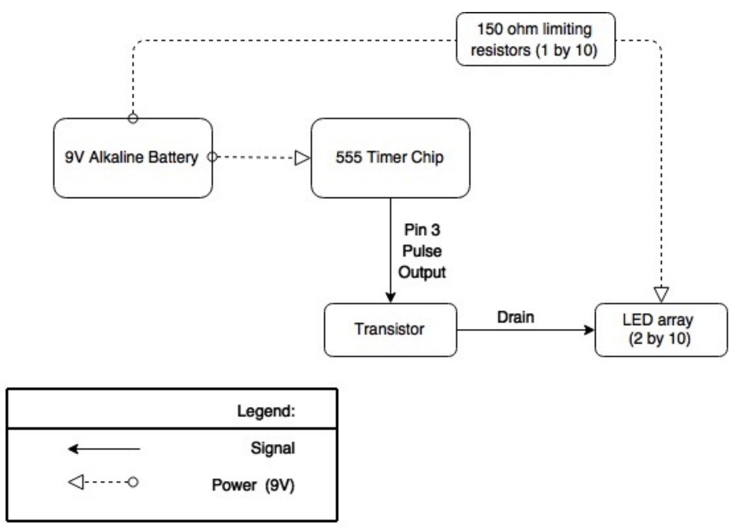


Figure 2: Block Diagram for our Water/User Interface Unit

The lighting unit is designed to produce strobing lights at exactly 100 Hz. We use a basic 555 timer circuit to produce the required waveform with a 10 percent duty cycle. The water unit, on the other hand, is designed to produce a range of frequencies between 90 Hz -110 Hz. We drive our speakers with varying frequencies provided by the sound generator, which vibrate our pipes accordingly. The water then flows out of the pipes through a hole at that frequency. When we synchronize these two modules together, we get our illusion.

The frequency values were altered by us through our design process because with some testing we were able to see that the overall illusion works better at 100 Hz. We explain this better in the following sections. We made other small changes such as the number of speakers and columns of water we wanted to work with. We also changed the number of LEDs used in the demo due to syncing considerations so as to not have interference from two separate PCBs.

2.2 Design

2.2.1 Design Procedure

Lighting Unit: Our original lighting unit was as shown below in Figure 3:

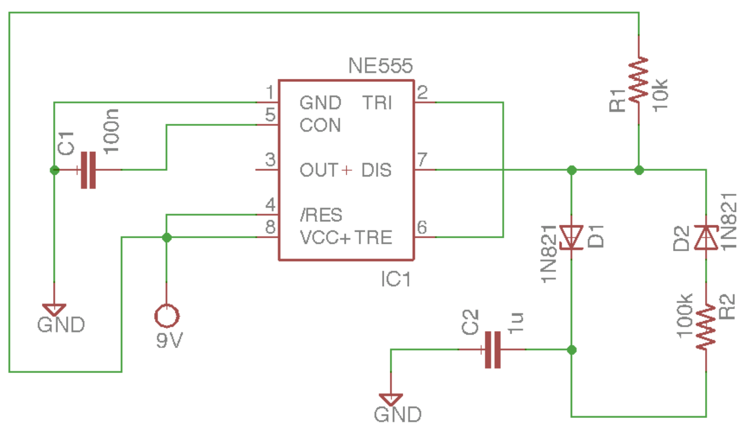


Figure 3: Schematic of LED strobe [2]

We made some changes to this design which made it easy to test the output for various combinations of R1, R2 and C2. We used 100 kOhm potentiometers instead of the fixed resistances R1 and R2. This gave us some flexibility in terms of choosing frequency and duty cycle. The main design equations involved in computing these values for 100 Hz frequency and 10 percent duty cycle are [3]:

$$f = \frac{1.44}{C} * (R_1 + 2 * R_2) \tag{1}$$

$$Dutycycle = \frac{R_1}{R_1 + R_2} \tag{2}$$

For our 10 percent duty cycle it is easy to see that the relationship between R1 and R2 is as follows [3]:

$$R_2 = 9 * R_1 \tag{3}$$

Our proposed frequency at the time of the design review was 60 Hz. However, this is very close to our threshold of human perception, and we would not see the illusion correctly. We still saw the light visibly strobing and this defeated the purpose of the illusion.We changed our frequency of operation to 100 Hz due to this. For our final demo, we used 10 LEDs (instead of our proposed 20 LEDs) sourced from a single PCB.

Due to the instability of the output of the 555 circuit, we decided not to use both the LED panels on either side of our contraption sourced from separate PCBs, since this could affect the resulting strobing effect due to the interference created by individual strobing. The water tub provided by the machine shop at the bottom of our contraption was also not wide enough to accommodate for all 3 speakers and having another panel on the other side would be redundant.

Water Unit: Our consideration for changing the operational frequency to 100 Hz also arises from this module. 60 Hz is very close to the lower limit of our chosen speaker frequency range (50 Hz - 20000 Hz) [4]. Due to this, we expected a very high level of distortion at this frequency and the speaker performance would not be ideal to serve our purpose. The speaker also has extremely high impedance at this frequency (45 ohms) which makes it hard for the amplifier to provide enough driving current at reasonable supply voltages [4]. At 60 Hz, we found that we could see the light flickering, which had a two-fold effect. Firstly, It took away from the illusion by letting the user know that the lights were strobing. Secondly, and more importantly, with the lights visibly flickering at 60Hz, there was a risk of seizure for anyone who looked directly at the lights. Lastly, the sound generator can produce 100 Hz tones more easily with the data bits it was supplied compared to our proposed 60 Hz.

