Senior Advanced Design Project: Final Report

PaPiya Radio

Team #6
Benjamin Ong
Danny Truong
Jamie Norvell
Zara Shoukat
Zayd Fazelyar
Zuha Abdul Wakeel

Faculty Sponsor: Dr. Jens-Peter Kaps

Table of Contents:

Executive Summary

While there are many Raspberry Pi audio systems on the market, most of these systems only contain a 2 channel amplifier. Additionally, the Raspberry Pi systems on the market do not include accompanying software. The aim of this project is to design a custom Pi-HAT consisting of a 2.1 channel amplifier with software that is tailored to the hardware. The music playing software is capable of NAS, internet radio, and USB connection. Furthermore, the entire system is placed within an enclosure that involves a touchscreen interface and hardware audio controls. This project allows audio enthusiasts to develop professional, high-quality audio systems using the Raspberry Pi.

Approach

Careful selection of components was required to fulfill the client's needs for the system outputs. The team gathered research on high-quality components capable of elevating the user's experience. The major requirement of the system was to design a custom Pi-HAT for the Raspberry Pi that featured a DAC as well as amplifiers for left, right and subwoofer channels. The heat generated by the system was a special consideration for the system design. The team utilized their previous knowledge of analog signals to determine that high pass filters would be needed for the stereo channels and low pass filters would be needed for the subwoofer. These filters would enable the system to produce high-quality audio without noise or distortion.

The amplifiers that were researched typically had a minimum input supply voltage of 20V; therefore, a power supply needed to be purchased that was capable of outputting at least 20V. Because the system implements the Raspberry Pi, a regulator was necessary to step down the voltage coming from the power supply to 5V. The team decided it would be best to place heat sinks directly onto the amplifiers for maximum heat dissipation. Sizing, mounting type, and dimensions of all components were taken into consideration as more PCB space occupied meant an increase in overall price.

There were several ideas for alternative and backup designs. For the amplifier, the first choice was the TPA3220DDWR chip. The team attempted to successfully breadboard the datasheet's schematic for this chip during the entire first semester and part of the second semester. However, this could not be accomplished. After two of the four chips were damaged and the component was out of stock due to the chip shortage, the decision was made to switch to the LM4766 amplifier. The LM4766 was the amplifier chip that was used in the final design for all three channels.

Initially, a preamplifier was chosen as a method of volume control. The LM386 was the first option, since it was immediately available. However, after further research and testing, it was determined that the LM386 could not be used because it is a power amplifier. The next option was to use the LM382, as it is a common choice for preamplifiers. However, the LM382 chip was sold out. Another alternative was the LM342, but this chip was too large and required too much power. Since the output of the DAC did not match the input requirements of most preamplifiers, the team opted to use a digital potentiometer to control the volume instead.

The final design that replaced the preamplifier includes a voltage adder after the DAC to perform stereo to mono conversion for the subwoofer. Originally, the LM386 was used as an active adder to add the right and left channels together. However, many difficulties arose such as inconsistencies in the frequency response, saturation, and excess power consumption. Hence, a passive adder using voltage dividers was implemented in the final system design.

Team members were split into groups based on individual skills and a desire to work on a certain aspect of the system. As the project manager, James handled the overarching administrative duties while delegating duties equally and fairly. A software team consisting of Benjamin and Danny were tasked with writing drivers, scripts, and code for the user interface (UI) and the DAC; they also managed the implementation of volume knob and power button functionalities. James, Zara, Zayd, and Zuha were on the hardware team and were assigned to construct a prototype of the system on breadboards. Splitting the team up further, Zara and Zuha conducted heavy research and experimentation regarding the various filters and dividers used throughout the system. This was done by measuring the input and output voltages to ensure that the audio signal was capable of passing through the varying electrical components in the system. James and Zayd constructed the PCB circuit schematic and layout for the system. It was important to properly draw the circuits and assign the appropriate footprints to the components within the schematic such that components were accurately soldered onto the board. The construction and assembly of the premade metal chassis was a group effort, which required tasks such as: sanding down all panels of the chassis, drilling of holes, painting, and discussing the layout of various user inputs.

The contributions stated above are each member's main responsibilities. However, all work efforts had overlapping responsibilities. Therefore, team members often contributed to all elements of the project.

Technical Section

Identification of Need:

There are a wide range of online tutorials that describe how to build an internet radio system with features such as an LCD touchscreen and USB connection. This project differs from other internet radios because it incorporates a passive subwoofer that enhances audio quality. This project also provides valuable experience in sound system development and embedded system interfacing for audio enthusiasts. Moreover, this project satisfies the client’s needs by reusing old speakers for a device with a modern user interface.

There are a few risks associated with the device. The device does not require a high operating voltage, which results in a low risk of electrical components overheating. Heat sinks and a metal chassis have also been used in the system to protect components from overheating.

There is a small risk of any of the components being damaged from excessive power because voltage dividers have been implemented to prevent saturation of any components in the system.

Operational Scenario and Requirements Specification:

Sample Input and Output Scenario:

Figure 1 depicts the sample input and output scenarios, which show how the user communicates with the system.

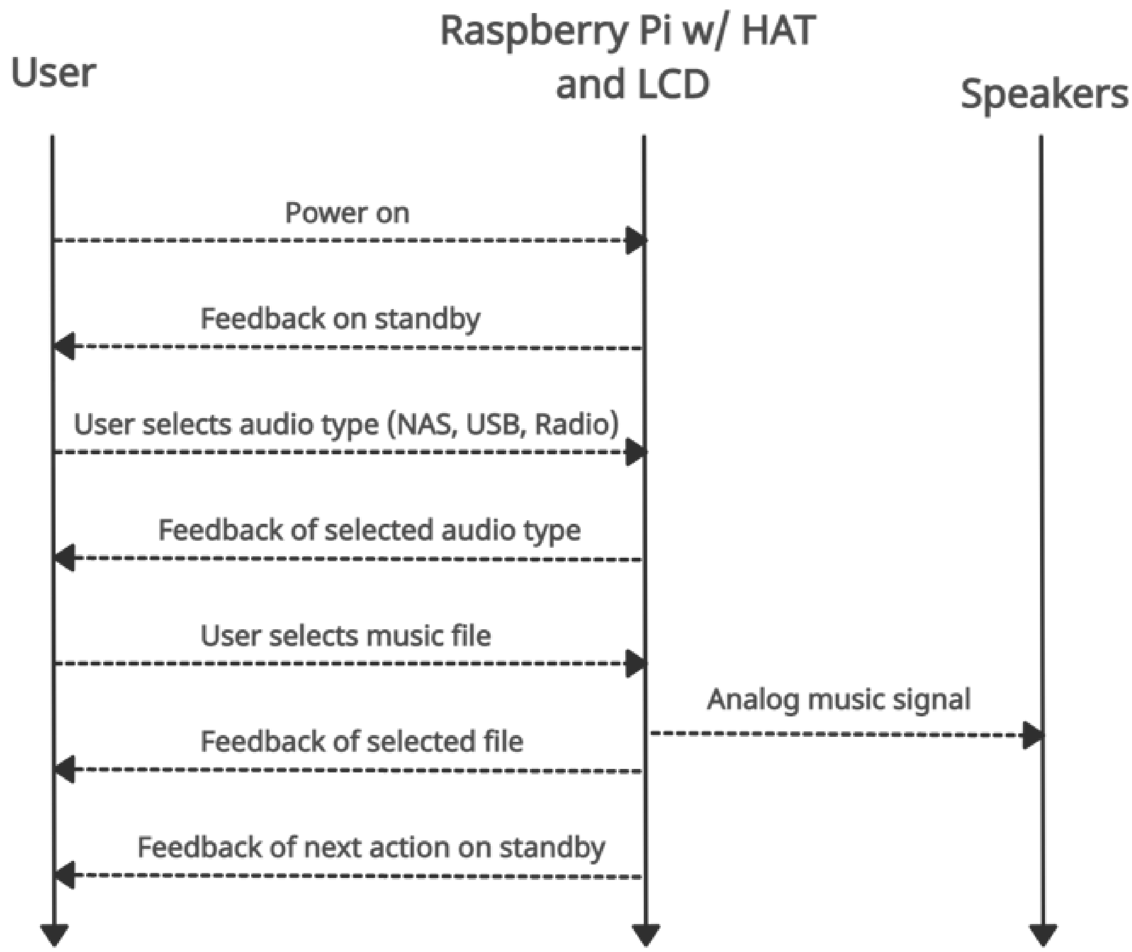

Figure 1: Sample Input and Output Scenario

External Systems Diagram:

Figure 2 depicts the external systems diagram. This demonstrates the high level operations of the system and exhibits the interactions between the user, the product, and the speaker output.

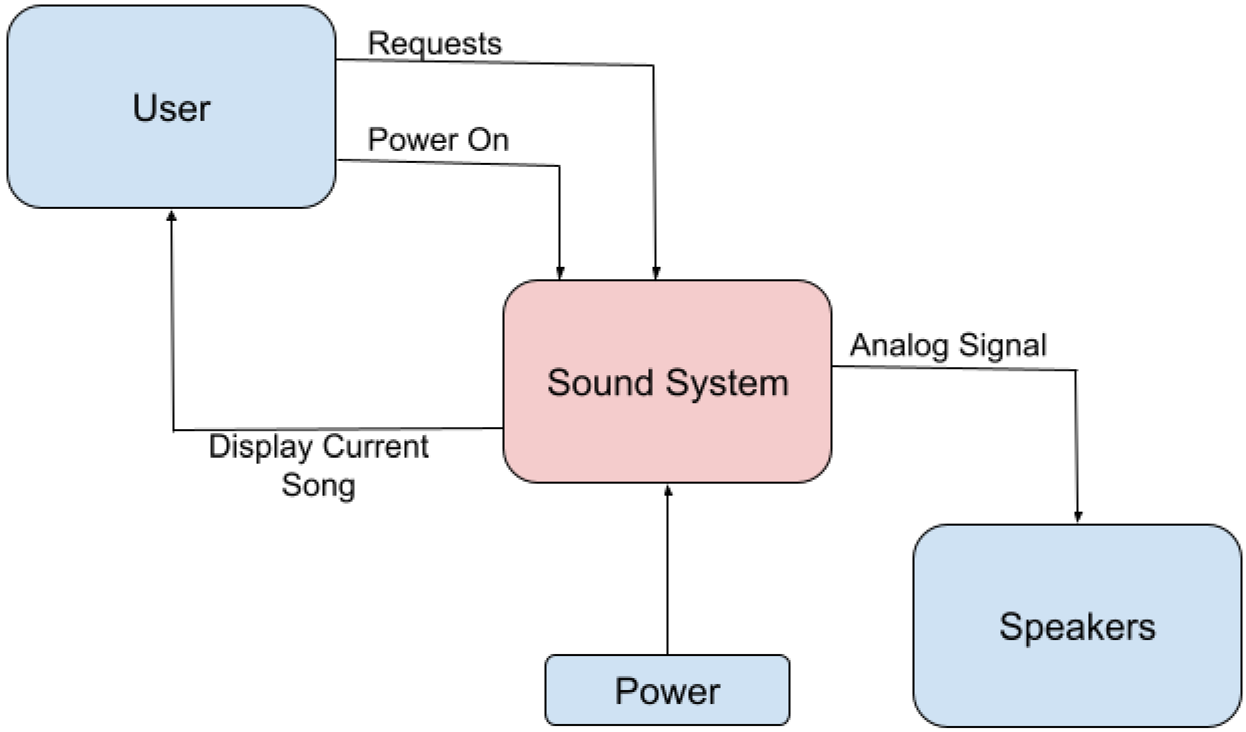

Figure 2: External Systems Diagram

Project Requirements:

The device is designed for home use. As such, the weight of the chassis is 20 pounds or less. All components are placed in a metal chassis that passively cools the system. The system also contains an LCD that allows the user to interact with the music player interface.

A Raspberry Pi 3B+ is implemented as the main microcomputer of the system. The custom Pi HAT incorporates the DAC and the 2.1 channel amplifier. The DAC accepts up to 24-bit audio sources at 192kHz. The 2.1 channel amplifier maintains a maximum power output of 50W per stereo channel and 100W for the subwoofer. The crossover frequency is between 100 and 200 Hz. The speakers connect to the system via speaker wire.

A combination of hardware controls and software controls are implemented to adjust the volume of the audio. The user can change the volume by turning the knob of a rotary encoder. The subwoofer also has its own volume control. In addition to the audio controls described previously, a power button is implemented to allow the user to turn the device on or off.

Once the device is powered on, the user can access his or her music through the music player interface, which is open-source software.Within the media player interface, a screen appears to allow the user to select his or her choice of audio input from a USB port as well as internet or NAS connection. After the user makes a choice, the user's music library is displayed and the user can choose a song to play. Song information and music controls are visible while a song is outputted through the speakers. Song information consists of the song title, artist name, and song duration; music controls consist of icons for play, pause, next-in-playlist, and previous-in-playlist.

Functional Decomposition:

The Level-0 design of the functional decomposition can be seen in Figure 3, and it describes the top level functions and user interactions of the system.

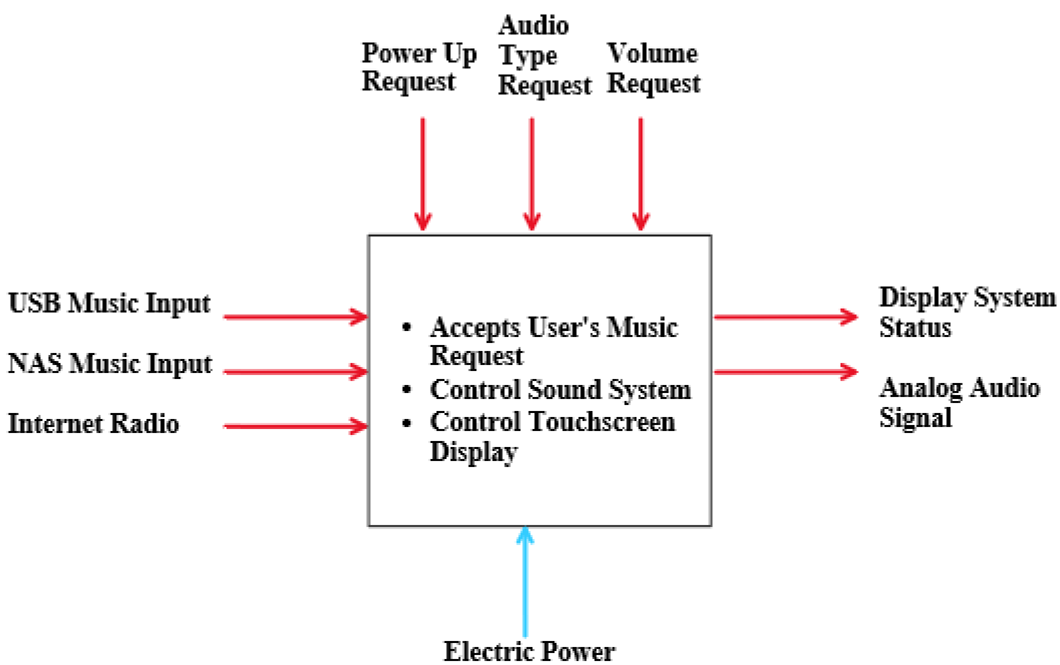

Figure 3: Top-Level Function (Level-0) Diagram

Figure 4 gives a generalized illustration of how each top level function operates within the system.

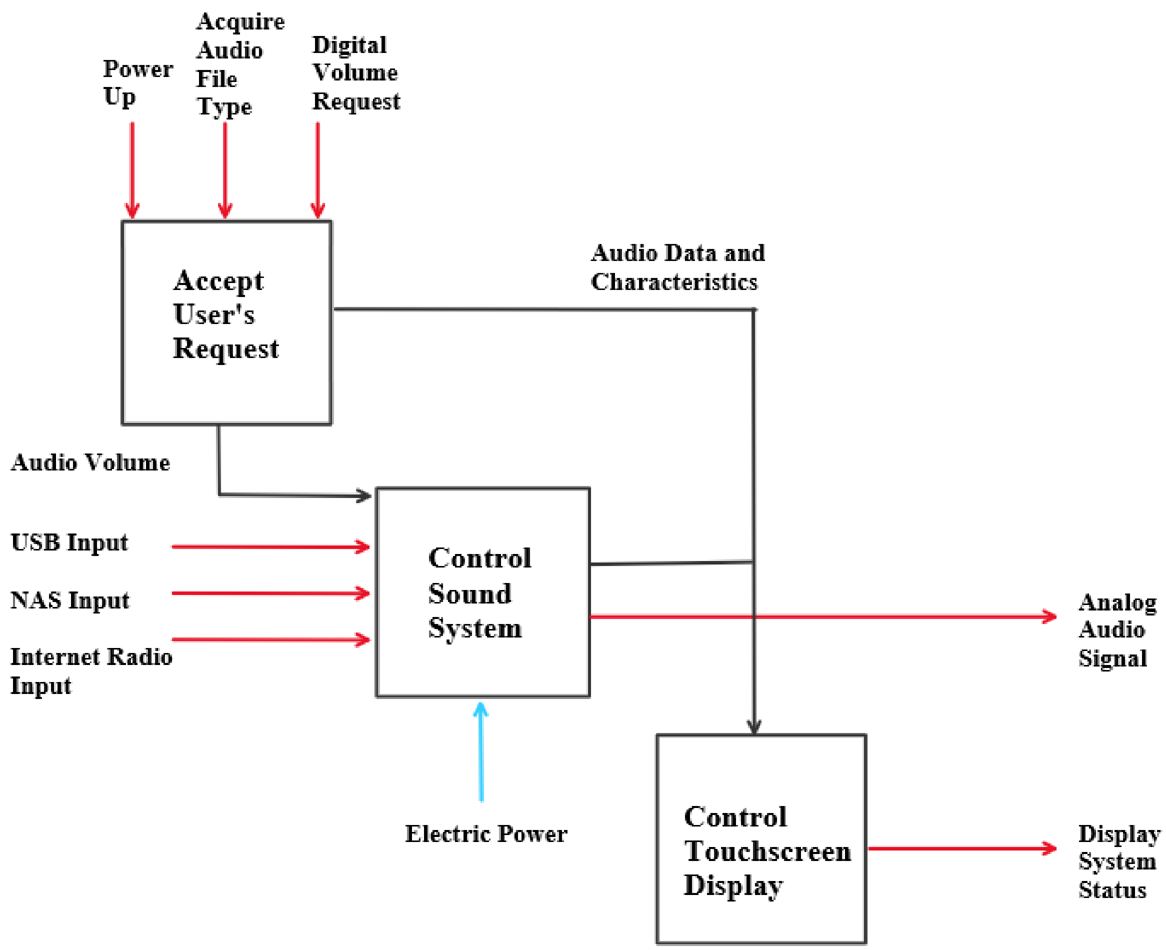

Figure 4: Functional Decomposition (Level-1) Diagram

The Level-1 functions are further decomposed in Figures 5, 6 and 7. Each figure represents the functionality of each top level operation.

Figure 5 establishes the interactions between the system and the user's requests.

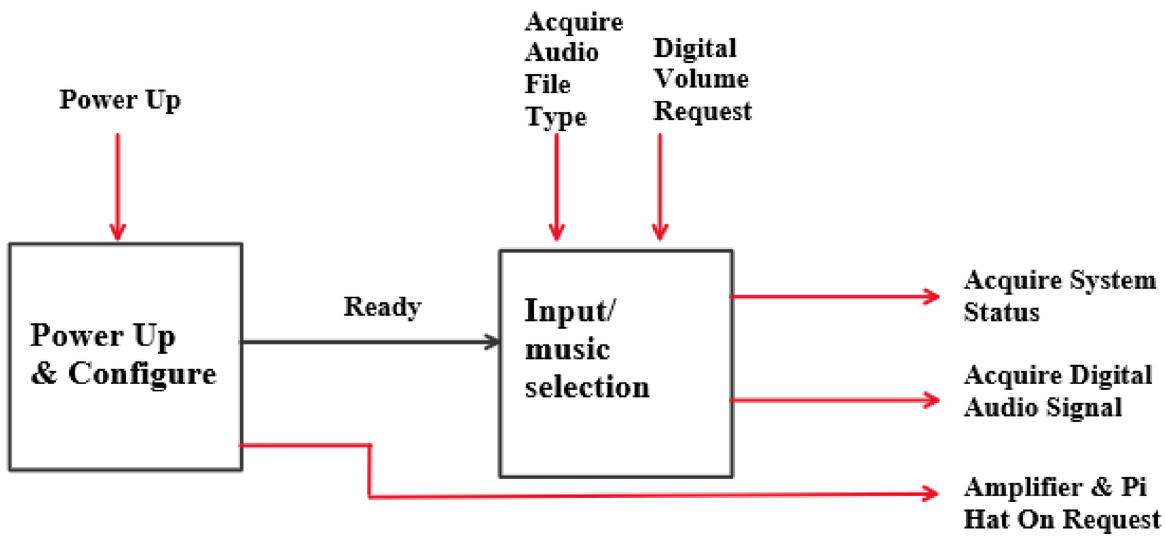

Figure 5: Functional Decomposition (Level-2) Diagram to Accept User's Request

Figure 6 highlights how the system handles digital audio input and outputs an analog music signal.

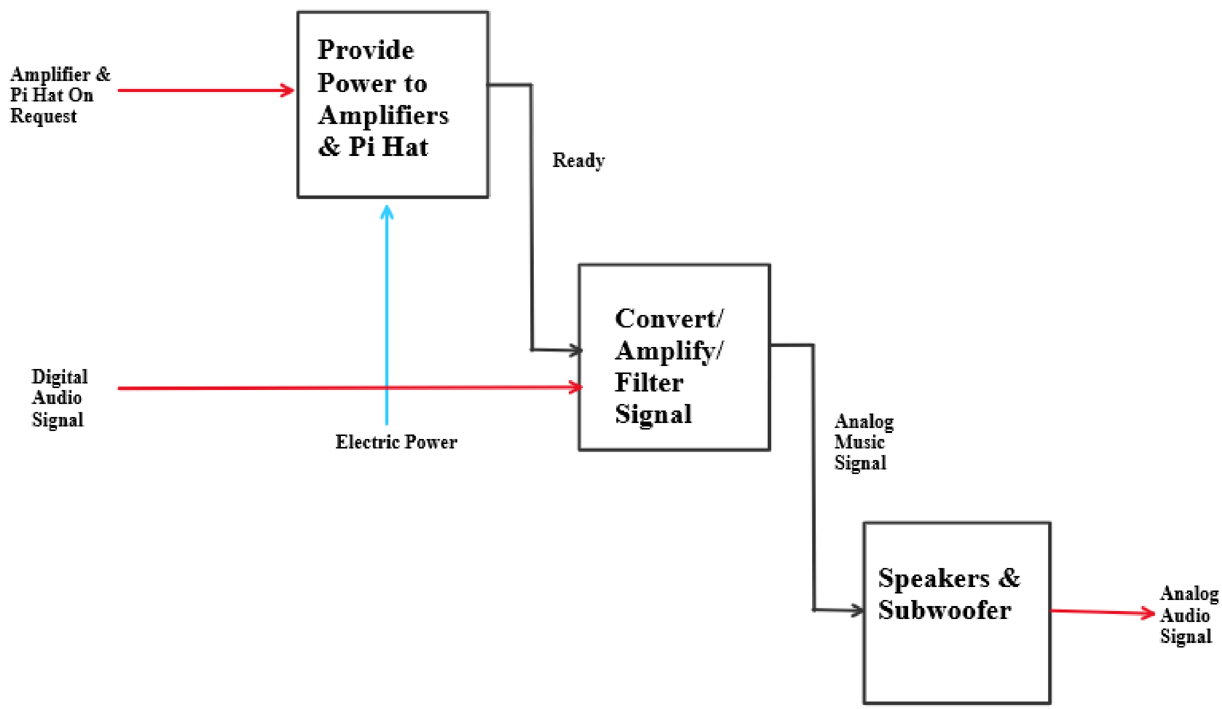

Figure 6: Functional Decomposition (Level-2) Diagram to Play Audio

Figure 7 details the uses the touchscreen will provide the client when operating the device.

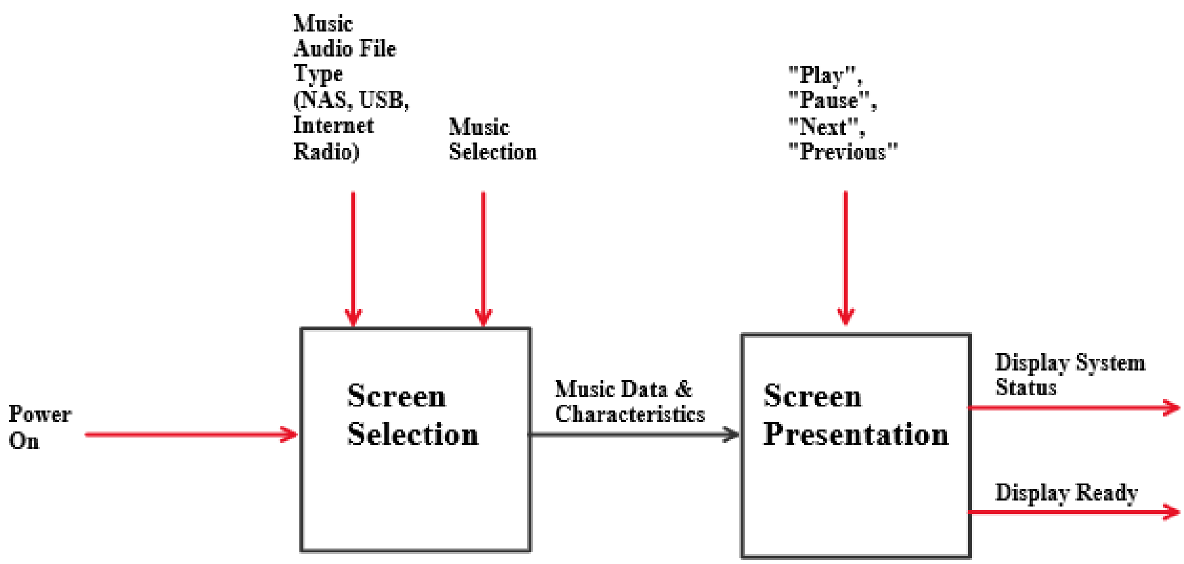

Figure 7: Functional Decomposition (Level-2) Diagram to Control System Display

System Architecture:

The generic physical architecture is shown in Figure 8 and highlights the resources used in the system. The overall system architecture is depicted in Figure 9, which consists of the top-level components of the design.

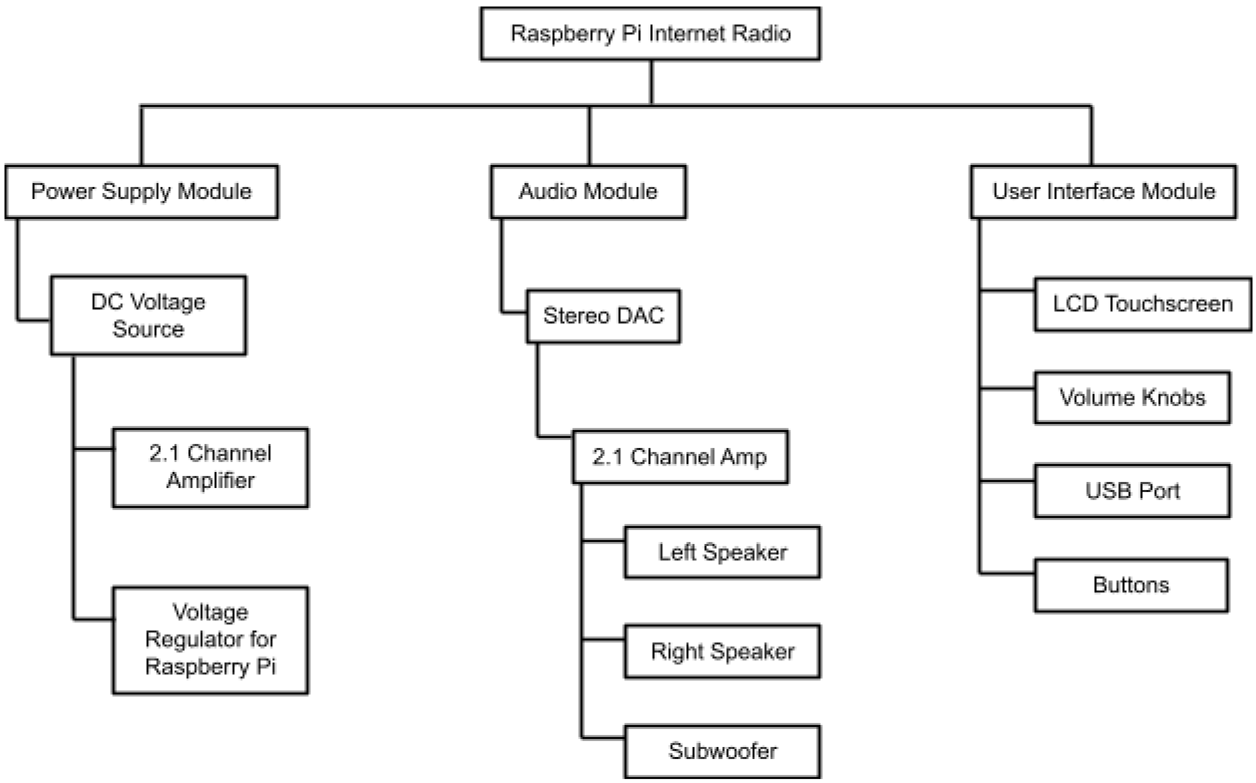

Figure 8: General Physical Architecture Diagram

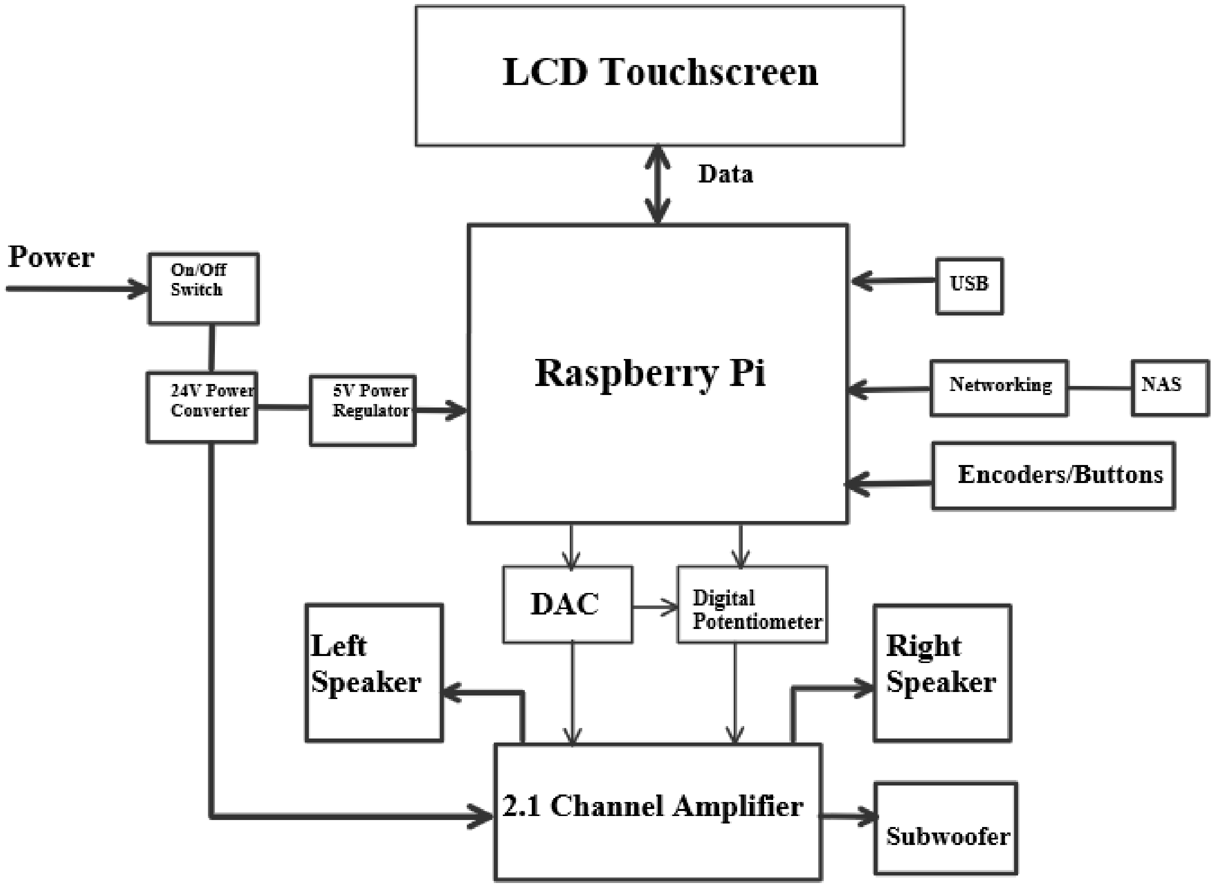

Figure 9: System Architecture Diagram

Elements of the System:

Raspberry Pi:

The Raspberry Pi drives all the hardware on the system and receives user input. User input can consist of media selection, song selection, volume control, play/pause selection, and other system interface interactions. This user input is then communicated to the Pi using the I^2C interface or the touchscreen. The Pi communicates to the corresponding chips, buttons, and rotary encoders of the system depending on the user's request.

Digital to Analog Converter (DAC):

The DAC receives digital audio data from the I^2S serial protocol and produces a stereo audio signal. The DAC supports audio data sampled at 384kHz [1], which surpasses the requirements for this project.

Amplifier:

Initially, the team selected to use the TPA3220DDWR amplifier in the design. This amplifier has a continuous output power of 60W in stereo operation and 110W when connected in mono operation [2]. The TPA3220DDWR also functions off of a supply voltage between 7V-32V [2]. It is a QFP package chip; therefore, breakout boards were manufactured for two of the four chips purchased for breadboarding during early prototyping. Unfortunately, this amplifier did not respond to early prototyping efforts, as both the two chips were damaged. The TPA3220DDWR was out of stock due to the chip shortage, which left only two chips for the final PCB design. The team decided to switch chips rather than implement the TPA3220DDWR on the PCB, which gave the team the ability to test a new chip before mounting it to the PCB.

The new amplifier chip that was selected was the LM4766. This amplifier has the capability to output 30W continuous power and 40W peak power per channel [3]. The chip can also function off of a supply voltage of 20V-60V [3]. Since the LM4766 is a Class AB amplifier, the transistors in the chip can conduct at the same time as the waveforms crossover and eliminate distortion [3]. This chip comes in a through-hole package, which simplified the prototyping process because the chip could be connected to the breadboard without the need for a breakout board. Upon early prototyping, the LM4766 chip performed properly, and both stereo and mono channel functionality were achieved.

Regulator:

The LT8609 5V regulator is a synchronous, monolithic, step-down switching regulator that consumes 1.7A of non-switching quiescent current [4]. The 24V power supply is stepped down to deliver a constant 5V at 3A of continuous current to power the Raspberry Pi. Prototyping efforts were not significant enough in ensuring this regulator's proper functionality. Thus, the final PCB design relies on an external 5V regulator that takes the supply voltage directly from the power supply and feeds it to the Pi through a micro-USB cable.

Digital Potentiometer:

The digital potentiometer in the system is the AD5253, and it is used to vary the strength of the signal fed into the subwoofer amplifier [5]. When the subwoofer volume knob is turned, the Pi updates the potentiometer value using the I^2C bus. The potentiometer has a total resistance of 50kΩ, and the wiper can be adjusted with a resolution of 64 steps.

One problem encountered when implementing this chip is that the voltage applied to the terminals of the potentiometer must be within the supply voltage range. The chip is powered from the 3.3V rail on the Pi. Hence, a DC offset voltage needs to be applied to the music signal before feeding it into the potentiometer. The software has been modified to enable a GPIO pin that applies the offset voltage and DC-blocking capacitors have to be added to accommodate this offset.

