Music-Visualization and Motion-Controlled LED Cube

Team 34: Hieu Tri Huynh, Islam Kadri, Zihan Yan

ECE 445 Project Proposal – Spring 2018

TA: Zhen Qin

1 Introduction

1.1 Objective

Our project's main inspiration came from an art piece called *Kinetic Rain* at Singapore's Changi Airport. The sculpture cycles over 16 pre-programmed shapes, through fluidlike movement. [1] The construction of the sculpture was made using over 1,200 bronze droplets, steel wires to hold the droplets, and cost several million dollars. [2][3] Although *Kinetic Rain* looks visually pleasing, we wanted to implement something that not only had visually pleasing animation but was based on real time sound input. The *Kinetic Rain* device's structure seems very complex and cumbersome to fix if any issues were to arise. Also, we wanted to implement this in a manner that would be very cost effective compared to *Kinetic Rain*.

Additionally, we'd like the user be able to interact with the LED grid to affect the animations within the device. By taking something that is traditionally 2-dimensional and implementing a 3-dimensional version, we can show off the benefits of user interaction as well as increase the entertainment factor of the device. We plan on doing this by making a Snake game that uses proximity sensors to control the snake's direction in a 3D manner.

Our team decided to build a device that took advantage of LED technology's efficient and aesthetically pleasing properties. We are going to build an 8x8x8 LED cube with two features: Music Visualization and a 3D Snake Game. The user will be able to switch between 2 modes (Music mode and Gaming mode) to use these 2 features.

For the first feature (Music Visualization), the device takes sound input from microphones and executes algorithms to extract the following information: frequency spectrum, amplitude, and direction of arrival. Then, it will display the information on the LED cube in a specific way, which will be discussed later. In order to determine the direction of arrival, we assume that there is only one sound source, otherwise the problem will become overly difficult to solve. Furthermore, please note that by saying determining the direction of arrival, we do not mean localizing the sound source or estimating the angle of arrival, which are much more difficult to solve and not suitable for our project. Instead, we mean that we will determine which side of the four surrounding faces of the cube is facing the sound source by comparing the amplitude of signals that is collected by microphones.

For the second feature (Snake Game), we will implement a 3D version of Snake. The device will display the Snake on the LEDs cube by turning off all LEDs except LEDs that represent the Snake and the LED for the food. The device captures the direction of user's hand movement when they move their hand above an array of proximity sensors. The device will then use this information to determine the motion and position of the Snake and map it to the LED grid.

1.2 Background

The *Kinetic Rain* sculpture's may be aesthetically pleasing but comes at a steep cost. The sculpture has a price tag of over $7 million and has expensive, unique parts that are hard to replace. [2] Our device would also be more cost efficient and be easy to fix if something were to go wrong with any of the pieces. Additionally, the sculpture took around 2 years to put together. [3] The *Kinetic Rain* was damaged on November 2nd, 2013 by a woman who climbed and hung onto the bronze droplets. The repair took engineers several months and was very costly. [4]

With our device, we'd have a low cost and easy to repair LED grid that would use live sound input to affect the animations, so visitors may be exposed to new unique animations based real time input instead of pre-programmed ones. Our device would be more scalable because of its modularity and cost. The tax payer burden, for example, would be less if a government municipality were to commission our device versus something like *Kinetic Rain.* Although there are devices that can display aesthetically pleasing animations, ours uses three sources of input to affect the animation. Our device will also have an interactive mode to allow users to affect animations through an entertaining game like Snake. The LED Cube ECE 445 project from the Fall 2013 semester used their phone application to affect user input. We will expand on this in terms of number of inputs, larger LED grid size, as well as improve the way a user interacts with the LEDs.

1.3 High-Level Requirements

- Device correctly extracts the information from the sound.
- The LEDs cube can reflect the frequency, amplitude, and direction of arrival of the sound.
- Device can correctly pick up the user's hand motion and properly control the Snake based on that information.

2 Design

2.1 Block Diagram

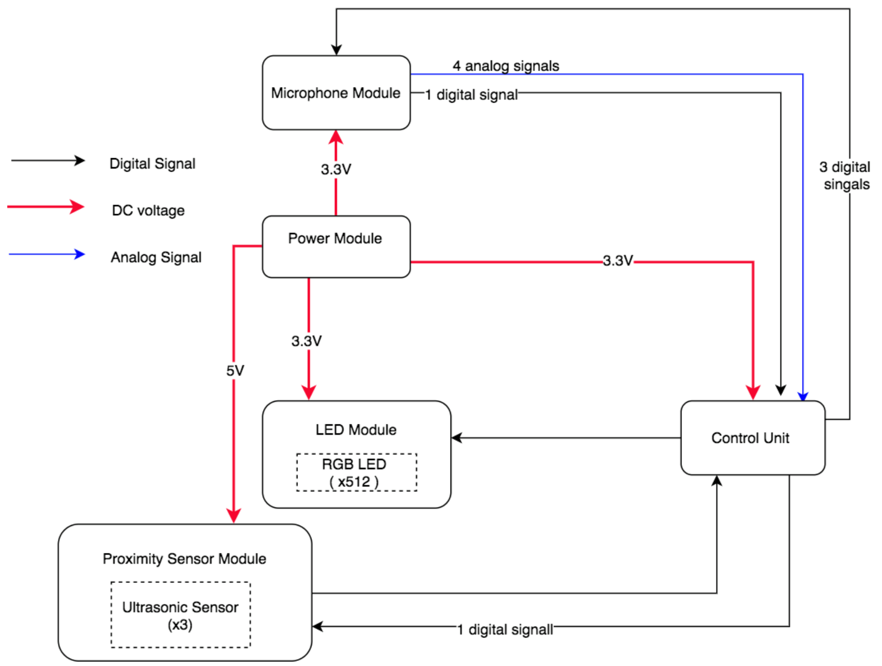


Fig. 1. Block Diagram

There are 5 modules in our design. First, the Microphone module will have 1 high quality MEMs digital microphone and 4 electret microphones. These microphones will be able to pick up the sound and send them to the Control Unit, which will extract the information of the signal (frequency spectrum, amplitude, and direction of arrival). With this design, we will satisfy the first high-level requirement. Next, the Control Unit will output 4 digital signals to drive the 512 LEDs in the LED Module. By doing this, the LED cube can reflect the information of the incoming sound, which satisfies the second high-level requirement. Finally, by having 3 Ultrasonic Sensors in the Proximity Sensor Module arranged in a specific way, the Control Unit will be able to determine the user's hand motion and properly drive the LED module. This will satisfy the third high-level requirement. In conclusion, our design will satisfy all three high-level requirements.