The physical contraption had to have a tub that was deep enough to prevent water from splashing [5]. In addition to this, the LEDs had to be placed in a manner such that they weren't at risk of being exposed to potential splashing. Hence, the tub we chose could fit 2 liters of water whereas we only needed a little less than 1.5 liters. The LEDs were fitted into panels on either side of the contraption, with only the heads being exposed to any potential splashing. The heads of the LEDs are not affected by water and the fact that they fit firmly into the LED panels prevents any water from leaking onto the PCB and the rest of the circuit.

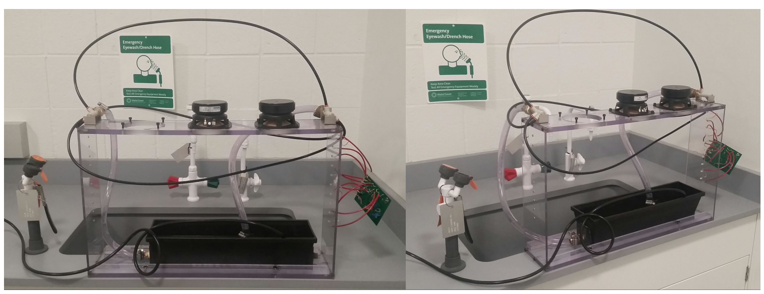


Figure 4: Physical Design - Left) Front View and Right) Isometric View

User Interface unit: The app in our user interface module is designed so that the user can control the illusion settings of each of the 3 streams of water independently of each other. Independence in this context means that the user is able to control a stream without causing any change in the functionality of any other stream. This app has been programmed in Java, using Android Studio.

2.2.2 Design Details

Strobe Lights: As mentioned in the above section, we used the equations for frequency and duty cycle for the 555 timer circuit to decide resistance and capacitance values. In Figure 5 below, we have the constructed circuit on the breadboard during testing 2 LEDs:

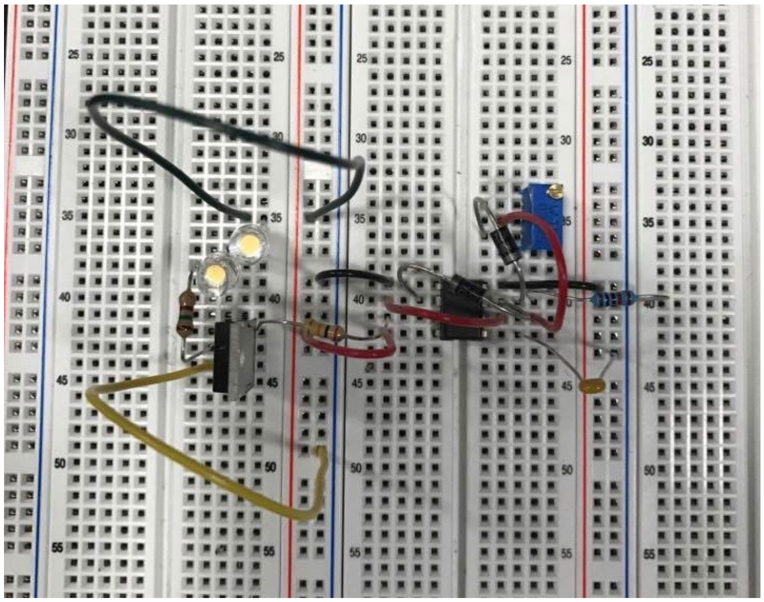


Figure 5: Breadboard Circuit for Strobe Light

Water Pump and Storage: The water pump of our choice was the 'Total Pond 140 GPH Fountain Pump'. It is a submersible pump that can pump water up to 3 feet in height at a maximum rate of 140 gallons per hour. It pumps water from a tank at the bottom of our water levitation device. The pipe that comes out of the pump has a 1/2-inch outer diameter and a 3/8-inch internal diameter. This pipe is fed into pipe splitter. The pipe splitter takes one input and 3 outputs. The 3 outputs of the pipe splitter are each fed into thinner pipes that have a 1/4-inch outer diameter and a 0.17-inch internal diameter. Each of these pipes goes across the face of a speaker. At the point where the speaker touches the pipe (diametrical center), a hole is made in the pipe to allow water droplets to flow. The height difference between the bottom of the bottom tank and the pipe splitter is less than 20 inches, making the pump we chose ideal for this job. The bottom tank can hold water up to a capacity of 2 liters. The reason we chose this capacity is that it is a combined total of the volume of the drops required to run the device correctly. The number of drops of water in motion per stream is 16, the calculation for which is:

$$S = ut + 0.5gt^2 \tag{4}$$

For the first droplet $u \simeq 0$, therefore $S = 0.5gt^2$

Evaluating:

$$S = 0.5 * 9.8\frac{m}{s^2} * (\frac{1}{100}s)^2 = 0.049cm = 0.0193inches \tag{5}$$

Thus, the distance between each droplet increases by approximately 0.0193 inches in a 100th of a second. The distance between the pipe nozzle and bottom water tank is approximated to be 15 inches. The number of droplets in each 15 inch column is then calculated as follows:

$$\sum_{n=1}^{x} n * 0.019315 \leq 15 \tag{6}$$

x is the maximum number of droplets, the quantity we need to find. The maximum value of x that satisfies the above equation is 39.