Filters:

For the stereo channels, a single-order high pass filter is used to block out lower frequencies. The values used for these filters are a resistor value of 1kΩ and a capacitor value of 1uF, and the cutoff frequency is approximately 122 Hz. The subwoofer channel has a single-order low pass filter to block out higher frequencies. This filter is implemented with a resistor value of 5.1kΩ and a capacitor value of 0.1uF, and the cutoff frequency is approximately 202 Hz.

Voltage Dividers:

Voltage dividers are necessary to step down the voltage from the DAC before feeding it into the amplifier. Based on the datasheet for the DAC, the typical output voltage from the DAC is 2.1V rms, or 3V; the output voltage from the DAC was also measured to verify the typical voltage. Then, the maximum input voltage to the amplifier was measured using an oscilloscope, which is approximately 350 mV. The measured voltages were used to design voltage dividers that step down the DAC output voltage by a factor of 10.

Touchscreen:

The LCD touchscreen mounted to the front of the chassis is a 5 inch display made by ELECROW [6]. The display is an HDMI monitor with an 800x480 LCD touchscreen, and simple installation. The touchscreen has a USB capacitive touch control, a free-driver with a micro USB interface for touch and power, and an HDMI interface for display. The LCD touchscreen works with the Raspberry Pi 4, 3 B+, BB Black, Banana Pi, and other popular microcomputers.

Physical Controls:

The system features a power button as a physical control. While the Raspberry Pi has no built-in power button, some Raspberry Pi models support a feature called “wake on GPIO” [7], which allows the Pi to receive a power on signal by shorting the GPIO3 pin to ground. A button was placed between GPIO3 and ground to allow the user to turn on the device without any difficulties. However, there were challenges with implementing the same button to trigger a shutdown. The solution was to use GPIO26 to listen for a shutdown signal and to connect a diode into the circuit; this was done to prevent GPIO3 from driving GPIO26 while still allowing the button to pull GPIO26 to ground.

Behavioral Modeling:

The function of the volume control program is modeled with the state machine seen in Figure 10.

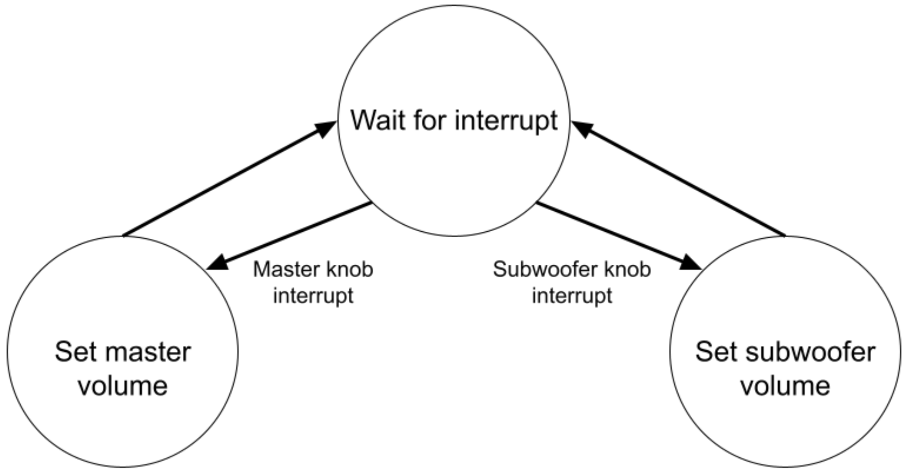

Figure 10: Volume Control State Machine

The procedure for controlling the master volume is demonstrated with the flowchart seen in Figure 11.

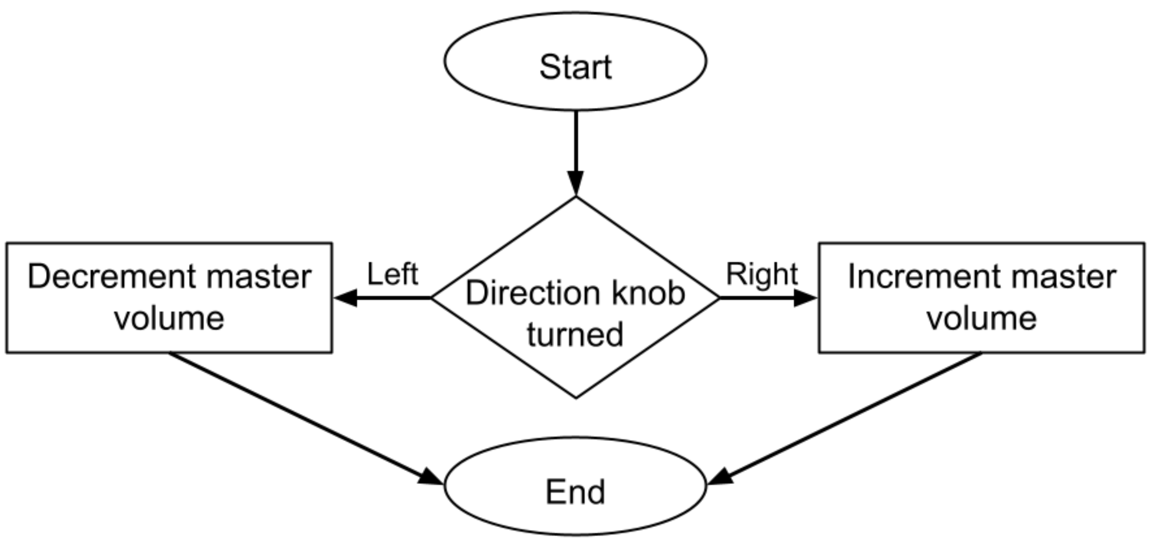

Figure 11: Master Volume Control

The procedure for controlling the subwoofer volume is represented with the flowchart seen in Figure 12.

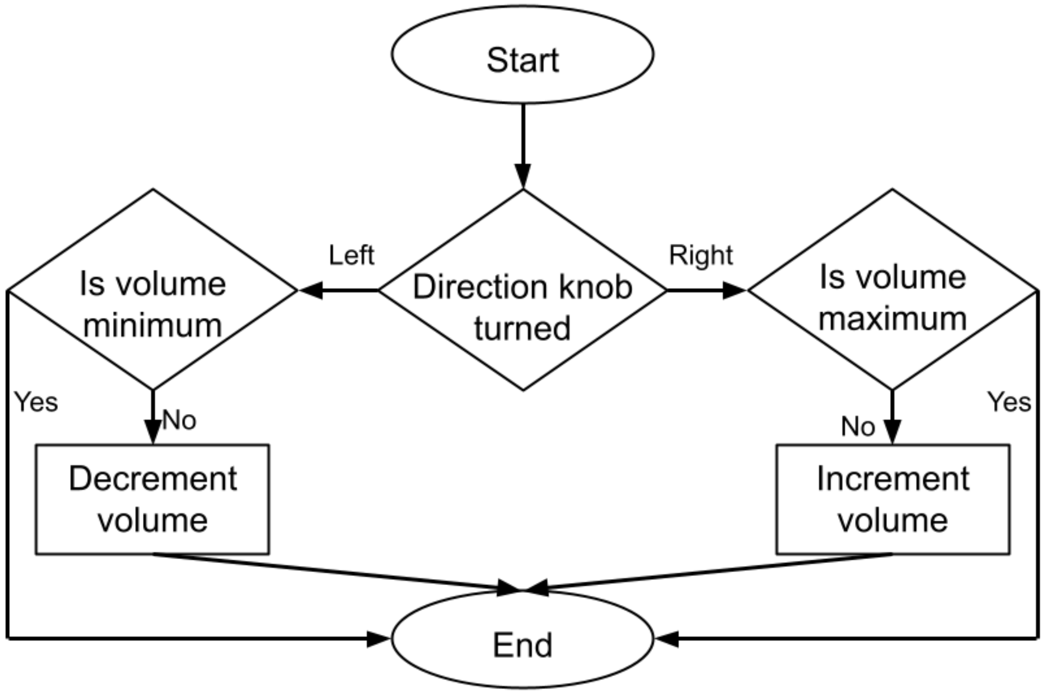

Figure 12: Subwoofer Volume Control

The procedure that the setup script follows for installing and configuring software is illustrated in Figure 13.

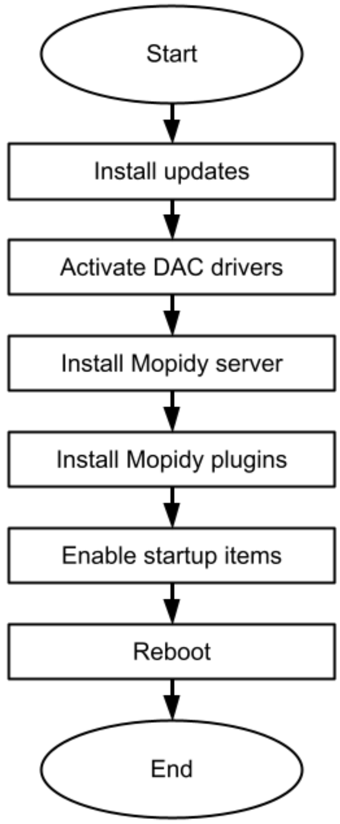

Figure 13: Setup Script Procedure

Background Phenomenology:

Heat Sinks:

The LM4766 requires the use of heat sinks to dissipate the heat generated during amplification. To determine the heat sink, the maximum thermal resistance from the heat sink to its surroundings, θ_{SA}, of the amplifier had to be calculated using the operation specifications used by the system. The thermal resistance is determined using Equation 1, where θ_{JC} is the thermal resistance from the junction to the case, θ_{CS} is the thermal resistance from the case to the sink, T_{JMAX} is the operating junction temperature, T_{AMB} is the ambient temperature, and P_{DMAX} is the power dissipation of the amplifier [3]. The datasheet specifies θ_{JC} = 1℃/W, θ_{CS} = 0.2℃/W, and T_{JMAX} = 150℃ [3]. The ambient temperature was taken to be 30℃ and the power dissipation was found using Equation 2, where the value of the input supply voltage V_{CC} = 24V and the value of the load impedance R_L = 4Ω [3]. P_{DMAX} was found to be 7.3W for one channel. Therefore, 15W was used for the total power dissipation for the chip. Using these values, it was determined that θ_{SA} = 15.238℃/W was the specification for the appropriate heat sink.

$$\theta_{SA} = \frac{(T_{JMAX} - T_{AMB}) - P_{DMAX}(\theta_{JC} + \theta_{CS})}{P_{DMAX}} \quad (1)$$

$$P_{DMAX} = \frac{V_{cc}^2}{2\pi^2 R_L} \quad (2)$$

Gain:

The amplifier is used to boost the analog audio signal that will be supplied by the DAC. The amplification of the signal is due to the gain, which is a constant factor that the device supplies the signal. This factor is used to amplify the various frequencies in the signal without changing the shape of the waveform or the quality of the sound. Gain is the ratio of the output voltage to the input voltage of the amplifier, as seen in Equation 3.

$$A_v = \frac{v_o}{v_i} \quad (3)$$

The forward voltage gain for audio amplifiers is typically 40dB, and the use of the forward gain with a feedback loop reduces noise and distortion from the power supply [8]. The amplifier can reach a saturation region, where it will no longer be able to amplify the signal. It is important to refrain from supplying the amplifier with signals that cause the amplifier to enter the saturation region, as long exposure in the saturation region could damage the amplifier.

The LM4766 uses an inverting amplifier to provide AC gain [3]. The formula for the gain of an inverting amplifier is represented in Equation 4, where R_i represents the input resistance and R_f represents the feedback resistance. The resistors used for the implemented amplifier were R_i = 1 kΩ and R_f = 32 kΩ. Therefore, the theoretical gain of the amplifier for each channel was 32 V/V.

$$|A_v| = \frac{R_f}{R_i} \quad (4)$$

Power:

The output power measured from the system is one of the major aspects of the project and is determined by the output voltage as well as the impedance of the speakers used. Equation 5 demonstrates how the power is calculated, where V is voltage and Z is impedance.

$$P = \frac{V^2}{Z} \quad (5)$$

Filters:

First order high pass filters are used in the stereo channels to pass high frequencies. For the subwoofer channels, a first order low pass filter is implemented to pass low frequencies. The cutoff for a first order high pass and low pass filter is given by Equation 6, where R is resistance and C is capacitance.

$$f_c = \frac{1}{2\pi RC} \tag{6}$$

Figures 14 and 15 below show the Pspice simulation results for the filters. The cutoff for the high pass filter is 122 Hz and the cutoff for the low pass filter is 202 Hz, as shown in the simulations below.

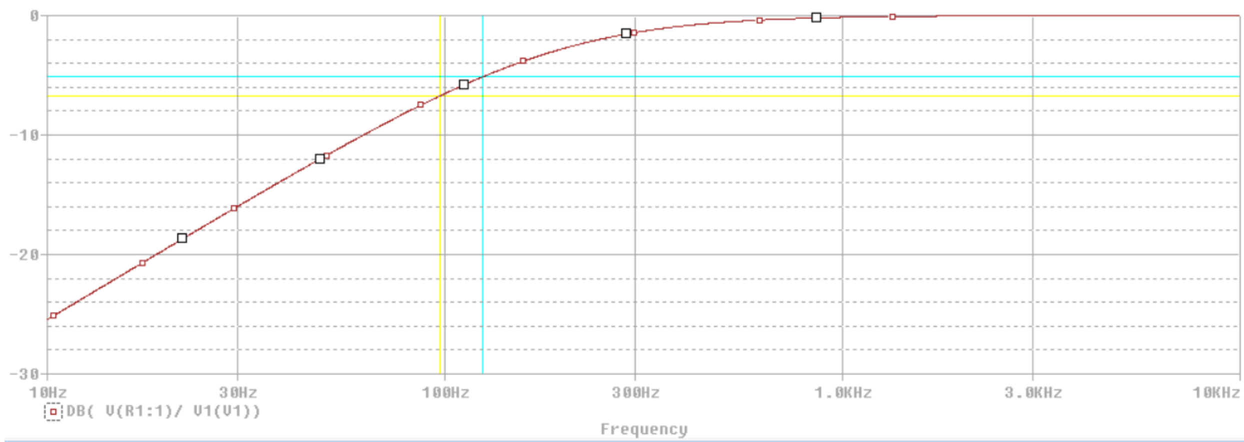

Figure 14: High Pass Filter Simulation for Stereo Channels

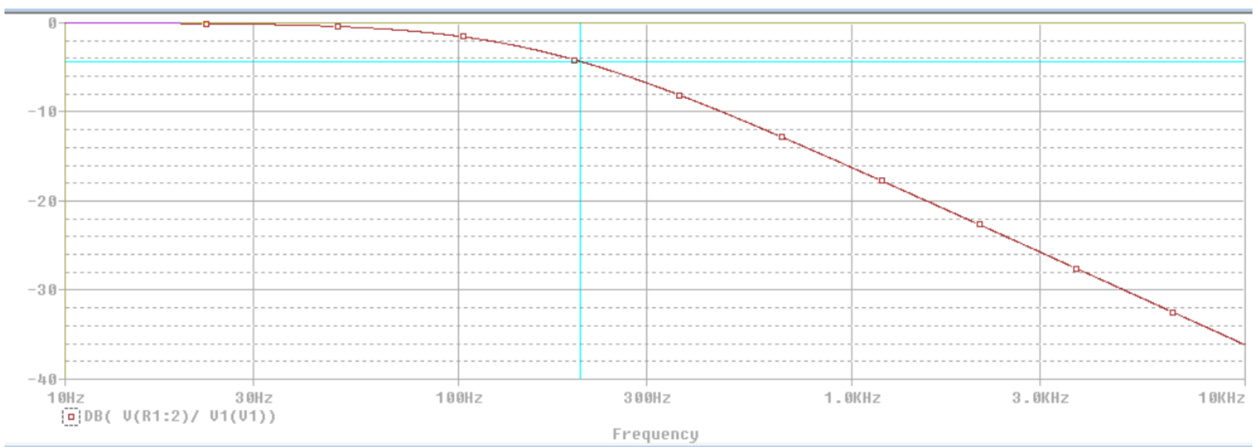

Figure 15: Low Pass filter Simulation for Subwoofer

Early Prototyping:

Early prototyping began with the initial amplifier, the TPA3220DDWR. The breadboarding of this amplifier can be seen in Figure 15. The team prototyped with the TPA3220DDWR for both ECE 492 as well as the beginning of ECE 493. A custom breakout board needed to be made in order to prototype with this chip. Additionally, circuit schematics of the TPA3220DDWR were created for the PCB design and the PCB layout. During prototyping, two out of the four chips that were purchased became damaged. As addressed previously, this amplifier was switched out for the LM4766 amplifier, which can be seen breadboarded in Figure

16. This amplifier did not require any special accommodations for prototyping, and the schematic in the datasheet was strictly followed in order to create a functioning circuit with the LM4766. Additionally, the PCM5122 DAC and the AD5253 digital potentiometer also had custom breakout boards created because both chips are QFP packages, and breakout boards allowed the team to prototype with these chips and interface them with the Raspberry Pi. The early prototyping breadboarding for the DAC and the digital potentiometer can be seen in Figures 17 and 18, respectively.

It is also important to note that in the prototyping stage, the team initially decided to use an active adder to combine the stereo outputs of the DAC into a mono input for the subwoofer. This was done using the LM386 amplifier, but it was decided to remove the LM386 and replace the active adder with a passive adder created from voltage dividers. This was done on account of the LM386 inducing noise in the system, requiring another source for the supply voltage, and consuming power from the system. The schematic for the active adder is shown in Figure 19. Furthermore, second-order RC filters were replaced by first-order RC filters for both the stereo and subwoofer channels. This decision was made to conserve space on the PCB and to increase the overall output power of the system.

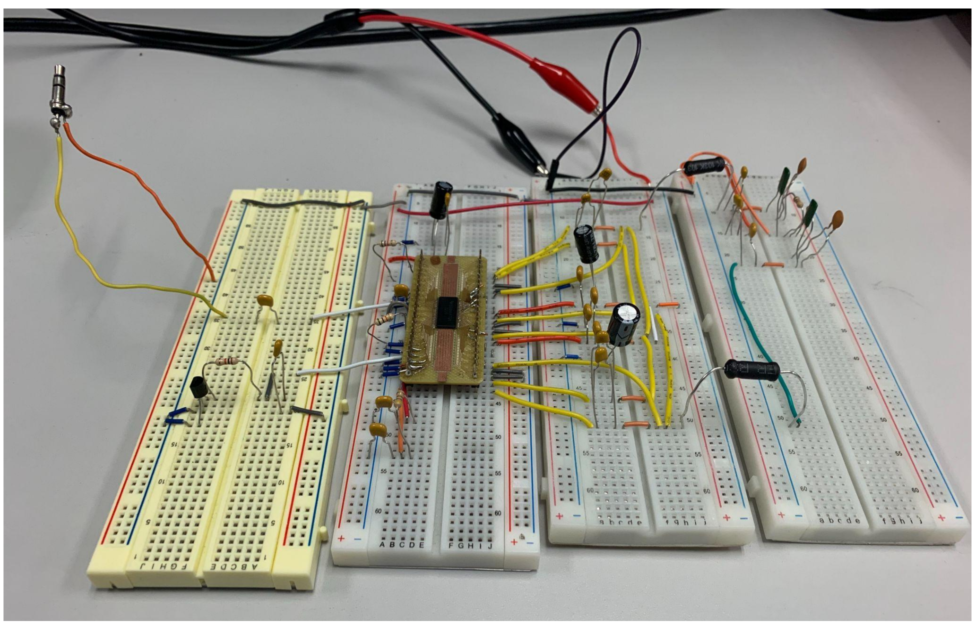

Figure 15: TPA3220DDWR Early Prototyping

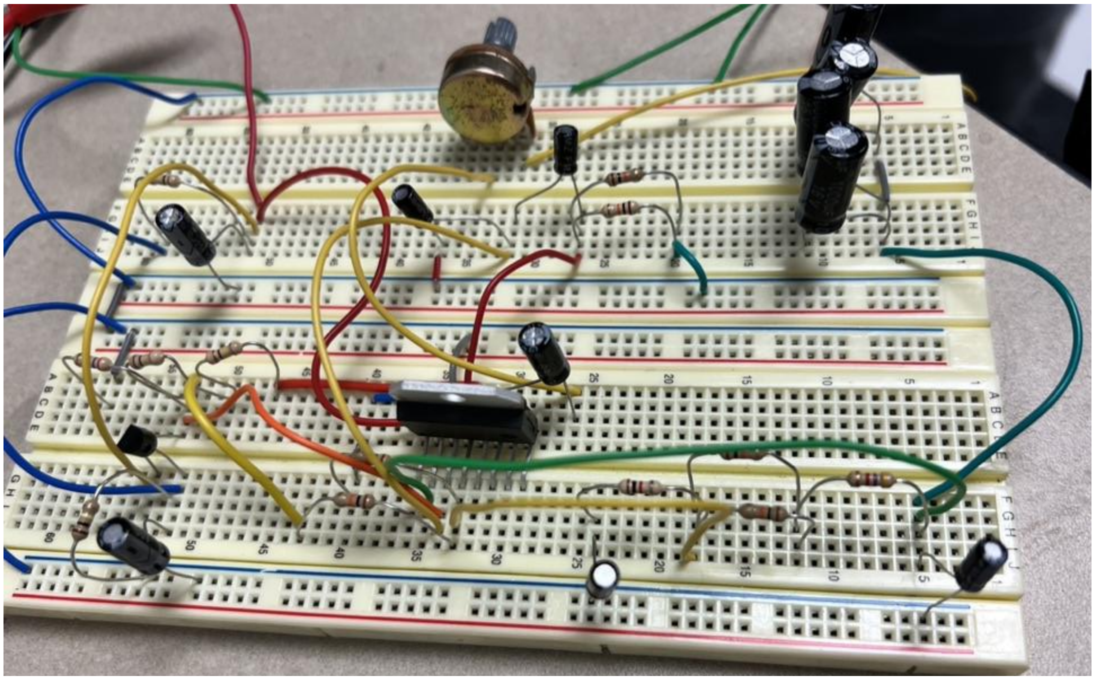

Figure 16: LM4766 Early Prototyping

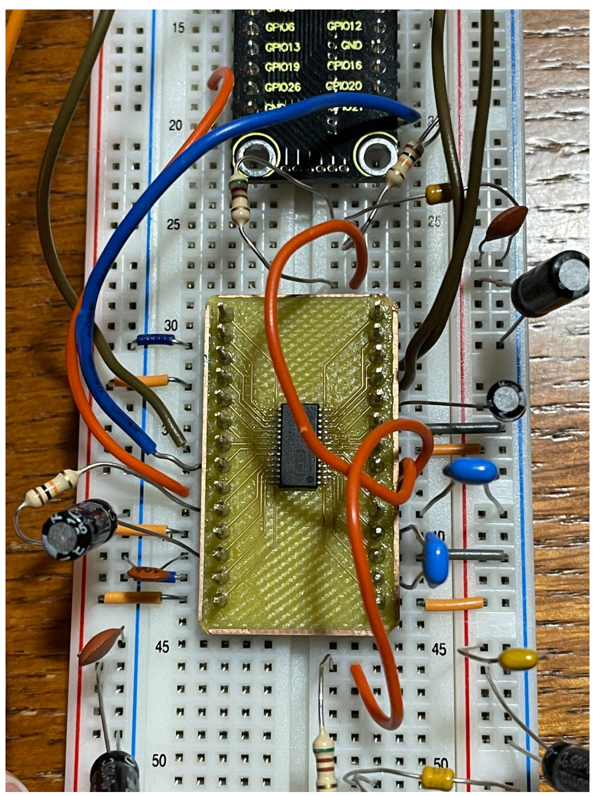

Figure 17: PCM5122 Early Prototyping

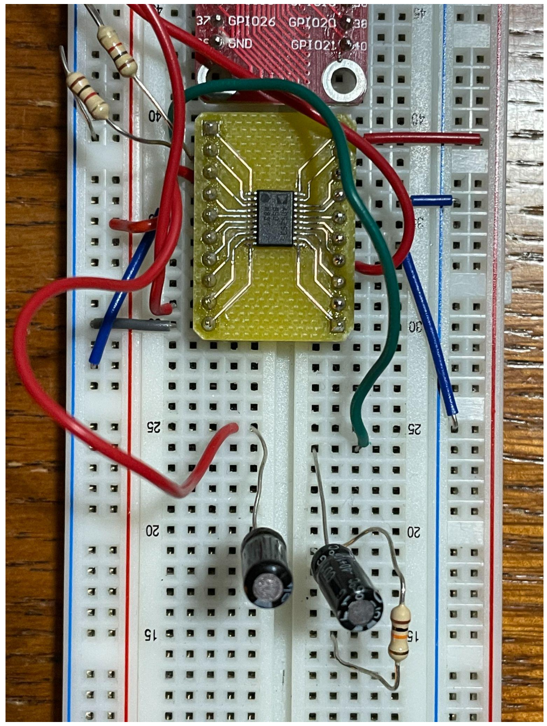

Figure 18: AD5253 Early Prototyping

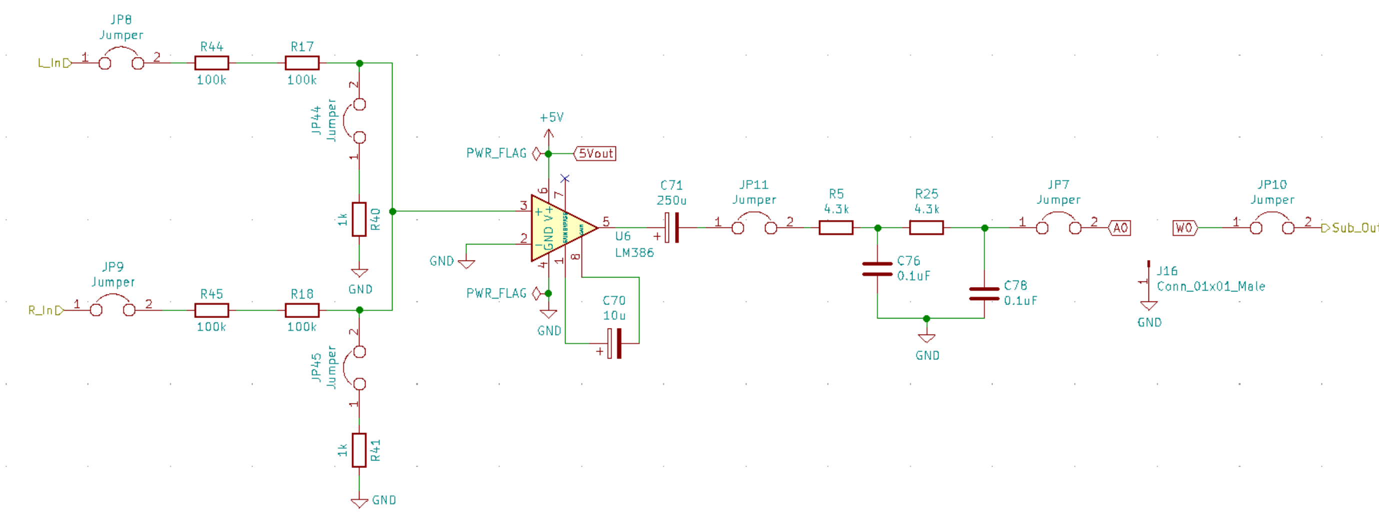

Figure 19: Schematic of Active Adder using LM386

Detailed Design:

Circuit Design:

DAC:

The DAC can be wired for operation in one of three modes: SPI mode, I^2C mode, and hardwired mode [1]. Because the circuit symbol was imported from a third party library, the pinout for the DAC in KiCad is in SPI. However, the I^2C mode was selected for this project and was configured by referring to the pinout on the DAC's datasheet and disregarding the pinout on the symbol in KiCad. The KiCad schematic is depicted in Figure 20. It is important to note that for the Raspberry Pi drivers to recognize the DAC, the I^2C address pins have to be set as ADR1 = 1, ADR2 = 0.

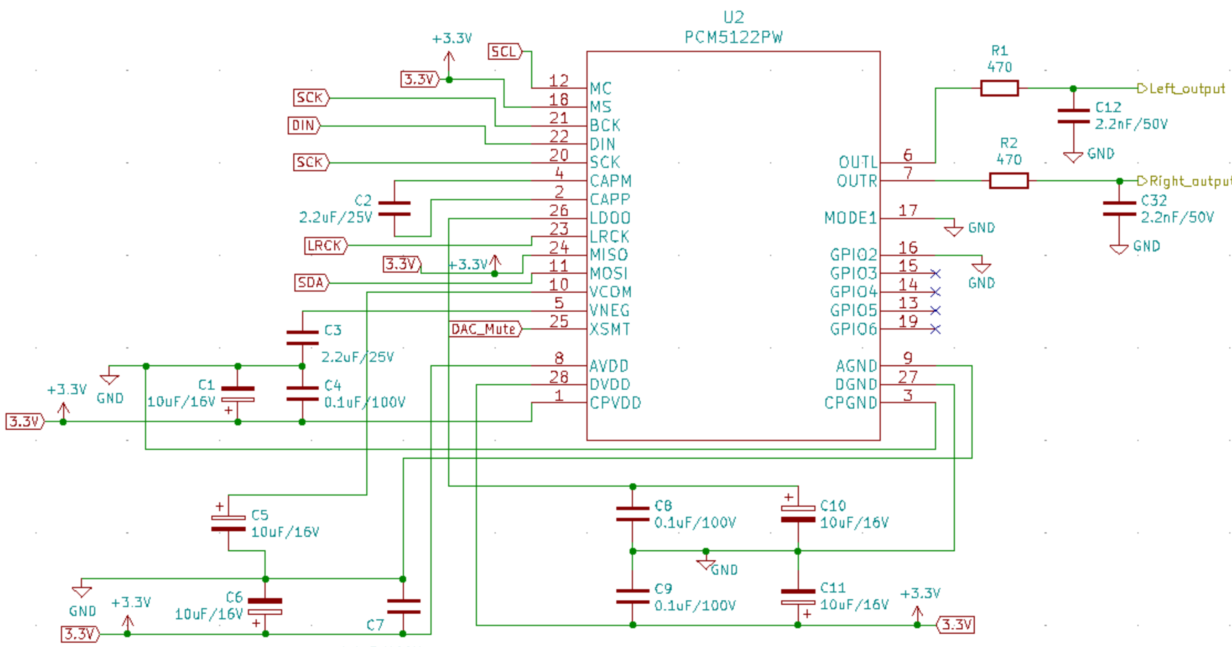

Figure 20: DAC Schematic in I^2C Mode

Amplifiers:

The amplifier used in the PCB design is the LM4766. Each LM4766 chip contains two amplifiers referred to as an A unit and a B unit. Therefore, two LM4766 chips are featured in the PCB design: one chip is implemented for the right and left channels, and the other is implemented for the subwoofer. Figure 21 displays the circuit schematic for a single channel on the amplifier. This schematic is duplicated for the right and left channels as well as the subwoofer. It is important to note that a bridge-tied load design was tested for the subwoofer but was unsuccessful as it required the use of a differential supply. Hence, it was decided a single channel would be used for the subwoofer. The schematic for the LM4766 is nearly identical to the one given in the datasheet for a single supply design. However, when experimenting with the chip on various breadboards, the team discovered it would be best to adjust the value of the feedback resistor connected to pins 1 and 12 to achieve a higher gain and a higher output power. Another change implemented by the team is taking the single electrolytic capacitor along the power rails and splitting it into a combination of electrolytic and ceramic capacitors in parallel.

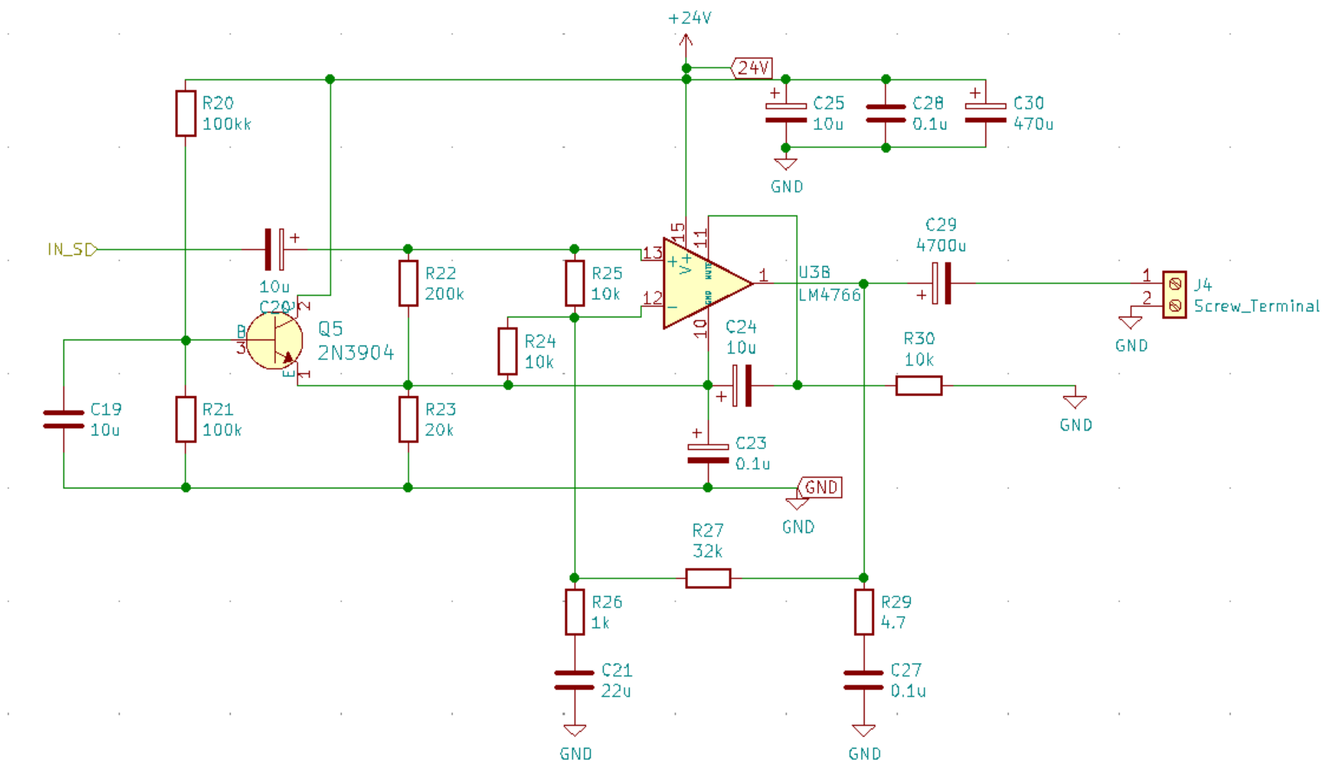

Figure 21: Single supply schematic for amplifier.

<u>Regulator:</u>

The LT8609 regulator accepts voltage inputs ranging from 6-40V [4]. At 2MHz, the regulator is designed to step down to 5V, which powers the GPIO pins and other external settings on the Raspberry Pi. In addition, the EN/UV pin on the regulator is active when it is high [4]. Thus, the EN/UV pin is tied high at the input and is connected to the voltage input to enable active high functionality.

The LT8609 regulator could not be effectively integrated in the system after extensive testing and experimentation. The team used jumper wires to completely bypass the regulator schematic on the PCB, as seen in Figure 22. Instead, a micro USB adapter was used to regulate the 24 VDC to 5 VDC. This was a last-minute arrangement that allowed a regulator to be wired directly into the 24 VDC power converter and placed directly into the chassis.