2.2 Physical Design

The LED module consists of 512, 5mm RGB LEDs, which are arranged into an 8x8x8 grid. We plan to arrange the LEDs such that the horizontal and vertical distances between any 2 LEDs is 1.25 inches. This will be placed on a flat rectangular surface approximately 24x18 inches^2, which is also what our sensors will be attached to. The cube will be placed in the upper middle part of the whole design with dimensions of 8.75 inches x 8.75 inches x 8.75 inches. We are going to place the microphones 4 inches away from the three edges of the board. The fourth edge is 2 inches away from the proximity sensor group, which also contains the fourth microphone. Three proximity sensors are arranged in a triangular fashion with a distance of 0.75 inches away from each other. This group's dimensions on the board is 2 by 4 inches^2. The diagrams of the physical design are shown below:

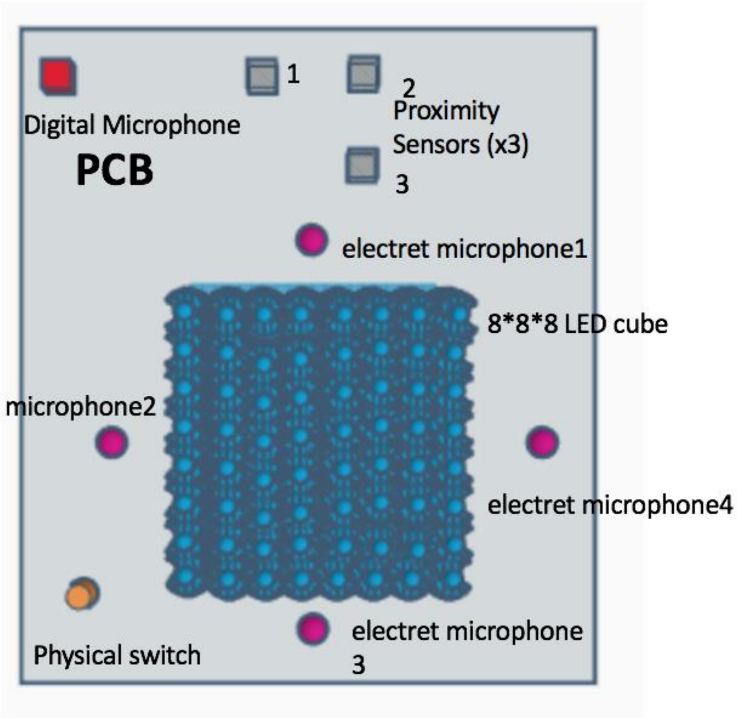


Fig. 2. Physical Design Top View

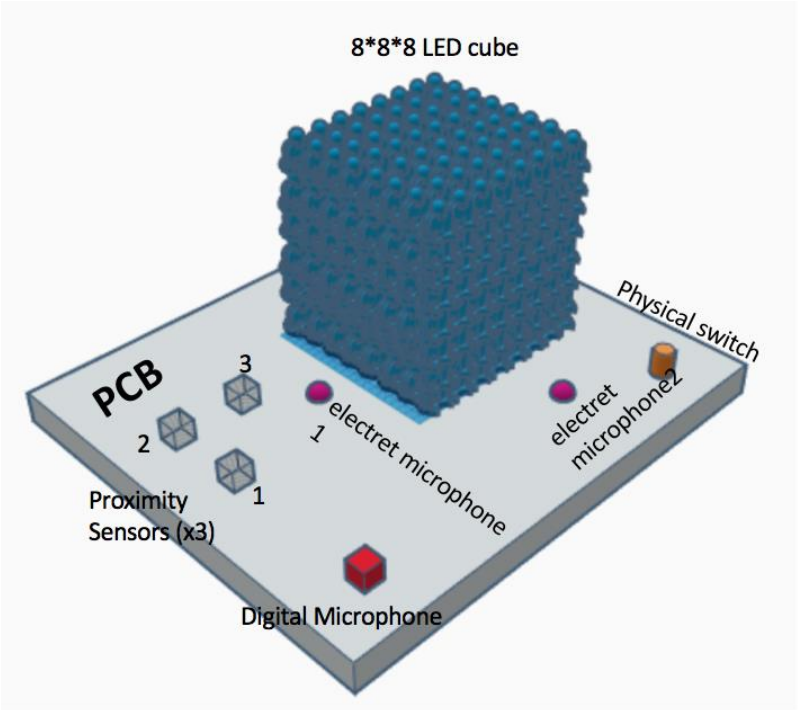


Fig. 3. Physical Design Right Side View

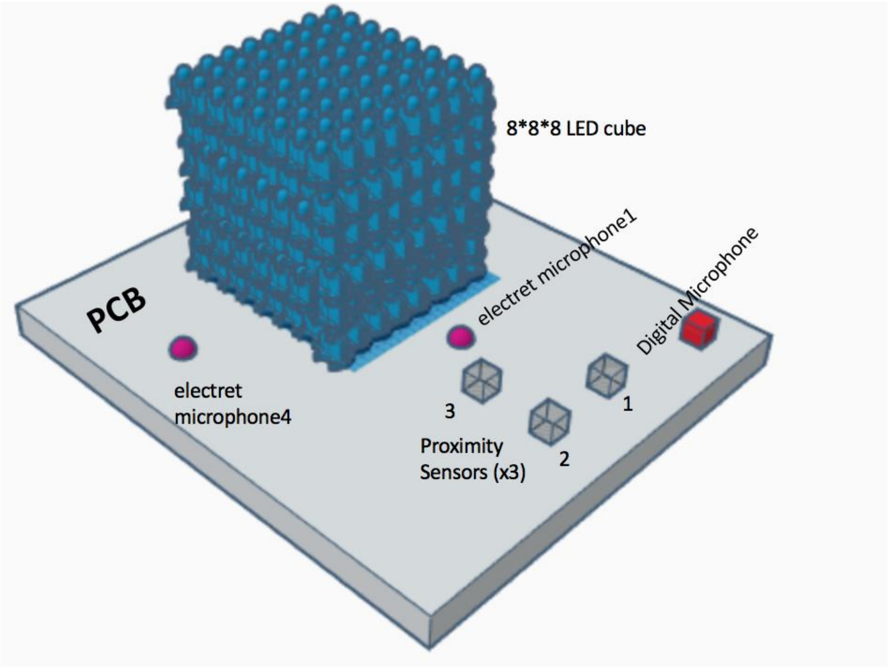


Fig. 4. Physical Design Left Side View

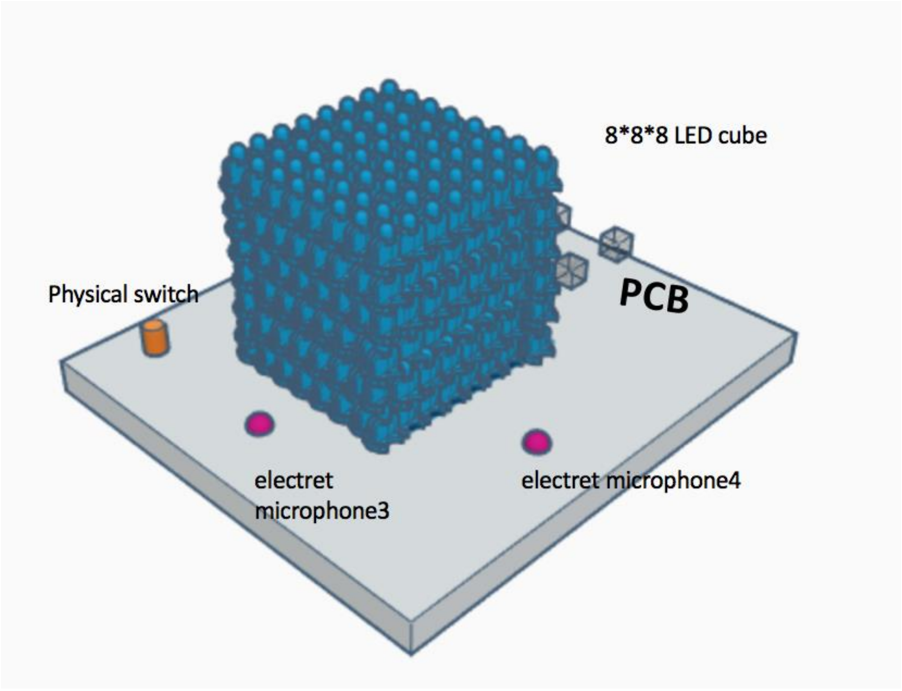


Fig. 5. Physical Design Back Side View

2.3 Functional Overview

2.3.1 Microphone Module

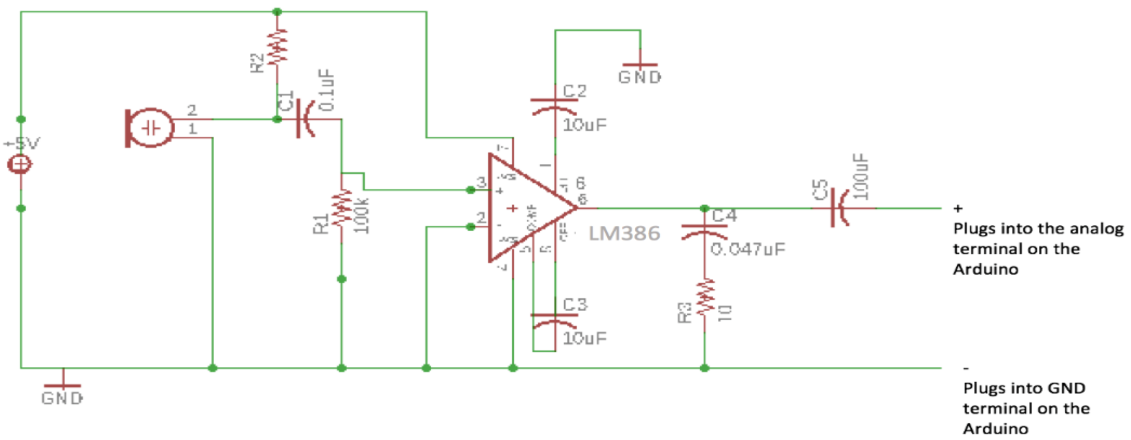
Components:

- 4 electret microphones (CMA-4544PF-W)
- 4 amplifier circuits
- 1 digital MEMS microphone (ICS-43434)

This module is used to collect sound from the environment and send the signals to the Control Unit. The Microphone Module must output signals so that the Control Unit can extract the 3 information: frequency, amplitude, and direction of arrival.

For the Control Unit to extract the frequency and amplitude, we need a high-quality microphone to capture sound so that the Control Unit can correctly extract the frequency and amplitude from the sound. Therefore, we decided to use just one MEMS microphone. We chose a digital MEMS microphone over an analog MEMS microphone because it will reduce the probability of error and complexity of the circuit. Specifically, we would need to choose the proper amplifier and ADC if we use an analog microphone. This will increase the probability of errors occurring and the Control Unit will not be able to extract the correct information. We chose the ICS-43434 digital MEMS microphone because it has frequency response range from 50Hz to 20kHz, which covers most of the frequency range of audible sound [5]. Furthermore, that model has a high Signal-to-Noise Ratio of 64 dBA, high sensitivity of -26 dB with +/-1 dB sensitivity tolerance, and sample rate range from 23 – 51.6 kHz. Because the highest frequency of audible sound is 20 kHz, we will set the microphone sample rate to be 40kHz, which satisfies the Nyquist rate [6]. The output of this microphone provides 24-bit data in I^2S format, which will provide high quality sound for the Control Unit to process.