We have 3 such streams, which add up to 117 drops of water. On average, we estimate 10 drops per ml [6] leading to the conclusion that about 12 ml of water is dripping at any instant in time. The bottom tank in actuality hold a little more than a liter of water so that there is enough depth that the water droplets fall into to prevent spillage due to splashing. The water pump must pump water to the top tank at least as fast as the water is dispensed from the system. The system is dispensing 3 drops every 100th of a second, or 300 drops every second. This approximates to a minimum of 30 ml/sec. Additionally, accounting for water to be forced to move horizontally across the face of the speakers for about 60cm, the pump must be able to exceed a speed of 65 ml/sec. The additional 35 ml/sec is given by the following equation:

$$V = \pi * r^2 * h = \pi * (0.004318)^2 * 0.6 = 35.18ml \tag{7}$$

The pump we chose easily meets these requirements by pumping water at about 100 ml/sec. The rate at which the pump functions can be adjusted via a knob on the pump.

HC-05 Bluetooth Module and Elegoo Mega 2560 Board: The HC-05 bluetooth module receives packets of data as bytes and transmits them to the Elegoo Mega 2560 Board [7]. Data can be sent to and from the ATMega 2560 microcontroller.

```
// Keep reading from HC-05 and send to Arduino Serial Monitor
if (BTserial.available()){
      b = BTserial.read();
      data_in = b;
      Serial.println(data_in);
}

if (Serial.available()){
  BTserial.write(Serial.read());
}
```

Figure 6: Code snippet detailing how data is being received and transmitted

The sendByteX(byte b) functions write the necessary data bits on the Xth sound generator chip. There are 3 sound generator chips. The function first writes the necessary data bits before enabling the write enable pin (active low), before disabling writing again.

```
void sendByte2(byte b)
{
  digitalWrite(PIN_NotWE_2, HIGH);
  delay(1);

  digitalWrite(PIN_D0_2, (b&1)?HIGH:LOW);
  digitalWrite(PIN_D1_2, (b&2)?HIGH:LOW);
  digitalWrite(PIN_D2_2, (b&4)?HIGH:LOW);
  digitalWrite(PIN_D3_2, (b&8)?HIGH:LOW);
  digitalWrite(PIN_D4_2, (b&16)?HIGH:LOW);
  digitalWrite(PIN_D5_2, (b&32)?HIGH:LOW);
  digitalWrite(PIN_D6_2, (b&64)?HIGH:LOW);
  digitalWrite(PIN_D7_2, (b&128)?HIGH:LOW);

  digitalWrite(PIN_NotWE_2, LOW);
  delay(2); // prevent races

  digitalWrite(PIN_NotWE_2, HIGH);
  delay(1);
}
```

Figure 7: The sendByte2(byte b) function writes byte ‘b’ to the 2nd sound Generator chip

Sound Generator: For our water unit, we had to produce frequencies in the range of 90 Hz - 110 Hz. We used a general SN76489 [9] sound generator chip (Datasheet in Appendix Section C. It generates sounds of frequencies given by the following equation [10] [11]:

$$f = \frac{N}{32 * n} \tag{8}$$

N is the frequency of the input clock, n is the desired frequency given by setting 10 bits (F9-F0) on the IC

We provided the chip a clock input of about 500 kHz from a signal generator. After a fair amount of testing, we arrived at the following frequencies we wished to work with - 98.27 Hz, 100.16 Hz, and 102.12 Hz. The binary bits chosen for these frequencies resulted in the integer values of 159, 156, and 153.

We had the following sound generator outputs for 97.6 Hz and 101 Hz corresponding to the upward and downward movement of water respectively:

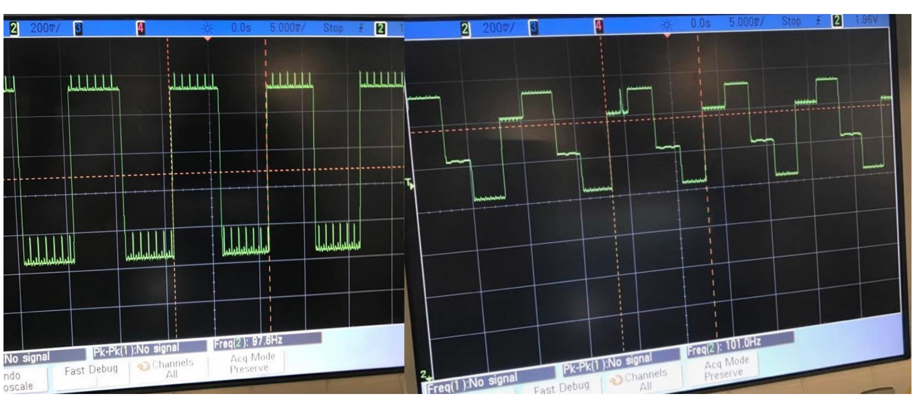


Figure 8: Sound Generator Outputs for Left) 97.6 Hz and Right) 101 Hz

As we can see in Figure 8 above, the outputs are approximately square waves of the frequencies that we need for our illusion. However, the oscilloscope was unable to detect the peak to peak voltages of these outputs, which can be translated to the signals being extremely weak. They also contain noise of the order of 440 kHz, and to tackle this issue we chose a speaker that filters out frequencies above 20 kHz. We would also not hear this since theses frequencies are above hearing range. The Dynavox LY401F has an asymptotic average 6 ohm impedance and a power handling RMS rating of 20 Watts [4].