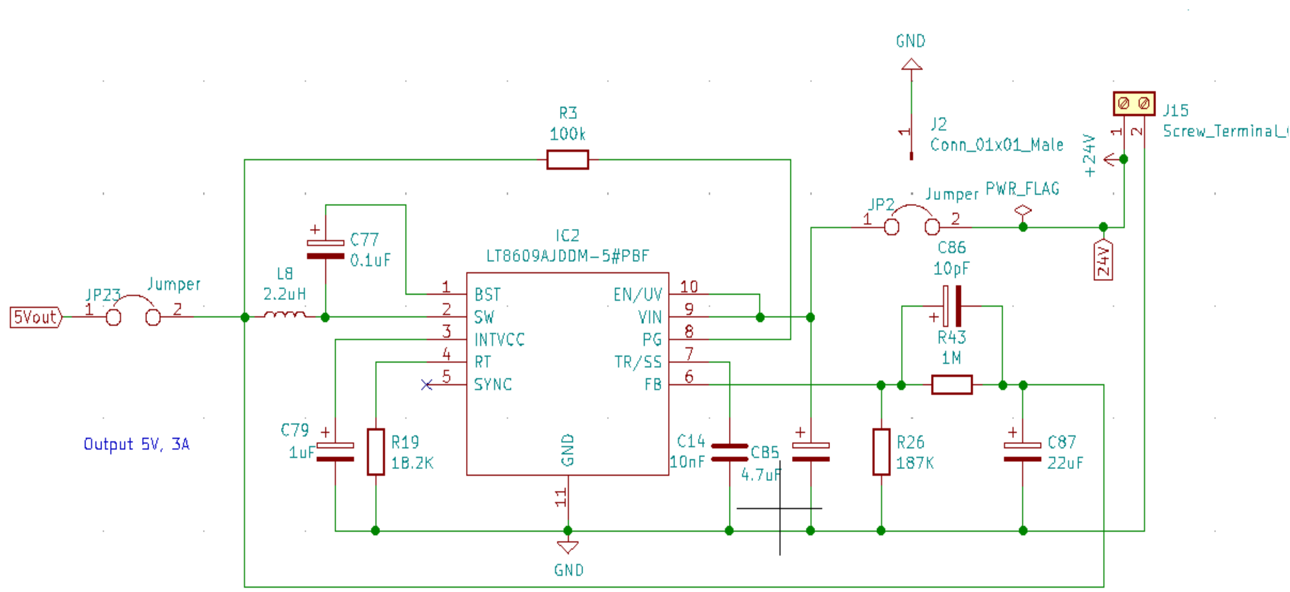

Figure 22: Schematic of 5V voltage regulator.

Digital Potentiometer:

The AD5253 chip is a quad-channel digital potentiometer, and each potentiometer channel functions independently [5]. Only one of the potentiometers is utilized in the final design; the others are left unconnected and may be used to implement other features in a future design revision. The schematic for the digital potentiometer can be seen in Figure 23.

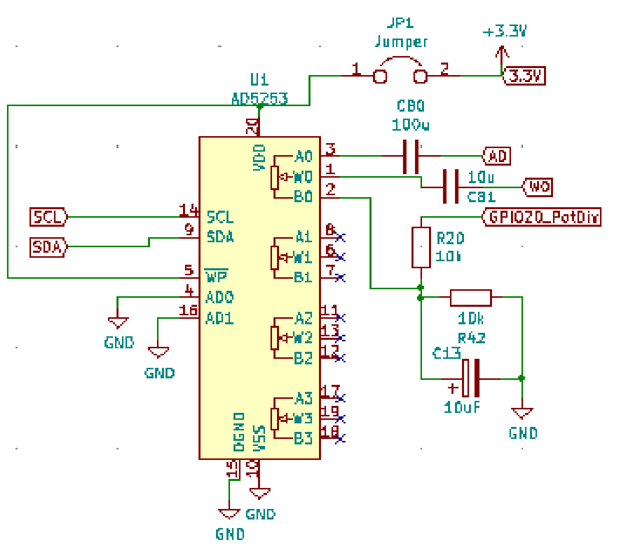

Figure 23: Digital Potentiometer Schematic

Filters:

Each stereo channel has a first order high pass filter with a resistor value of 1 kΩ and a capacitor value of 1uF. The high pass filter is shown in Figure 24.

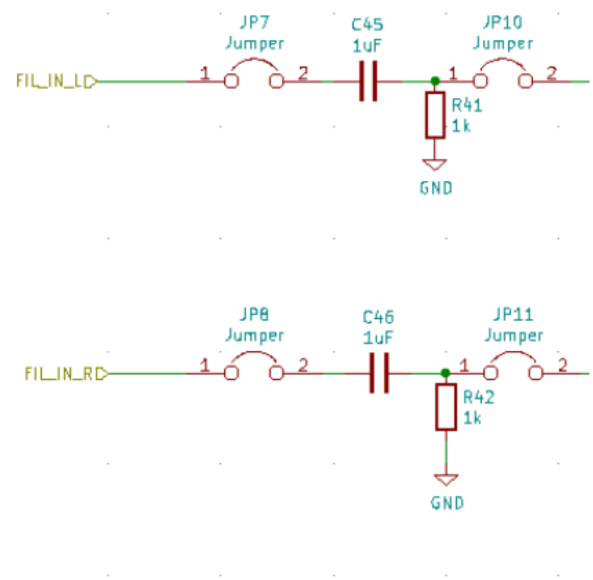

Figure 24: High Pass Filter Schematic for Each Stereo Channel

The subwoofer channel has a first order low pass filter with a resistor of 5.1kΩ and a capacitor of 0.2uF. The low pass filter is shown in Figure 25.

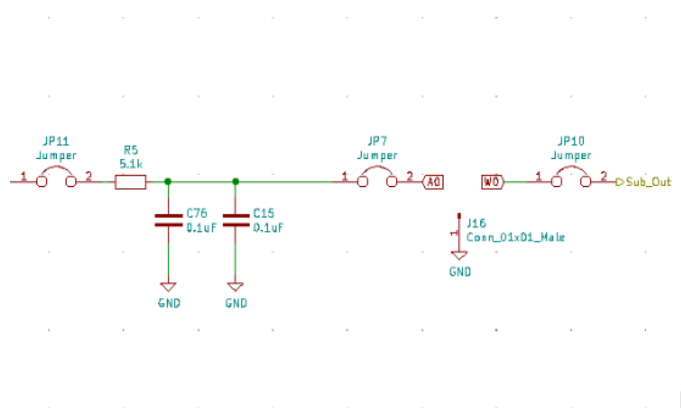

Figure 25: Low Pass Filter Schematic for Subwoofer

Voltage Dividers and Adders:

The voltage dividers on the stereo channels use 4.7kΩ and 1kΩ resistors. The passive adder on the subwoofer channels use 5.8kΩ and 1kΩ resistors. The voltage dividers for the stereo and the subwoofer channels can be seen in Figures 26 and 27, respectively.

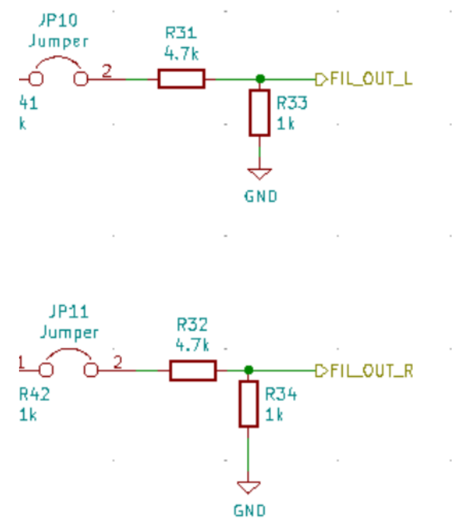

Figure 26: Voltage Divider for Each Stereo Channel

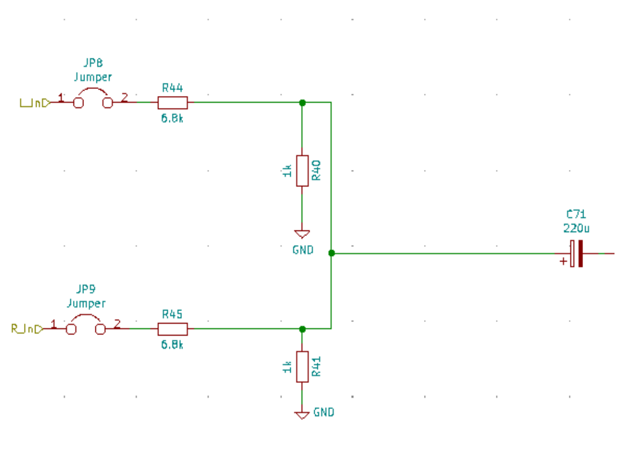

Figure 27: Passive Adder for the Subwoofer

PCB Design and Layout:

Due to the change of amplifiers, it was decided to make the PCB design modular. The design was split into two boards, one that is referenced as the Digital Board and one that is referenced as the Analog Board. The modular design was implemented to allow for design flexibility. If individuals in the future feel the need to use the Digital Board and redesign the Analog Board with the TPA3220DDWR or with other audio amplifiers, that can be achieved. The modularity also allows for an individual to redesign the Digital Board and connect it to the Analog Board containing the LM4766 directly.

The Digital Board acts as the Raspberry Pi-HAT and has headers that connect directly to the Raspberry Pi's 40 GPIO header pins. It consists of the circuits for the DAC, the digital potentiometer, the power supply regulator, and voltage divider as well as the filters for the subwoofer. The PCB layout for the Digital Board can be seen in Figure 28, and the 3D view of this board is depicted in Figure 29.

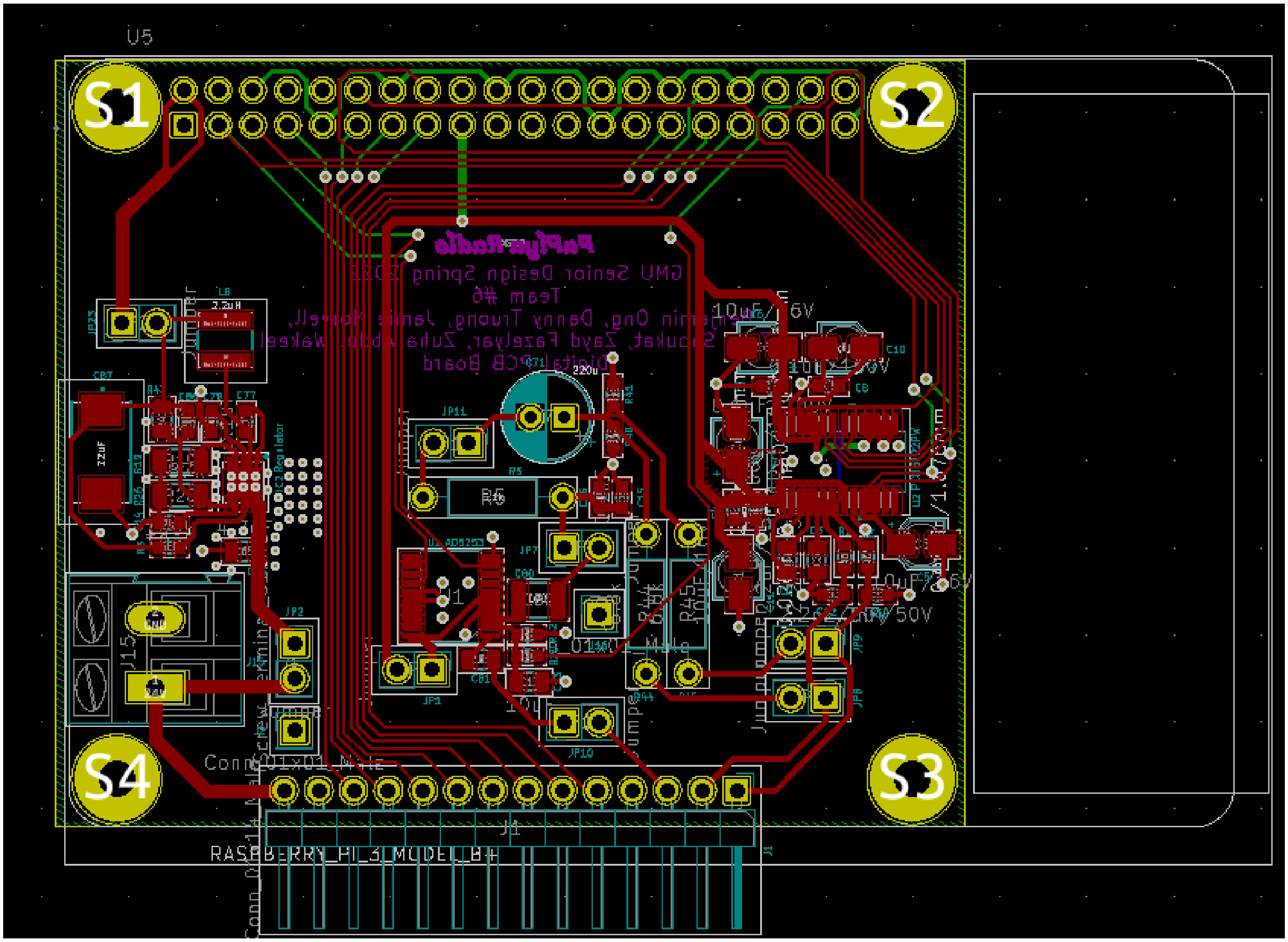

Figure 28: Digital Board PCB Layout

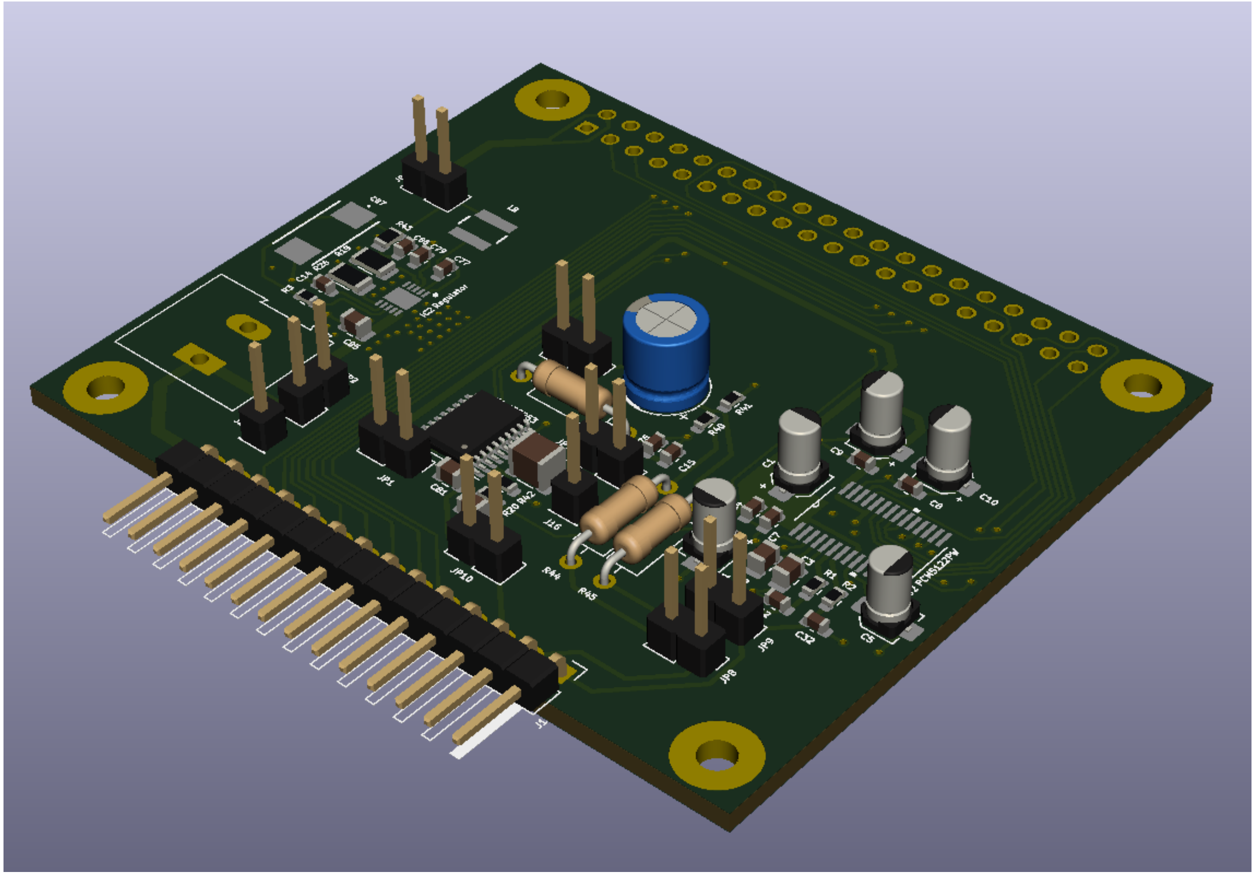

Figure 29: 3D View of Digital Board

The Analog Board contains the following components: the filters for the right and left channels; the two LM4766 chips for the right, left, and subwoofer channels; the heat sinks for each amplifier; and the screw terminals to connect to the exterior speakers. This board also contains four mounting holes necessary to keep the board level with the Digital Board. The PCB layout for the Analog Board is depicted in Figure 30, and the 3D view can be seen in Figure 31.

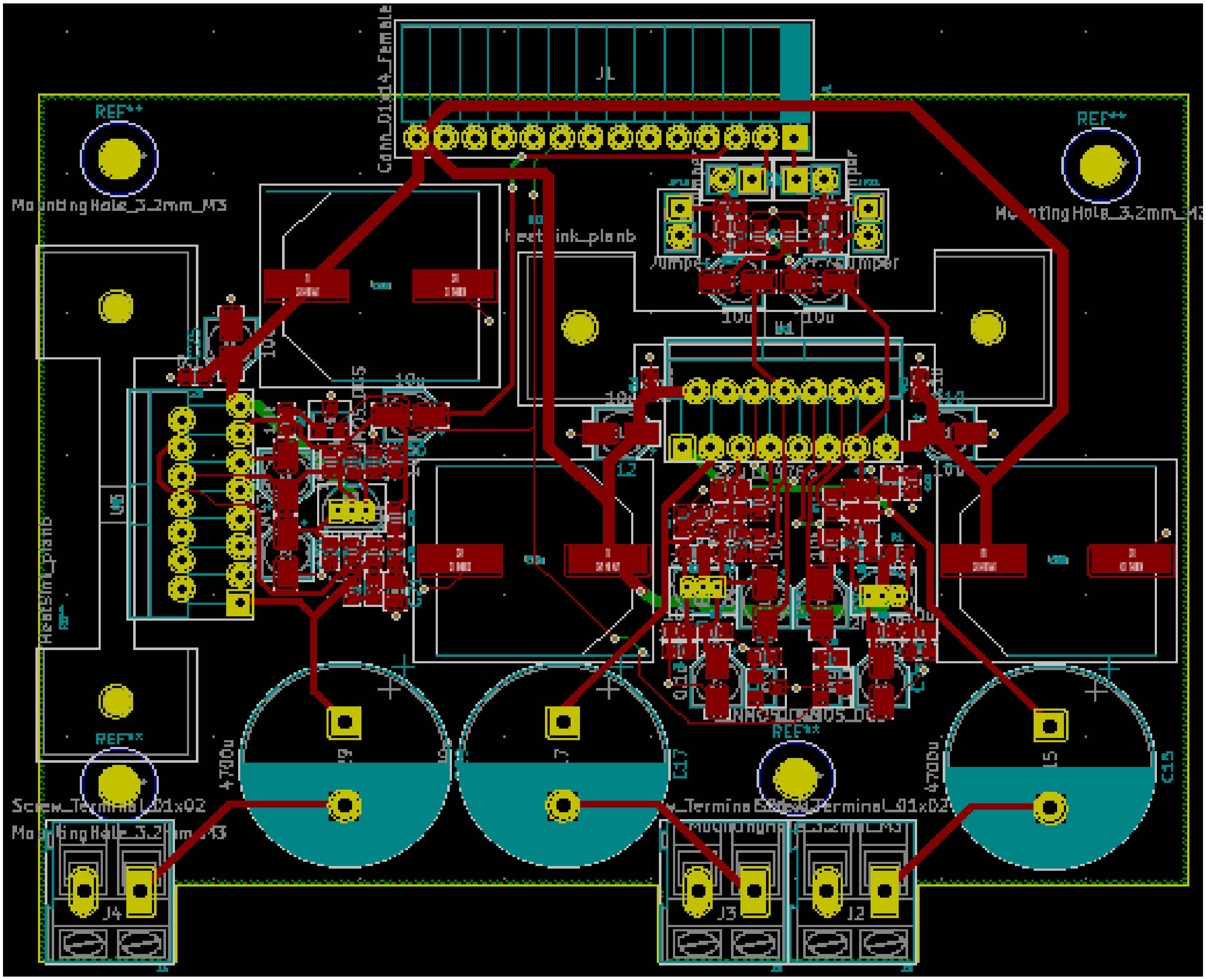

Figure 30: Analog PCB Board

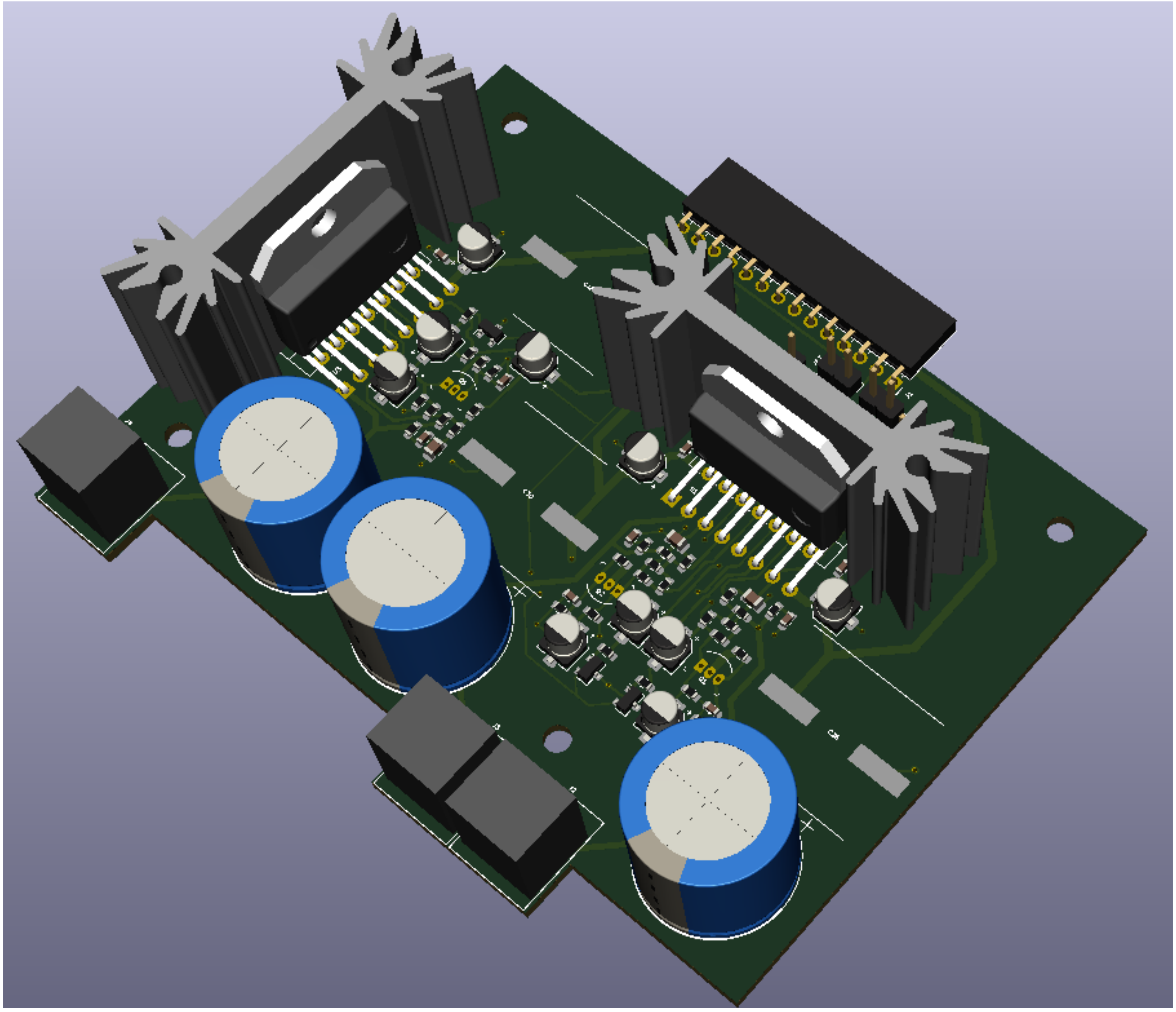

Figure 31: 3D View of Analog PCB

Software Structure:

Figure 32 illustrates the architecture of how the various software components connect.

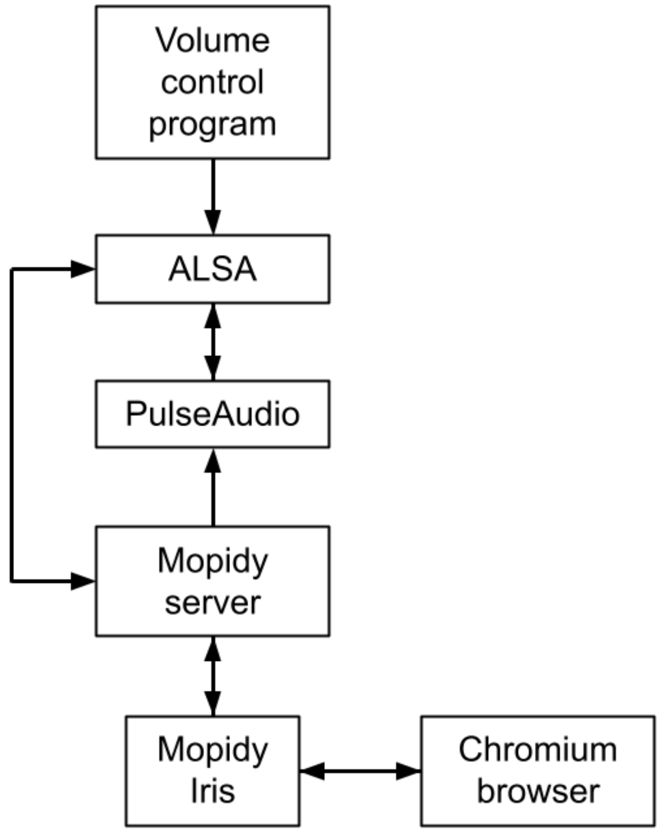

Figure 32: Software Architecture

The discrepancy between the functional decompositions and the software architecture is because the original functional decompositions were developed before the team found and expanded upon open-source software that met the project requirements. Therefore, the software architecture is an accurate reflection of the software running on the device. Both ALSA and PulseAudio software were used to fully develop the software [9].

The front end is served to the user through the Chromium browser as seen in Figure 33.

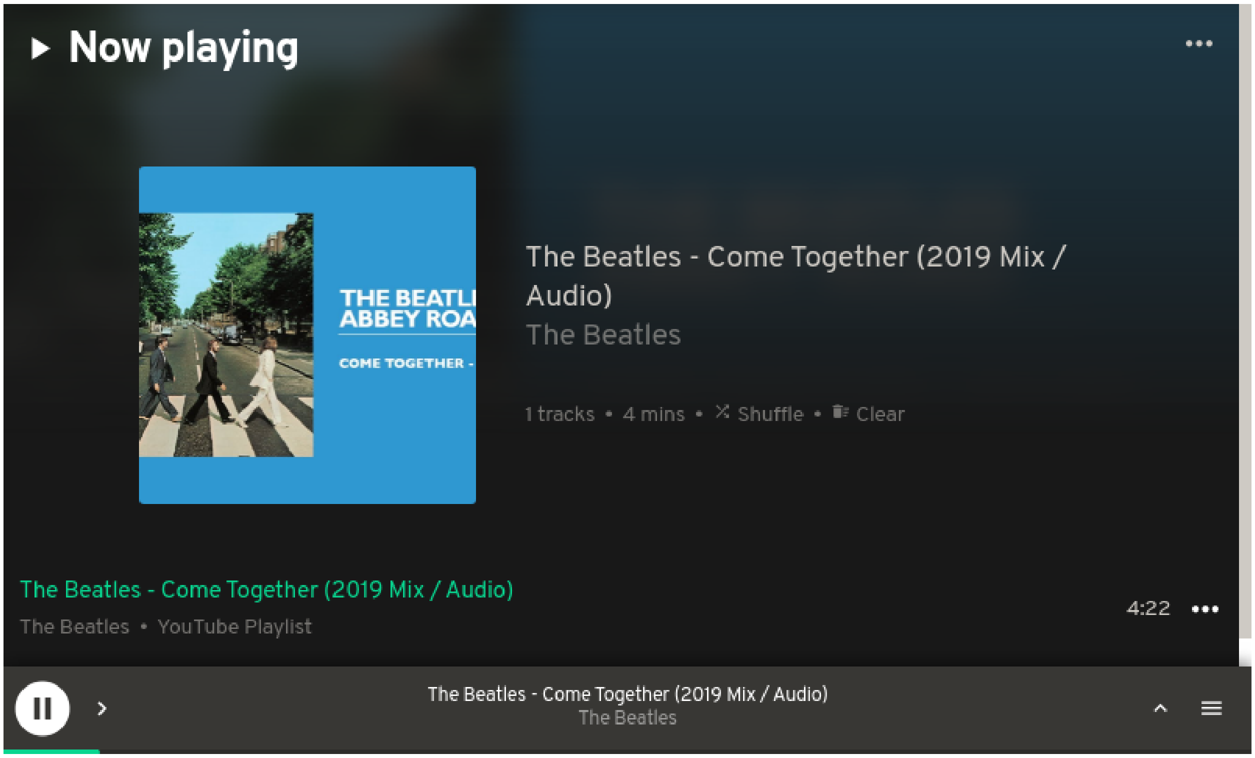

Figure 33: Mopidy Iris Interface

Experimentation and Validation

The team had three experiments planned to validate that the final product meets the requirements for power output, crossover frequency, and media compatibility.

The procedure for the first test case is as follows:

Test Case 1:

1. What is being tested?
    a. Requirements:
        i. The device shall drive stereo speakers and a subwoofer
        ii. The device shall be capable of driving 50W through each stereo speaker ranging between 4Ω to 8Ω and 100W through the subwoofer ranging between 4Ω to 8Ω
            1. A higher power output than stated above is acceptable
2. How is it tested?
    a. Connect a resistor ranging between 4Ω and 8Ω to each of the speaker terminals as a method of simulating the speakers
    b. Connect an oscilloscope across each resistor

c. Play a monotone signal within the human range of hearing at the input of each amplifier channel
d. Gradually turn up the volume in increments of 5%

3. Which data to be collected?
    a. Measure the voltage drop (Vrms) across each resistor for each volume level
4. How are raw data processed?
    a. Use the voltage measurements to calculate the power draw of each resistor for each volume level
5. Characteristics
    a. Plot a graph of power vs. volume for each speaker. An example of the graph format is displayed in Figure 34.

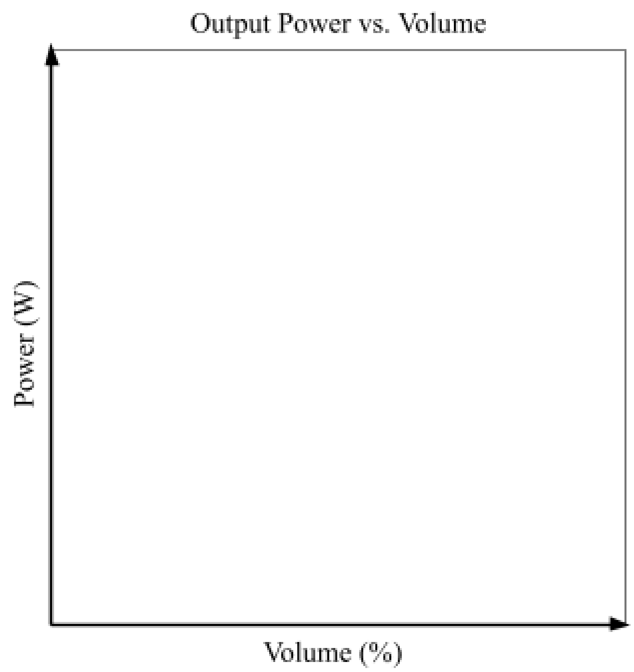

Figure 34: Example Format Graph for Power vs Volume

6. Evaluating success
    a. Answer the following questions:
        i. Does the power draw across each stereo speaker reach at least 50W?
        ii. Does the power draw across the subwoofer reach at least 100W?
        iii. Can the device output a combined 200W out of all speakers?

Instead of measuring the relationship between volume and power output, the maximum power output of the amplifiers was obtained. Figure 36 shows the maximum average power observed for the stereo channels. Figure 37 shows the stereo channel input and output voltages. These values are used to calculate the gain and power displayed in Table 1. Similarly, Figure 38 and Figure 39 show the maximum average power and input and output voltage for the subwoofer, respectively. Table 2 shows the corresponding gain and power calculations.

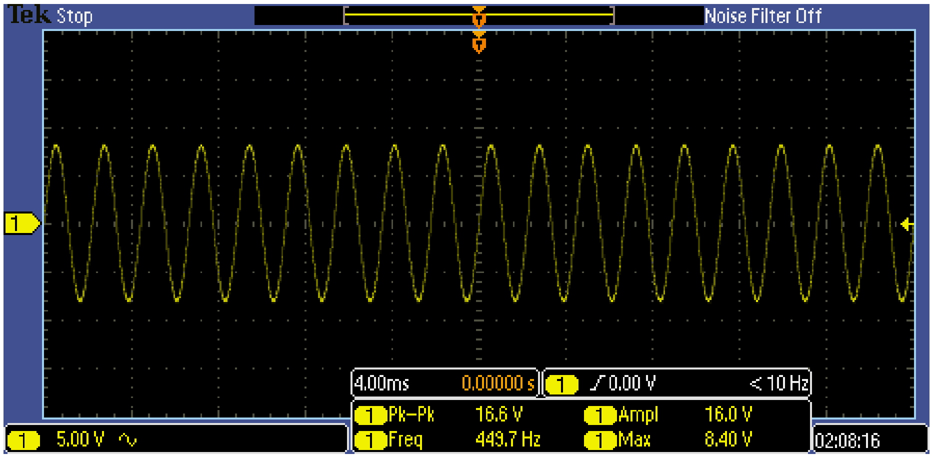

Figure 36: Stereo Channel Maximum Average Power Output

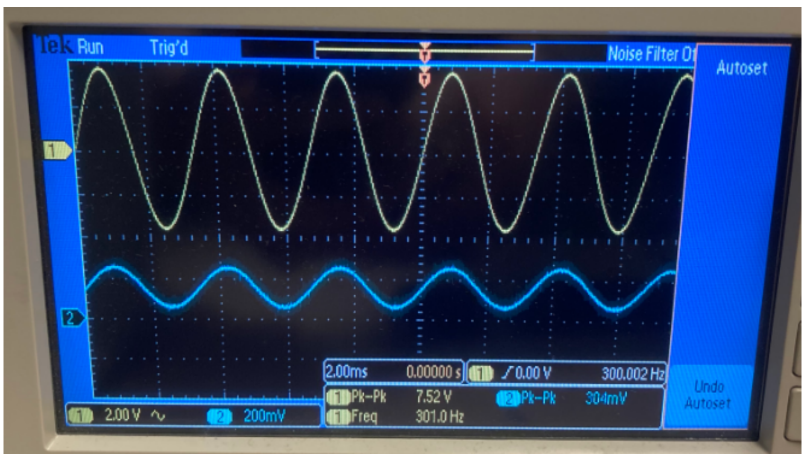

Figure 37: Stereo Channel Output

Table 1: Stereo Channel Power and Gain Calculations

Stereo Speakers			
A	A_{dB}	$P_{6\Omega}$	$P_{4\Omega}$
24.737 V/V	27.867 dB	11.76 W	17.64 W

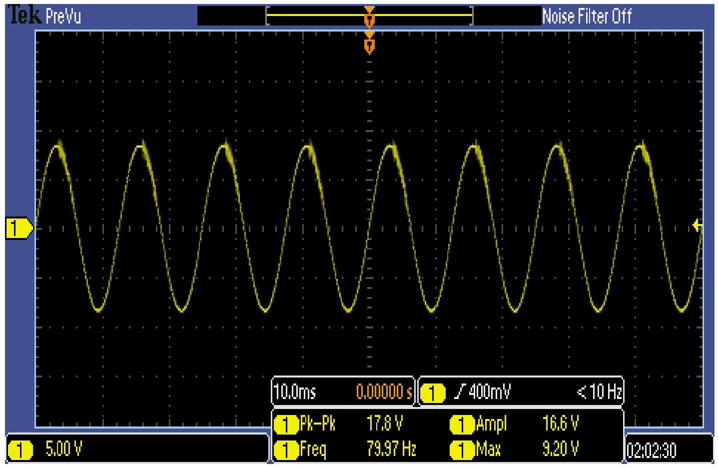

Figure 38: Subwoofer Maximum Average Power Output

Figure 39: Subwoofer Output

Table 2: Subwoofer Power and Gain Calculations

Subwoofer Speaker			
A	A_{dB}	$P_{6\Omega}$	$P_{4\Omega}$
26.923 V/V	28.602 dB	14.10 W	21.60 W

The original power requirements could not be met once the team switched to the LM4766. However, the LM4766 is capable of outputting 30W continuous, and 40W peak, according to its datasheet. Although the resulting power outputs of the system do not meet the datasheet value 30W, this was expected because the system uses a single supply instead of a split supply. The split supply does not provide the full voltage swing for the amplifier, which prevents the amplifier from taking in its maximum input voltage. The lower maximum input voltage results in a lower output voltage, which also leads to a lower output power.