To extract the direction of arrival to the Control Unit, we will need to have an array of at least 3 microphones so that the Control Unit can compare the amplitude of signals from these microphones to determine the direction of the arrival. The signals from these 3 microphones don’t have to be of high quality because we only compare their amplitudes. Therefore, we decided to use electret microphones which are cheaper than MEMS microphones. We also decided to use 4 electret microphones instead of 3 electret microphones in order to increase the accuracy. The positions of the 4 electret microphones are shown in the Physical Design section. The reason we cannot utilize the digital MEMS microphone mentioned above as one of the 4 microphones in the microphone array is that the Control Unit wouldn't be able compare the amplitudes of the digital and the electret microphone together. We decide to use the CMA-4544PF-W electret microphone because it has an applicable frequency range (20Hz to 20 kHz), reasonable sensitivity (-44dB), high SNR (60dBA), and is inexpensive. The output of these microphones needs to be amplified before going to the Control Unit.



Plugs into the analog terminal on the Arduino

Plugs into GND terminal on the Arduino

Fig. 6. Amplifier Circuit

Input	Output
1. Sound from the environment 2. 3 digital signals (WS, LR, SCK) to drive the digital microphone	1. Serial data for I2S interface going to the Control Unit 2. 4 Analog Signals going to Control Unit

Requirements	Verification
1. Digital MEMS microphone must have sampling rate at least 40 kHz	1. Use the Control Unit to count the number of samples received in on second.

2. Analog signals from 4 electret microphones must be in range 0-5V after begin amplified	2. Use an oscilloscope to measure the amplitude analog signals
3. Analog signals must change according to sound	3. Playing the sound with different amplitude and measure the output voltage

2.3.2 Proximity Sensor Module

Components:

- 3 Ultrasonic Sensors (HC-SR04)

The Proximity Sensor Module is used to pick up the user’s hand positions to provide information for the Control Unit to make decision about the user’ hand motion. The Control Unit will determine the user’s hand motion based on the change in distance between the user’s hand and the sensors. We will discuss the method in greater detail in the Control Unit section. This module must output the signals so that the Control Unit can calculate the distance between a user's hand and each sensor when the user's hand is in the specified range of 3 cm to 1 m. Based on the requirement, we choose to use an Ultrasonic Sensor because it can measure the distance of object in range from 2cm to 4m with a high accuracy (about 3mm). Furthermore, it can work well regardless of the lighting condition since our device might be used in both low and normal light conditions. We need 3 Ultrasonic Sensors because we want to detect the motion in 6 directions. These 3 Ultrasonic Sensors are located at certain positions so that we extract the motion of the user’ s hand.

Input	**Output**
1. Digital signal from Control Unit into TRIG_IN pins	1. Digital signal from ECHO_OUT to Control Unit