Amplifier: The sound generator output, however, will not be able to drive our speaker. For this reason we needed to use an amplifier chip to construct a speaker driver. The supply voltages provided to this amplification circuit was initially chosen to be 25V to give us approximately 25W of power. This is because the amplifier has to be 25 percent more powerful than the speaker.

However, we realised that the frequencies we wished to operate at would provide very high impedance and not the aforementioned 6 ohms. This changed our calculations and led us to have a smaller voltage difference across the positive and negative terminals of our LM1875 chip.

Below in Figure 9 we have the schematic for our proposed circuit [13]:

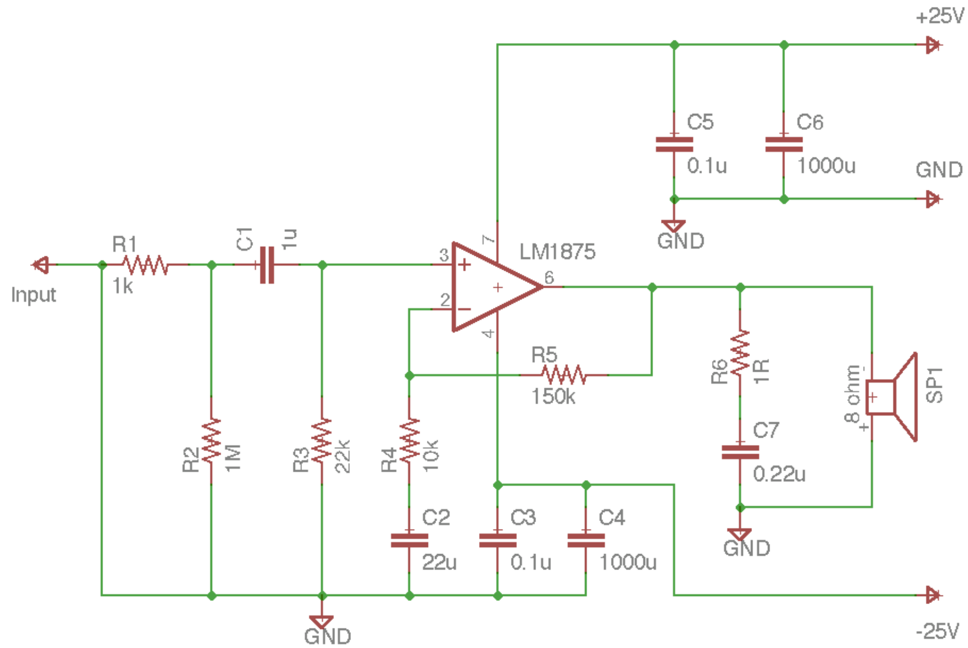


Figure 9: Circuit Schematic of audio amplifier circuit

We chose the LM1875 chip specifically since it provides low distortion at our operational frequencies. It has protection consists of both internal current limit and thermal shutdown. Other features is high gain, fast slew rate and a wide bandwidth, large output voltage swing, high current capability, and a very wide supply range [13].

Since this circuit involved high currents (up to $\sim 2A$) and high supply voltages in the order of $\pm$10V to $\pm$25V, we constructed this circuit on a perfboard. Even though our circuit worked with a supply potential difference of about 20V, a capacitor on our board blew up as we stepped up the difference to 30V. We performed the continuity test on the circuit after the first trial to make sure it was still intact, except the burnt capacitor. We then replaced it and ran the system again, however, we blew up another capacitor. We attribute this problem to having picked up the wrong capacitors or having soldered the wires incorrectly. We, however, did not have enough time to know where the problem was.

Speakers: For our speakers, we chose the Dynavox LY401F (Datasheet in Appendix Section B). We required speakers that would be able to vibrate our pipes effectively. At its maximum volume, we found that this speaker has a vibration amplitude of 1.55 mm which would serve our purpose. The speaker has an operational range of 50 - 20000 Hz and since we needed an output range of 90 - 110 Hz, this was ideal. At these frequencies we found that the impedance was a little higher than its rated 6 ohms at about 12 ohms. However, this was more acceptable than the 45 ohms at 60Hz that we had previously considered for the design review.

Since our amplifier was unable to serve the purpose of providing input to the speakers, we had to rely on signal generators to emulate our sound generator outputs. We provided the positive and negative terminals with a set sine wave of 100 Hz to see the output of the speaker. The vibrations were excellent and worked very well with the pipes of our choice. They were also very silent, which was an added feature. They however created some sound on coming in contact with the pipes due to the material of the pipes. If we used our original vinyl pipes, this would not be an issue.

The portion of the speaker that is in touch with the pipes to provide the required vibration is a conical structure that very effectively focuses all the mechanical movement at its tip. We punched holes into the pipe and held it right below this cone to get maximum displacement. This was better than having a spread out diaphragm which may have not been able to isolate all the movement at the exact point on the pipe.