The procedure for the second test case is described below:

Test Case 2:

1. What is being tested?
   a. Requirements:
      i. The device shall support two stereo speakers and a subwoofer
      ii. The crossover frequency between the stereo speakers and subwoofer should be between 100Hz and 200Hz
      iii. The crossover frequency should not change significantly as volume changes
2. How is it tested?
   a. Construct the setup from Test Case #1
   b. Set the volume to 25%
   c. Do a frequency sweep at each amplifier input from 20Hz to 20kHz
   d. Increment the volume by 25%
   e. Repeat the procedure for each increment of 25% up to 100%
3. Which data to be collected?
   a. Measure the voltage drop (Vrms) across each output resistor for each frequency tested to obtain the frequency response for each output
4. How are raw data processed?
   a. For each volume level, graph the frequency response of all three speakers on the same plot
   b. The plot will have three curves; find the intersection between the subwoofer curve and each stereo speaker curve
   c. There will be a pair of intersections for each volume level for six total data points
   d. Take note of the frequency of each data point: these are the crossover frequencies
5. Characteristics
   a. Graph the frequency response curves for all three speakers on the same plot for each volume level. An example of the graph format is seen in Figure 35.

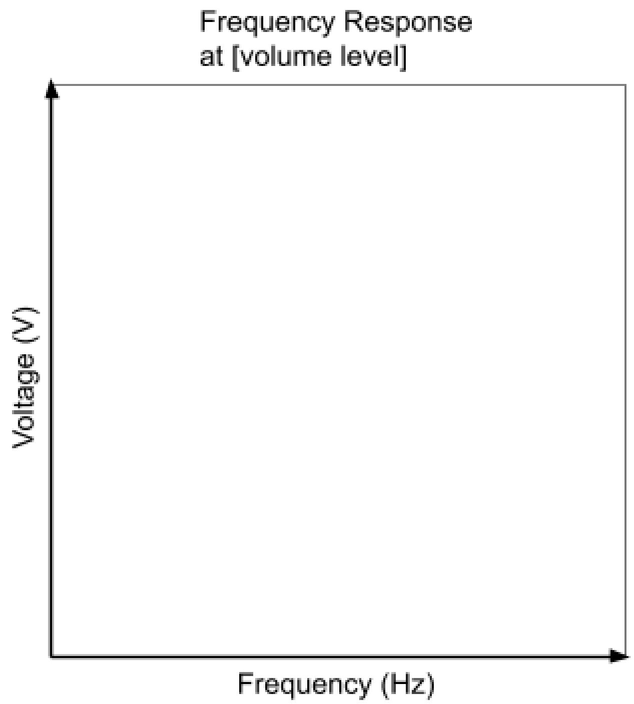

Figure 35: Example Format Graph for Voltage vs Frequency

6. Evaluating success
   a. Through answering the following questions:
      i. Are all of the noted crossover frequencies similar to each other?

Instead of measuring the relationship between volume and crossover frequency for multiple volume levels, only two volume levels were measured. Table 3 shows the data taken for the frequency response at 100% volume, and Table 4 shows the data taken for the frequency response at 70 % volume. Additionally, Figures 40 and 41 depict the crossover frequency of the subwoofer system at various volumes: one at full volume and one at 70% volume, respectively.

Table 3: Frequency at 100% Volume

Frequency (Hz)	Stereo Voltage (V_{pp})	Subwoofer Voltage (V_{pp})	Stereo Gain (dB)	Subwoofer Gain (dB)
30	3.4	10.6	10.62957834	20.50611731
40	4.6	12.6	13.25515663	22.0074109
50	6.2	13.4	15.84783379	22.54209597
60	7.4	13.8	17.38463439	22.79758173
70	8.6	14.2	18.68996902	23.04576689
80	9.8	14.4	19.82452151	23.16724984
90	10.8	14.2	20.66847511	23.04576689
100	11.6	14.2	21.28915978	23.04576689
110	12.2	13.8	21.72719661	22.79758173
120	13	13.4	22.27886705	22.54209597
130	13.6	13	22.67077817	22.27886705
140	14.2	12.6	23.04576689	22.0074109
150	14.6	12.2	23.28705712	21.72719661
160	15	11.6	23.52182518	21.28915978
170	15.2	11.2	23.63687176	20.98436045
180	15.4	10.8	23.75041442	20.66847511
190	16.2	10.6	24.19030029	20.50611731
200	16.2	10.2	24.19030029	20.17200344
210	16.6	9.8	24.40216176	19.82452151
220	17	9.6	24.60897843	19.64542466
230	17.2	9.4	24.71056894	19.46255707
240	17.4	9.2	24.81098497	19.27575655
250	17.4	8.8	24.81098497	18.88965344
260	17.8	8.6	25.00840005	18.68996902
270	17.8	8.4	25.00840005	18.48558572
280	17.8	8.2	25.00840005	18.27627705
290	18	8	25.1054501	18.06179974
300	18.2	7.8	25.20142776	17.84189205

310	18.2	7.4	25.20142776	17.38463439
320	18.2	7.2	25.20142776	17.14664993
330	18.2	7	25.20142776	16.9019608
340	18.4	7	25.29635646	16.9019608
350	18.6	6.6	25.39025888	16.39087871
360	18.6	6.6	25.39025888	16.39087871
370	18.6	6.2	25.39025888	15.84783379
380	18.6	6.2	25.39025888	15.84783379
390	18.6	6.2	25.39025888	15.84783379
400	18.6	6	25.39025888	15.56302501

Table 4: Frequency at 70% Volume

Frequency (Hz):	Stereo Voltage (V_{pp})	Subwoofer Voltage (V_{pp})	Stereo Gain (dB)	Subwoofer Gain (dB)
30	0.28	0.606	-11.05683937	-4.350547517
40	0.368	0.752	-8.683043627	-2.475643188
50	0.44	0.8	-7.13094647	-1.93820026
60	0.472	0.776	-6.521160027	-2.202765575
70	0.57	0.84	-4.882502887	-1.514414279
80	0.65	0.84	-3.741732867	-1.514414279
90	0.72	0.84	-2.853350071	-1.514414279
100	0.76	0.84	-2.383728154	-1.514414279
110	0.816	0.832	-1.766196825	-1.597533474
120	0.84	0.816	-1.514414279	-1.766196825
130	0.904	0.78	-0.8766313905	-2.158107946
140	0.952	0.76	-0.4272610323	-2.383728154
150	0.984	0.744	-0.1400980314	-2.568541289
160	1	0.72	0	-2.853350071
170	1.03	0.704	0.2567444941	-3.048546817
180	1.02	0.688	0.1720034352	-3.248231235
190	1.08	0.688	0.6684751097	-3.248231235
200	1.11	0.656	0.9064595757	-3.661923212
210	1.14	0.624	1.138097027	-4.096308206
220	1.15	0.608	1.213956807	-4.321928415
230	1.16	0.6	1.289159785	-4.436974992
240	1.14	0.6	1.138097027	-4.436974992
250	1.18	0.592	1.437640146	-4.553565866

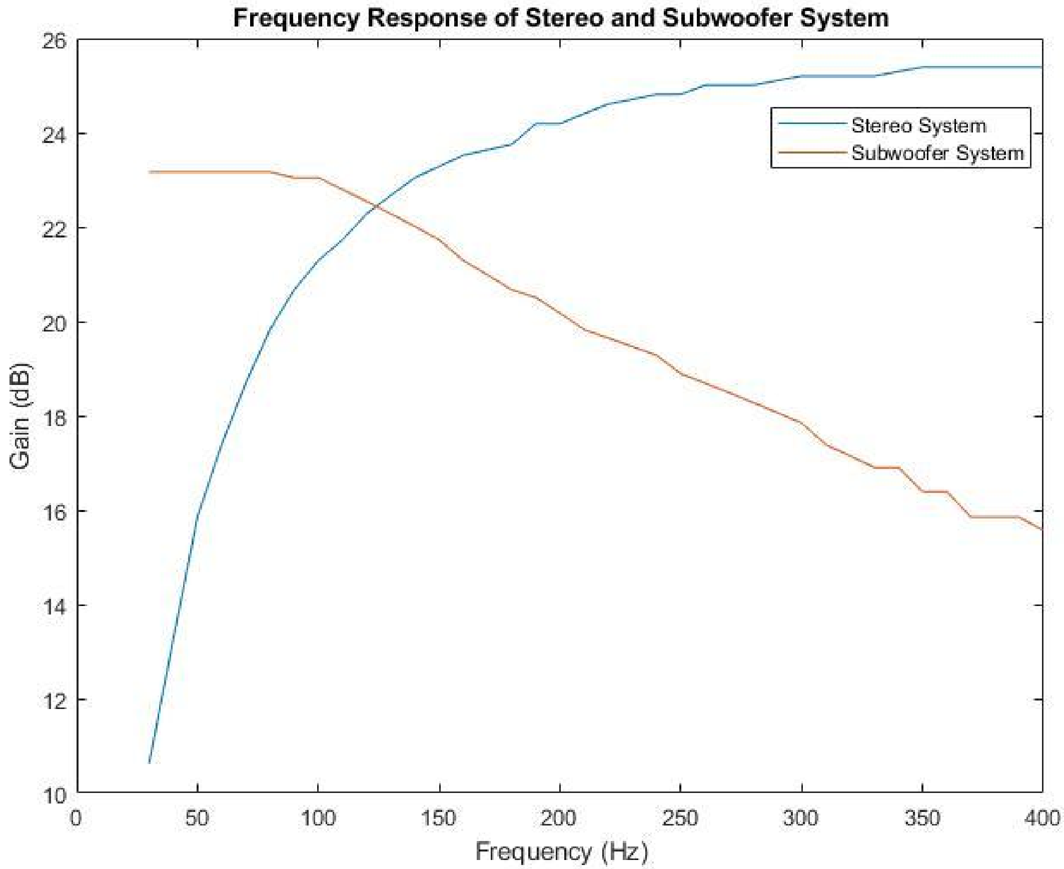

Figure 40: Frequency Response of Subwoofer at Full Volume

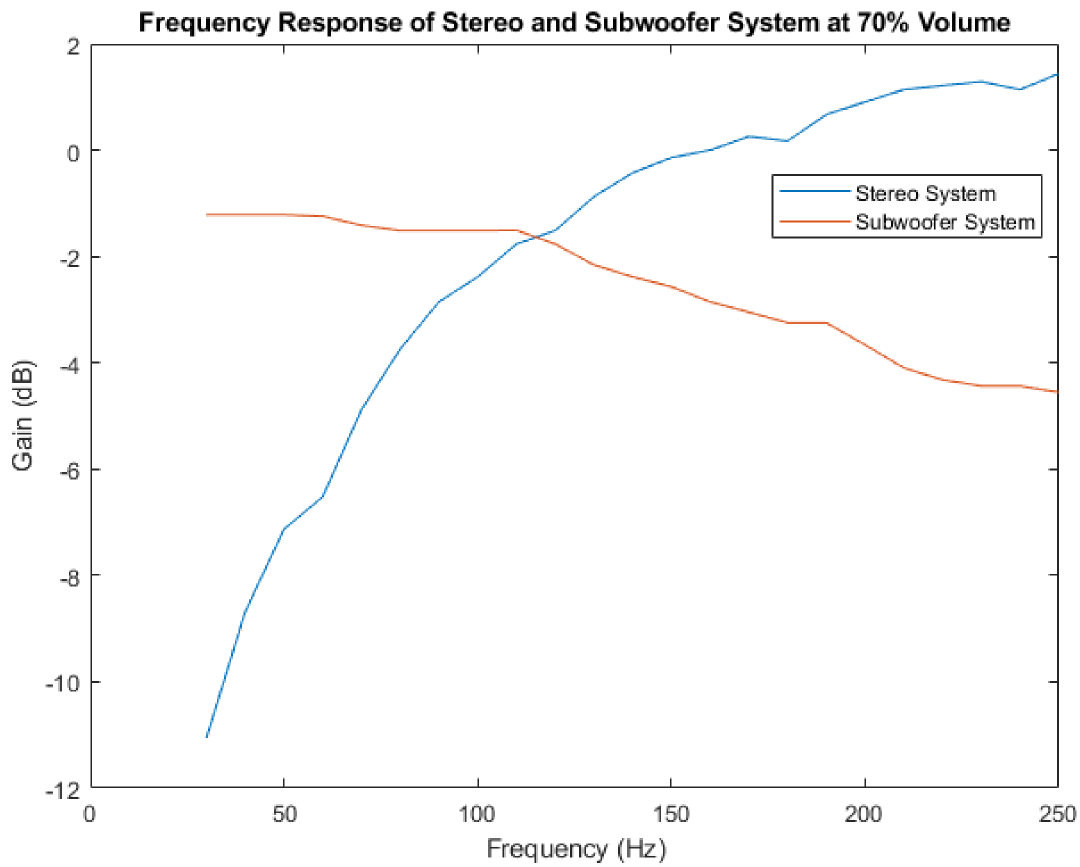

Figure 41: Frequency Response of Subwoofer at 70% Volume

From these data, the team concluded that the volume does not affect the crossover frequency. The crossover frequency was within the range of 100Hz and 200Hz. However, the cutoff frequency for the low pass filter of the subwoofer was not the same as the cutoff frequency for the high pass filter of the stereo channels.

The procedure for the third test case is detailed below:

Test Case 3:

1. What is being tested?
    a. Requirements:
        i. The device shall play music from a NAS, USB drive, and Internet radio
2. How do you test it?
    a. There is no experimental setup here
    b. Simply demonstrate that a variety of music sources are compatible with the device
3. Which data are to be collected?
    a. n/a
4. How are raw data processed?
    a. n/a

5. Characteristics
    a. n/a
6. Evaluating success
    a. Answer the following questions:
        i. Can the device play music stored on a NAS?
        ii. Can the device play music stored on a USB drive?
        iii. Can the device play music from Internet radio stations?

The tests determined that the system can play media from the aforementioned sources.

Due to time constraints, the in-depth experiments that were described in the test cases could not be performed for Test Case 1 and Test Case 2. However, the results are still conclusive enough to determine whether the system requirements were met.

Other Matters

Reasons for Project:

Sound quality and pricing are the most important elements to consider when evaluating modern speaker systems. A high-end home speaker system can cost between $300 to $2700 [10]. Moreover, many speaker systems do not have a subwoofer; instead, they include a left and right speaker or a two-channel speaker bar. Most speaker systems also lack an interface that allows users to engage directly with the device. A product that more closely resembles this project is the Pimoroni Pirate Radio Pi Zero W Project Kit [11]. This kit is the most advanced system on the market and allows local files such as MP3 and FLAC to be played through a built-in 1W speaker. Thus, the kit provides subpar audio quality due to its mono setup.

The PaPiya Radio adopts the beneficial features of the products that are currently on the market, while aiming to be on the lower end of typical speaker system prices. These elements are combined with the client's requirements to produce a sound system that is simple to use and produces quality audio. Because of this project's unique characteristic of integrating all aspects from hardware to software, the PaPiya Radio is a one-stop solution for many at-home Raspberry Pi radio enthusiasts.

Potential Uses:

Many Pi-HATs and speaker systems in general do not feature a subwoofer in their design, which limits their systems to 2 channels. The PaPiya Radio features the subwoofer within its 2.1 channel design, and comes in a ready-made chassis, which offers an aesthetic design that is easy to market. The cost of the system is comparable to what is currently on the market. Individuals who are looking to revitalize either old speakers or spare speakers currently not in use can purchase a PaPiya radio to meet their needs.

This project is also a great option for college students or individuals looking to gain more hands-on experience with skills such as: circuit schematic analysis and design; breadboarding;

interfacing with the Raspberry Pi; processing and analysis of analog signals in the time domain and frequency domains; PCB design; soldering, electronic component selection; and mechanical work for chassis construction. These skills are diverse and have proven to be challenging, but achievable.

Cost Figures:

The breakdown of costs for the product and its components can be seen in Tables 5, 6, and 7. The cost breakdown is separated into three different tables. Table 5 is for the cost of the components for the Digital Board, and Table 6 displays the price of the components necessary to complete the Analog Board. Table 7 displays the price for the remaining components to complete the construction of the overall system within the chassis. The overall cost of the product is approximately $354.38. The time spent on development and coding totaled around 915.25 hours, which is displayed in Table 11 under the man-hours section. The cost breakdown for the time spent developing the project is approximately $10,067.75, given that the team was compensated at the Virginia minimum wage of $11 an hour [12]. This would cause the full development cost of the project to be approximately $10,422.13.

Table 5: Digital PCB Cost

Digital PCB Board Cost			
Item	Quantity	Price Per Unit	Total Price
DAC PCM5122PW	1	$7.83	$7.83
Digital Potentiometer AD5253	1	$7.75	$7.75
Regulator LT8609AJDDM	1	$6.54	$6.54
1k Resistor	2	$0.10	$0.20
5.1k Resistor	1	$0.10	$0.10
6.8k Resistor	2	$0.10	$0.20
10k Resistor	2	$0.13	$0.26
18.2k Resistor	1	$0.11	$0.11
100k Resistor	1	$0.02	$0.02
187k Resistor	1	$0.11	$0.11
1M Resistor	1	$0.29	$0.29
10pF Ceramic Capacitor	1	$0.56	$0.56
2.2nF Ceramic Capacitor	2	$0.14	$0.28
10nF Ceramic Capacitor	1	$0.12	$0.12
0.1uF Ceramic Capacitor	7	$0.15	$1.02
1uF Ceramic Capacitor	1	$0.15	$0.15
2.2uF Ceramic Capacitor	2	$0.20	$0.40
4.7uF Ceramic Capacitor	1	$0.71	$0.71
10uF Ceramic Capacitor	2	$0.29	$0.58
10uF Electrolytic Capacitor	5	$0.28	$1.42
22uF Electrolytic Capacitor	1	$1.79	$1.79
100uF Ceramic Capacitor	1	$1.33	$1.33
220uF Electrolytic Capacitor	1	$0.23	$0.23
2.2uH Inductor	1	$0.46	$0.46
Screw Terminal	1	$0.68	$0.68
Header Pin	18	$0.05	$0.90
14 Pin Horizontal Header	1	$1.29	$1.29

40 Pin Extra Long Socket Header	1	$6.05	$6.05
Mounting Screw	2	$0.12	$0.24
Mounting Nut	2	$0.16	$0.32
Separator	2	$0.27	$0.54
OshPark Fabrication	1	$9.57	$9.57

Table 6: Analog PCB Cost

Analog PCB Board Cost			
Item	Quantity	Price Per Unit	Total Price
LM4766 Amplifier	3	$6.65	$19.95
4.7 Resistor	3	$0.44	$1.32
1k Resistor	7	$0.10	$0.70
4.7k Resistor	2	$0.10	$0.20
32k Resistor	3	$0.10	$0.30
10k Resistor	9	$0.13	$1.19
20k Resistor	3	$0.03	$0.10
100k Resistor	6	$0.02	$0.10
200k Resistor	3	$0.15	$0.45
0.1uF Ceramic Capacitor	6	$0.15	$0.88
1uF Ceramic Capacitor	2	$0.15	$0.30
10uF Ceramic Capacitor	3	$0.34	$1.02
22uF Ceramic Capacitor	3	$0.34	$1.02
0.1uF Electrolytic Capacitor	3	$0.18	$0.54
10uF Electrolytic Capacitor	9	$0.19	$1.71
470uF Electrolytic Capacitor	3	$1.68	$5.04
4700uF Electrolytic Capacitor	3	$3.22	$9.66
Heat Sink	2	$1.07	$2.14
Screw Terminal	3	$0.68	$2.04
2N3904 Transistor	3	$0.52	$1.56
Header Pin	8	$0.05	$0.40
14 Pin Horizontal Socket Header	1	$1.92	$1.92
Mounting Screw	4	$0.12	$0.48
Mounting Nut	4	$0.16	$0.64
Separator	4	$0.27	$1.08
OshPark Fabrication	1	$20.25	$20.25

Table 7: Other System Costs

Other System Costs			
Item	Quantity	Price Per Unit	Total Price
Raspberry Pi	1	$35.00	$35.00
Power Supply	1	$35.90	$35.90
LCD touchscreen	1	$55.00	$55.00
Chassis	1	$20.00	$20.00
Exterior Regulator	1	$14.83	$14.83
Rotary Encoder	2	$4.50	$9.00
Rotary Encoder Adapter	2	$5.95	$11.90
USB Socket	1	$13.89	$13.89
Ethernet Socket	1	$9.99	$9.99
Power Socket	1	$1.90	$1.90
Speaker Output Terminals	3	$3.00	$9.00
Buttons	2	$0.29	$0.58
Button Caps	2	$0.70	$1.40
Diode	1	$0.50	$0.50
Ethernet Cable	1	$2.00	$2.00
USB-USB Cable	1	$1.49	$1.49
Right Angle HDMI Adapter	1	$6.99	$6.99
HDMI Cable	1	$0.00	$0.00
Micro-USB Cable	1	$0.00	$0.00
Jumper Wires	15	$0.10	$1.46
Speaker Wire	1.5	$0.04	$0.06
Crimps	9	$0.10	$0.90
Pack of Mounting Screws	1	$1.29	$1.29
Pack of Screws for Shelf	1	$1.29	$1.29
Pack of Wingnuts	1	$1.29	$1.29
Pack of Nuts	1	$1.29	$1.29
Sheet Metal Shelf	1	$1.25	$1.25

Alternatives to the Implemented Design:

The team considered 3D printing a custom enclosure with a slanted front panel to house all the components necessary for the system to function. This would have been done utilizing knowledge of CAD software and printing the design using a 3D printer located at the MIX at George Mason University (GMU). This design was not implemented because the metal chassis introduced passive cooling to the system. Furthermore, the choice of a ready-made metal enclosure saved design time that was then used for other aspects of the project.

The selection of the LCD touchscreen on the front panel is another alternative design. Initially, a 7-inch touchscreen created by Adafruit for the Raspberry Pi was chosen as the user's main form of interaction with the system. However, this touchscreen required several connections to the RPi's GPIO pins, and this was not ideal as the team preferred to have as many GPIO pins available for the rest of the system. Instead, a 5-inch touchscreen display requiring only HDMI and USB connections to the Pi was chosen at a similar price of nearly $60 [6].

Maintenance:

Presently, there is no straightforward way to perform device maintenance.

For the hardware, the modularity of the PCB opens the possibility of replacing only part of the PCB if some component on it were to stop functioning. But each individual module is not designed with repairability in mind. The PCB would have to be redesigned with even more modularity in mind to increase repairability.

For software, there is no mechanism for serving updates to the device. However, most of the software running on the device is installed from the package manager, which already handles package updates. Any updates for the custom software components can be pulled from the team's Github repository. These tasks can be implemented with a simple shell script that calls the appropriate commands to install updates from the package manager and pull the Github repository. The shell script can be set to run on a schedule to periodically check for updates without user intervention.

Retirement and Replacement of Product:

The overall system contains electrical components, therefore the lifetime of the system is determined by the components that will age the fastest. In the PCB design, the 4700uF electrolytic capacitors have an operation lifespan of 2000 hours at 85°C. Since the product is meant for household use, it is unlikely that the ambient temperature of the system will exceed 85°C. Using Equation 7, the lifespan of these capacitors can be calculated, where L is the estimated lifetime in hours, L_0 is the lifetime at the rated temperature, T_{max} is the rated temperature, and T_a is the ambient temperature [13]. If room temperature is taken at 20°C, then the estimated lifetime of the 4700uF capacitors is estimated to be 181,019 hours, or approximately 20 years, according to Equation 7. However, the system does produce heat within the chassis which can contribute to a shorter lifespan of the capacitors.

Additionally, the lifespan of the Raspberry Pi is estimated to be approximately seven to ten years, since the Pi's SD card has a lifetime in this range [14]. This reduces the lifespan of the overall system to approximately seven to ten years. Once the end of life (EOL) of the system is reached, the product can be disposed of at a local electronic recycling facility by the user. According to Virginia law, the product can be returned to the manufacturer to be disposed of at their recycling facility [15].

$$L = L_0 \times 2^{\left(\frac{T_{max} - T_a}{10}\right)} \tag{7}$$

Administrative Information

Project Progress:

The overall project has progressed as planned, since a functioning 2.1 channel amplifier was produced with the capability to play songs from internet radio, USB, and NAS. However, various requirements could not be met, and minor setbacks occurred during the development of the system. First, it is important to note that there has been a chip shortage throughout the entire development of the system. This external issue has led to problems in acquiring components and their reserves.

Initial development of the system made use of a Class D amplifier chip, the TPA3220DDWR. This chip was a QFP package chip and surpassed the power output requirements specified, producing 60W in stereo mode and 110W in mono-channel mode. Four of these chips were purchased: two for early prototyping efforts and two for the final PCB design. During early prototyping, the two chips were soldered onto breakout boards for testing purposes. The team constructed the stereo and mono channel schematics from the datasheets.

During experimentation in the Fall 2021 semester, both the chips did not function as expected and could not produce an output signal. It was determined that these two chips must have been damaged. Due to the chip shortage, the TPA3220DDWR was out of stock and only the two chips used for the final product were the only reserves left. The group had to decide to either proceed without testing the amplifier chips or to select a new amplifier chip that could be tested and used in the design.

Upon return for the Spring 2022 semester, the team decided to test these chips one more time, while also selecting a backup amplifier. The TPA3220DDWR chips were tested one last time and determined to be nonfunctioning. Chip selection for the new amplifier was limited due to the chip shortage. Furthermore, a through-hole chip was preferred for the new amplifier to simplify breadboard testing. Therefore, the team proceeded with the LM4766 amplifier. The LM4766 amplifier was chosen for the following reasons: it could be supplied with the same power supply as the TPA322DDWR, it had a simplified circuit schematic, it was through-hole, and its schematic contained some components that would have been used with the previous amplifier. However, the amplifier did not meet the output power requirements, producing 30W of continuous power per channel. The switch to this amplifier was reasonably seamless, as the breadboarded amplifier functioned properly.

It is also important to note that the testing and implementation of the DAC and digital potentiometer raised some minor issues, but were easily resolved. For instance, a DC offset had to be introduced to the music signal to be able to pass it through the digital potentiometer. Additionally, the DAC introduced some minor issues regarding the establishment of the address to the device. In the tutorials for how to activate DAC drivers on the Raspberry Pi, there is no indication that the DAC needs to be at a specific I^2C address [16][17]. However, hours of troubleshooting proved that the problem with the DAC narrowed down to the address. A forum post was used to fix the issue with the address, and the DAC was able to operate properly [18].

Additionally, an active adder was used to combine the right and left signals for the subwoofer output during the circuit design phase of development. The LM386 audio power amplifier was used to actively add these signals, but it introduced various problems into the system. One problem was that the device induced noise and had to be supplied with a higher voltage of 9V to pass music signals without saturation. Another problem was that a voltage divider had to be used on the input of this device to reduce the music signal, as the LM386 could only receive a maximum of 0.4V while the DAC had a typical output of 3V. The voltage dividers for the LM386 caused the output signal of the system to have a very low power output. An active adder was expected to avoid crosstalk between the channels more efficiently than a passive adder. Therefore, a significant amount of time was spent attempting to eliminate the problems that the LM386 introduced into the system. Eventually, it was decided to test the system without the LM386. A passive adder was implemented using two voltage dividers to reduce the music signal from 3V to approximately 350 mV for the LM4766 amplifier. The passive adder was tested to see if crosstalk was observed, and it was determined that crosstalk did not occur. Furthermore, the output power was increased, saturation did not occur, and the noise was eliminated after removing the LM386. The experimentation with the LM386 took approximately a month, which did setback the development of the circuit and PCB designs.

Once the circuit schematics were finalized, the PCB development was able to begin. A modular design was chosen, where one board contains all the digital components such as the DAC, the digital potentiometer, and voltage regulator, while the other board contains the amplifier circuits. This was done so that the Digital Board could be completed while the amplifier breadboard circuit could still be experimented with. This also allowed for the amplifier board to be swapped out in the future with another amplifier PCB that could meet the power requirements. PCB design and development went seamlessly, with various checks of the layout before sending out the digital board for fabrication. Due to the various setbacks previously discussed, the amplifier board had to be expedited which increased the cost of fabrication. Both the boards came in approximately at the same time, and the components were soldered within three days.

Initial testing of the voltage regulator revealed that it did not function properly. The regulator did not step down the 24V from the power supply to 5V for the Raspberry Pi. Instead, the regulator passed the total voltage. Since testing of the regulator did not occur before it was on the PCB, jumpers were included to bypass the regulator circuit. This allowed for an external regulator to be used to step down the voltage for the Pi. The team proceeded with the external regulator to supply the Pi and began testing the digital components which worked seamlessly.

The amplifier board did not initially function as designed. It was found that the footprints used for the transistors were incorrect. Since through-hole technology was used for the transistor, the pins were able to be moved to fix the error. Additionally, it was discovered that the input capacitors were flipped on the schematic and therefore had to be flipped on the PCB. After these errors were resolved, the amplifier board functioned properly. The system was then assembled and validation testing was performed, demonstrating that the system worked as expected.

In addition to the hardware setbacks, there were also significant challenges regarding software development. The major software development tasks were: driver development, user interface development, and automation.

Each of the chips communicating on the I^2C line required drivers to either be written by the team or used from other sources. The drivers for the rotary encoders originally came from an Adafruit library [19] that includes drivers for many devices sold by Adafruit. The rotary encoder driver was reimplemented by the team to only recover the parts of the drivers that were necessary for this project. The drivers for the digital potentiometers were completely written by the team according to the specifications from the datasheet. These drivers were able to function without any issues, as the datasheet clearly described the messaging protocol that the chip expects over the I^2C bus.

The most difficult chip to configure was the DAC. Once the DAC was verified to function in hardwired mode, the challenge was to structure the DAC to operate in I^2C control mode. The datasheet for the DAC provides a detailed description of how the chip is configured and controlled over I^2C, but the process was not intuitive. Open-source drivers for connecting the DAC to the ESP32 series of microcontrollers were found online [20], and the team considered porting that code to Raspberry Pi. But, the team discovered that the Pi has the drivers for the DAC built into the OS and configured the Pi to recognize and utilize the DAC [16][17]. After fixing hardware issues related to the proper address of the DAC, the drivers worked as intended.

The user interface is a Mopidy music server that can be controlled from a web interface called Iris [21]. The team attempted to set up Mopidy as a kiosk on Chromium [22] but was unsuccessful, despite using three different Raspberry Pi devices. The next solution attempted was to use Ubuntu Frame, which is a software package designed for turning a computer into a kiosk [23]. However, this solution proved unreliable as it would sometimes prevent the user from interacting with the device. The working solution for a kiosk mode involved shell scripting to prevent Chromium from malfunctioning. However, some additional functionality such as an on-screen keyboard is unavailable in the kiosk mode.

Additionally, the requirement of being able to play songs sampled at 192kHz was not achieved due to the constraints of the Raspberry Pi. The DAC has the capability of playing signals sampled at 384kHz. However, the Raspberry Pi downsamples the audio signal to 44.1kHz when outputting to the DAC. This problem can be partially resolved by adjusting the output sampling frequency of the Pi. This would cause songs that are recorded at lower resampling frequencies to be upsampled [24]. Therefore, it was decided to keep the sampling frequency set to the default of 44.1kHz.

The final challenge was to develop a setup script which installs and configures all of the software, which was achieved after researching automation tasks in Linux.Once launched, the script installs packages, edits configuration files, and copies files into the correct locations. When the script finishes, it reboots the device and the device automatically launches the Mopidy Iris interface upon boot.

Even with all these setbacks, the final product was able to be constructed and tested successfully. Although it does not meet all the requirements, the product still outputs signals at a high enough output power to produce quality audio. Furthermore, the project was also designed for potential legacy applications to enable other groups to produce an amplifier board that could meet the power requirements.

Funds Spent:

The funds spent were broken down into two tables. Table 8 displays the funds spent during ECE 492, and Table 9 contains the funds spent during ECE 493. The total cost spent by the team was $813.54. This was above the $600 budget by $213.54. The incurred extra cost of $231.54 can be accounted for by the switch in amplifiers, the expedited shipping costs, the purchase of reserve components, damaged components, and experimental components. As addressed previously, it was decided to switch to the LM4766 amplifier. Therefore, the team had to purchase the new amplifiers, as well as some new components needed for the amplifier circuits. The team attempted to select an amplifier where the previously purchased components for the initial amplifier could be applied to some of the components of the new amplifier in order to save costs. Additionally, because of the switch in amplifiers, there was a delay in producing the amplifier PCB board, which resulted in the team needing to pay for expedited fabrication of the board to receive it on time. It was also essential to buy reserves for all components due to the chip shortage. Therefore, there were extra components that were unused. In the developmental process, there were some damaged components. In particular, the rotary encoders needed to be replaced and incurred another extra cost in development. Lastly, the team purchased a practice regulator, digital potentiometer, and amplifiers to help learn more about how to design the system and its operation.

Table 8: ECE 492 Cost Breakdown

Date	Item	Qty	Cost	Total
10/10/21	LCD touchscreen	1	$60.41	$60.41
10/10/21	Voltage converters	2	$5.99	$11.98
10/15/21	LM386 audio amp	3	$0.66	$1.98
10/17/21	3.5mm audio jack	2	$0.50	$1.00
10/20/21	32-bit DAC	2	$11.61	$23.22
10/20/21	24-bit DAC Practice DAC	1	$19.04	$19.04
10/19/21	TPA3220DDWR Amplifier	4	$7.84	$31.36
10/26/21	Subwoofer Speaker	1	$18.58	$18.58
11/5/21	80 breadboard header pins	1	$2.00	$2.00
11/12/21	4.7 ohm resistors	10	$0.05	$0.50
11/20/21	Stereo Sony speakers	1	$10.00	$10.00
			Total:	180.07

Table 9: ECE 493 Cost Breakdown

Date	Item	Cost	Qty	Total
1/19/2022	SMD 5V Regulator	$21.85	1	$21.85
2/5/2022	PCB Components	$115.83	1	$115.83
1/20/2022	Rotary encoder	$4.50	4	$18.00
1/20/2022	Rotary encoder adapter	$5.95	4	$23.80
1/30/2022	Rotary encoder	$4.50	3	$13.50
1/30/2022	Rotary encoder adapter	$5.95	3	$17.85
1/20/2022	Digital potentiometer on breakout (Practice)	$4.95	2	$9.90
11/30/2021	USB to TTL adapter	$9.95	1	$9.95
1/18/2022	Digital potentiometer SMD	$7.75	3	$23.25
1/30/2022	Total Shipping and tax	$54.21	1	$14.73
1/27/2022	Power supply	$35.90	1	$35.90
3/11/2022	External 5V Power Supply Regulator	$14.83	1	$14.83
2/27/2022	Chassis	$20.00	1	$20.00
4/1/2022	Digital PCB Board Fabrication	$36.05	1	$36.05
4/5/2022	More PCB Components	$33.02	1	$33.02
4/6/2022	Analog PCB Board Fabrication	$128.85	1	$128.85
4/8/2022	Sockets For Chassis	$35.38	1	$35.38
4/8/2022	Power socket crimps	$0.90	1	$0.90
4/29/2022	Right Angle HDMI Adapter	$20.00	1	$20.00
			Total:	$633.47

Man-Hours Devoted:

The team of six has devoted a significant amount of time to complete the project successfully. The team would meet twice a week, with a virtual stand-up meeting typically on Monday and an in-person meeting with the Faculty Advisor (FA) on Thursday. Tasks would be assigned weekly to each team member on Thursdays. These tasks would typically take one to two weeks to complete, but sometimes they take longer if there was a setback or if approval from the faculty advisor was needed. The Man-Hours spent developing and working on the tasks assigned are highlighted in Table 10. This table is broken up by week, and the team member was responsible for entering the amount of time spent working on project development. The total hours spent per person can be seen in Table 11 with the whole team devoting approximately 915 hours to the completion of the project. It is important to note that the team worked on the project during winter break. Some team members were traveling during winter break and could not contribute to the project during that time.