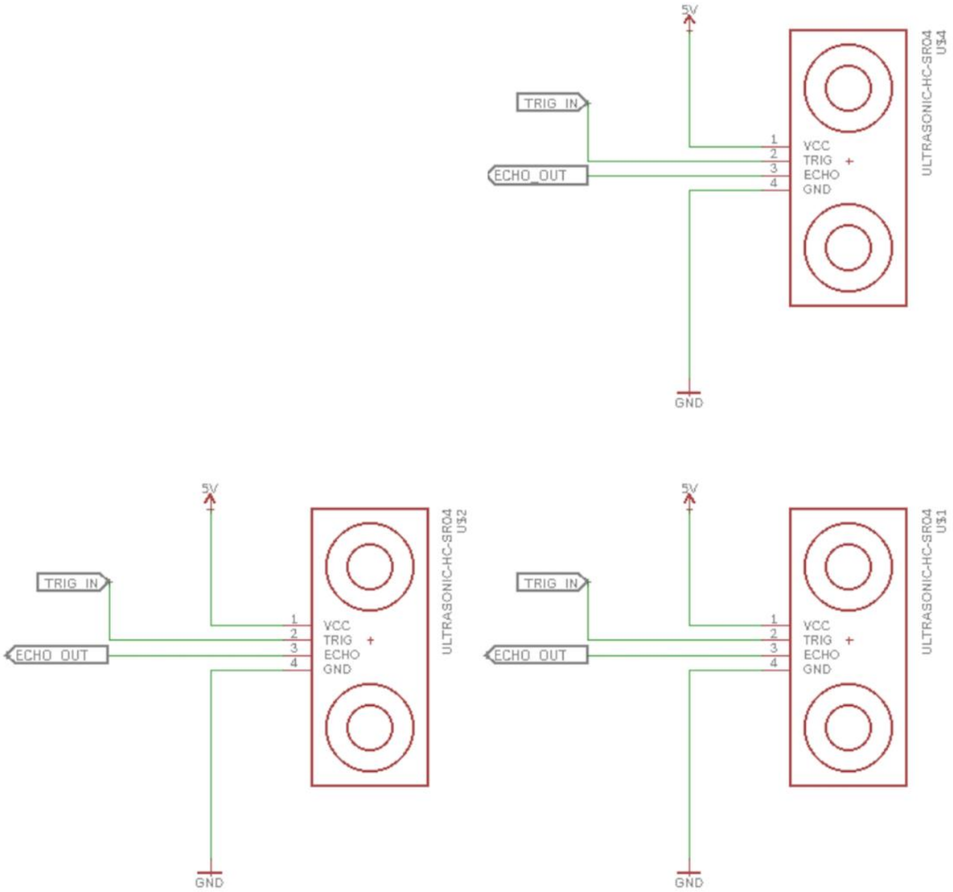


Fig. 7. Proximity Sensors Circuit

Requirements	Verification
1. Ensure that sensors can detect the position of hand when hand is in range of 3cm to 1m. (Hand is not moving).	1. Using Arduino to measure and calculate the distance between hand and sensors when hand is placed at 8 positions: 3cm, 20cm, 40cm, 60cm, 70cm, 80cm, 90cm, 100cm.
2. Ensure that sensors can detect the position of hand when hand is moving and the distance is in range of 3cm to 1m.	2. Using Arduino to measure and calculate the distance between hand and sensor when hand is moving at normal speed and the distances between hand and sensor are : 3cm, 20cm, 40cm, 60cm, 70cm, 80cm, 90cm, 100cm.

2.3.3 LED Module

Components:

- 512 5mm RGB common cathode LEDs
- 25 8-bit Shift Registers (SN74HC595)

This module is used to display the frequency spectrum of the sound inputted and the Snake game. Each LEDs has 4 pins, so, we have a total of 512*4 = 2048 pins. Instead of having 2048 outputs from the controller unit to control these LEDs, we will use 25 8-bit shift registers to control these 512 LEDs. These registers are connected in the way such that the data from one register will shift out to the next one, i.e. data from the 1st register will shift out to the 2nd register, data from 2nd register will shift out to the 3rd register, and so on. The circuit of the first 4 shift registers is shown in Fig. 8. We only show 4 shift registers in our figure because showing all 25 registers will make the picture hard to read.

Connecting LEDs:

We will connect all 64 cathode pins of the 64 LEDs on the same layer together and use the 1st shift register to control the status of these 8 layers (required for all 512 LEDs). Then, for LEDs on different layers but with the same (x, y) coordinates, their Red anodes, Green anodes, and Blue anodes will be connected together. The 64 Red anodes will be controlled by the next 8 registers. The 64 Green anodes will be controlled by the next 8 registers. The 64 Blue anodes will be controlled by the last 8 shift registers. The circuit of 8 LEDs with same (x, y) coordinates is shown in Fig. 9. The circuit of 9 LEDS on layer 0 is shown in Fig. 10. We only show part of the circuit because showing all 512 LEDs will make the picture very difficult to read.

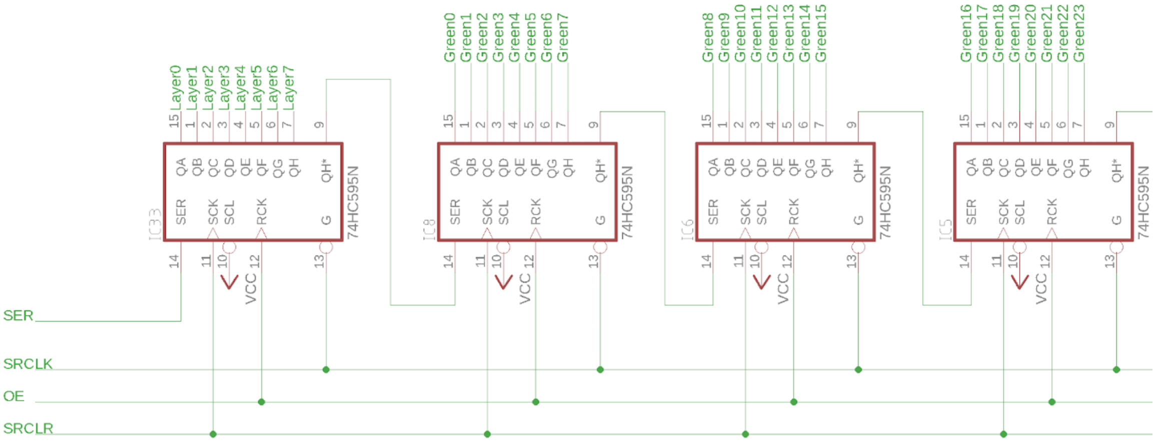


Fig. 8. Circuit of The First 4 Shift Registers

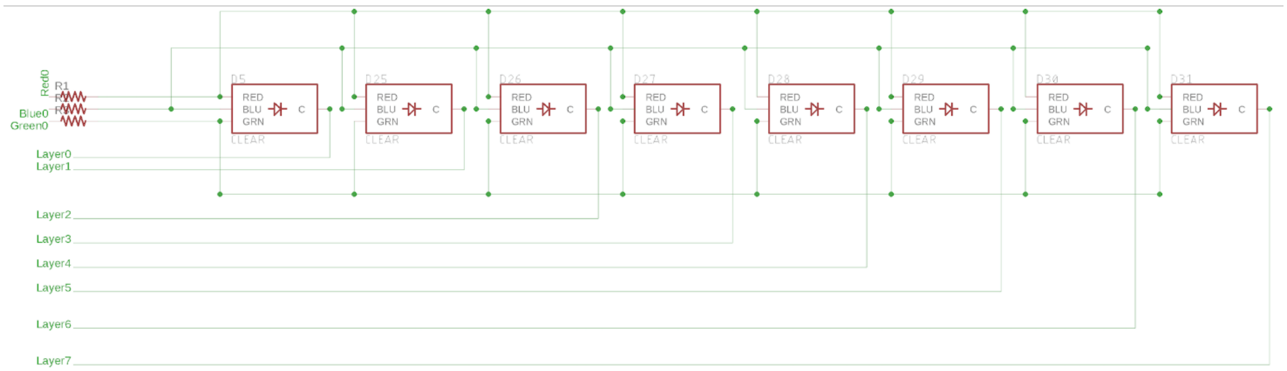


Fig. 9. Circuit of 8 LEDs with Same Horizontal Coordinates on 8 Layers

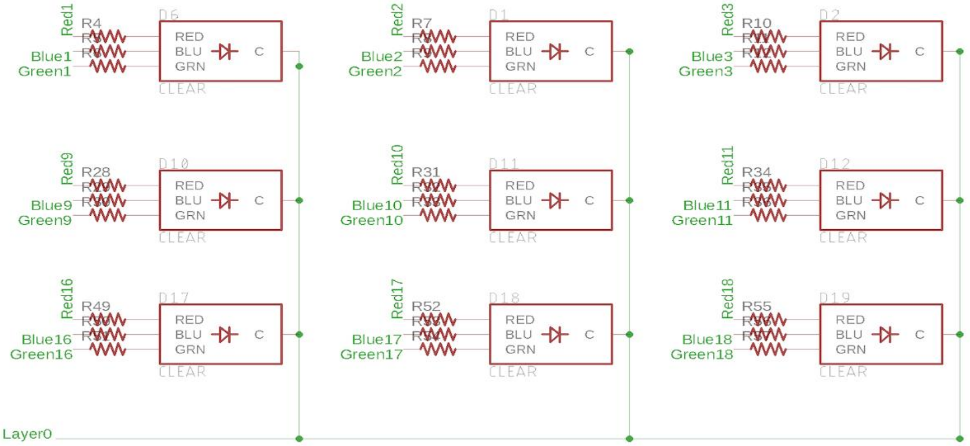


Fig. 10. Circuit of 9 LEDs on Layer 0

Input	Output
1. 4 Digital signals (SER, SRCLK, OE, and SRCLR) from Control Unit	1. Display LEDs

Requirements	Verification
1. LEDs is ON/OFF correctly based on Control Unit input signals. 2. There is no flickering on the LEDs. Animation appears smoothly.	1. Use Arduino to turn on and off each LED to check if LED is follow by the signal from the Arduino. 2. Turn on 1 LED and observe to see if there is any flickering or not.