Pipes: The pipes we initially chose were clear vinyl pipes that were very flexible and easy to wrap around our system. Unfortunately, these pipes did not fit perfectly into the pipe-splitter provided by the Machine Shop. Hence, these pipes ended up leaking water. We had to compromise and finish our project using pipes made of a less flexible and much harder material. The new pipes made much more noise while vibrating as compared to the previous vinyl pipes. Additionally, the water droplets produced were slightly more messy as compared to those produced by using the vinyl pipes. Ideally, we would use the vinyl pipe with the correct pipe-splitter.

User Interface: The app has a total of 11 buttons as part of the User Interface:

Figure 10: Screenshot of the User Interface

There is a button labeled ‘OPEN to find only the HC-05 Bluetooth module we chose and to open the Bluetooth connection between the two so that data can be transmitted back and forth. If the Bluetooth module has not been paired with the Android device beforehand, it will carry out this functionality too. There is also a button labeled ‘CLOSE to close the Bluetooth connection between the Android device and the Bluetooth module. Once the connection has been closed, no data will be sent or received.

There is a 3x3 grid of 9 buttons. Each column consists of 3 buttons, labeled ‘UP, ‘LEVITATE, and ‘DOWN’. Each column corresponds to one stream of water. Pressing one of the aforementioned 9 buttons will cause the corresponding stream of water to appear to move in the direction specified by the label. The default operation is to levitate the water droplets.

Confirmation messages and status updates are displayed in a text box. This is a useful tool for debugging and to know the state of the Bluetooth connection.

3 Verification

3.1 Lighting Unit

The verification for this portion of our Table 1 in the Appendix involved getting an output from the 555 circuit of 100 Hz and 10% duty cycle. The voltage level was expected to be $9V \pm 0.8V$. Below is an oscilloscope capture for the same:

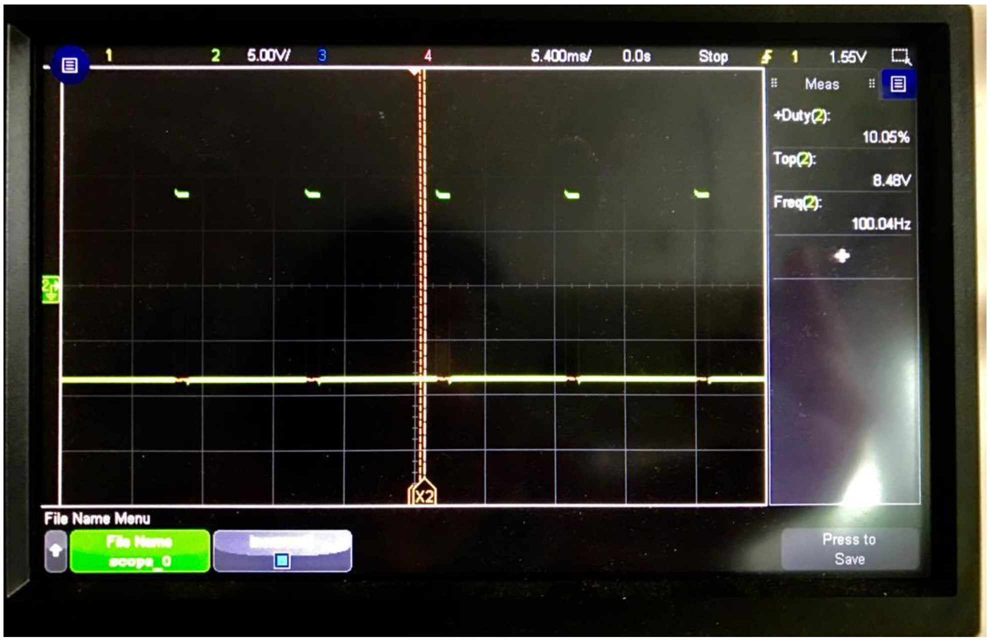


Figure 11: Light strobing frequency of 100 Hz with 10% duty cycle

We checked the current in each branch of the LED array and it was approximately 16mA and this was less than our desired limit of 20 mA.

The current output of the 555 chip was not very significant as to damage our transistor, which was capable of currents up to 1 A running through it. However, we did place a 10 kOhm resistor between the 555 and the transistor to make sure not to cause a breakdown within the latter. Due to this, the current entering the transistor was about 10 mA.

3.2 Water Unit and User Interface

The verification for the user interface involved pressing the app buttons and observing the corresponding effects as specified in the Appendix: Section A.

Clicking on the 'OPEN' button would cause the LED on the HC-05 to blink twice in quick succession followed by it being off for approximately one second. This indicated that a connection was successfully established. Clicking on the 'CLOSE' button would cause the LED on the HC-05 to blink very quickly, almost 5 times per second. This indicated that the HC-05 is ready to pair with a device, since it does not have an established connection.

Clicking on any of the other 9 buttons on the app would send bytes of data to the microcontroller via the HC-05. While the serial monitor was open, we verified the sent and received data by printing it to the screen on the app and on the serial monitor. Additionally, one button always sent the same information each time it was pressed, while each button sent different information. For example, when pressing the leftmost 'UP' button, the data sent was always the character '1'. Whereas if the rightmost 'UP' button was pressed, the data sent was always the character '7'. This way the microcontroller was able to differentiate the data sent to it and effect changes in the system accordingly.