Table 10: ECE 493 Man-Hours Per Week

Date	**1/10**	**1/17**	**1/24**	**1/31**	**2/7**	**2/14**	**2/21**	**2/28**	**3/7**	**3/14**	**3/21**	**3/28**	**4/4**	**4/11**	**4/18**	**4/25**	**5/2**
Benjamin Ong	3	3	2	10	12	20	3	10	10	12	10	10	10	15	15	20	10
Danny Troung	4	4	4	4	4	20	1	2	4	3	15	9	15	2	18	20	20
James Norvell	6	4	6	5	4	24	2	4	17.5	10	24	23	6	3	10	20	10
Zara Shoukat	2	3	5	12	8	8.25	5	4	4.5	9	24	22.5	3	3	18.5	19	10
Zayd Fazelyar	2	4	5	6	6	12	4	4	5	7	19	11	4	3	17	17	4
Zuha Abdul Wakeel			6	6	8	9	5	4.5	4	6	15	8	6	2	17.5	18	7

Table 11: Total Man-Hours Accumulated

	Total Hours
Benjamin Ong	175
Danny Troung	149
James Norvell	178.5
Zara Shoukat	160.75
Zayd Fazelyar	130
Zuha Abdul Wakeel	122
Total Man Hours	915.25

Lessons Learned

Additional Knowledge and Skills Acquired:

Hardware:

Hardware development was dependent on the availability of components and the specific requirements of each component. Due to the chip shortage, the group quickly learned to have backup components for the various operations of the system. Furthermore, three group members were working on the schematics at the same time during PCB design. To perform this task, GitHub was utilized to provide a seamless collaborative setting. As this was the first time to use GitHub for some of the group members, the team had some challenges using it. KiCad files and file paths for footprints were especially hard to use with GitHub. By the end of PCB development, the team became fairly versed in GitHub and its usage with KiCad.

Additionally, the three members designing the PCB took ECE 436 at GMU, a PCB design class, to prepare them for PCB development. Although this class focused on PCB design, the complex circuits that were implemented in the project were not covered. Therefore, the team needed to learn common industry practices in PCB design.

Although surface-mount components are more desirable, they tend to require external assistance. This cost the team additional time and effort, as the PCB Fabrication Lab at GMU was needed. The team learned how to solder surface-mount components onto a PCB. While system-mount components are smaller than through-hole components, through-hole components are easier to debug. In hindsight, implementing more through-hole components would have been much simpler for the team to debug the amplifier and regulator circuits on the PCBs.

When purchasing PCB circuit components, especially during a chip shortage, it is best practice to reasonably overestimate the specific amount of components required for proper testing and implementation of the design. The team learned that there are few designs capable of operating on the first attempt. Every component featured within the PaPiya Radio failed to function as intended at some point during the development process. The initial amplifier selected for this design never made it past the breadboarding stage of testing and delayed project progress by several weeks as the chip went out of stock. The same can be said for the DAC, as it failed to work initially and the chip went out of stock. However, the chip became available later on. The team made sure to purchase a proper amount of DAC chips during the second purchase to ensure that the progress of the project would not be delayed. The team learned that the availability of smaller components such as resistors and capacitors is never guaranteed either.

Software:

To give the user a seamless experience, the software running on the Pi had to be completely automated. The goal was to create a system that did not require any manual launches or configurations. Enabling all of the functionality to work at startup without requiring any intervention from the command line was challenging and required a lot of research as well as trial and error.

Another challenge was sourcing or writing device drivers for the various chips on board. For the DAC, the Pi had drivers preinstalled [16][17]. For the rotary encoders, drivers available from Adafruit [19] were rewritten for increased simplicity and portability. The potentiometer driver was written entirely by the team, and it was based on the specifications of the datasheet. The group had never studied device registers or written code that was compliant with datasheet specifications. All of the digital devices communicate with the Pi via the I^2C bus, but each has its own messaging protocol to implement. This made driver development difficult.

The final challenge in software development was to integrate the various installed software. Aggregating disparate software together with shell scripts and plugins was a challenge that required much research and experimentation. While most of the software running on the Pi came from open source software, the task of implementing those solutions together was significant.

Another skill that had to be learned was the development of a NAS server. One of the requirements of this project was to allow the device to connect to a NAS. Since a NAS is expensive to purchase, a NAS server was developed on one of the team member's laptops to demonstrate NAS functionality. The team first needed to learn how to set up a Samba server on a Windows laptop. The team also needed to learn how to connect the Pi to the Samba server.

References

[1] “PCM5122,” PCM5122 data sheet, product information and support | TI.com. [Online]. Available: https://www.ti.com/product/PCM5122.
[2] “TPA3220 SymLink - semiconductor company | ti.com.” [Online]. Available: https://www.ti.com/lit/ds/symlink/tpa3220.pdf.
[3] “LM4766 Symlink - Texas Instruments.” [Online]. Available: https://www.ti.com/lit/ds/symlink/lm4766.pdf.
[4] “LT8609/LT8609A/LT8609B (rev. J) - mouser.com.” [Online]. Available: https://www.mouser.com/datasheet/2/609/LT8609_8609A_8609B-1551398.pdf.
[5] “AD5253,” Datasheet and Product Info | Analog Devices. [Online]. Available: https://www.analog.com/en/products/ad5253.html.
[6] Kuklovskaia Elizaveta, “ELECROW Raspberry Pi Touchscreen Monitor,” Amazon. [Online]. Available: https://www.amazon.com/dp/B07FDYXPT7?ref=ppx_yo2ov_dt_b_product_details&th=1.
[7] J. Galt, “How to add a power button to your Raspberry Pi,” Howchoo, 25-Nov-2020. [Online]. Available: https://howchoo.com/g/mwnlytk3zmm/how-to-add-a-power-button-to-your-raspberry-pi.
[8] G. N. Gibson, "6.1 Gain and Feedback," University of Connecticut, [Online]. Available: https://www.phys.uconn.edu/~gibson/Notes/Section6_1/Sec6_1.htm.
[9] “What is the relation between alsa and pulseaudio sound architecture?,” Ask Ubuntu, 01-Oct-1962. [Online]. Available: https://askubuntu.com/questions/581128/what-is-the-relation-between-alsa-and-pulseaudio-sound-architecture.
[10] N. Pino and O. Tambini, "The best wireless speaker 2021: our top picks from Sonos, Bose, Google, and more," Future Publishing Limited Quay House, 8 November 2021. [Online]. Available: https://www.techradar.com/in/news/best-speaker.
[11] “Pimoroni pirate radio - pi zero W with headers project kit,” adafruit industries blog RSS. [Online]. Available: https://www.adafruit.com/product/3477.
[12] “Virginia law,” Code of Virginia Code - Article 1.1. Virginia Minimum Wage Act. [Online]. Available: https://law.lis.virginia.gov/vacodefull/title40.1/chapter3/article1.1/.
[13] G. Bocock, “Electrolytic capacitors determine the lifetime of a power supply,” XP Power. [Online]. Available: https://www.xppower.com/resources/blog/electrolytic-capacitor-lifetime-in-power-supplies.
[14] “FAQ: SD Association,” SD Association | The SD Association, 25-Mar-2021. [Online]. Available: https://www.sdcard.org/consumers/faq/#:~:text=SD%20standards%2Dbased%20memory%20cards%2C%20like%20most%20semiconductor%20cards%2C,and%20reduce%20consumer%20electronic%20waste.
[15] “Public Works and Environmental Services,” Fairfax County Virginia. [Online]. Available: https://www.fairfaxcounty.gov/publicworks/recycling-trash/electronics.

[16] "Configuring linux 4.x or higher," HiFiBerry, 14-Feb-2022. [Online]. Available: https://www.hifiberry.com/docs/software/configuring-linux-3-18-x/.
[17] "Datasheet Dac2 Pro," HiFiBerry, 21-Oct-2020. [Online]. Available: https://www.hifiberry.com/docs/data-sheets/datasheet-dac2-pro/.
[18] "DAC+ pro and RTC module – hifiberry." [Online]. Available: https://support.hifiberry.com/hc/en-us/community/posts/205329129-DAC-Pro-and-RTC-module.
[19] "Adafruit/ADAFRUIT_CIRCUITPYTHON_SEESAW: Seesaw Helper IC driver for circuitpython," GitHub. [Online]. Available: https://github.com/adafruit/Adafruit_CircuitPython_seesaw.
[20] Tommag, "Tommag/pcm51xx_arduino: Arduino Library for TI PCM51XX DAC ICS software configuration via I2C.," GitHub. [Online]. Available: https://github.com/tommag/PCM51xx_Arduino.
[21] "Getting started · jaedb/iris wiki," GitHub. [Online]. Available: https://github.com/jaedb/Iris/wiki/Getting-started#installing.
[22] "Raspberry Pi kiosk using chromium," Pi My Life Up, 23-Feb-2022. [Online]. Available: https://pimylifeup.com/raspberry-pi-kiosk/.
[23] "Canonical launches ubuntu frame, the Foundation for embedded displays," Ubuntu, 06-Oct-2021. [Online]. Available: https://ubuntu.com/blog/canonical-launches-ubuntu-frame-the-foundation-for-embedded-displays.
[24] "How to change audio bit depth and sampling rate?," Ask Ubuntu, 01-Feb-1960. [Online]. Available: https://askubuntu.com/questions/138611/how-to-change-audio-bit-depth-and-sampling-rate.

Appendix A: Proposal

Project Proposal

Kapstone Raspberry Pi Radio
(KRPiR)

Team #6
Benjamin Ong
Danny Truong
Jamie Norvell
Zara Shoukat
Zayd Fazelyar
Zuha Abdul Wakeel

Faculty Sponsor: Dr. Kaps

Executive Summary

Currently, there are many speaker systems on the market that allow users to play music from internet radio stations or streaming services and output audio through a 2.1 channel amplifier. However, these systems are often expensive or do not have a user interface that enables the user to interact with the device directly: the user can only control the speakers via Bluetooth or remote. Furthermore, many affordable stereo sound systems do not include a subwoofer. The aim of this project is to develop an affordable device that has the rich sound quality of an expensive speaker system through the use of a 2.1 channel amplifier that includes two stereo speakers as well as a subwoofer. In addition, the device will have an LCD touchscreen and hardware volume knobs to allow the user to access audio controls through the device directly. The use of a subwoofer will enhance the sound quality of the speaker system, and the touchscreen and hardware audio controls will increase the user's ease of use. Thus, this project is recommended to deliver an accessible, high-quality sound system to the average consumer.

Problem Statement

Motivation:

The motivation for this project is to fulfill a client's request to revitalize old speakers. In addition, the goal of this project is to create a low-cost system with the rich sound quality of a high-end speaker system by utilizing a 2.1 channel amplifier with two stereo speakers and a subwoofer.

Identification of need:

Currently, there are many online tutorials that describe how to construct an internet radio system with similar features to the proposed device, such as an LCD touchscreen and USB connection. The proposed device differs from other internet radios because it will incorporate a passive subwoofer, which will enrich the sound quality of the music. The device will also use a digital-to-analog converter (DAC). The implementation of the DAC within a Pi HAT for this project will provide valuable experience in sound system development and embedded system interfacing for future students and engineers alike. Moreover, this project will satisfy the client's needs by reusing old speakers for a device with a modern user interface.

The estimated cost of all components is at most $150 because the device does not include the speakers or subwoofer. The development of the device is dependent on the amount of time that it will take to acquire all components. Once all of the components have been obtained and fully developed, the device should be assembled and fully operational within a short time frame.

Prototyping will be used to evaluate whether the device has met the objectives of the project. The components of this project will be chosen based on their performance. A working prototype will be delivered halfway through the duration of the project. The prototype will indicate that the device is operational, and it can be modified to meet the client's requirements during a final evaluation.

Market/application review:

Sound quality and price are the main factors to consider when analyzing speaker systems that are currently on the market. The cost of a high quality home speaker system can range from $300 to $2700 [1]. Often, such speaker systems do not include a subwoofer: the speaker system either has a left and right speaker or a speaker bar with 2 channels. Furthermore, these speaker systems do not contain an interface to allow the user to interact with the device directly. A product that more closely resembles the proposed device is the Pimoroni Pirate Radio Pi Zero W Project Kit [2]. This kit is the most advanced system on the market and allows local files such as MP3 and FLAC to be played through a built-in 1 W speaker [2]. However, this kit suffers from poor audio quality due to the limited wattage of the speaker system. The proposed device will adopt the beneficial features of the products that are currently on the market. These features will be implemented with the client's needs to create a sound system that is simple to use yet challenging to create.

Approach

Problem Analysis:

The technology in recent sound systems has become more advanced, which has led to outdated speakers that lack certain capabilities and interfaces. The objective of this project is to restore these outdated speakers through an updated interface. The sound system will be capable of connecting to older speakers via speaker wire. The device will accept music from various inputs such as the internet, a NAS, and a USB drive. The device will also have an LCD touchscreen and hardware controls. These features will allow older speakers to compete with current sound systems. An exemplary embodiment of the device will also implement Bluetooth connection, audio jack connection, and a hardware equalizer. Figure 1 demonstrates a simple user interaction with the system.

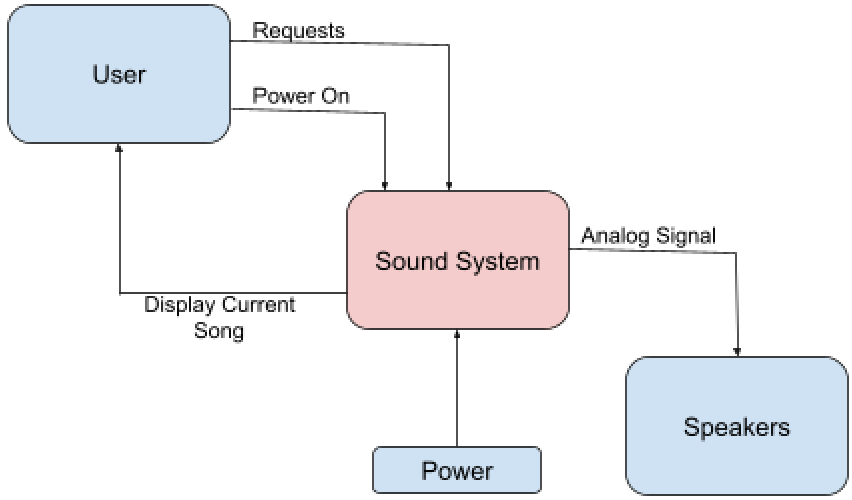

Figure 1: External System Diagram

Approach:

The design process was planned in order to implement all of the client's needs. The foundation of the system will be the Pi. A custom Pi-hat with a DAC will be connected to a 2.1 channel amplifier. The 2.1 channel amplifier will amplify signals for a left stereo speaker, right stereo speaker, and a subwoofer with a crossover. As described previously, the system will be able to play music from various inputs. A touchscreen interface will allow the user to control the device, and the software that will be displayed on the touchscreen will provide the user interface and media playback. The device will have a modern, polished appearance with features to enhance the user experience. The system will be placed in an enclosure with the LCD touchscreen and hardware knobs on the main face, and the enclosure will also include a USB port.

Alternative approaches:

There are a variety of free and open-source software that are capable of driving the device. While these open-source software contain many built-in features, they can also be modified to serve the functional needs of the project. The primary choice of software is Mopidy, which is a highly extensible Python-based music server [3]. Mopidy allows users to access music from network access, local disk, and streaming services; users can also edit playlists from another computer or smart device [3]. If Mopidy does not suffice, then Kodi is the secondary option for software. Kodi is a free and open-source media player software that can playback audio files and stream music from the internet directly [4]. Another option under consideration is piCorePlayer, which can play music from local files as well as internet music streaming services via a headless connection [5].

There is a wide range of affordable DACs that are available on the market. The chosen DAC is the PCM5102A from Texas Instruments. This DAC is a dual-channel DAC that supports audio sources of up to 32-bit, 384kHz quality and connects via the I^2S interface on the Pi. [6]. The second option that is being considered is the PCM5122, which is also from Texas Instruments. Like the PCM5102A, the PCM5122 is a dual-channel DAC which supports up to 32-bit, 384kHz audio sources [7]. The PCM5122 differs from the PCM5102A due to its built-in equalization and filtering; the utilization of these features would increase the development time of the project. Therefore, the PCM5122 is the secondary choice. Table 1 lists the potential choices for the DAC.

Table 1: DAC Choices [6] [7]

Name	*PCM5102A*	*PCM5122*
Max bit depth	32	32
Max sample rate	384kHz	384kHz
Cost	$5.53	$5.24

Amplifier chip functionality is a major element in the design process for the system. Various amplifiers can be used depending on the number of channels supported. For instance, a three channel amplifier integrated-circuit (IC) can be used to implement one channel for each speaker in the system. It is important to note that the use of a separate amplifier for the subwoofer will require an active low pass filter. A list of potential amplifiers can be seen below in Table 2.

Table 2: Potential Amplifier Components [8] [9]

Name	*TPA3220DDWR*	*TPA3156D2DAD*	*Y1AA*
Supply Voltage	7 V and 30 V	4.5-26V	18-24V
Max Output Power	110W(1) 89W(2)	140W(1) 70W(2)	80W
Channels	2	2	3
Cost	$3.30	$4.13	$16.94

The main factors for choosing an LCD touchscreen are its compatibility with the Pi and its functionality based on its price. Although the RPi touchscreen in Table 3 is the most expensive option, this touchscreen is made specifically for the Pi. Therefore, the use of the RPi touchscreen will limit any potential issues with compatibility. The UCTRONICS 5-inch display and UCTRONICS 3.5-inch display are under consideration due to their affordable prices, but both displays are not guaranteed to be compatible with the Pi. Furthermore, the 3.5-inch display is a TFT LCD that has a significantly smaller size and resolution compared to the other two options.

Table 3: Touch screen display options [10] [11] [12]

Name	*RPi 7-in. touch screen display*	*UCTRONICS 5-in. touch screen display*	*UCTRONICS 3.5-in. touch screen display*
Display size	7 inches	5 inches	3.5 inches
Resolution	800 x 480p	800 x 480p	480 x 320p
Compatible w/ Pi?	Yes	Yes	Yes
Connection type	Ribbon cable to DSI port	USB to micro USB	HDMI
Cost	$83.99	$52.99	$36.99
Notes	Offered at discounted price	N/A	Cheapest option, uses TFT

Background Knowledge/Phenomenology:

Communication Protocols:

The Pi needs to connect to a specific server to receive music from the internet. For example, a user's decision to connect to YouTube would require the Pi to establish a connection to a YouTube server. This connection between the Pi and the server is described using the TCP/IP protocol. HTTPS is the application layer protocol of TCP/IP that allows the user to access music data from a website [13]. The Pi connects to a HTTPS server through port 443 [14]. Once a connection has been established, the Pi acts as a client and requests music files from the server, and the server responds by providing the music files that were requested.

The Pi also uses an application layer protocol on top of TCP/IP to communicate with the NAS; this protocol is referred to as SMB/CIFS [15]. Like HTTPS, SMB/CIFS is a request-response protocol. The Pi connects to the SMB/CIFS server through port 445 to establish a secure connection [15]. Since the NAS is located in the user's home, the NAS and Pi are connected on the same LAN and can communicate via said LAN.

Amplifiers:

The amplifiers for the system will need to generate a signal to supply two 50 W speakers and one 100 W subwoofer. The amplification of the signal is due to the gain, which is the ratio of the output voltage to the input voltage of the amplifier, as seen in equation 1. This factor is used to amplify the signal without changing the shape of the waveform or the quality of the sound [16]. The amplifier can reach a saturation region, where it will no longer be able to amplify the signal [17]. It is important to refrain from supplying the amplifier with signals that cause the amplifier to enter the saturation region, as long exposure in the saturation region could damage the amplifier.

$$A_v = \frac{v_o}{v_i} \tag{1}$$

The subwoofer is intended to output bass audio at low frequencies. The standard crossover frequency, which is analogous to the cutoff frequency, ranges between 70 Hz and 80 Hz for a subwoofer. [18]. The crossover frequency will be 80 Hz for this system. Therefore, a low pass filter is needed to produce the bass audio of the subwoofer. The type of low pass filter that has been chosen is the Butterworth RC filter because it is more commonly used for filtering audio and has a flat passband [19].

For the Butterworth RC filter, the equation for the cutoff frequency is defined in equation (2).

$$f_C = \frac{1}{2\pi RC} \tag{2}$$

The capacitor values that are being considered for this system are 220 μF and 200 nF because these capacitors are readily available. Assuming that the value of *C* will equal to either 220μF or 200 nF, the values of *R* for each corresponding *C* can be calculated using equation 3.

$$R = \frac{1}{2\pi f_C C} \tag{3}$$

$$\text{For } C = 220\ \mu\text{F}, R = \frac{1}{2\pi\ (80)\ (220 \times 10^{-6})} = 9.043\ \Omega.$$

$$\text{For } C = 200\text{nF}, R = \frac{1}{2\pi\ (80)\ (200 \times 10^{-9})} = 9.947\text{k}\Omega.$$

In practice, these resistor values would be implemented as *R* = 9.4 Ω and *R* = 10 kΩ. The actual frequencies for the implemented resistor values are 76.961 Hz and 79.577 Hz, respectively. This information is summarized in Table 4 below.

Table 4: Parameters of Butterworth RC Filter

Value of C	*Calculated Value of R*	*Actual Value of R*	*Actual Frequency*	*Percent Error*
220 μF	9.043 Ω	9.4 Ω	76.961 Hz	3.799 %

200 nF	9.947 kΩ	10 kΩ	79.577 Hz	0.529 %

The stereo speakers will use a high pass filter to output frequencies above 80 Hz to prevent speaker damage. A high pass Butterworth Filter will be used with the same parameters that are calculated above.

Touch screen:

Most modern touch screens are capacitive. When supplied with a voltage, a capacitor takes time to fully charge and subsequently takes time to discharge when the voltage supply is removed. The charge-discharge times change when the capacitance of the circuit itself is changing, which is the key principle in how touchscreens work. A human finger acts as a dielectric when it makes contact with a capacitive plate. This increases the capacitance in the circuit, and the ensuing change in charge-discharge time registers the presence of human touch [20].

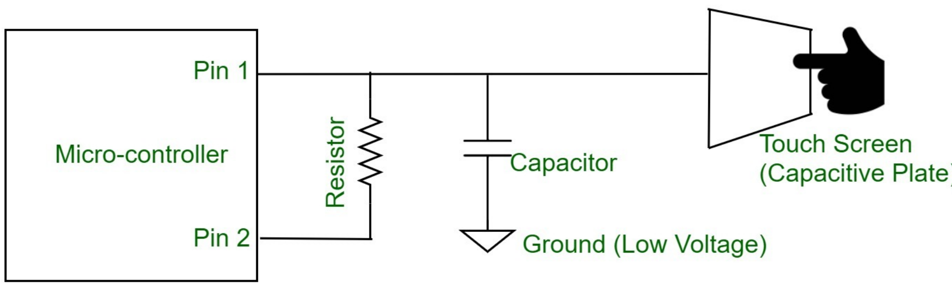

Figure 2: Capacitive touch sensing circuit

Digital to Analog Converter:

A DAC is a vital part of the sound system for the device because the music is stored as digital data. The purpose of a DAC is to take digital data and produce an analog voltage. The analog voltage is fed into amplifiers, which drives the speakers. A simple DAC can be made from resistors and an op amp, and there are two basic types: the weighted resistor, and the R-2R ladder circuit [21].

The design of a simple 4-bit weighted resistor DAC is depicted in Figure 3.

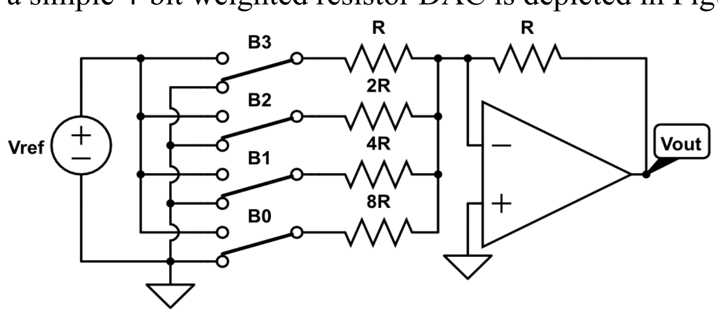

Figure 3: 4-bit weighted resistor DAC

For this 4-bit DAC, the voltage produced at the output is shown by equation 4.

$$V_{out} = \; -V_{ref} \times (\frac{B_0}{2^3} + \frac{B_1}{2^2} + \frac{B_2}{2^1} + \frac{B_3}{2^0}) \qquad (4)$$

For an N-bit DAC, the voltage produced at the output is shown by equation 5.

$$V_{out} = \; -V_{ref} \times (\frac{B_0}{2^{N-1}} + \frac{B_1}{2^{N-2}} + ... + \frac{B_{N-1}}{2^0}) \qquad (5)$$

The alternative design is a R-2R DAC; the design for a simple 4-bit R-2R DAC is illustrated in Figure 4.

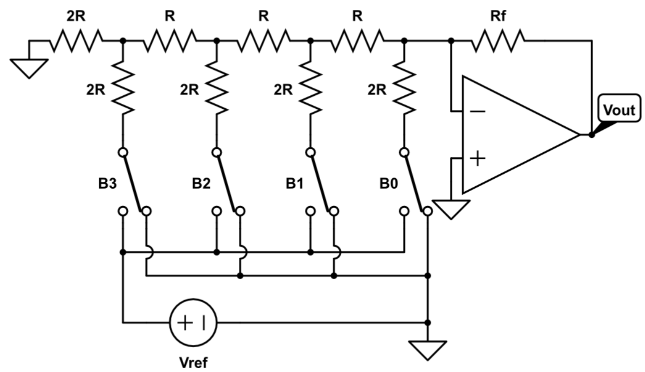

Figure 4: 4-bit 4-bit R-2R DAC

For this 4-bit DAC, the voltage produced at the output is shown by equation 6.

$$V_{out} =- V_{ref} \times (\frac{B_0}{2^4} + \frac{B_1}{2^3} + \frac{B_2}{2^2} + \frac{B_3}{2^1}) \qquad (6)$$

For an N-bit DAC, the voltage produced at the output is represented by equation 7.

$$V_{out} =- V_{ref} \times (\frac{B_0}{2^N} + \frac{B_1}{2^{N-1}} + ... + \frac{B_{N-1}}{2^1}) \qquad (7)$$

Project Requirements Specification:

The device is designed for home use. As such, the weight and price constraints have been determined based on the needs of the average consumer. The weight of the device should be 20 pounds or less. The total cost of the device, including the speakers and subwoofer, should range between 200 and 300 dollars. A Raspberry Pi 3B+ or Raspberry Pi 4 shall be implemented as the main processor of the device. All components will be placed in a metal casing that shall passively cool the device. The device will also contain an LCD that will allow the user to interact with the music player interface.

A custom Pi HAT will be used to supply power for the Pi. The custom Pi HAT shall incorporate power converters as well as the DAC and 2.1 channel amplifier on the same PCB. The DAC shall accept up to 24-bit audio sources at 192kHz. The 2.1 channel amplifier shall maintain a maximum power output of 50W per stereo channel and 100W for the subwoofer. The subwoofer should have an adjustable crossover frequency; the standard crossover frequency will be 80 Hz. The Pi HAT shall support speakers of up to 4 ohms impedance, and the speakers shall connect to the speakers via speaker wire. Additionally, the device shall accept audio inputs from a USB port as well as an internet or NAS connection.

A combination of hardware controls and software controls will be implemented to adjust the volume of the audio. The user should be able to change the volume by turning the knob of a rotary encoder. The subwoofer will also have its own volume control. In addition to the audio controls described previously, a power button will be implemented to allow the user to turn the device on or off. Once the device is powered on, the user shall only have the ability to interact with the interface of the music player, allowing the user to readily access his or her music. The media player interface will be software that is either open-source or programmed internally.

Within the media player interface, a screen will appear to allow the user to select his or her choice of audio input. After the user makes a choice, the user’s music library will be displayed and the user will choose a song to play. Song information and music controls will be visible while a song is outputted through the speakers. Song information should consist of the song title, artist name, song duration, and album cover art; music controls should consist of icons for play, pause, next-in-playlist, and previous-in-playlist. The user should also be able to view a split-screen if desired, with one side of the screen showing song information and music controls while the other side displays the user’s music library.

System Design

Functional Decomposition:

The Level-0 design of the functional decomposition can be seen in Figure 5. This figure describes the top level functions and user interactions of the system.

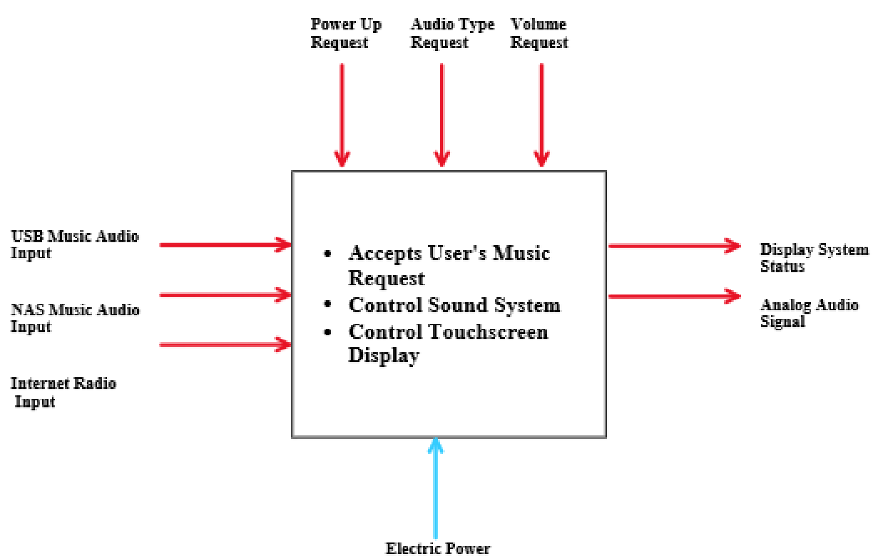

Figure 5: Level-0 of Functional Decomposition

Figure 6 gives a clearer depiction of how each top level will function within the system.

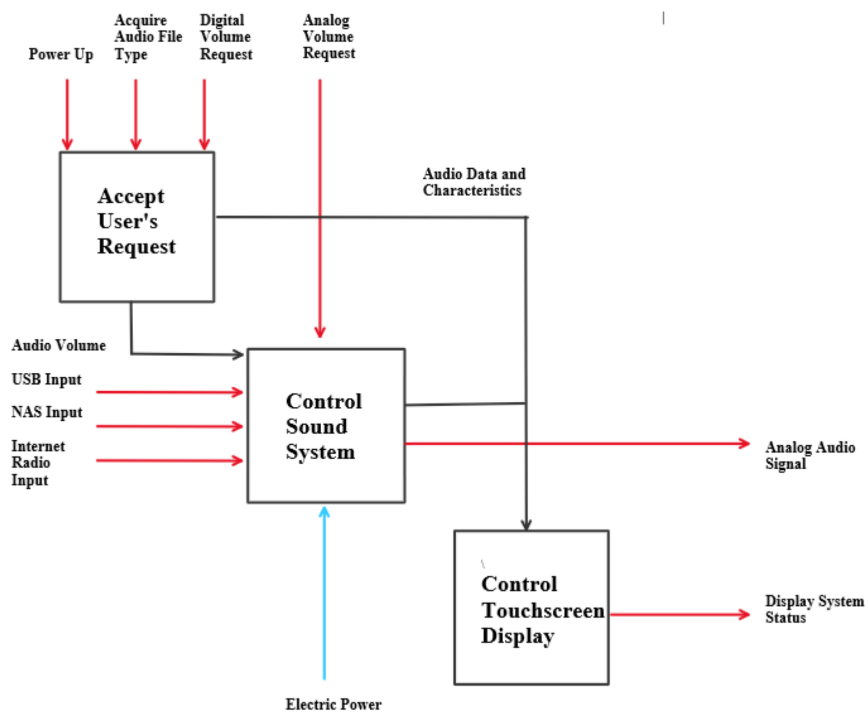

Figure 6: Level-1 of Functional Decomposition

The Level-1 functions are further decomposed in Figures 7, 8 and 9. Each figure represents the functionality of each top level operation.

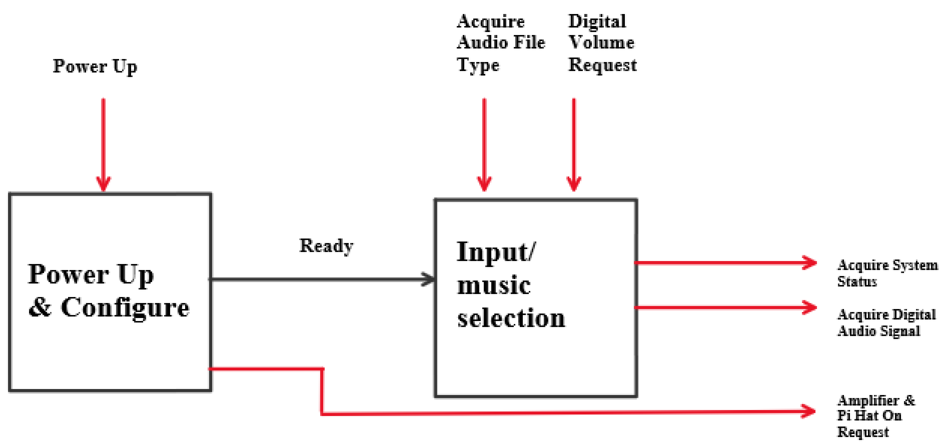

Figure 7: Level-2 Function Accept User Requests

Figure 7 establishes the interactions between the system and the user's requests.

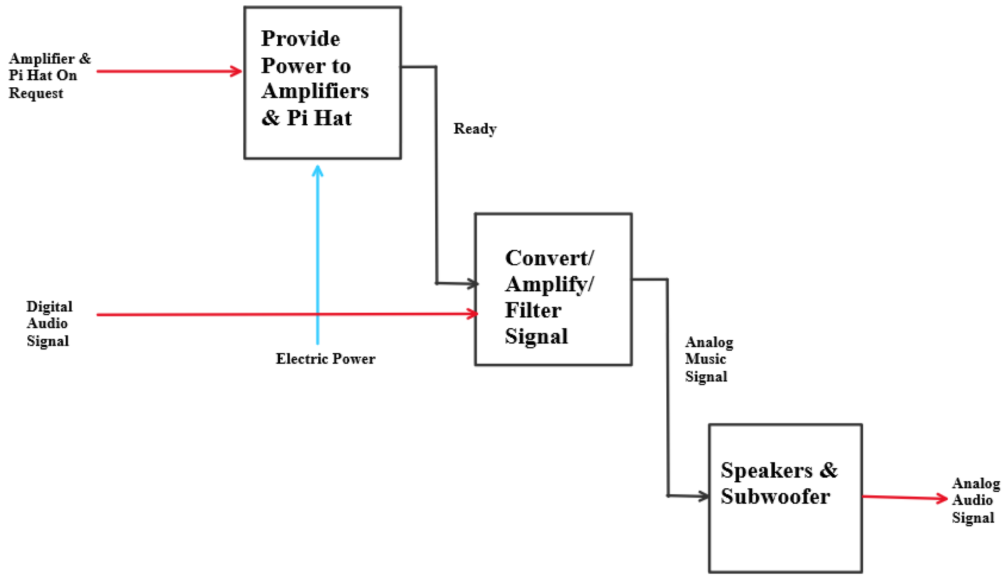

Figure 8: Level-2 Function Control Sound System

Figure 8 highlights how the system handles digital audio input and outputs the analog music signal.