2.3.4 Power Module

Components:

- Adaptor (dfds25212)
- 5 V Voltage regulator (7805 TO-220)
- 3.3V Voltage regulators (LD1117-3.3 TO-220)

This module is responsible for supplying power to all components of the device. Because our device has 512 LEDs and will consume lots of power, we decided to use power directly from the wall outlet. Therefore, we will need an adaptor to convert the 110VAC to a desired VDC to supply our other modules. This module contains a DC power adaptor, which will take 110 VAC from a wall outlet and output 12 VDC with 4A current. This power module will provide 3.3V to LED module, Microphones module, and Control Unit, and supply 5V to the Proximity Sensors Module.

For the LED module, each single-color LED inside a RGB LED's typically operate at 20mA, meaning that each single RGB LED needs 60mA. Although we have 512, we only turn on 1 layer of 64 LEDs at a time, as discussed in the LED module. Therefore, the total current is 64 × 40mA = 3.84A. Therefore, the total power that is needed for LED module is 3.3 × 3.84 = 12.672W. Besides that, the Microphone Module consumes 55mW, the Proximity Sensor Module consumes 49.5mW, and the Control Unit consumes 2.64W. Adding up all the power consumed by all modules, we have the total power consumption is 15.4165W. Therefore, our adaptor can supply enough power for the whole system.

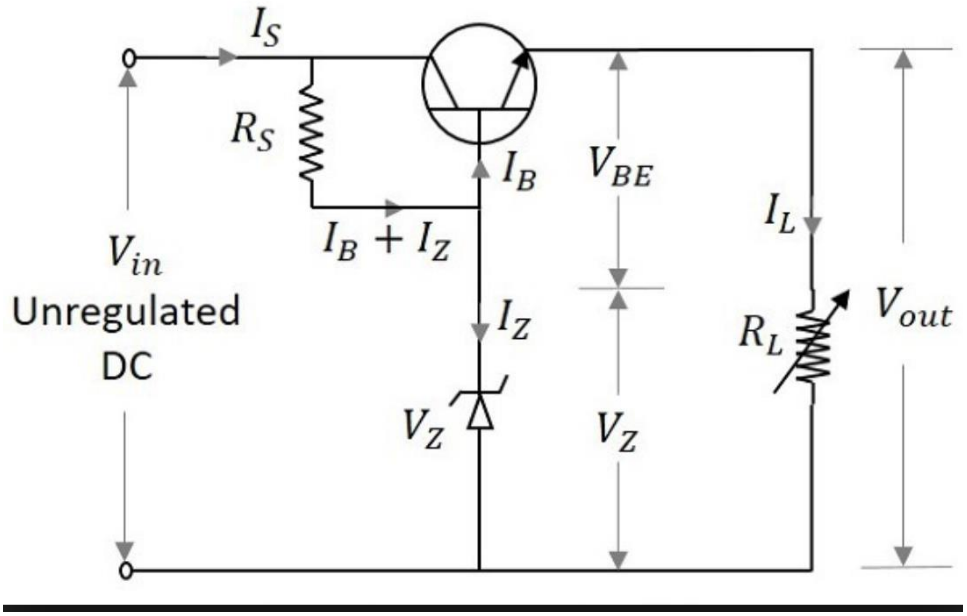


Fig. 12. Transistor Series Regulator circuit

We are going to use five voltage regulators that each supply 3.3V to the system and one 5V voltage regulator to supply 5V directly into the proximity sensors. The five 3.3V regulators in parallel will take in the output of 12V adaptor and have that output go to all the modules in our design except proximity sensor module.

As shown in the circuit above, the circuit consists of an NPN transistor and Zener diode. The majority of the input voltage will be distributed across the transistor, thus keeping the output voltage consistent.

Input	Output
1. 110 VAC	1. 3.3V goes to control unit 2. 3.3V goes to microphone module 3. 5V goes to proximity sensor module 4. 3.3V goes to LED module

Requirements	Verification
1. Supply enough power for each module. • 12.672W for LED module • 55mW for Microphone module • 49.5mW for Proximity Sensor module • 2.64W for Control Unit	1. Use oscilloscope to measure the short circuit current and Voc. Then calculate power using P = V×I.
2. Supply 5±0.5V output voltage to Proximity Sensors; 3±0.5V to all other modules. The ripple must be smaller than 0.5V.	2. Measure the voltage that goes to each module.
3. Supply voltage and current to each component in operational range.	3. Measure the voltage and current that go to each component and juxtapose with datasheet requirements. • LED: 3-3.4V and 20mA • Arduino Zero: 3.3V and 800mA

	Ultrasonic Sensors: 3.3V and 15mA Digital Microphone: 3.3V and 490uA Electret Microphone: 3.3V and 0.5mA

2.3.5 Control Unit

Components:

- Arduino Zero
- Physical Button

The Control Unit is used to perform calculations based on inputs from the Microphone Module and the Proximity Sensor Module to output the proper signals to drive the LED Module. We decide to use the Arduino Zero for the following reasons. Firstly, the Arduino has great libraries that we can utilize to help optimize the software and hardware processes. Secondly, we choose the Arduino Zero instead of the older Arduino Uno due to the Zero's increased computational power so that we can run more complex computations. Furthermore, the Arduino Zero supports I2S interface which allows us to connect directly to our digital MEMS microphone. The physical button is used to allow the user to switch between 2 modes (Gaming Mode and Music Visualization Mode). Based on the current mode, the Arduino will handle inputs differently and output the signals to drive the LED module.

Music Mode

When the device is in Music Visualization Mode, the Arduino uses 1 digital signal and 4 analog signals received from the Microphone module to extract the frequency, amplitude, and direction of arrival.

To calculate frequency spectrum of the signal, we will use an FFT algorithm that uses 256 samples. The reason we choose a 256-point FFT is that this will give us the frequency resolution of Fs/256 = 40000/256 = 156.25 Hz [7]. The resolution can be increased by collecting more samples before applying FFT. However, because our device has limited space, we can only display 8 ranges of frequency from 50Hz to 20kHz. Thus, there is no point in increasing the resolution. We decide to divide the frequencies into the following 8 ranges: [50 - 200], [201 - 800], [801 - 1200], [1201 - 5000], [5001 - 8000], [8001 - 11000], [11001 - 15000], [15001 - 20000]. After calculating the FFT, for each range listed above, we will pick the amplitude of the most dominant frequency (highest amplitude) in that range. After collecting 8 values from 8 ranges, we will use them to calculate the height of LED columns represented by those 8 ranges. Let the 8 values be x_1, x_2, … x_8, we will calculate the height of 8 columns by using Eq.1.

$$\text{height}[i] = 8 \times x_i/\max(x_1, x_2,\dots x_8) \qquad \text{Eq.1}$$

Next, we will calculate the amplitude and use it to control LED color. We will find the maximum value of each of the range data received from the Microphone module. Let m be the maximum value of this 256-point data and v_old, which initially equals 0, is the value that is currently used to set the color of the LEDs. We update the v value based on m by using formula Eq.2. By using this formula, we can prevent fluctuations caused by noise while being able to update the v value quickly. Based on the datasheet of the digital microphone [8], the output value will have a range from 0-2V. We will divide this range into 6 subranges: [0 – 0], (0 – 0.4], (0.4 – 0.8], (0.8 – 1.2], (1.2 – 1.6], (1.6 – 2.0]. If the calculated value v is in the first subrange, we turn off all LEDs. We will map the 5 latter subranges with the 5 colors respectively: BLUE, GREEN, YELLOW, ORANGE, RED.