The changes effected by the microcontroller were clearly visible by probing the output of the sound generator chip. On pressing a button, the output frequency would accordingly change. Another verification of the water unit working correctly was to turn on the strobing LEDs and alter the frequency of the tone sent to the speaker. Each new frequency setting in the 97 Hz - 103 Hz range changed the speed and direction of the movement of water droplets.

4 Costs

4.1 Labor

Table 1: Labor Costs

Name	**Hours Invested**	**Hourly Rate**	**Total cost** $= HourlyRate * HoursInvested * 2.5$
Atreyee	300	$20.00	$15,000.00
Siddharth	300	$20.00	$15,000.00
Total			$30,000.00

4.2 Parts

Table 2: Parts list and cost

Parts and Quantity	**Part Number**	**Unit Cost**	**Total cost**
LED (x30)	1080-1008-ND	$0.4496	$13.488
Timer IC (x4)	NE555P	$0.39	$1.56
Audio Amplifier IC (x3)	LM1875	$2.75	$8.25
Sound generator IC (x3)	SN76489AN	$0.94	$2.82
Bluetooth module	HC=05	$4.83	$4.83
Elegoo Mega 2560 R3	Elegoo Mega 2560 R3	$11.99	$11.99
Dynavox (x2)	LY401F	$19.99	$39.98
TotalPond Fountain Pump	MD11130	$21.97	$21.97
Transistor (x5)	FQP30N06	$1.09	$5.45
Wall Power Adaptor	iMBA-9V500MA-1PK	$8.79	$8.79
Total			$119.128

4.3 Grand total

Table 3: Labor Costs

Section	**Cost**
Labor	$30,000.00
Parts	$119.128
Total	$30,119.128

5 Conclusion

In conclusion, we gained a lot of experience through this project, both as a team as well as individuals. We learned some very important skills such as PCB design and soldering on the hardware side, as well as app development and microcontroller programming on the software side. Neither of us had any experience with arduino programming or bluetooth networking, and this was a good way to get hands-on experience in those areas. We had most of our project working and we were able to show the final effect with some necessary tweaks to our demo. We met almost all of the goals we had set for ourselves.

We faced a lot of difficulty initially with trying to figure out how to vibrate our speakers and had to go over multiple options such as basic transducers, bigger subwoofers and contraptions involving solenoids and magnets. However, once we figured that out, the rest of the design process was fairly straightforward. We did ample research for the parts we would need, and this ensured that most of our tests ran smoothly and successfully as well. Our PCBs passed the first revision itself and were fully functional.

We learned a lot over the course of the semester and even though we had trouble visualising the project due to its intuitive and experimental nature, we had a functioning demo and it was a very rewarding experience.

5.1 Safety Considerations

Safety was an important concern for us throughout this project and we had to change a lot of our design and materials used to account for this. We refrained from testing in any of the labs with water, and received access to one of the rooms in the basement with a drain and no electrical equipment to carry out all water related testing. We kept in mind the IEEE code of ethics [14] (1) while doing this: " ... making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might endanger the public or the environment", since our project may have endangered the environment or the safety of others who may be helping us.

We were also mindful of wastage and conservation of water, since this is a water heavy project. We used a water pump that can pump water to about 2 feet, and the entire system had 1.5L of water continuously circulating. This is in accordance with the IEEE code (1).

We also kept in mind the IEEE Code of Ethics (7): "to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others" and (9): "to assist colleagues and co-workers in their professional development…". These were important factors to ensure a smooth collaboration between the two groups involved in water aliasing this semester. We were respectful of their timings regarding our shared demo space.

We were also aware of all the resources available to us during the entire course of this semester and adhering to the ACM Code of Ethics and Professional Conduct [15] (2.4): " Accept and provide appropriate professional review", we will perform honestly and proficiently at peer reviews and provide our views as critically as possible. We took any and all critique pertaining to our work from our teaching assistants and professors to improve our product and this ensured a successful project.

5.2 Future Work

For our future work, we would definitely fix our issues pertaining to the amplifier circuit to achieve the "waterbending" we were looking forward to. We would also like to use better pipes, which are more flexible to keep in line with our design requirements. There is now gesture recognising technology available, and if we could integrate it with our design, we could control our streams of water with palm movements instead of an app and this would really add a "wow!" factor to this project.

References

[1] Amazon.com, 'Levitating Waters', 2017. [Online]. Available: https://www.amazon.com/Levitating-Waters-Plug-Funtime-Gifts/dp/B01CRU01PY/ref=sr_1_3?s=home-garden&ie=UTF8&qid=1486601806&sr=1-3&keywords=levitating+water [Accessed: 03-May-2017]

[2] Texas Instruments, 'LM 555 Timer Data Sheet', 2015. [Online]. Available: http://www.ti.com/lit/ds/symlink/lm555.pdf [Accessed: 03-May-2017]

[3] Circuit Basics, '555 Timer Basics - Astable Mode', 2015. [Online]. Available: http://www.circuitbasics.com/555-timer-basics-astable-mode/ [Accessed: 03-May-2017]

[4] MCM Electronics, 'Dynavox LY401F-M01 Data Sheet', 2015. [Online]. Available: http://www.mcmelectronics.com/content/ProductData/Spec\%20Sheets/55-5535.pdf [Accessed: 03-May-2017]

[5] Wikipedia.com, 'Splash', 2016. [Online]. Available: https://en.wikipedia.org/wiki/Splash_(fluid_mechanics) [Accessed: 03-May-2017]