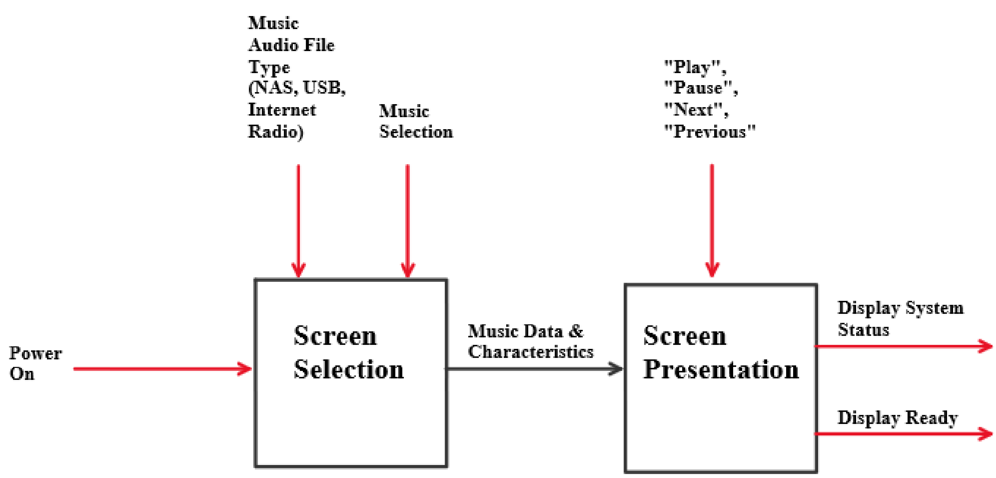

Figure 9: Level-2 Function Control Touchscreen Display

Figure 9 details the usability the touchscreen will provide the client when operating the device.

System Architecture:

The generic physical architecture is shown in Figure 10 and highlights the resources used in the system. The overall system architecture is depicted in Figure 11, which consists of the top level components needed to complete the design.

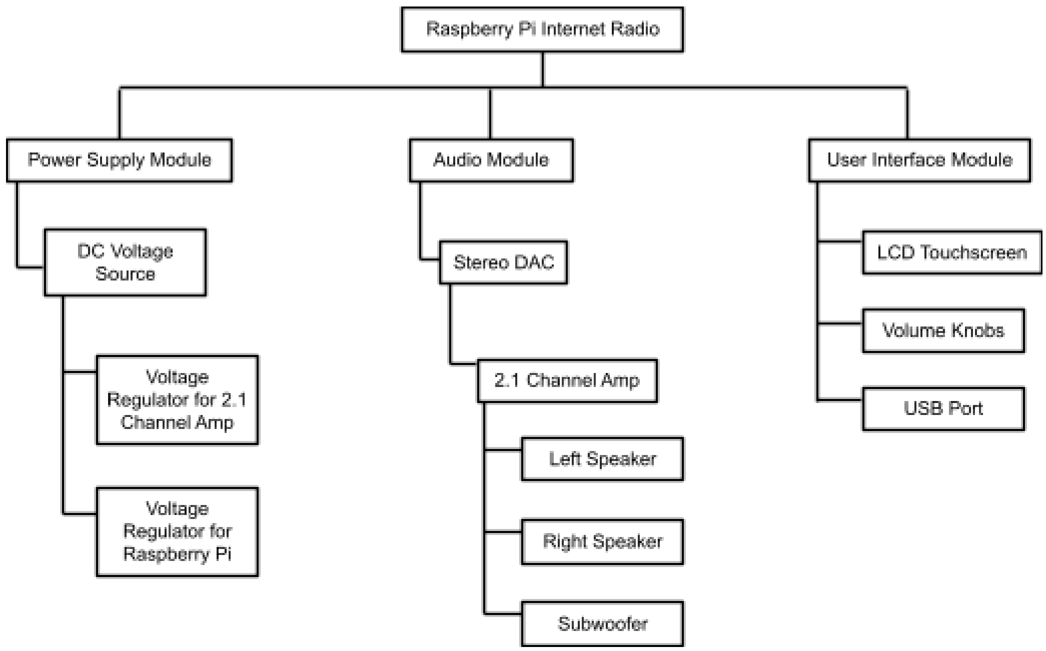

Figure 10: Generic Physical Architecture

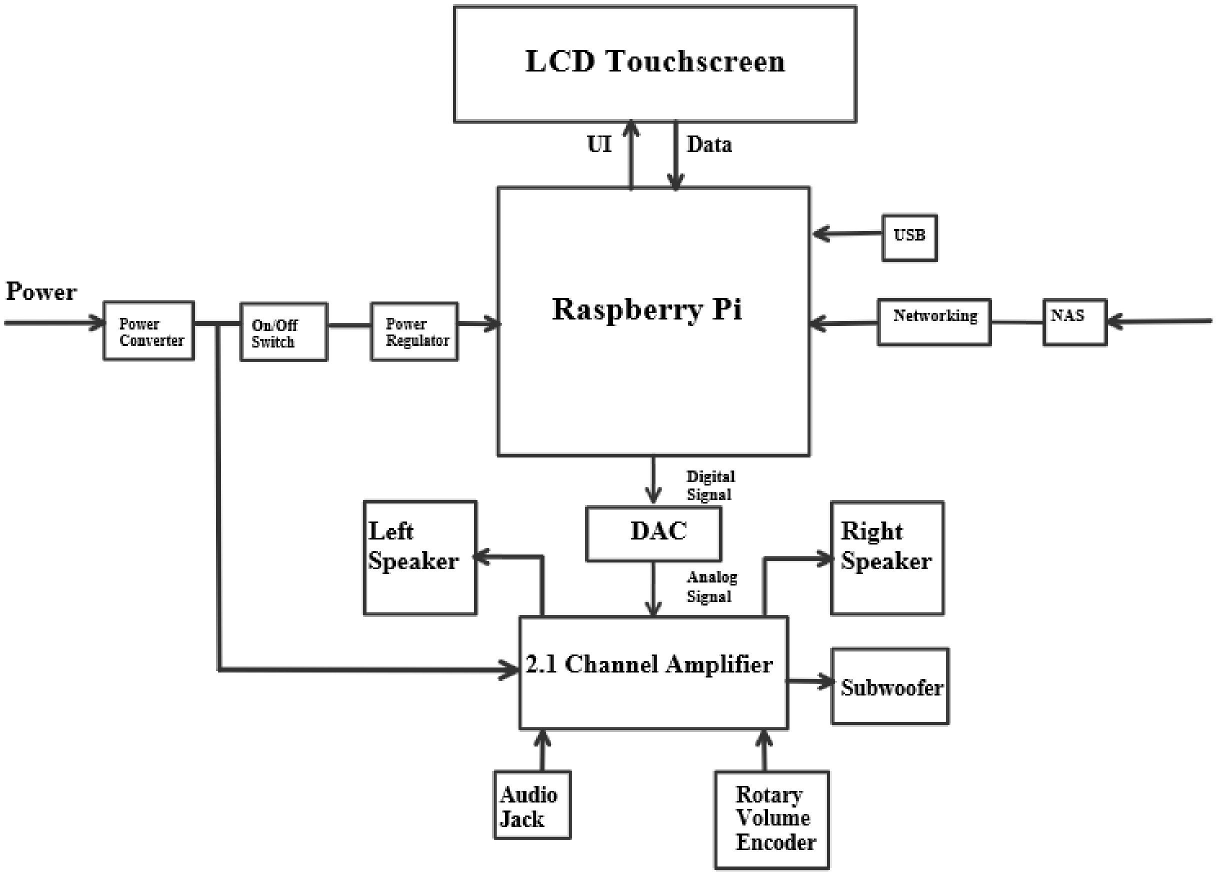

Figure 11: System Architecture

Preliminary Experimental Plan

Several experiments have been planned to verify the system performance. These experiments will test the power supply, crossover, operating temperature, and power output.

The power supply must be tested to minimize the noise from the voltage regulator. To test the noise level, the output of the regulator will be connected to a dummy load of resistors, and the voltage drop across those resistors will be monitored by an oscilloscope. The output of the regulator should be a DC signal, but may have some AC characteristics in the form of a ripple. Ripple produces a buzzing noise in the speakers, and different capacitors can be applied across the voltage output to reduce ripple to an acceptable level.

The crossover must be tested to verify that its cutoff frequency matches the project specifications. To test the crossover, the frequency response will be plotted for the range of human hearing. The crossover will be adjusted until the frequency response has a desirable rolloff at the specified cutoff frequency.

It must be ensured that the system can operate at a safe temperature at full load. To test this requirement, the system will operate at maximum volume. Then, the temperatures of the hottest components, such as the Pi and the amplifiers, will be monitored. If the system does not remain cool, then heat sinks will be implemented to reduce the temperature of the system.

The maximum power output of the system must meet the project requirements without clipping. To test this, the maximum amplifier input will connect to a function generator and the amplifier output will connect to a dummy load of resistors. The voltage drop across the resistors will be measured with an oscilloscope. If the voltage drop corresponds to the desired power output and the waveform does not clip, then the amplifier power output meets specifications.

Preliminary Project Plan

Future Tasks:

Development, implementation of the design specifications, and testing will occur after prototyping the design. A list of the major tasks and their subcategories for the coming months are listed below. This list is preliminary and is subject to change.

1. **Hardware Development**
   a. PCB Design
   b. PCB Assembly
   c. Analog Volume Control Setup and Testing
   d. Touchscreen Setup and Testing
2. **Software Development**
   a. Open-source Interfacing
   b. Request Module
   c. Display Module
   d. Music Module
3. **System Integration**
   a. Request Functionality
   b. Display Functionality
   c. Music Functionality
   d. Digital Volume Control
4. **Testing**
   a. Experiment #1
   b. Experiment #2
   c. Experiment #3
   d. Experiment #4
5. **Reporting**
   a. Progress Report
   b. In-Progress Presentation
   c. Final Report and Presentation
6. **Milestones/Demos**
   a. Demo #1 Hardware
   b. Demo #2 Software
   c. Demo #3 Integrated System

Allocation of Responsibilities:

The allocation of responsibilities is subject to change depending on member interests and capabilities for a given task assignment. The team of six will be split into two groups. Ben and

Danny will be tasked with Software Development, and Zara will help with implementing the SMB/CIFS protocol for the NAS. The second group will focus on Hardware Development and will consist of James, Zayd, Zuha, and Zara. These teams will accomplish both the hardware and software development simultaneously and then work together on system integration.

Project Problems

Knowledge and Skills to be Learned

The team consists of six team members: five electrical engineers and one double major in computer engineering and computer science. Three electrical engineers are concentrated in embedded systems, one electrical engineer is concentrated in communications/signals processing and one electrical engineer is concentrated in computer engineering. Most of the team currently possesses knowledge on basic filter design, which will play a key role in creating the amplifiers for this system. Furthermore, the team has completed various signal processing courses and are familiar with the Raspberry Pi.

The DAC and amplifiers will be integrated onto one PCB to maximize the space within the metal enclosure. The design and development of the PCB is expected to take a significant amount of time. While the team has limited experience with PCB design, three team members are currently taking a PCB design course and look to gain the knowledge and experience required to accomplish this task.

Project Risk Analysis:

There are potential risks that arise with building this speaker system. It is important to consider the current chip shortage due to the COVID-19 pandemic and how it will impact the price and shipment times of DAC and amplifier components.

While the risk of the system overheating is low, it is still a possibility and necessary precautions should be taken. The heat output from the components, particularly the Pi and amplifier, can cause damage to the circuit or potentially escalate to be a fire hazard: the Pi is known to overheat when nearing its limits [22]. There is no salvaging of components if they are damaged due to overheating, so it is best if this problem is dealt with early on or avoided entirely.

References

[1] N. Pino and O. Tambini, “The best wireless speaker 2021: Find the best connected speakers for your home,” *TechRadar India*, 16-Aug-2021. [Online]. Available: https://www.techradar.com/in/news/best-speaker.

[2] "Pimoroni pirate radio - pi zero W with headers project kit," *adafruit industries blog RSS*. [Online]. Available: https://www.adafruit.com/product/3477.

[3] P. J. Evans, "Make a raspberry pi audio player with Mopidy Music," *The MagPi magazine*. [Online]. Available: https://magpi.raspberrypi.org/articles/make-a-raspberry-pi-audio-player-with-mopidy-music.

[4] E. Odunlade, "Top 10 media server software for music streaming on Raspberry Pi," *Circuit Digest*, 02-Sep-2019. [Online]. Available: https://circuitdigest.com/article/top-media-server-software-for-music-streaming-on-raspberry-pi.

[5] "Picoreplayer," *piCorePlayer*. [Online]. Available: https://www.picoreplayer.org/.

[6] Texas Instruments, "PCM510xA 2.1 VRMS, 112/106/100 dB Audio Stereo DAC with PLL and 32-bit, 384 kHz PCM Interface", PCM510 datasheet, May 2012 [Revised May 2015]

[7] Texas Instruments, "PCM512x 2-VRMS DirectPath™ , 112-dB and 106-dB Audio Stereo DACs With 32-Bit, 384-kHz PCM Interface", PCM512 datasheet, Aug. 2012 [Revised Oct. 2018]

[8] Texas Instruments, "TPA3156D2 2 x 70-W, Analog Input, Stereo, Class-D Audio Amplifier With Low Idle Power Dissipation", TPA3156D2 datasheet, Dec. 2017

[9] Texas Instruments, "TPA3220 60-W Stereo, 110-W Peak HD Analog-Input, Pad-Down Class-D Amplifier", TPA3220 datasheet, Jan. 2018 [Revised Aug. 2018]

[10] "Raspberry pi 7' Touch screen display - amazon.com." *Amazon* [Online]. Available: https://www.amazon.com/Raspberry-Pi-7-Touchscreen-Display/dp/B0153R2A9I.

[11] "Amazon.com: UCTRONICS 5 inch touch screen for Raspberry Pi ..." *Amazon*. [Online]. Available:https://www.amazon.com/UCTRONICS-Raspberry-Portable-Capacitive-Touchscreen/dp/B07VV7RL7Y.

[12] "Amazon.com: UCTRONICS 3.5 inch touch screen for Raspberry Pi ..." *Amazon* [Online]. Available:https://www.amazon.com/UCTRONICS-Raspberry-Portable-Capacitive-Touchscreen/dp/B07VV7RL7Y.

[13] "An overview of HTTP - http: MDN," *HTTP | MDN*. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview.

[14] "Port - MDN Web Docs Glossary: Definitions of Web-related terms: MDN," *MDN Web Docs Glossary: Definitions of Web-related terms | MDN*. [Online]. Available: https://developer.mozilla.org/en-US/docs/Glossary/Port.

[15] "1.1 understanding CIFS," *Understanding CIFS - OES 2018 SP2: OES CIFS for Linux Administration Guide*. [Online]. Available: https://www.novell.com/documentation/open-enterprise-server-2018/file_cifs_lx/data/bld9p53.html.

[16] "Gain and feedback." [Online]. Available: https://www.phys.uconn.edu/~gibson/Notes/Section6_1/Sec6_1.htm.

[17] C. R. Nave, "Amplifiers," *Audio Amplifiers*. [Online]. Available: http://hyperphysics.phy-astr.gsu.edu/hbase/Audio/amp.html.

[18] D. Hawley, "How to set crossover frequency for a car audio system," *J.D. Power*. [Online]. Available: https://www.jdpower.com/cars/shopping-guides/how-to-set-crossover-frequency-for-a-car-audio-system.

[19] D. Raja, “Butterworth Filter: First Order and Second Order Low Pass Butterworth Filter,” *Circuit Digest*, 25-Jun-2020. [Online]. Available: https://circuitdigest.com/tutorial/butterworth-filters-first-order-and-second-order-low-pass-butterworth-filters.

[20] GeeksforGeeks, 2021, *Capacitive Touch Technology - GeeksforGeeks*. 6 October 2021. [online] Available: https://www.geeksforgeeks.org/capacitive-touch-technology.

[21] “Digital to analog converter (DAC) - types, working & applications,” *ELECTRICAL TECHNOLOGY*, 29-Apr-2020. [Online]. Available: https://www.electricaltechnology.org/2020/04/digital-to-analog-converter-dac.html.

[22] N. Heath, “Raspberry pi overheating? here's how to stop it says Microsoft,” *TechRepublic*, 18-Oct-2017. [Online]. Available: https://www.techrepublic.com/article/raspberry-pi-overheating-heres-how-to-stop-it-says-microsoft/.

Appendix B: Design Document

Design Review

Kapstone Raspberry Pi Radio
(KRPiR)

Team #6

Benjamin Ong

Danny Truong

Jamie Norvell

Zara Shoukat

Zayd Fazelyar

Zuha Abdul Wakeel

Faculty Sponsor: Dr. Kaps

Table of Contents

Executive Summary

Currently, there are many speaker systems on the market that allow users to play music from internet radio stations or streaming services and output audio through a 2.1 channel amplifier. However, these systems are often expensive or do not have a user interface that enables the user to interact with the device directly: the user can only control the speakers via Bluetooth or remote. Furthermore, many affordable stereo sound systems do not include a

subwoofer. The aim of this project is to develop an affordable device that has the rich sound quality of an expensive speaker system through the use of a 2.1 channel amplifier with two stereo speakers as well as a subwoofer. In addition, the device will have an LCD touchscreen and hardware volume knobs to allow the user to access audio controls through the device directly. The use of a subwoofer will enhance the sound quality of the speaker system, and the touchscreen and hardware audio controls will increase the ease of use. Thus, this project is recommended to deliver an accessible, high-quality sound system to the average consumer.

Problem Statement:

Motivation:

The motivation for this project is to fulfill a client's request to revitalize old speakers. Furthermore, by combining a 2.1 channel amplifier with two stereo speakers and a subwoofer, the system will be affordable while still providing the rich sound quality of a high-end speaker system.

Identification of need:

There are many online instructions that describe how to develop an internet radio system with characteristics comparable to the proposed device. The proposed system will vary from current internet radios because it will have a passive subwoofer, which will improve the sound quality. The implementation of a digital-to-analog converter DAC within a Pi HAT for this project will provide valuable experience in sound system development and embedded system interfacing for future students and engineers alike. Moreover, this project will satisfy the client's needs by reusing old speakers for a device with a modern user interface.

The system will be evaluated via prototyping to determine if the overall design has been met with the consumer. The components of this project will be chosen based on their performance. A working prototype will be delivered halfway through the duration of the project. The prototype will indicate that the device is operational, and it can be modified to meet the client's requirements during final evaluations.

Market/application review:

Sound quality and price are the main factors to consider when analyzing speaker systems that are currently on the market. The cost of a high quality home speaker system can range from $300 to $2700 [1]. Often, such speaker systems do not include a subwoofer: the speaker system either has a left and right speaker or a speaker bar with 2 channels. Furthermore, these speaker systems do not contain an interface to allow the user to interact with the device directly. A product that more closely resembles the proposed device is the Pimoroni Pirate Radio Pi Zero W Project Kit [2]. This kit is the most advanced system on the market and allows local files such as MP3 and FLAC to be played through a built-in 1 W speaker [2]. However, this kit suffers from poor audio quality due to the limited wattage of the speaker system. The proposed device will adopt the beneficial features of the products that are currently on the market. These features will be implemented with the client's needs to create a sound system that is simple to use yet challenging to create.

Requirements Specification:

The device is designed for home use. As such, the weight and price constraints have been determined based on the needs of the average consumer. The weight of the device should be 20 pounds or less. The total cost of the device, excluding the speakers and subwoofer, should range between 200 and 300 dollars. A Raspberry Pi 3B+ is implemented as the main controller of the device. All components will be placed in a metal casing that shall passively cool the device. The device also contains an LCD that allows the user to interact with the music player interface.

A custom Pi HAT will be used to supply power for the Pi. The custom Pi HAT shall incorporate power converters as well as the DAC and 2.1 channel amplifier on the same PCB. The DAC shall accept up to 24-bit audio sources at 192kHz. The 2.1 channel amplifier shall maintain a maximum power output of 50W per stereo channel and 100W for the subwoofer. The crossover frequency will be 200 Hz. The Pi HAT shall support speakers of 8 ohms impedance, and the speakers shall connect to the speakers via speaker wire. Additionally, the device shall accept audio inputs from a USB port as well as an internet or NAS connection.

A combination of hardware controls and software controls will be implemented to adjust the volume of the audio. The user should be able to change the volume by turning the knob of a rotary encoder. The subwoofer will also have its own volume control. In addition to the audio controls described previously, a power button will be implemented to allow the user to turn the device on or off. Once the device is powered on, the user shall only have the ability to interact with the interface of the music player, allowing the user to readily access his or her music. The media player interface uses open-source software.

Within the media player interface, a screen appears to allow the user to select his or her choice of audio input. After the user makes a choice, the user's music library is displayed and the user will choose a song to play. Song information and music controls are visible while a song is outputted through the speakers. Song information should consist of the song title, artist name, song duration, and album cover art; music controls should consist of icons for play, pause, next-in-playlist, and previous-in-playlist.

System Decomposition:

Functional Decomposition:

The Level-0 design of the functional decomposition can be seen in Figure 1. This figure describes the top level functions and user interactions of the system.

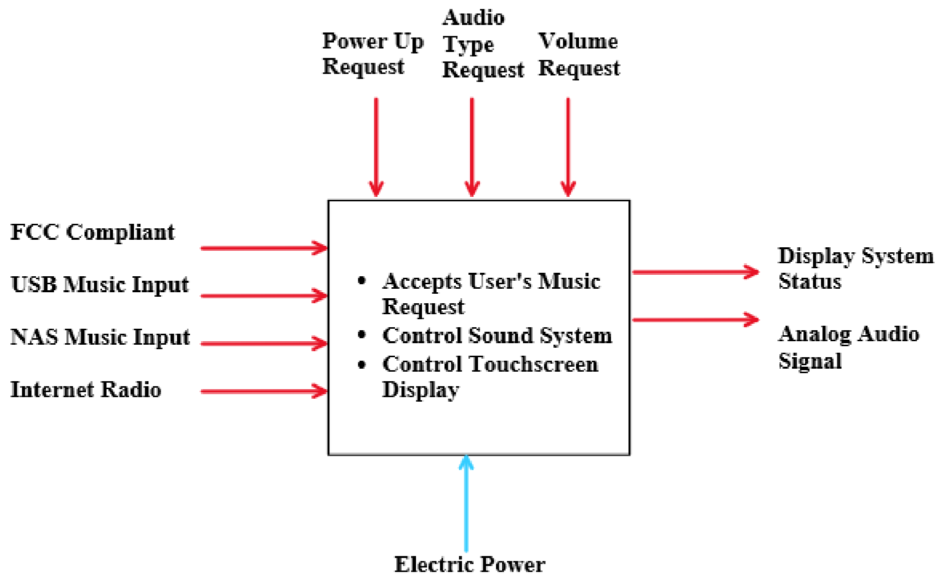

Figure 1: Level-0 of Functional Decomposition

Figure 2 gives a clearer depiction of how each top level will function within the system.

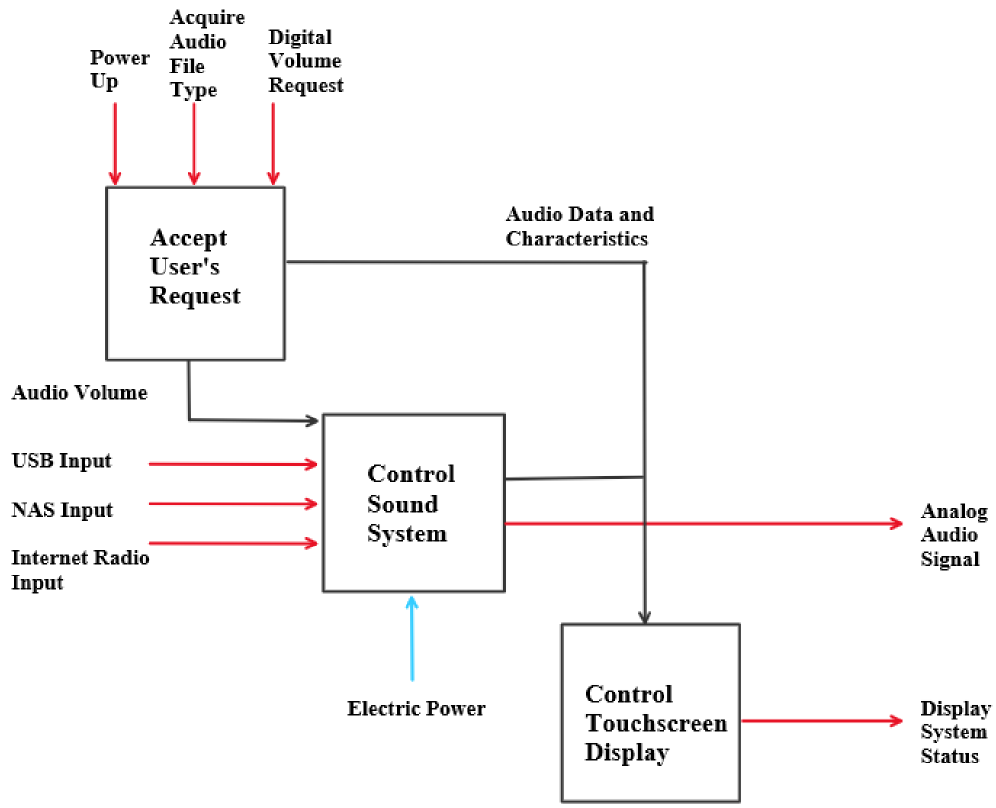

Figure 2: Level-1 of Functional Decomposition

The Level-1 functions are further decomposed in Figure 3, Figure 4, and Figure 5. Each figure represents the functionality of each top level operation.

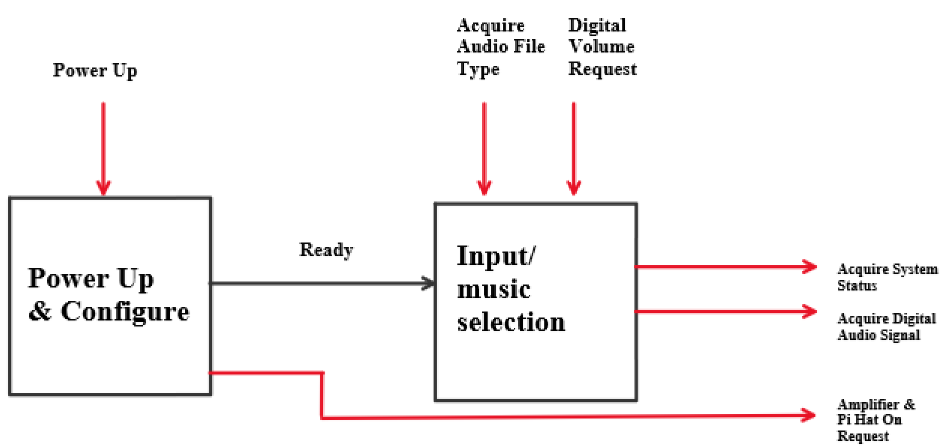

Figure 3: Level-2 Function Accept User Requests

Figure 3 establishes the interactions between the system and the user's requests.

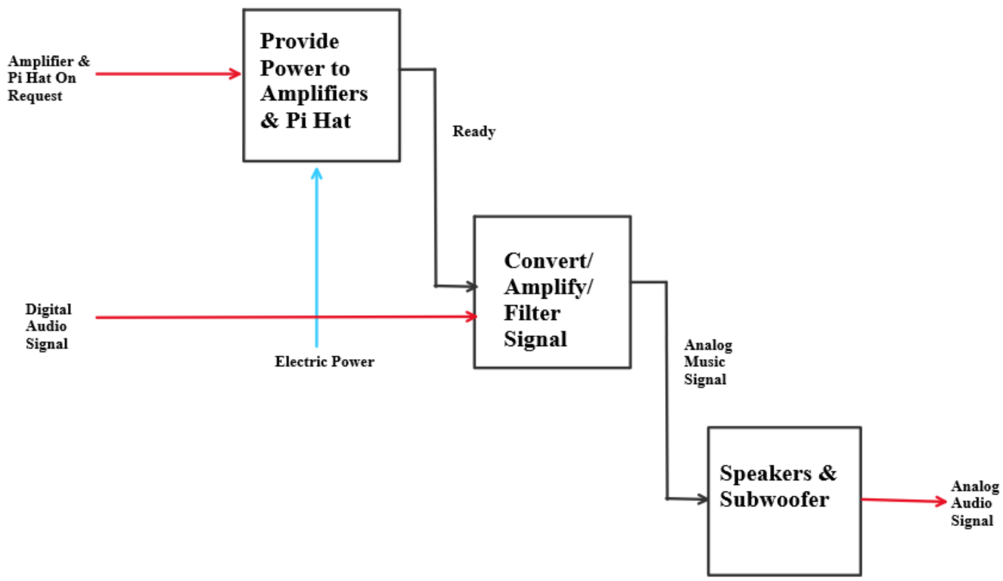

Figure 4: Level-2 Function Control Sound System

Figure 4 highlights how the system handles digital audio input and outputs an analog music signal.

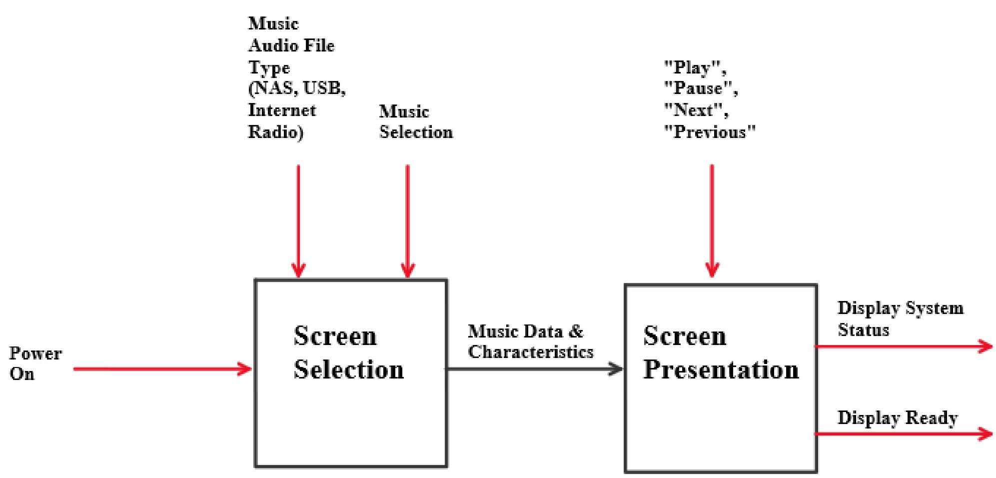

Figure 5: Level-2 Function Control Touchscreen Display

Figure 5 details the usability the touchscreen will provide the client when operating the device.

Generic Physical Architecture and System Architecture:

The generic physical architecture is shown in Figure 6 and highlights the resources used in the system. The overall system architecture is depicted in Figure 7, which consists of the top level components needed to complete the design.

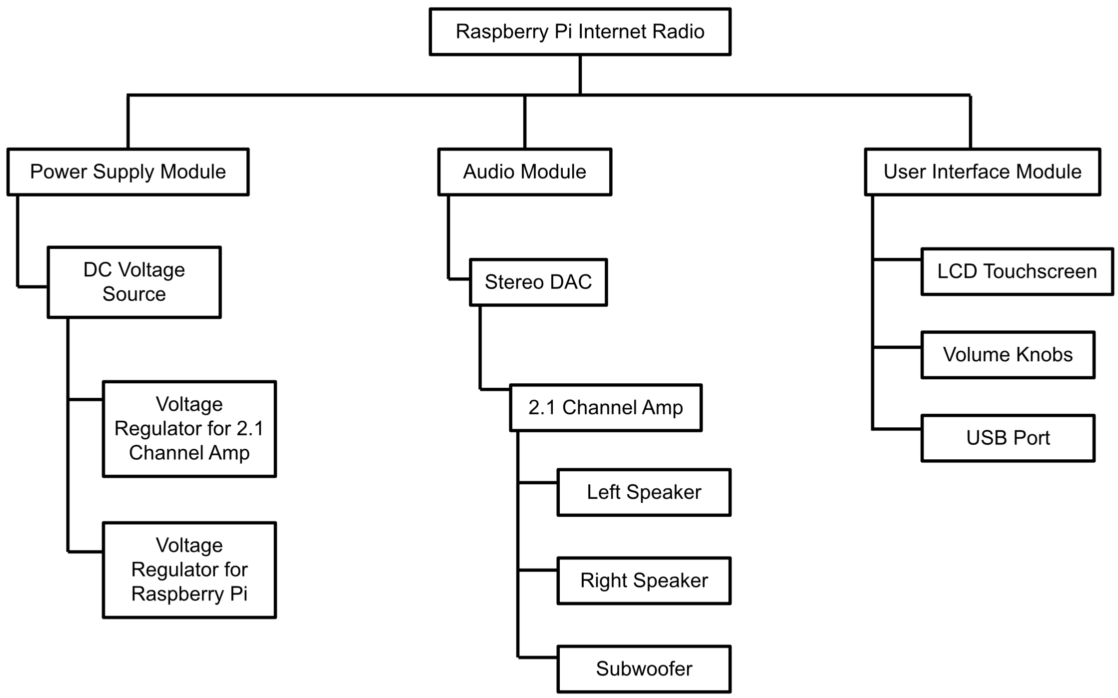

Figure 6: Generic Physical Architecture

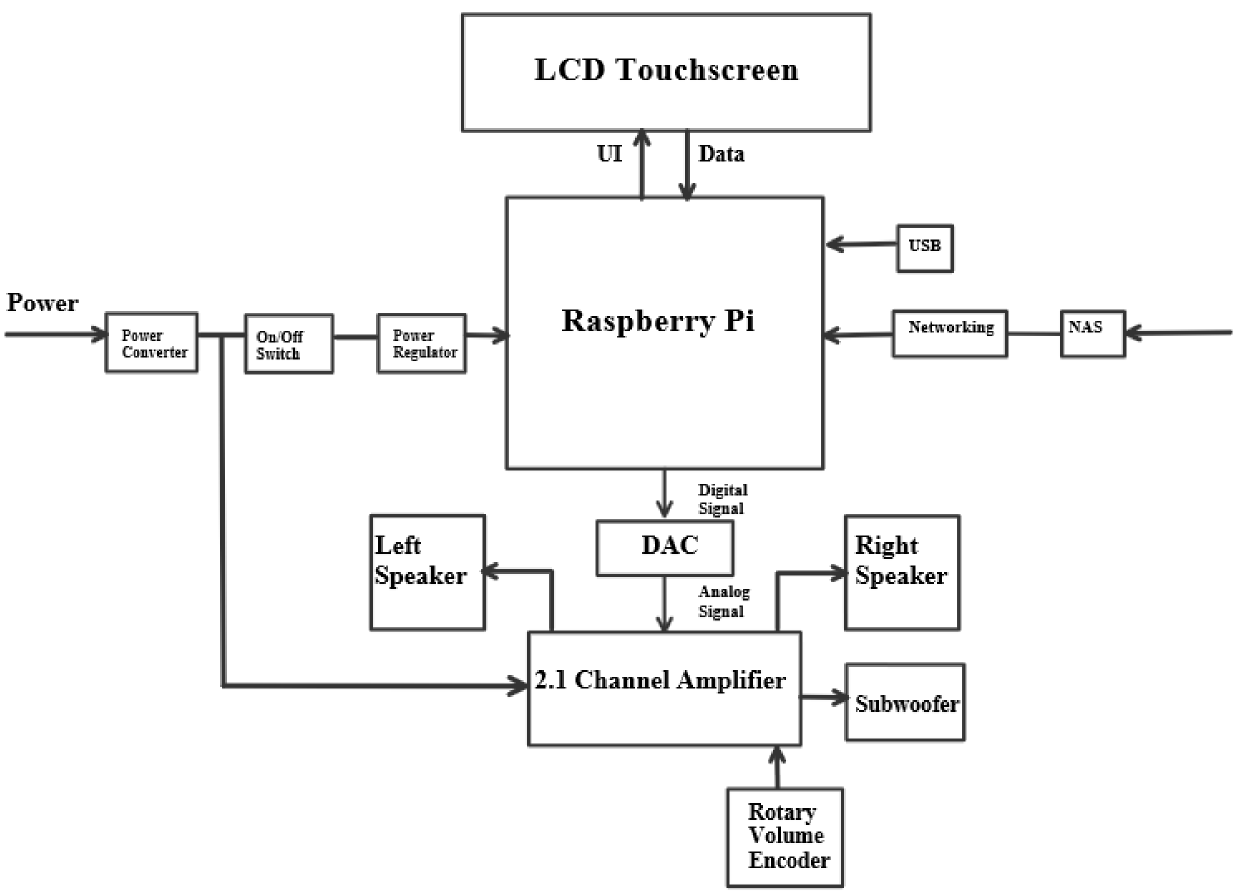

FIGURE 7: SYSTEM ARCHITECTURE

Background Knowledge/Phenomenology:

Communication Protocols:

The Pi needs to connect to a specific server to receive music from the internet. For example, a user's decision to connect to YouTube would require the Pi to establish a connection to a YouTube server. This connection between the Pi and the server is described using the TCP/IP protocol. HTTPS is the application layer protocol of TCP/IP that allows the user to access music data from a website [3]. The Pi connects to a HTTPS server through port 443 [4]. Once a connection has been established, the Pi acts as a client and requests music files from the server, and the server responds by providing the music files that were requested. Such requests and responses travel from the Pi to the server via the cloud, since the Pi and the server are connected to different networks.