$$v\_new = v\_old/2 \times m/2 \qquad \text{Eq.2}$$

Finally, to calculate the direction of arrival, we use the 4 analog signals received from the 4 electret microphones in the Microphone Module. The Arduino will be used to compare the amplitudes of these 4 signals. The microphone which has the highest amplitude signal is the microphone that is closest to the sound source. Using this logic, the Arduino will determine that the microphone whose amplitude signal is highest is also facing the sound source. In order to prevent fluctuations in the result caused by echo and noise, we will do a comparison on the amplitude signal every 5ms, which give us 200 results per second. We will pick the most frequent result among the 200, and use that as the direction of arrival. After determining the direction of arrival, the Arduino will execute the rotation pattern for the LEDs so that the frequency spectrum will be displaced on the surface facing the sound source.

Gaming Mode

When the device is in Gaming Mode, the Arduino only uses the 4 digital signals received from Proximity Sensor Module. To calculate the distance between the sensor and user's hand, we use the Eq. 3. Let's the distances between the sensor1, sensor2, sensor3 and user's hand be called d1, d2, d3 be. We determine the motion of hand by comparing the distances between user's hand and sensors. For example, while the user is moving their hand from the right to the left, the user's hand can be at 3 positions: above sensor 1 but not sensor 2 (at the beginning); above both sensor 1 and sensor 2 (during movement); above sensor 2 but not sensor 1 (at the end). The value of d1 and d2 will be changed according to the position of the user's hand. When the user's hand is above sensor 1 but not sensor 2, d1 is smaller than d2. When the user's hand is above both sensor 1 and sensor 2, d1 and d2 are roughly equal. Finally, when user's hand is above sensor 2 but not sensor 1, d2 is smaller than d1. Utilizing this pattern, we can detect that the user's hand is moving to the left. Using the same method, we can determine if the user's hand is moving to the right or toward/away from the LED Cube. Next, we will observe and check that when the user moves their hand upward, all the values d1, d2, d3 will increase and subsequently decrease if the user moves their hand downward. Therefore, we can determine the motion of the user's hand and will use this to drive the Snake's movement as described in Fig. 12. We will update the position of the Snake every 500ms.

$$\text{distance} = 330\text{m/s} * t \qquad \text{Eq. 3}$$

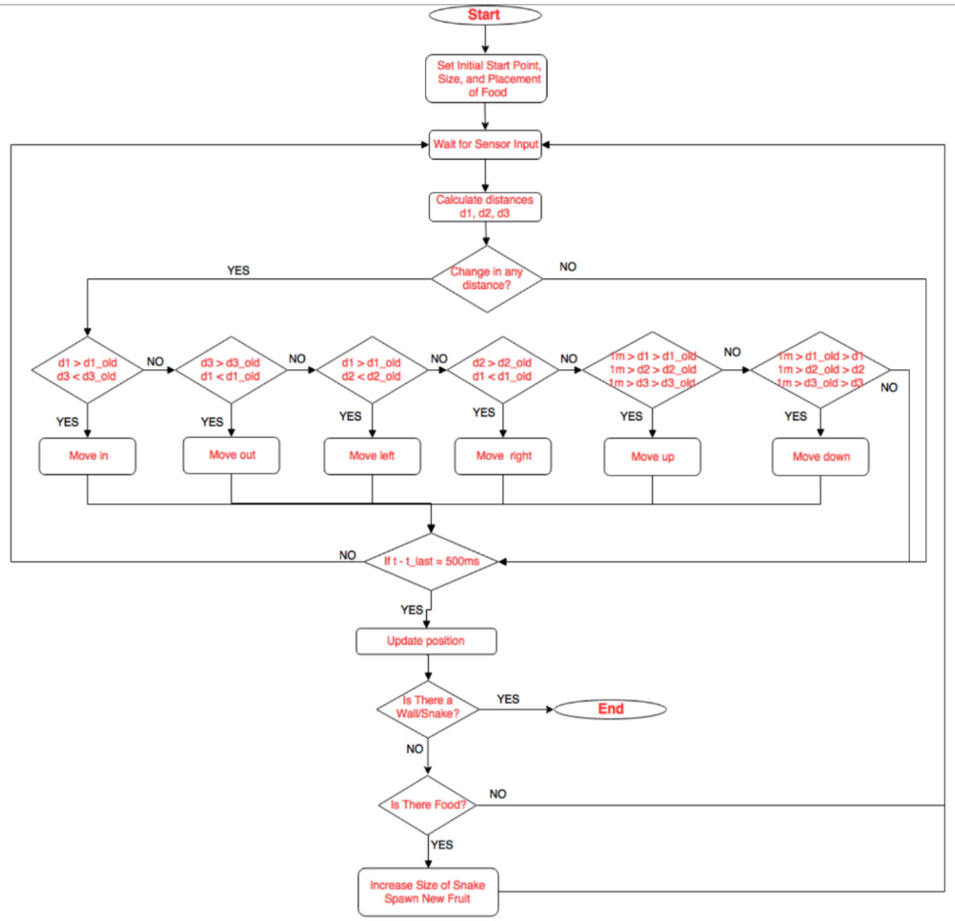


Fig. 13. Flow Chart of Snake Game

Physical Button Circuit

The following circuit is used to switch between 2 modes (Gaming and Music modes). We will use a de-bounced circuit as shown in Fig. 13.

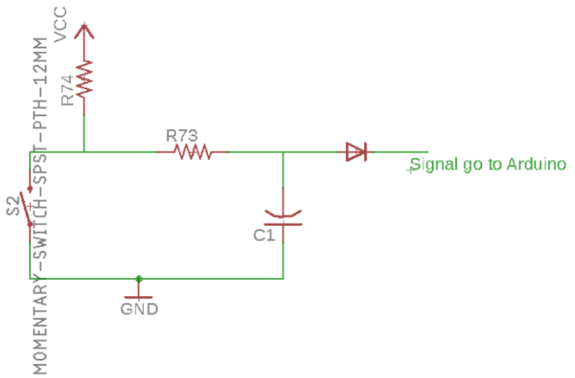
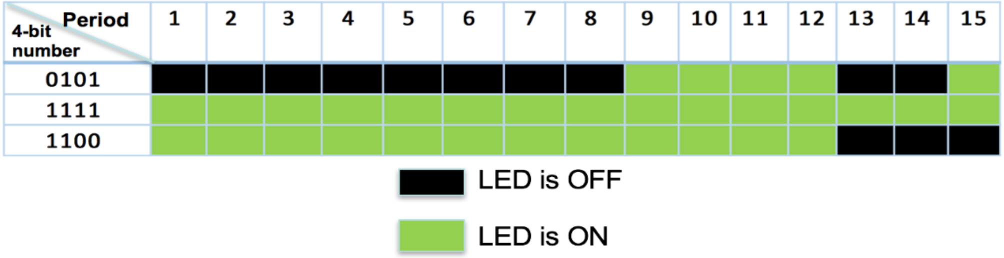


Fig. 14. De-Bounced Switch Circuit

Control LEDs

As discussed in the LED Module section, we will control 512 LEDs by using 25 8-bit shift registers. The idea is that we will turn on only 1 layer of LEDs at a time with the rate of 100 Hz so that human eyes will perceive that all LEDs are ON at the same time. The Control Unit will change the LEDs status by shifting out 25×8= 200 bits to these 25 shift registers, then set the OE (Output Enable) signal to low in order to load the registers' values to the LEDs. Because of the way we connect 25 shift registers, bits [0-7] are used to control which level is ON; bits [8-71] are used to control Red Anodes; bits [72-135] are used to control Green Anodes; bits [136, 199] are used to control Blue Anodes.