[6] Wikipedia.com, 'Drop (Unit)', 2017. [Online]. Available: https://en.wikipedia.org/wiki/Drop_(unit) [Accessed: 03-May-2017]

[7] Fairchild Semiconductor Corporation, 'FQP27P06 P-Channel QFET MOSFET'. [Online]. Available: https://www.fairchildsemi.com/datasheets/FQ/FQP27P06.pdf [Accessed: 03-May-2017]

[8] Arduino-info.wikispaces.com, 'HC-05', 2017. [Online]. Available: https://arduino-info.wikispaces.com/BlueTooth-HC05-HC06-Modules-How-To [Accessed: 03-May-2017]

[9] Arduino.cc, 'Arduino Board Mega 2560 Rev3'. [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardMega2560 [Accessed: 03-May-2017]

[10] Members.casema.nl, 'SN76489 Sound Generator Chip'. [Online]. Available: http://members.casema.nl/hhaydn/howel/parts/76489.htm [Accessed: 03-May-2017]

[11] Dancewithferrets.org, 'Experimenting with an Arduino and a SN76489', 2014. [Online]. Available: http://danceswithferrets.org/geekblog/?p=93 [Accessed: 03-May-2017]

[12] Suzushoweb.com, 'Texas Instruments SN76489AN Datasheet'. [Online]. Available: http://www.suzushoweb.com/pdf_file/4644200529e19.pdf [Accessed: 03-May-2017]

[13] Eleccircuit.com, 'LM1875 Datasheet - 25W Hifi Audio Amplifier Circuit'. [Online]. Available: http://www.eleccircuit.com/25w-hi-fi-audio-amplifier-using-ic-lm1875/ [Accessed: 03-May-2017]

[14] Ieee.org, 'IEEE Code of Ethics', 2016. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html [Accessed: 03-May-2017]

[15] Ethics.acm.org, 'ACM Code of Ethics and Professional Conduct', 1992. [Online]. Available: http://ethics.acm.org/code-of-ethics/ [Accessed: 03-May-2017]

Appendix

Section A Requirements and Verification Table

Requirement	Verification	Points
Module: Strobe light		
1) The duty cycle of the output from the 555 timer circuit must be 10%±3% for good strobing effect.	a) Choose resistor values such that discharging resistor has a value 9 times the charging resistor. b) Set up oscilloscope probes to measure the output voltage at pin 3 of the 555 chip to verify this duty cycle. c) Use divisions of oscilloscope to calculate if time for which output pulse is high = 10±3% of 0.01667s.	8
2) The frequency of the output wave must be constant at 100 Hz±0.8Hz.	a) Choose resistor and capacitor values, while keeping ratio between resistors R1 and R2 constant at all times. (9:1) b) Check frequency of output wave at pin 3 of 555 to verify that frequency lies between 100±0.8 Hz.	7
3) Current through each branch of the LED array must never exceed 20mA (peak brightness). We will use 150 ohm limiting resistor in each branch to achieve this.	a) Use DMM to measure voltage drop across LEDs in each branch. This should be 3.2±0.2V for each LED. b) Measure the current in each branch with the DMM. This will be 17.33±1mA. c) Total current at the drain of the transistor will be 173.33±10mA. This can be verified with the DMM by probing at the drain.	6
4) High current must not enter the	a) Probe at the gate of the NMOS	3

transistor gate from the output of the 555. We will place a 10k ohm resistor before the gate to ensure this.	transistor to check current at gate. This must be <20mA.	
Module: Water Unit <u>Sub module: Water pump and Tank</u>		
1) Both water tanks must not leak any water.	a) Fill both the water tanks to maximum capacity and check if any water leaks out.	0
2) The bottom water tank must be resistant to splashes, to prevent the water droplets from splashing outside the tank.	a) With just the skeletal structure of the tank (without any electrical component), drip drops of water from a height of 15 inches and check to see if any water spilled outside the bottom tank.	1
3) The total amount of water in the device must never exceed 200ml. Create an overflow point at the 200 ml mark inside the tank, beyond which water is emptied from the tank.	a) Add water to the tank until user sees the water overflowing to an external location (like a drain) without the risk of the water expelled coming in contact with any part of the device.	0
4) The water pump must be able to pump water at the rate of at least 18ml/sec.	a) This comes from the data sheet or specs of the pump purchased.	0
<u>Sub module: Microcontroller</u>		
1) Must be able to send data to sound generator chip and do so without causing distortion due to presence of old data while setting new data on sound generator chip.	a) Using testing LEDs, see that the correct LEDs turn on when setting data bits to high or low b) Measure the output of the SN76489	3