The Pi also uses an application layer protocol of TCP/IP to communicate with the NAS; this protocol is referred to as SMB/CIFS [5]. As with HTTPS, SMB/CIFS is a request-response protocol. The Pi connects to the SMB/CIFS server through port 445 to establish a secure connection [6]. The NAS is the server and stores the user's music files. Since the NAS is located in the user's home, the NAS and Pi can communicate on the same local subnet.

Amplifiers:

As stated previously, the amplifiers for the design of the system will need to generate a signal that will be strong enough to supply two 50 W speakers and one 100 W speaker. The amplification of the signal is due to the gain, which is a constant factor that the device supplies the signal [7]. This factor is used to amplify the various frequencies in the signal without changing the shape of the waveform or the quality of the sound [8]. Gain is the ratio of the output voltage to the input voltage of the amplifier, as seen in equation (1). The forward voltage gain for audio amplifiers is typically 40dB, and the use of the forward gain with a feedback loop reduces noise and distortion from the power supply [9]. The amplifier can reach a saturation region, where it will no longer be able to amplify the signal [7]. It is important to refrain from supplying the amplifier with signals that cause the amplifier to enter the saturation region, as long exposure in the saturation region could damage the amplifier. For the left and right speakers in the design, the amplifier should have a frequency response that can cover a range of 20Hz to 20kHz, as these are the frequency ranges for most audio inputs [8]. Lower frequencies will be used for the subwoofer.

$$A_v = \frac{v_o}{v_i} \qquad (1)$$

This equation represents the transfer function for both low pass and high pass filters. The transfer function for the low pass filter can be found using nodal analysis as shown:

The transfer function for a low pass RC filter can be found by representing the filter in the Laplace domain and using nodal analysis as shown in Figure 8.

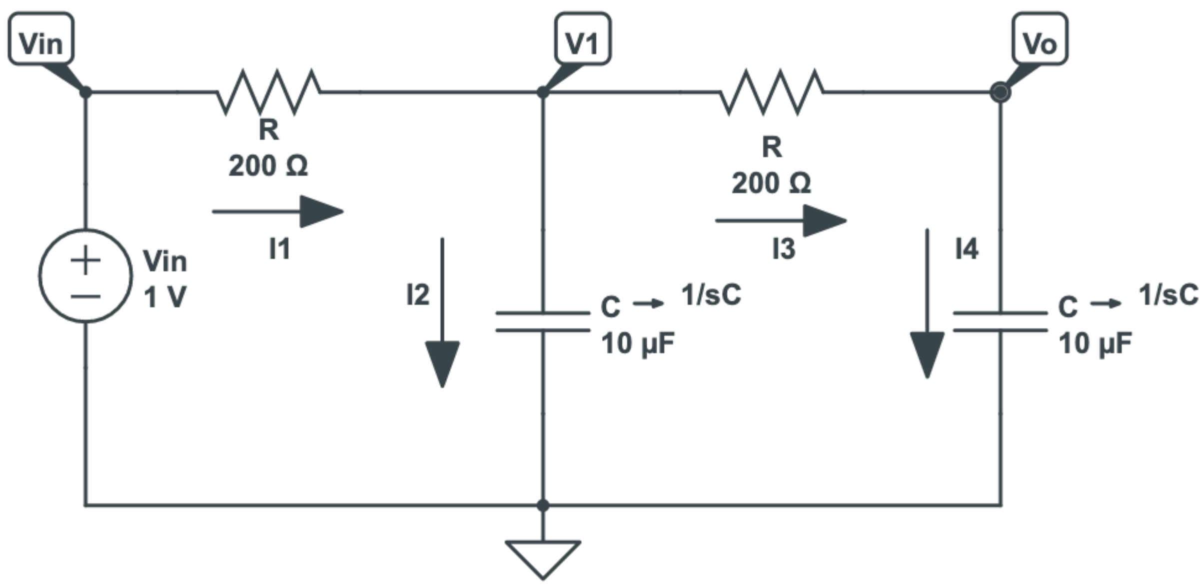

FIGURE 8: NODAL ANALYSIS OF RC LOW PASS FILTER

For node V1,

$$\frac{V_{in} - V_1}{R} = \frac{V_1}{\frac{1}{sC}} + \frac{V_1 - V_0}{R} \quad (2)$$

$$\frac{V_{in} - V_1}{R} = sCV_1 + \frac{V_1 - V_0}{R} \quad (3)$$

$$V_{in} - V_1 = sRCV_1 + V_1 - V_0 \quad (4)$$

$$V_{in} - V_1 + V_0 = sRCV_1 \quad (5)$$

$$V_{in} + V_0 = sRCV_1 \quad (6)$$

$$V_1 = \frac{V_{in} + V_0}{sRC} \quad (7)$$

For node V_0,

$$\frac{V_1 - V_0}{R} = \frac{V_0 - 0}{\frac{1}{sC}} \quad (8)$$

$$V_1 - V_0 = sRC \cdot V_0 \quad (9)$$

$$V_1 = sRC \cdot V_0 \quad (10)$$

Substituting (10) into (7), the transfer function is

$$\frac{V_0}{V_{in}} = \frac{1}{s^2R^2C^2 + sRC + 1} \quad (11)$$

The transfer function of a high pass RC filter can be found in a similar way as shown in Figure 9.

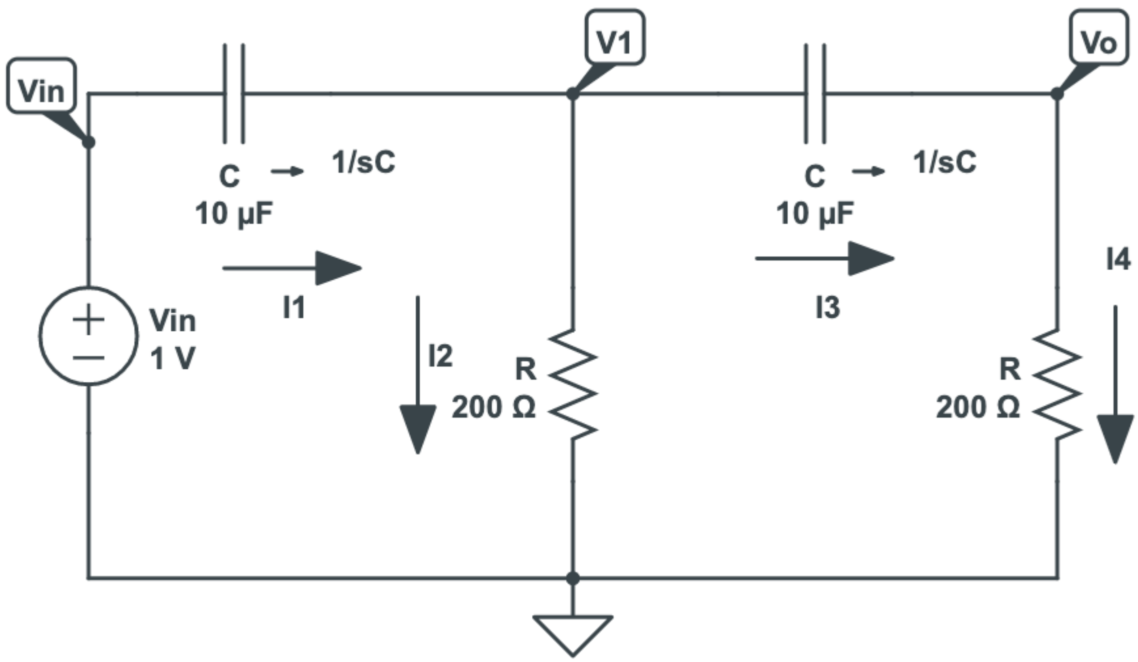

Figure 9: Nodal Analysis of RC High Pass Filter

For node V1,

$$\frac{V_{in} - V_1}{\frac{1}{sC}} = \frac{V_1}{R} + \frac{V_1 - V_0}{\frac{1}{sC}} \quad (12)$$

$$V_! = \frac{sRC \cdot V_{in} + sRC \cdot V_0}{2sRC + 1} \quad (13)$$

For node V_0,

$$(V_1 - V_0)sC = \frac{V_0}{R} \quad (14)$$

$$sRCV_1 = V_0 + sRCV_0 \quad (15)$$

Substituting (13) in (15), the transfer function is

$$\frac{V_0}{V_{in}} = \frac{s^2R^2C^2}{s^2R^sC^2 + sRC + 1} \quad (16)$$

The subwoofer is intended to output bass audio at low frequencies, as stated above. The crossover frequency will be 200 Hz for this system. Therefore, a low pass filter is needed to produce the bass audio of the subwoofer. A high pass filter is also needed for the stereo channels of the sound system. Both the low pass filter and the high pass filter have a cutoff frequency of 200 Hz for this system. The type of filter being used is a RC filter because RC filters are more commonly used for filtering audio [10]. A second order passive RC filter is being used because of its steep roll-off of -40 dB/decade and unity gain.

The values of R were calculated for $C = 220$ µF, $C = 10$ µF and $C = 100$ nF by plugging in the desired C as well as the desired cutoff angular frequency into equation (11). The information in Table 1 describes the calculated R, the R that would be implemented in practice, the actual frequency, and the percent error.

TABLE 1: PARAMETERS OF RC LOW PASS FILTER

Value of C	*Calculated Value of R*	*Actual Value of R*	*Actual Frequency*	*Percent Error*
220 µF	1.139 Ω	1 Ω	227.781 Hz	13.891 %
10 µF	25.056 Ω	25 Ω	200.447 Hz	0.224 %
100 nF	2.506 kΩ	2.5 kΩ	200.447 Hz	0.224 %

The same capacitor values were plugged into equation (16) for the high pass filter. The results are shown in Table 2.

TABLE 2: PARAMETERS OF RC HIGH PASS FILTER

Value of C	*Calculated Value of R*	*Actual Value of R*	*Actual Frequency*	*Percent Error*
220 µF	11.479 Ω	11 Ω	208.700 Hz	4.35 %
10 µF	252.527 Ω	252 Ω	200.419 Hz	0.210 %
100 nF	25.253 kΩ	25 kΩ	202.022 Hz	1.011 %

The values that were chosen for the filters were C = 100 nF and R = 2.5 kΩ for the low pass filter, and C = 100 nF and R = 25 kΩ for the high pass filter. These parameters were chosen because they have a low percent error and are readily available components.

The types of capacitors were also taken into consideration for the filter calculations. Electrolytic capacitors will be used in the filters, which have a tolerance of 20%. Table 3 depicts the varied capacitor values based on the tolerance and their corresponding resistor values.

Table 3: Capacitor Tolerance and Corresponding Resistance

C (nF)	Low Pass R (kΩ)	High Pass R (kΩ)
80	3.135	31.566
90	2.786	28.059
100	2.508	25.253
110	2.28	22.957
120	2.09	21.044

For the low pass filter, the lowest value of the capacitor tolerance was taken into account and the highest resistor value was used to compensate. For the high pass filter, the highest value of the capacitor tolerance was taken into account and the lowest resistor value was used to compensate.

Touchscreen:

An LCD touchscreen will be included to provide a user-friendly interface that will enable the user to access his or her music. This touch screen features an 800 x 480 capacitive touch display [11]. Capacitive touch screens, seen in Figure 8, are implemented in smartphones and tablets due to their increased brightness and sensitivity [12]. Moreover, capacitive touch screens have gained popularity in the last decade, as 90% of manufacturers in 2013 used this method of display [13]. Resistive touch screens are another type of touchscreen that are better suited for displays set up in an outdoor environment, since these touch screens provide much better protection against moisture and harsh temperatures.

Most modern touch screens are capacitive. When supplied with a voltage, a capacitor takes time to fully charge and subsequently takes time to discharge when the voltage supply is removed. The charge-discharge times change when the capacitance of the circuit itself is changing, which is the key principle in how touchscreens work. A human finger acts as a dielectric when it makes contact with a capacitive plate. This increases the capacitance in the circuit, and the ensuing change in charge-discharge time registers the presence of human touch [14]. This process is depicted in Figure 10 below.

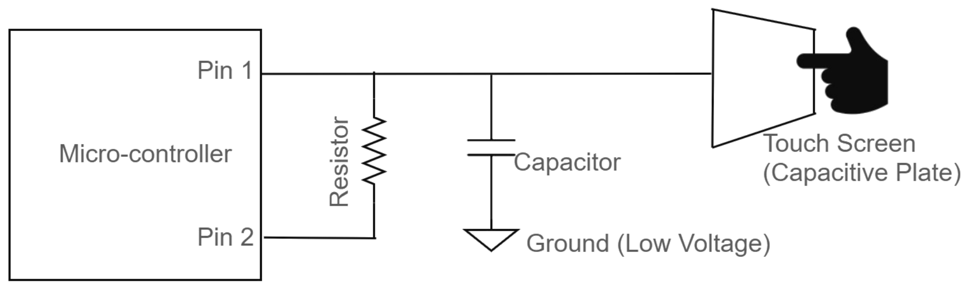

Figure 10: Capacitive Touch Sensing Circuit

Digital to Analog Converter:

A DAC is a vital part of the sound system for the device because the music is stored as digital data. The purpose of a DAC is to take digital data and produce an analog voltage. In the sound system, the analog voltage will be fed into amplifiers which will drive the speakers. A simple DAC can be made from resistors and an op amp, and there are two basic types: the weighted resistor, and the R-2R ladder circuit [15].

Another important specification of a DAC is the supported sampling rate. Sampling rate is a characteristic of the original audio recording. Sound is an analog signal, so to record it digitally, samples are captured and stored at a particular rate [16]. Ideally, those samples are recreated by the DAC at that same sampling rate to produce an audio signal. The maximum sampling rate supported by the chosen DAC is 384 kHz [17].

The sampling rate for audio is chosen based on the range of human hearing. Humans can generally hear sounds between 20 Hz to 20 kHz [18]. According to the Nyquist theorem, the audio source must be sampled at a frequency that is greater than or equal to twice the maximum frequency [18]. By sampling at least twice the maximum frequency, all frequencies lower than that maximum are properly recorded. Equation 17 shows the sampling rate f_s needed to encompass all frequencies of 20 kHz and lower.

$$f_s = 2 \times 20\ kHz = 40\ kHz \qquad (17)$$

A common sampling frequency is 44.1 kHz, and higher quality audio recordings are captured at even higher sampling rates.

Voltage Supply:

Voltage regulators are necessary to step down the supply voltage to the appropriate level for each component in the system. There are typically two types of voltage regulators: linear, and switching.

A linear regulator steps down a voltage by dissipating excess power as heat [19]. The efficiency of a linear regulator follows a linear relationship [19], where the efficiency decreases as the output of the regulator decreases.

A switching regulator steps down voltage by rapidly pulsing on and off the voltage [20]. Ripple is reduced and the voltage is stabilized by feeding the pulsed voltage through a low pass filter. The voltage level at the output is adjusted by varying the duty cycle of the pulse.

Switching regulators are preferred because of the reduced heat dissipation. However, switching creates noise, which must be accounted for if the components are sensitive to variations in supply voltage.

Detailed Design:

Circuit Schematics:

The amplifier circuit for the stereo speakers can be seen in Figure 11 and the circuit for the subwoofer can be seen in Figure 12.. It is important to take note of the RC filters being used to filter the analog signal on the output before feeding it to the speakers. Other functionalities the chip provides include mute functionality (pin 17) and gain setting (pin 2). In particular, a Fault pin and an OTW_CLIP pin provide error readings that allow for troubleshooting.

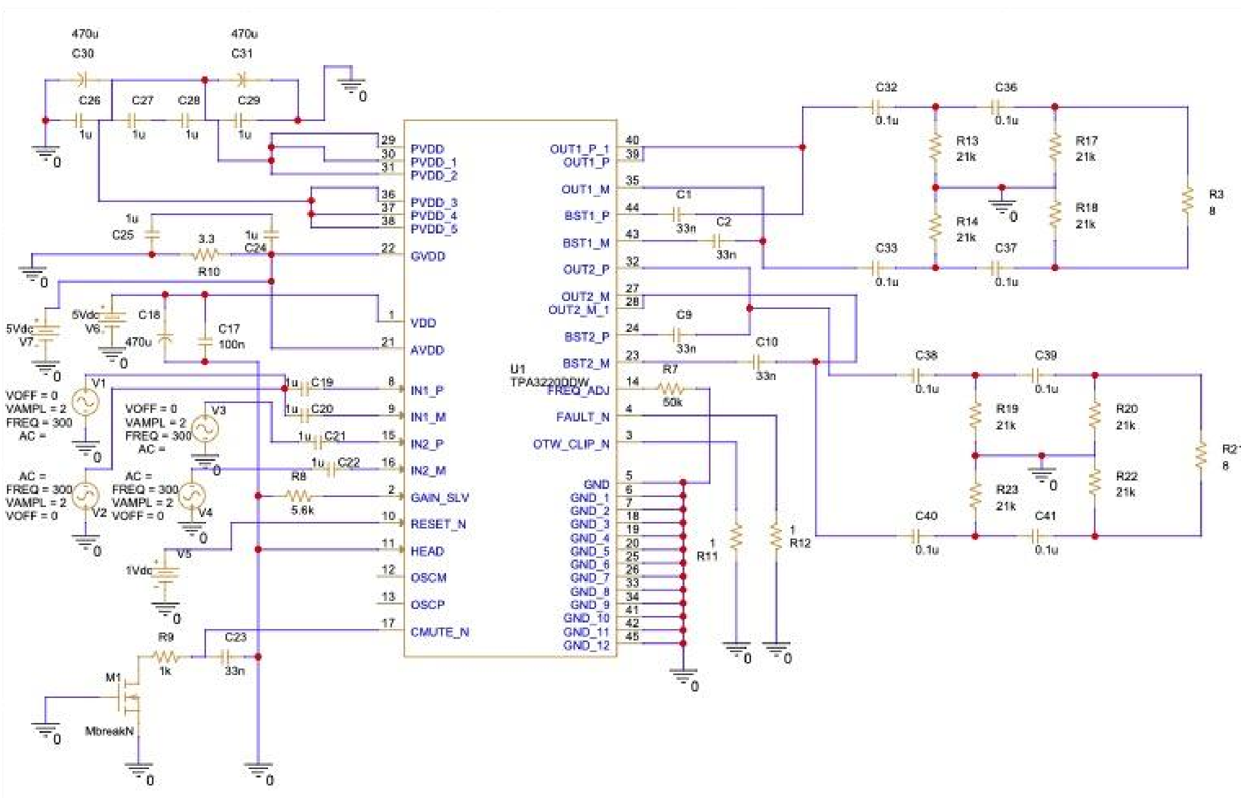

FIGURE 11: STEREO SPEAKER AMPLIFIER CIRCUIT

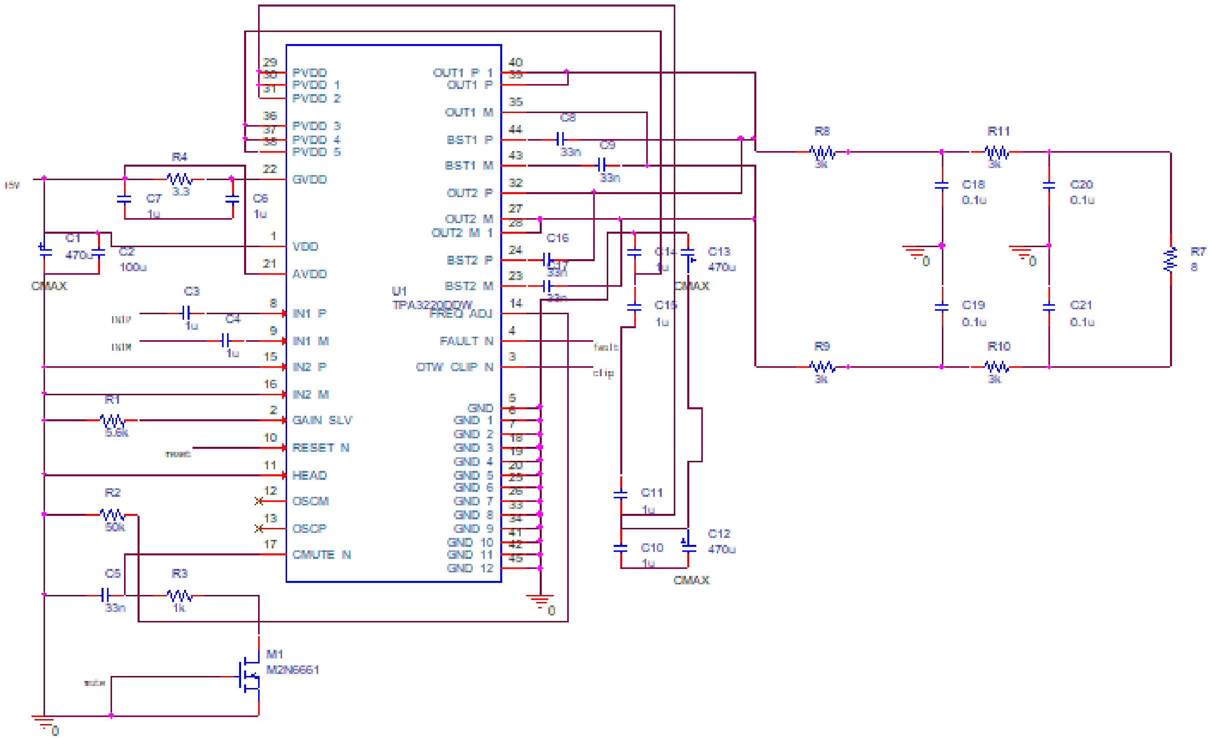

FIGURE 12: SUBWOOFER AMPLIFIER CIRCUIT

The circuit schematics for the low pass filter and the high pass filter are shown in Figures 13 and 14, respectively. These filters are designed to support a bridge-tied load (BTL), which is consistent with the filters used in the amplifier circuits in Figures 13 and 14.

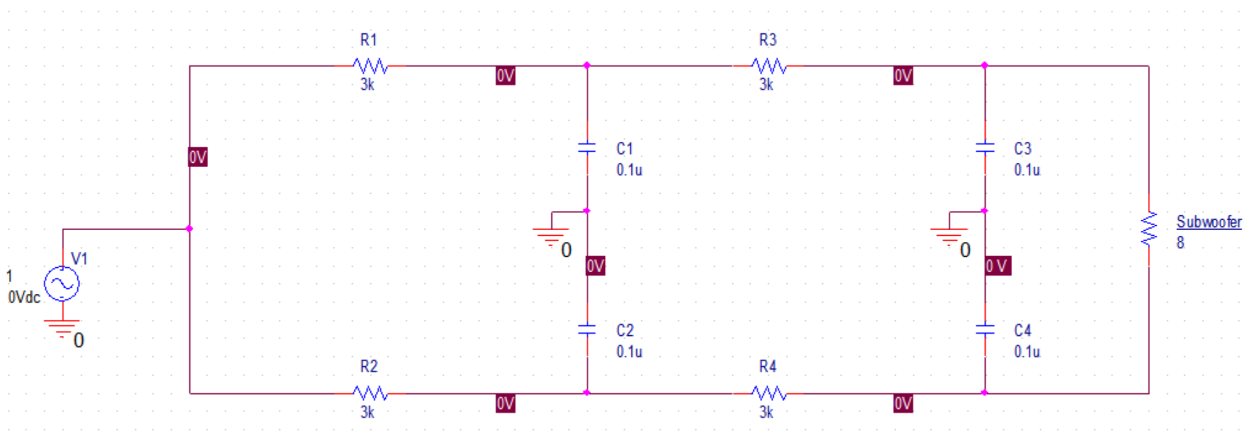

FIGURE 13: LOW PASS FILTER CIRCUIT

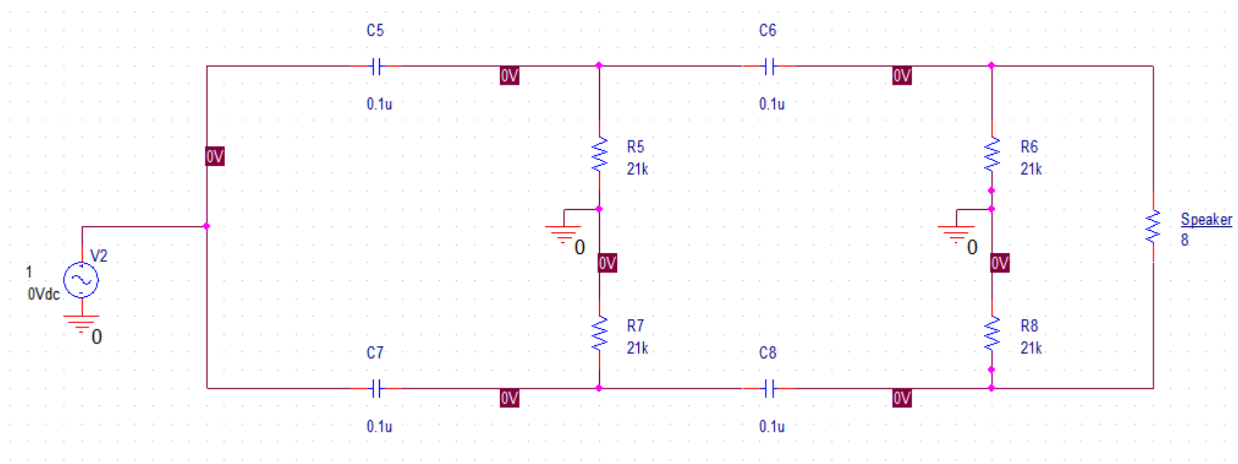

Figure 14: High Pass Filter Circuit

Components and Interfaces:

A list of the finalized components used in the system can be seen in Table 4.

Table 4: System Components

Component	Device Name
LCD Touchscreen	ELECROW Raspberry Pi 5" touch screen
Raspberry Pi	Raspberry Pi 3B+
DAC	PCM5122 Audio Stereo DAC
Amplifier	TPA3220DDW Audio Amplifier
Speakers	JBL 2500 Speaker NIPPON Subwoofer
Open Source Software	Mopidy

Multiple interfaces will be used for the Pi to communicate with other components. The Pi communicates with the DAC over both I^2C and I^2S; I^2C is for setting the DAC control registers, and I^2S is for sending audio data. The Pi also transmits information with digital potentiometers over I^2C to set control registers that correspond to the wiper position. Both the DAC and digital potentiometers connect to the Pi via the GPIO pins. In addition, the touchscreen will receive a video signal from the Pi via HDMI and receive power via one of the USB ports.

Description of Software:

Mopidy is an open source music server, and supports all required functions either by default or with free plugins. The server handles music playback, the file system, and the user interface. Therefore, it is unexpected that the source code will have to be modified to fit the requirements.

There are several ways to install the base version of Mopidy. To begin on a Debian based Linux distribution like Raspbian:

1. Add the archive's GPG key:
   ```
   wget -q -O - https://apt.mopidy.com/mopidy.gpg | sudo apt-key add
   ```
2. Add the APT repo to source packages:
   ```
   sudo wget -q -O /etc/apt/sources.list.d/mopidy.list
   https://apt.mopidy.com/buster.list
   ```
3. Install Mopidy and all dependencies:
   ```
   sudo apt update
   sudo apt install mopidy
   ```

Mopidy is now ready to run. However, to install extensions like Spotify or Soundcloud:

1. Install extensions:
   ```
   apt search mopidy
   ```
2. Install one of the listed packages, e.g. mopidy-mpd:
   ```
   sudo apt install mopidy-mpd
   ```

The source code of Mopidy must be modified to add the ability to interact with the GPIO. The source code will be copied into the team's Github repository, where all customizations and modifications will be stored. The installation for the final, modified version of Mopidy will involve cloning the repository onto the Pi, then launching the program.

The flowchart for a typical user interaction with Mopidy can be seen in Figure 15.

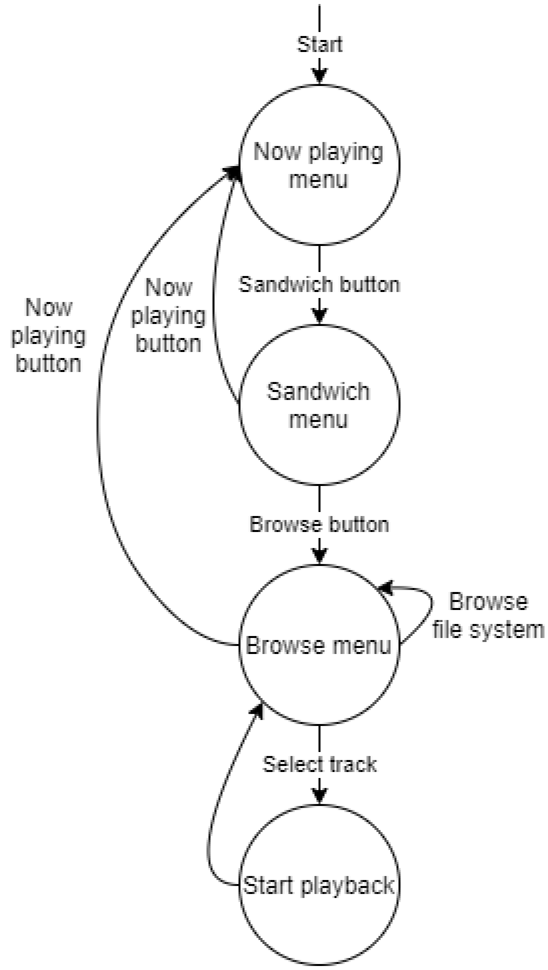

FIGURE 15: FLOWCHART OF USER INTERACTION WITH MOPIDY

Custom software needs to be developed for the Pi to control the DAC, rotary encoders, and digital potentiometers. The state diagram for the control software is depicted in Figure 16.

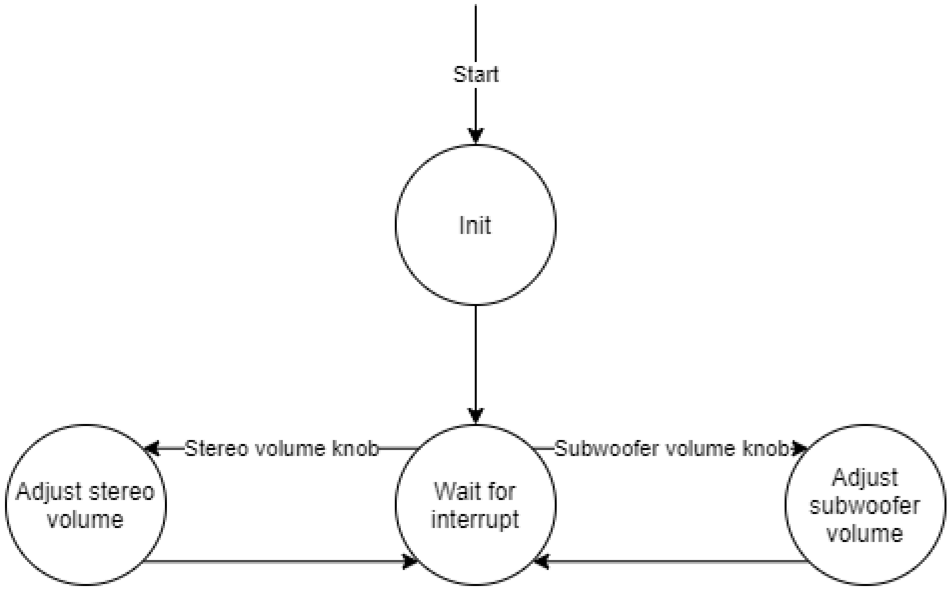

FIGURE 16: STATE DIAGRAM FOR CONTROL SOFTWARE

The flowcharts for each control state are shown in Figures 17 through 19.

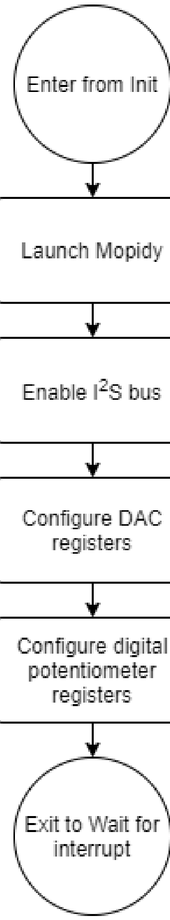

Figure 17: Flowchart for Init State

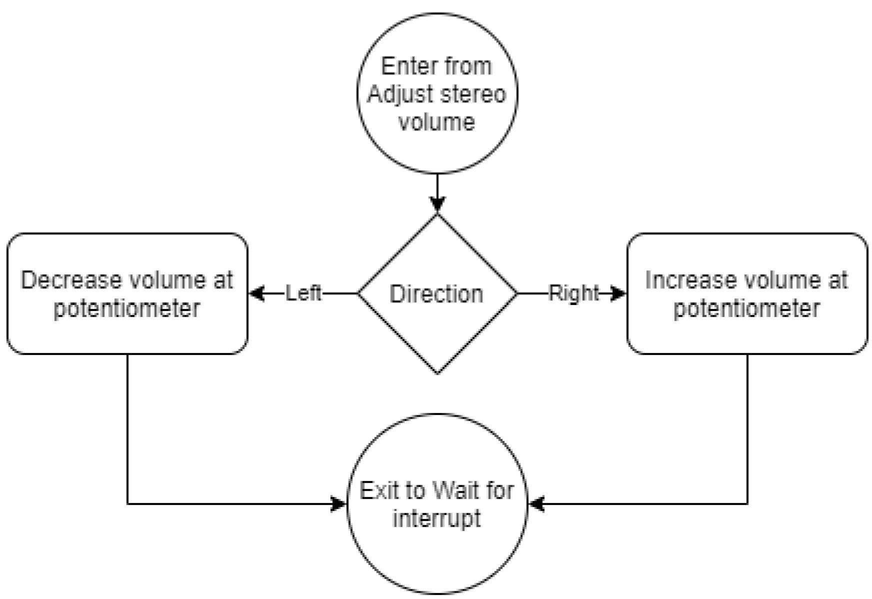

Figure 18: Flowchart for Adjust Stereo Volume State

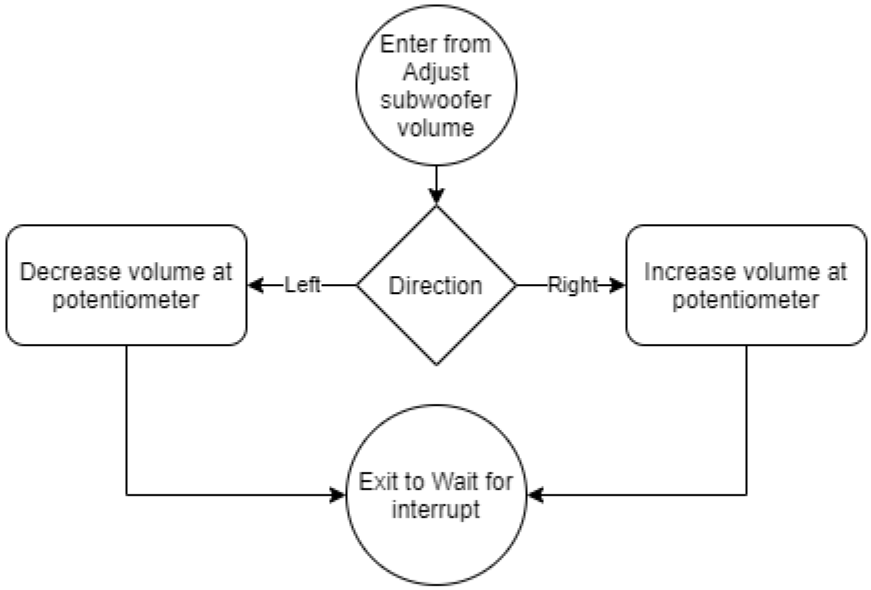

Figure 19: Flowchart for Adjust Subwoofer Volume State

Prototyping Progress:

DAC:

One of the first steps when prototyping was connecting a DAC to a Raspberry Pi. For testing purposes, the Adafruit I2S Stereo Decoder [21] was configured to a breadboard and connected to the Raspberry Pi. The I^2S audio drivers were enabled on the Pi based on a guide from Adafruit Industries [22]. The output of the DAC was connected to a simple stereo amp constructed with LM386 amps [23], which was connected to speakers. The system was capable of playing stereo sound as expected. The setup was demonstrated at the design review presentation as seen in Figure 20.