To control the color of the we will use the BAM (Bit Angle Modulation) method. We will use 4-bit BAM, which gives us 16 Red levels, 16 Green levels, and 16 Blue levels. Therefore, we will have 16×16×16 = 4096 colors. Using 4-bit BAM means that we will divide time into equal periods and use a 4-bit number to control the duty cycle of the LEDs in 15 consecutive periods. The 1st bit (MSB) in the number has a weight of 8, the 2nd bit in the number has a weight of 4, the 3rd bit in the number has a weight of 2, and the last bit has a weight of 1. When a bit is set to 1, the LEDs will be turn ON for the number of periods corresponding to the bit number. A few examples are illustrated in Fig.14.



Period / 4-bit number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0101	⬛	⬛	⬛	⬛	⬛	⬛	⬛	⬛	🟩	🟩	🟩	🟩	⬛	⬛	🟩
1111	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
1100	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	⬛	⬛	⬛

⬛ LED is OFF

🟩 LED is ON

Fig. 15. 4-bit BAM examples

According to *High Frame-Rate Television*, the human eye cannot see the more than 60 frames per second [9]. Therefore, the LEDs need to update with a rate at least 60Hz so that human cannot perceive the flicker. We choose the rate to be 100Hz to make the animation smoother. In each period, we need to update the 8 LED layers, meaning that each layer will be updated at rate 8×100Hz = 800 Hz. The Arduino Zero's clock speed is 48MHz so we need to update the LEDs every 48MHz/800Hz = 60,000 clock cycles. Because the shift register's maximum clock frequency is 25MHz [10], we will set it to 24MHz. Every time we want to update the LEDs" status, we need to shift out 200 bits, which will take 200 clock cycles from the shift registers (24MHz clock) and equal to 400 clock cycles of the Arduino Zero (48MHz clock). Therefore, we will have 60000 – 400 = 59600 clock cycles to perform all the calculations.

Input	Output
1. 4 Digital Signals (ECHO_OUT) from Proximity Sensor Module 2. 4 Analog Signals from Microphone Module 3. 1 Digital Signals (I2S format) from Microphone Module	1. 4 Digital signals (SER, SRCLK, OE, and SRCLR) to LED Module 2. 3 digital signals (WS, LR, MIC_SCK) to Microphone Module 3. 1 digital signal (TRIG_IN) to Proximity Sensor module

Requirement	Verification
1. Correctly estimate the movement of user hand in 6 directions (left, right, up, down, inward, outward) with accuracy at least 70%.	1. Using the Arduino to write a program that calculate the direction when hand is moving above the sensors (in range 3cm to 1m) in 6 directions (left, right, up, down, inward, and outward), and print out the output on terminal. Test each direction 10 times.
2. Correctly compute the frequency spectrum of the signal in the 50Hz to 20kHz range with frequency resolution of at least 160Hz.	2. Generate sine waves with frequencies from 50Hz to 20 kHz with 150Hz separation.

3. Correctly estimate the direction of arrival when sound source is directly facing 4 surrounding faces with accuracy at least 60%.	3. Put sound source at 4 different positions directly facing 4 surrounding faces and use the Arduino to estimate and displace result. Test each position 10 times.
4. Switch can switch between 2 modes.	4. Pushing the switch and test if the Arduino can receive the signal.
5. Can correctly control each LED in the LED cube.	5. Write program to turn ON and OFF each LED, and check if all LEDs behave correctly.

2.4 Risk Analysis

The most difficult part of the project will be making sure all the LEDs in our grid are properly connected and functioning. Because we have so many LEDs, making sure all of them work immaculately is pivotal to our project succeeding. Some LEDs may also be faulty so replacing them could be very cumbersome. Additionally, making sure that the proximity sensors pick up the right signals and that we can translate them to directions for the Snake game will be somewhat difficult due to the nature of a sensor potentially not picking up the correct signal. Lastly, the microphones may not pick up the sound accurately and that could impair the visualizations made on the LED grid.

2.5 Tolerance Analysis

2.5.1 LED Module

One of the critical parts of our project is the LED grid that we will use to display the outputs of our sound input and snake game.

For the music visualization mode, the Blue, Green, and Red LEDs must have at least 3.3V and 2.0V respectively to be optimally powered and ensuring that they have the optimal current as well. Because each single-color LED needs 20mA to function, a single RGB LED will need:

$$20\text{mA} \times 3 \text{ (for each single-color LED inside an RGB LED)} = 60\text{mA}$$

Because only 1 layer of LEDs is turned on at any one time for our visualization mode, the total current calculates to:

$$60\text{mA} \times 64 \text{ LEDs (1 layer)} = 3.84\text{A}$$

Therefore, the total power we need is:

$$3.3 \times 3.84 = 12.672\text{W}$$

Using the proper AC adapter is pertinent to how our device functions. Deciding to go with a 48W adapter will allow us to use more power for the rest of the device in a safer manner. The adapter has a DC output of 12V and 4A. This is in line with the way we plan on using the voltage regulators and will have enough current to drive our LEDs as displayed above.

2.5.2 Ultrasonic Sensors

Another important part of our project pertains to how a user controls the Snake in gaming mode. The HC-SR04 sensor has a detection range of 2cm to 4m. However, the angle at which the sensing is done can affect the accuracy of the sensor itself.

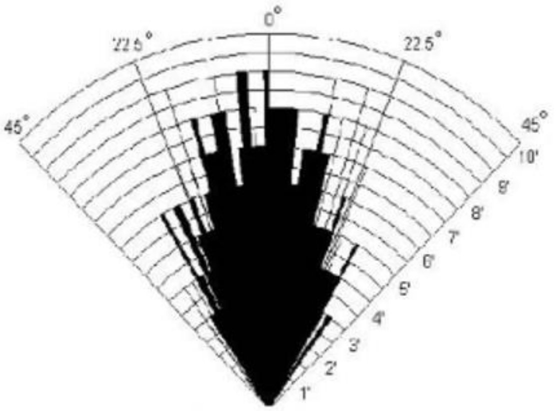


Fig. 16. [11]

In this figure, we see that the sensor's accuracy differs based on the distance and angle that it gets input from. We plan on having the user interact with the sensors from no more than 1 foot away from the sensors. As displayed in the figure, the sensor picks up much of the signal at an optimal level in the 1 to 7-foot range.

2.5.3 Frequency Analysis

Since we update our LEDs with rate of 100Hz, the delay for displaying the sound's information is 10ms. As we discussed above, to compute frequency, we use 256-point FFT, which takes about 3ms [12]. We have 40k samples per second, meaning that we have 400 samples every 10ms. Therefore, for every 10ms, we need to perform two FFT, which will take us 2×3ms = 6ms. We still have 4ms for other computations. Therefore, the algorithm that we use is possible for our system to achieve 10ms delay.