	in response to changing data sent from microcontroller.	
2) Must be able to unpack and read data correctly from Bluetooth module.	a) Test using laptop, connect to Arduino and print data (characters) to screen relayed from HC-05 to Arduino.	2
<u>Sub module: Sound Generator</u>		
1) Must be able to generate sounds of frequencies 90 Hz to 110 Hz.	a) Set the data bits and input clock to theoretically produce a frequency of 100 Hz. Check the output on the Oscilloscope. Repeat for the range using same input clock.	3
3) The frequency must be able to change in less than 0.5 seconds of pressing a button on the app.	a) Measure the time between pressing the button and seeing a visible change in the movement of water droplets.	2
<u>Sub module: Audio Amplifier</u>		
1) It should not cause too much distortion (<0.2 Hz) for audio input in the frequency range 90-110 Hz.	a) Run the output of the amplifier through a network analyzer (VSA) and test the signal. b) To check the stability of the frequency, use the continuous peak search option on the VSA and reduce the resolution bandwidth and span. c) Observe the largest frequency drift, it should be on the order of 0.2 Hz.	8
<u>Sub module: Speakers</u>		
1) Speakers must be able to produce frequencies in range of 90-110 Hz.	a) This comes from the datasheet of the speakers [13].	0
2) Speakers must be able to vibrate the pipe with water. It should not be vibrating	a) Play speaker on full volume, attach a pipe and see if it vibrates with a large	2

with a large amplitude such that water sprays outside of the physical framework built.	amplitude. If the vibration is too large, reduce volume. Test again with water (This time increase the volume because with the added weight of water, the pipe will vibrate less).	
Module: User Interface App 1) Must be able to send data regarding position of slider to Bluetooth module. 2) Each button must cause changes in strictly only their corresponding pipes.	 a) Print data sent from phone to screen and print data received by Bluetooth module to screen. a) Press the button for one pipe and print the data received. Press the button for another pipe then. Check if the data received is different.	 2 3
Total		**50**

Section B Speaker Datasheet

DYNAVOX

Model #: LY401F-M01

ITEMS:	SPECIFICATION:
Diameter	4”
Impedance	6 ohms
Input	RMS 20 W, MAX 40 W
Free air resonance	50 Hz
Sensitivity	88 dB ± 2 2.45 V
Frequency range	50 ~ 20000 Hz
Sine wave test	5.5 V
Weight	1.76 lbs.

Basket: die cast (aluminum)

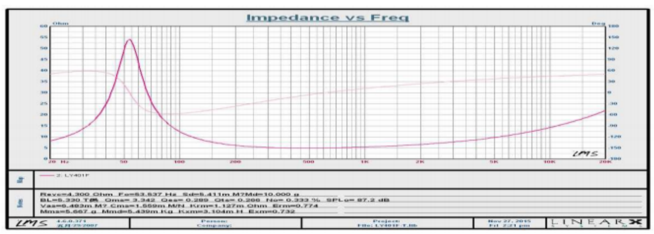
VOICE COIL:		
Diameter	25.4	(mm)
Winding Length	7.1	(mm)
Layer	2	
Former material	Kraft	KSV
Wire material	Copper	SVW

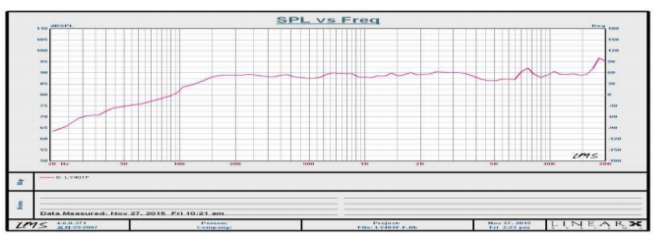
CONE:	Treated paper
SURROUND:	Treated cloth

MAGNET:	(1)	(2)	
Diameter	85		(mm)
Height	15		(mm)
Material	Ferrite		Y30
Quantity	1		pcs
Weight	354		(grams)
Gap(H)	4		(mm)

PARAMETERS:

DC resistance	Re:	4.3	(Ω)	Moving mass	Mms:	5.66	(grams)
Resonance frequency	Fs:	53.53	(Hz)	Equivalent volume	Vas:	6.48	(l)
Maximum impedance	Zm:	54.2	(Ω)	Sensitivity	SPL:	87.2	(dB)
Mechanical Q factor	Qms:	3.34		Suspension compliance	Cms:	1.56	(mm/N)
Electrical Q factor	Qes:	0.29		BL product	Bl:	5.33	(N/A)
Total Q factor	Qts:	0.26		Driver piston diameter	D:	83	(mm)
Linear Displacement	Xmax:	1.55	(mm)	Voice Coil Inductance	Le(1K):	0.38	(mH)





Data Presented by: MCM Electronics

Figure 12: Datasheet for Speaker

Section C Sound Generator Datasheet

8. **PIN ASSIGNMENT**

The table below defines the SN76489AN pin assignment and describes the function of each pin.

SIGNATURE	*PIN*	*I/O*	*DESCRIPTION*
$\overline{CE}$	6	IN	Chip Enable – when active (low) date may be transferred from CPU to the SN76489AN.
DO(MSB)	3	IN	D0 through D7 – Input data bus through which the control data is input.
D1	2	IN	
D2	1	IN	
D3	15	IN	
D4	13	IN	
D5	12	IN	
D6	11	IN	
D7	10	IN	
VCC	16	IN	Supply Voltage (5V nom)
GND	8	OUT	Ground Reference
CLOCK	14	IN	Input Clock
$\overline{WE}$	5	IN	Write Enable – when active (low), $\overline{WE}$ indicates that data is available from the CPU to the SN76489AN.
READY	4	OUT	When active (high), READY indicates that the data has been read. When READY is low, the microprocessor should enter a wait state until READY is high.
N.C.	9		No external connection should be made in this pin.
AOUT	7	OUT	Audio Drive Out

Figure 13: Datasheet for Sound Generator