3 Cost & Schedule

3.1 Cost Analysis

We assume our hourly salary is $40/hour based on average income for an ECE at Illinois new graduate [13] and work on average 20 hours a week for each member of our team:

$$\text{Total} = 3 \times \frac{\$40}{hr} \times \frac{20\ hr}{week} \times 10 \text{ weeks} \times 2.5 = \$60{,}000$$

Description	Manufacturer	Part Number	Quantity	Cost/Unit	Total Cost
Digital MEMS Microphone	TDK InvenSense	ICS-43434	1	$2.20	$2.20

Electret Condenser Microphone	CUI Inc.	CMA-4544PF-W	4	$0.82	$3.28
8-bit Shift Registers	Fairchild Semiconductor	74HC595	25	$5.49 (10)	$16.47
Low Voltage Audio Power Amplifier	Texas Instruments	LM386-4/NOPB	4	$1.01	$4.04
Linear Voltage Regulator 3.3V	STMicroelectronics	LD1117V33	5	$0.55	$2.75
Linear Voltage Regulator 5V	STMicroelectronics	L7805 TO-220	1	$0.75	$0.75
Arduino Zero Microcontroller	Arduino	N/A	1	$39.00	$39.00
Ultrasonic Range Sensor	ElecFreaks	HC-SR04	3	$3.95	$11.85
Common Cathode RGB 5mm LED	Chanzon	100F5W-YT-RGB-CC	600	$8.96 (100)	$53.76
48W 12V 4A AC Adapter	Kastar	B004FLJ37Q	1	$10.49	$10.49
Total					**$144.59**

Grand total = $60,000 + $144.59 = $60,144.59

3.2 Schedule

Week	Tasks
2/18/2018	• Order parts needed to begin building the project • Finish calculations needed for our device, as well as power requirements • Check in with TA regarding the Arduino Zero usage • Finalize schematics of our modules to start work on PCB • Mock Design Review
2/25/2018	Zihan Yan: • Finalize parameters for Microphone Module schematics Islam Kadri: • Begin building LED grid and wiring

	Hieu Huynh: • Finalize schematics for LED Module Common Tasks: • Prepare for Design Review • Work on soldering assignment • Order parts needed to begin building the project
3/4/2018	Zihan Yan: • Continue building LED grid • Build and test the microphone amplifier circuit Islam Kadri: • Build a simple 2x2x2 LEDs cubes and test the logic of controlling LEDs Hieu Huynh: • Start implementing schematics on PCB Common Tasks: • Talk to Machine Shop regarding a frame for our LED grid, as well as getting some thicker wires • Finish soldering assignment
3/11/2018	Zihan Yan: • Write program to estimate the direction of arrival Islam Kadri: • Write program to calculate Frequency spectrum and amplitude Hieu Huynh: • Write program to drive LEDs cube Common tasks: • Revise PCB and get ready for audit check • Complete LED grid and start setting up sensors/modules • Check in with TA before Spring Break to make sure we are making good progress and to ask about any concerns we may have • Finish building the LEDs cube
3/18/2018	• Spring Break
3/25/2018	Zihan Yan: • Soldering components onto PCB Islam Kadri: • Write software for the Snake Game

	Hieu Huynh: • Test and debug the Music Mode after putting everything together Common Tasks: • Finish individual progress reports for team members • Verify all requirements of all modules • Revise the PCB if needed
4/1/2018	Zihan Yan: • Test the Gaming Mode after putting everything together Islam Kadri: • Start writing Final Report Hieu Huynh: • Continue debugging any problem related to software/hardware Common Tasks: • Make sure project is on schedule and discuss any resolve any enduring issues
4/8/2018	Common Tasks: • Prepare for mock demo to TA • Continue writing Final Report • Test and record a functioning project for official demonstration
4/15/2018	Common Tasks: • Present mock demo • Continue test and record a functioning project for official demonstration • Sign up for demonstration and mock final presentation • Revise final report and practice final presentation and present to class staff
4/22/2018	Common Tasks: • Prepare for final presentation • Perform demo • Finalize final report
4/29/2018	Common Tasks: • Finalize and turn in lab notebooks • Perform lab checkout • Perform final presentation

At the submission time of this design document, the distribution of work between the three group members will be decided based on time availability. Furthermore, although we will try our best to finish both features in our project, we will focus the priority on the music visualization feature.

4 Ethics & Safety

For this project we will abide by IEEE's Code of Ethics [14] and use our moral judgement when needed. We will respect all intellectual property laws and be honest with all the estimations and results during the entire process of our project. The public's safety, as well as ours, will be a priority and we will make sure that we thoroughly test and mitigate hazards.

We will be using an AC wall outlet to power our device so we need to make sure the voltage regulator is working properly. We must never allow a short circuit to occur as this could make the device overheat and potentially make LEDs explode. There could also be dangers from a shock hazard as we are dealing with a high amount of electricity running through the LEDs and other modules. To mitigate hazard to people who interact and view our device, we will try to safely cover the amount of internal circuitry that is exposed to a potential user. Our design will contain hundreds of LEDs which will be needed to be wired properly. The glass material and the fact that LEDs could be fragile could make the process challenging and dangerous. Also, our device will require heavy use with the soldering iron to connect the device's modules together as well as the hundreds of LEDs that we must put together. We will follow the lab guidelines very closely because making a mistake during soldering can be very harmful. Because the device will be viewed by the public, safety and proper containment of the device will be followed.

5 References

[1] ‘Kinetic Rain’, Changai Airport Singapore, 2012. [Online]. Available: http://www.changiairport.com/en/airport-experience/attractions-and-services/kinetic-rain.html [Accessed: 2-Feb-2018].

[2] ‘Metalier saves $6.3 million dollars on kinetic sculpture’, Metalier Coatings, 2015. [Online]. Available: https://www.metaliercoatings.com/save6-3m-on-kinetic-sculpture/ [Accessed: 2-Feb-2018].

[3] Emilie Chalcraft, ‘Kinetic Rain by ART+COM’, dezeen, 2012. [Online]. Available: https://www.dezeen.com/2012/07/19/kinetic-rain-artcom/ [Accessed: 2-Feb-2018].

[4] Chai Hung Yin, 'Woman arrested for damaging Changi Airport's kinetic rain sculpture', 2013. [Online]. Available: http://www.asiaone.com/singapore/woman-arrested-damaging-changi-airports-kinetic-rain-sculpture [Accessed: 2-Feb-2018].

[5] Rosen, Stuart (2011). 'Signals and Systems for Speech and Hearing (2nd ed.)'. BRILL. p. 163.

[6] John W. Leis (2011). Digital Signal Processing Using MATLAB for Students and Researchers. John Wiley & Sons. p. 82. ISBN 9781118033807.

[7] Duhamel, Pierre (1990). "Algorithms meeting the lower bounds on the multiplicative complexity of length-2^n DFTs and their connection with practical algorithms". IEEE

Transactions on Acoustics, Speech, and Signal Processing. **38** (9): 1504–1511. doi:10.1109/29.60070.

[8] TDK InvenSense, Mic Memes Multi-mode omni – 26dB, ICS-43434 datasheet

[9] High Frame-Rate Television(September 2008), BBC White Paper WHP 169, , M Armstrong, D Flynn, M Hammond, S Jolly, R Salmon

[10] Texas Instruments, "High speed CMOS logic analog multiplexers/demultiplexers," 5 8-Bit Shift Registers with 3-State Output Registers datasheet, DECEMBER 1982–REVISED SEPTEMBER 2015

[11] 'Ultrasonic Ranging Module HC SR04', 2014 [Online]. Available: https://www.itead.cc/wiki/Ultrasonic_Ranging_Module_HC-SR04[Accessed: 20-Feb-2018]

[12] 'Arduino FHT Library', 2016 [Online]. Available: http://wiki.openmusiclabs.com/wiki/ArduinoFHT [Accessed: 20-Feb-2018]

[13] 'ECE GRADUATE STARTING SALARIES', 2015. [Online]. Available: https://ece.illinois.edu/admissions/why-ece/salary-averages.asp [Accessed: 2-Feb-2018].

[14] IEEE.org, "IEEE IEEE Code of Ethics", 2018. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html [Accessed: 4-Feb-2018].