Figure 20: DAC Demonstration

Adapters:

The amplifiers and the DAC are both QFP package chips; thus, prototyping on a bread board with them poses a challenge. PCB adapters were designed, as seen in Figure 21 and Figure 22, to allow for the breadboarding of these chips and completion of a functional prototype. Note that the large copper pad in the design of the amplifier PCB layout functions as a heat sink by dissipating the heat over the large area.

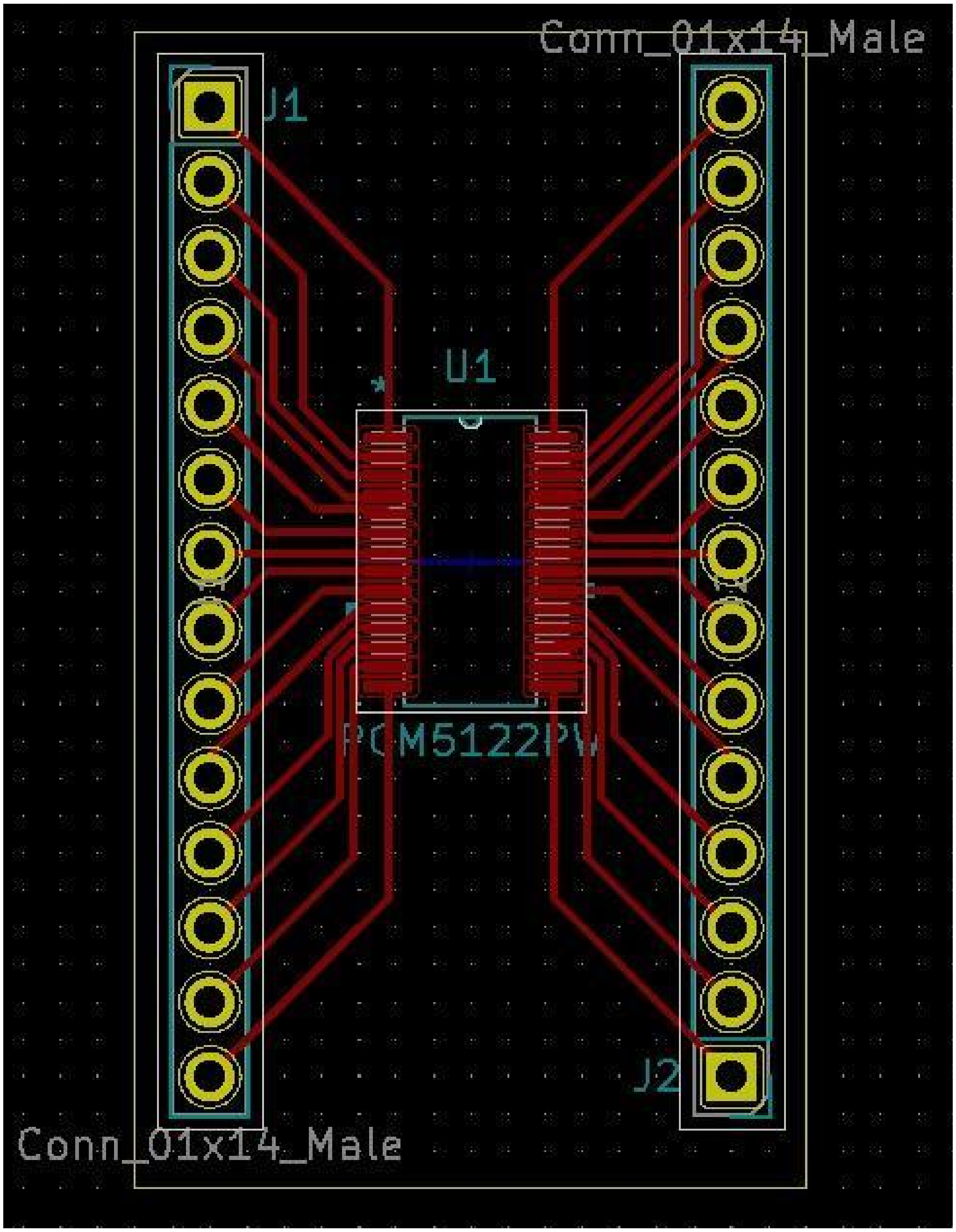

FIGURE 21: DAC PCB BREADBOARD ADAPTER

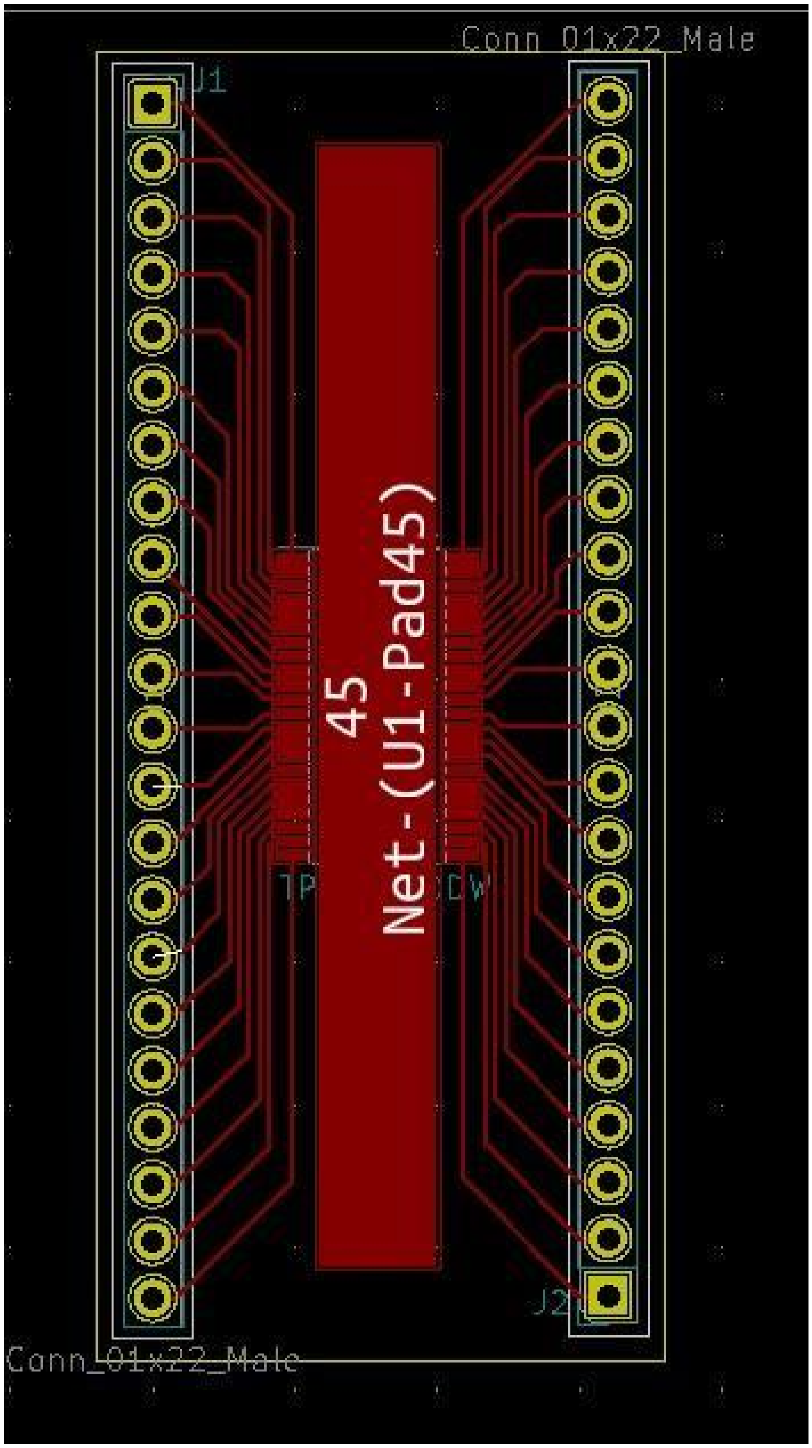

FIGURE 22: AMPLIFIER BREADBOARD ADAPTER

Amplifier:

Figure 23 depicts the initial prototyping configuration for the TPA322DDWR amplifier. The configuration is based on the typical differential AD-Mode Parallel Bridge Tied Load (PBTL) amplifier circuit diagram, which is described in the datasheet for a subwoofer application. The prototyping effort for this device is currently underway. There have been some challenges in achieving the output for the device such as minor circuitry errors, and trace/pin header stability of the PCB. Ultimately, it was decided to advance with manufacturing a more stable PCB that will contain this circuit and other surface mount technology for further prototyping.

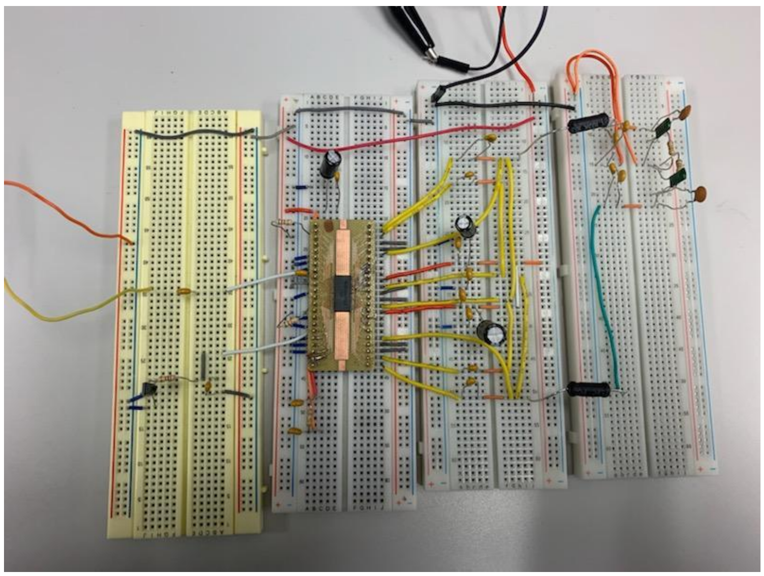

FIGURE 23: AMPLIFIER PROTOTYPING SETUP

Software:

Mopidy and Iris [24], a front-end for Mopidy, was installed on a Pi. The menus of the software were explored to make state diagrams that represent the software.

LCD Touchscreen:

The LCD touchscreen plugged into the Raspberry Pi and worked with no configuration required. Furthermore, the Iris interface for Mopidy has buttons large enough to accommodate a touch interface.

Filters:

The low pass RC filter was simulated in PSPICE using $C = 0.1\mu F$ and $R = 3\ k\Omega$. Similarly, the high pass RC filter was simulated using $C = 0.1\ \mu F$ and $R = 21\ k\Omega$. There were three main factors to consider when analyzing the simulated results for both filters: gain, cutoff frequency, and roll-off. The desired properties of the filter were a unity gain, a cutoff frequency of 200 Hz, and a roll-off of −40 dB/decade.

The schematic for the low pass filter is shown in Figure 24.

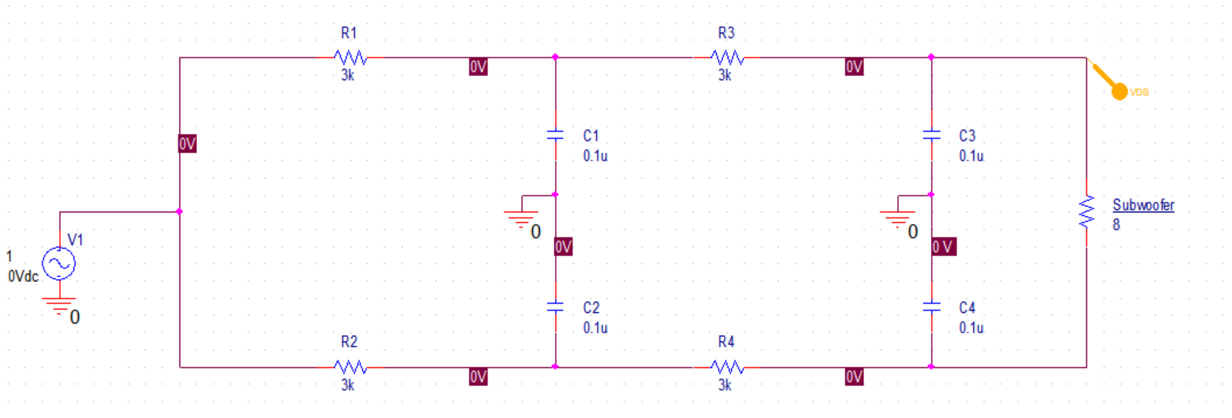

FIGURE 24: SCHEMATIC FOR LPF

The simulated frequency response is shown in Figure 25. The frequency response demonstrates unity gain in the passband. The simulated cutoff frequency is 198.06368 Hz and the roll-off is -35.470 dB/decade.

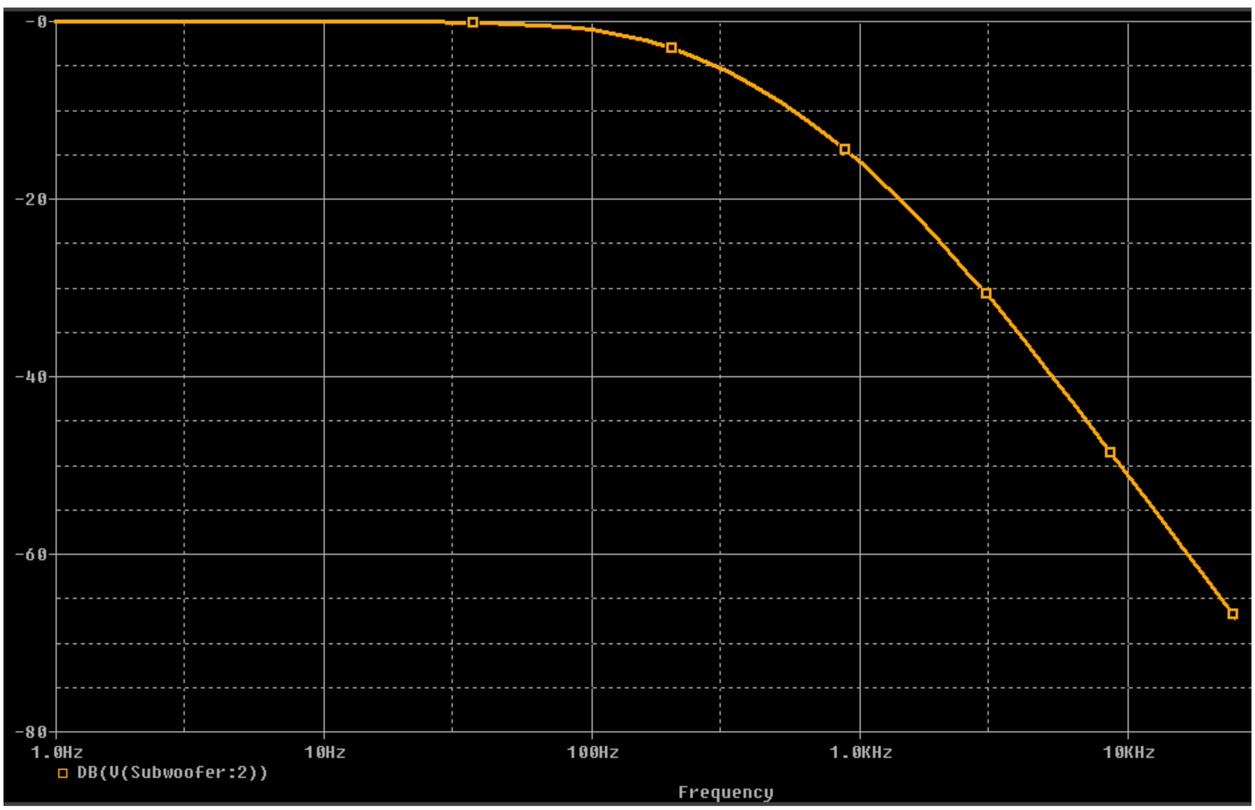

FIGURE 25: SIMULATED FREQUENCY RESPONSE FOR LPF

The schematic for the high pass filter is shown in Figure 26.

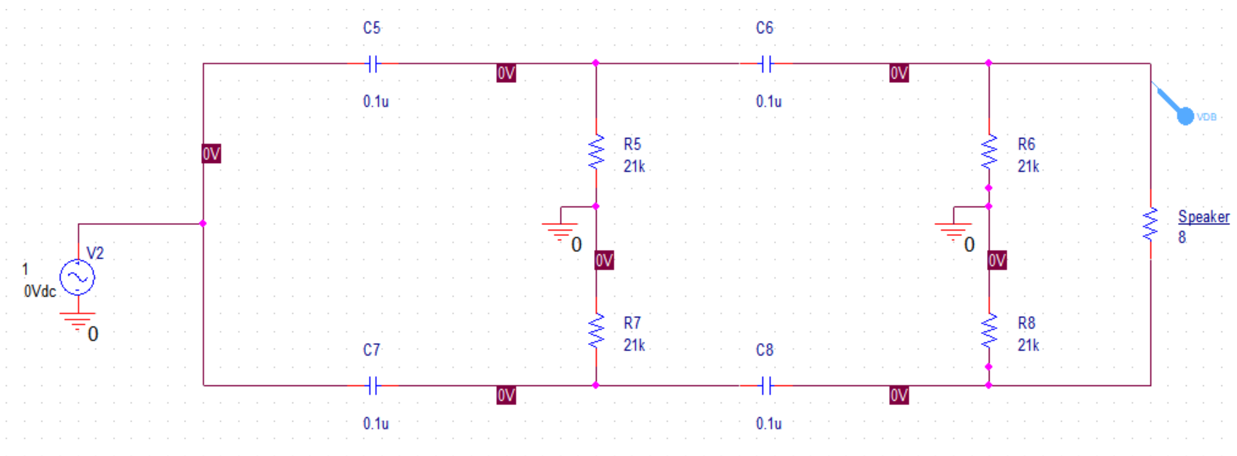

Figure 26: HPF Schematic

The simulated results for the high pass filter are shown in Figure 27. The frequency response for this filter also produces a unity gain in the passband. The simulated cutoff frequency is 203.84228 Hz and the roll-off is -36.420 dB/decade.

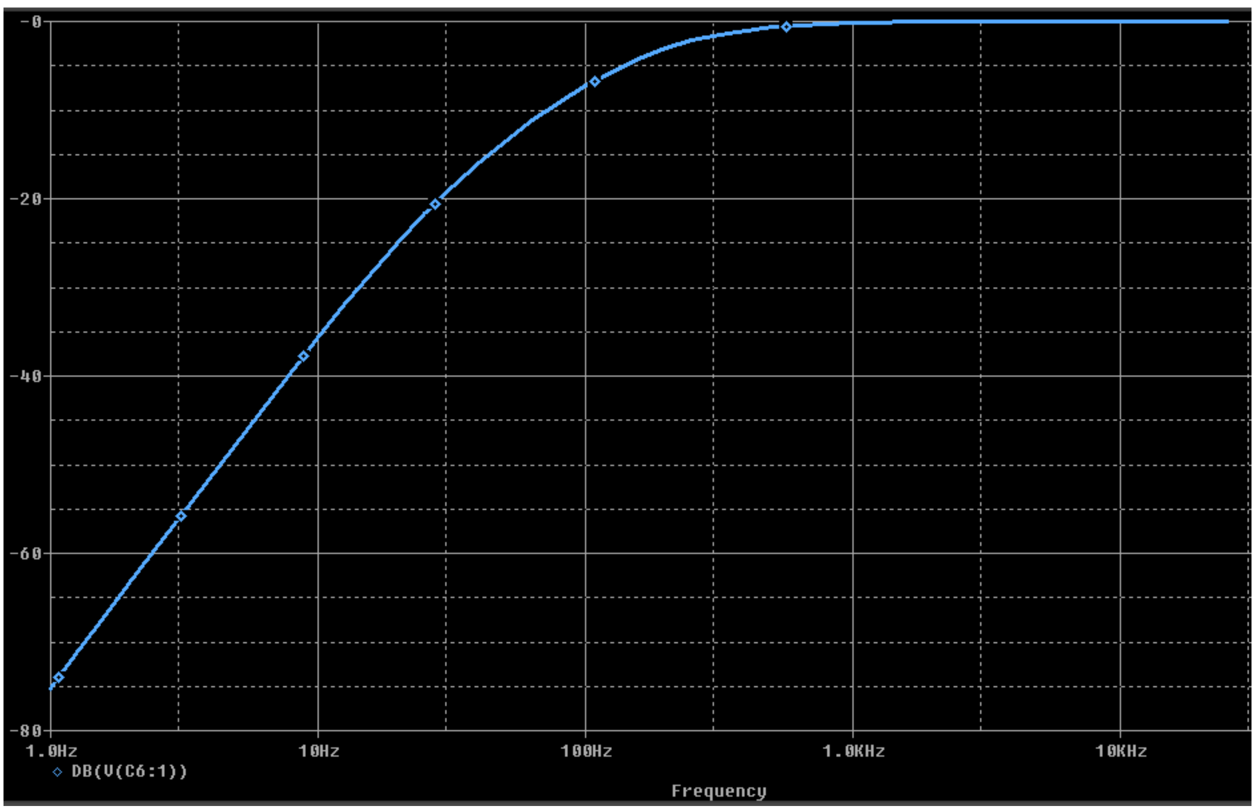

Figure 27: Simulated frequency response for HPF

Figure 28 shows the simulated crossover for both filters. The crossover occurs at 202.307 Hz, where the low pass filter has a gain of -3.1014 dB and the high pass filter has a gain of -2.9993 dB.

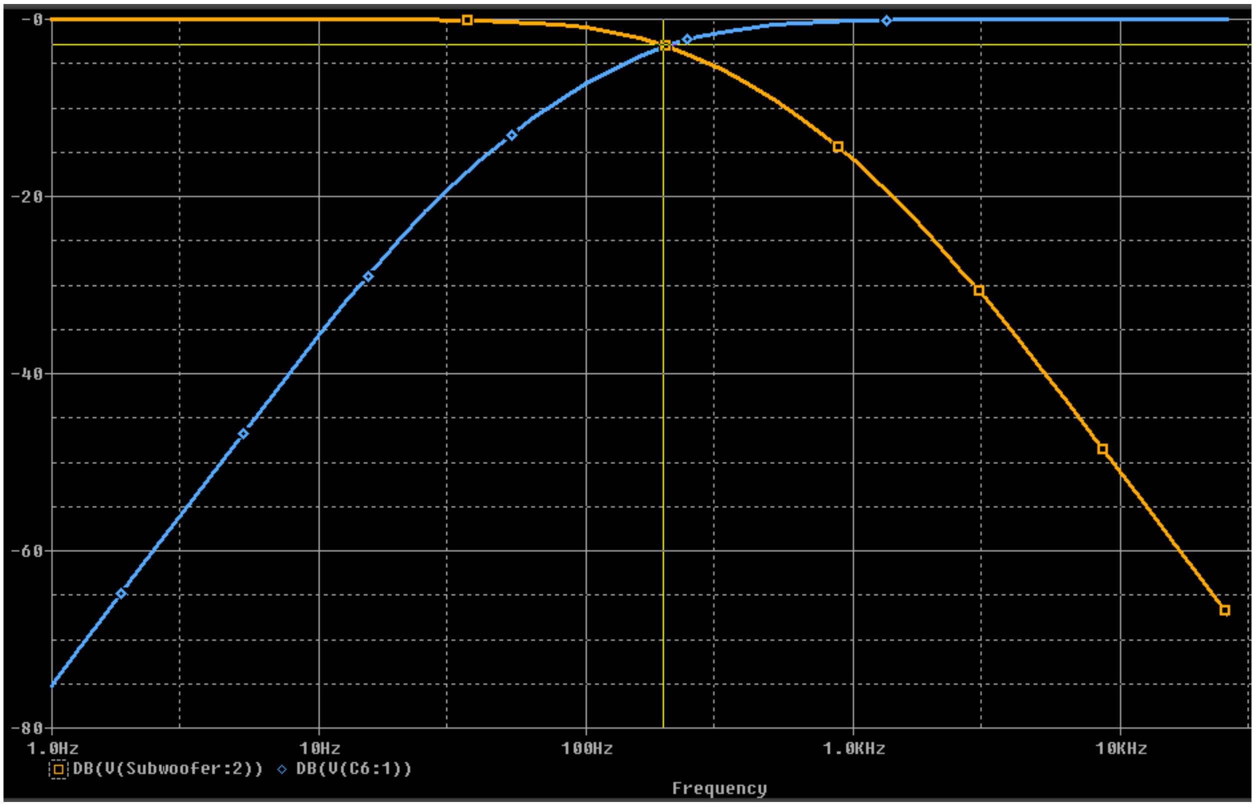

FIGURE 28: SIMULATED FREQUENCY RESPONSE FOR HPF AND LPF

The crossover for the filters without a load is shown in Figure 29. The frequency response demonstrates that the crossover is at 193.21 Hz. This crossover occurs when the low pass filter has a gain of -3.927 dB and the high pass filter has a gain of -3.386 dB.

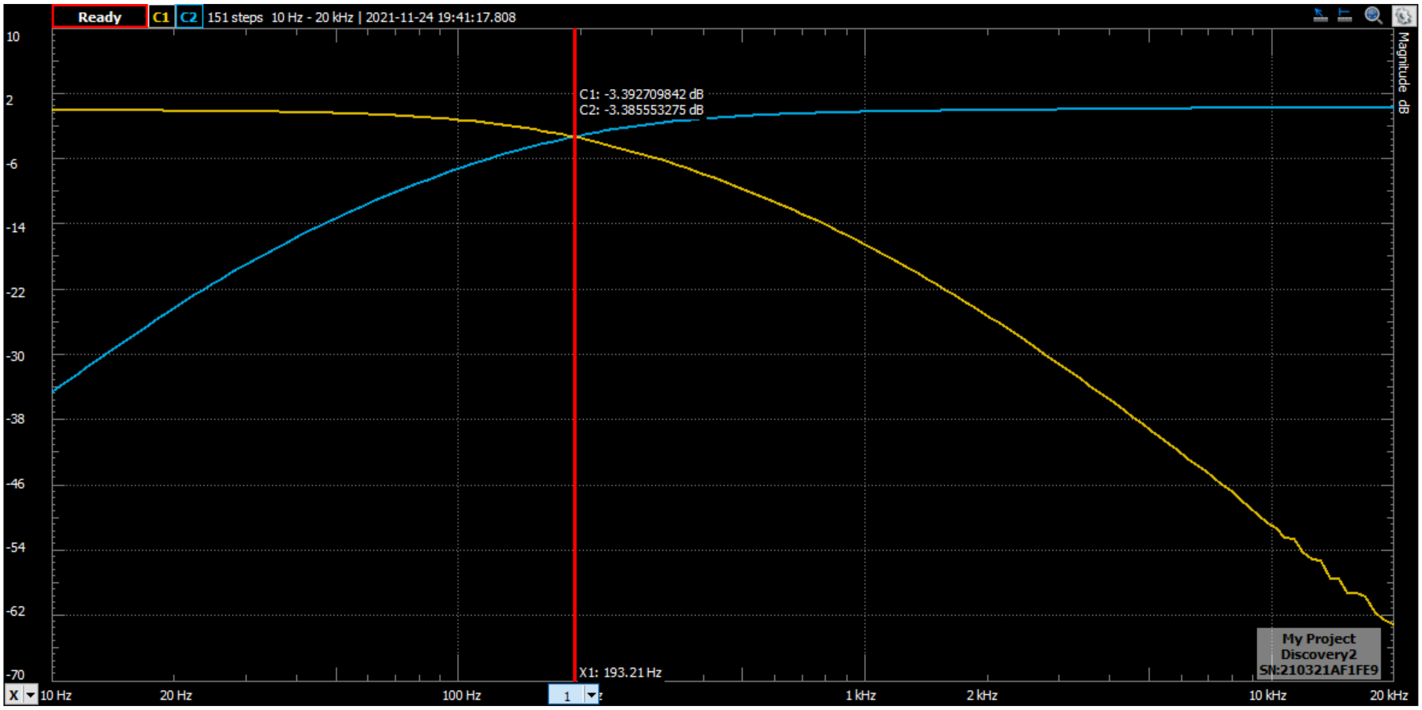

FIGURE 29: CROSSOVER FOR FILTERS WITHOUT A LOAD

When a load is implemented, the gain reduces from unity gain to approximately -6 dB for both the low pass filters and high pass filters. The crossover for the filters with a load is shown in Figure 30. The frequency response shows that the crossover frequency is at 195.91 Hz when the low pass filter has a gain of -9.419 dB and the high pass filter has a gain of -9.431 dB.

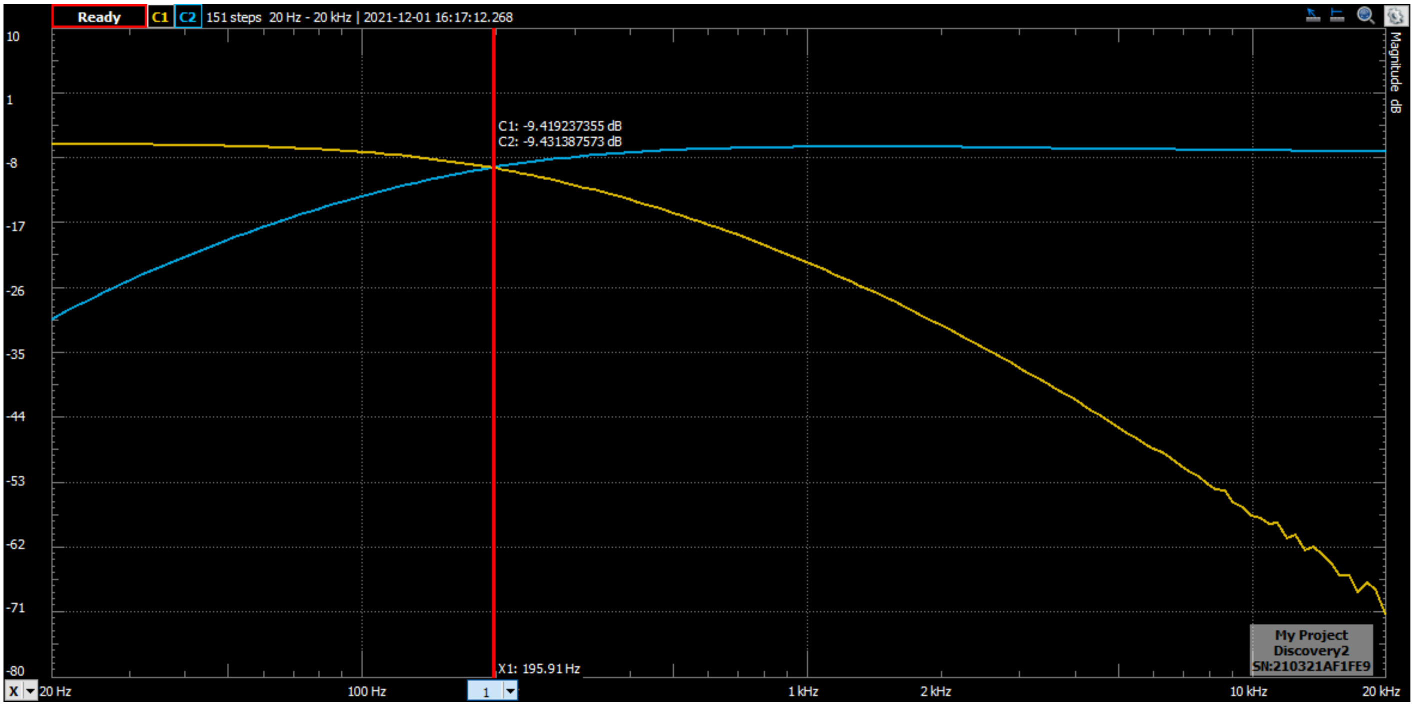

Figure 30: Crossover for Filters With A Load

The frequency response for the low pass filter with a load is shown in Figure 31. The cutoff frequency is 184.08 Hz and the gain in the passband is -6.1 dB.

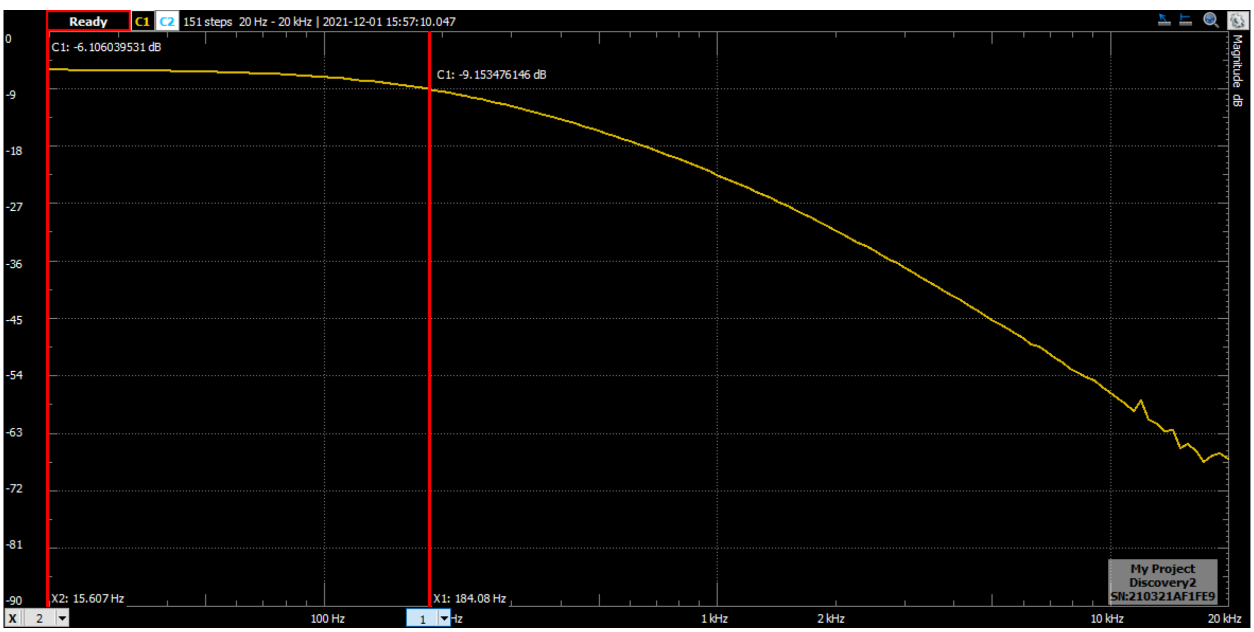

Figure 31: Frequency Response for LPF With A Load

The frequency response for the high pass filter is shown in Figure 32. The cutoff frequency is at 194.68 Hz and the gain is -6.469 dB in the passband.

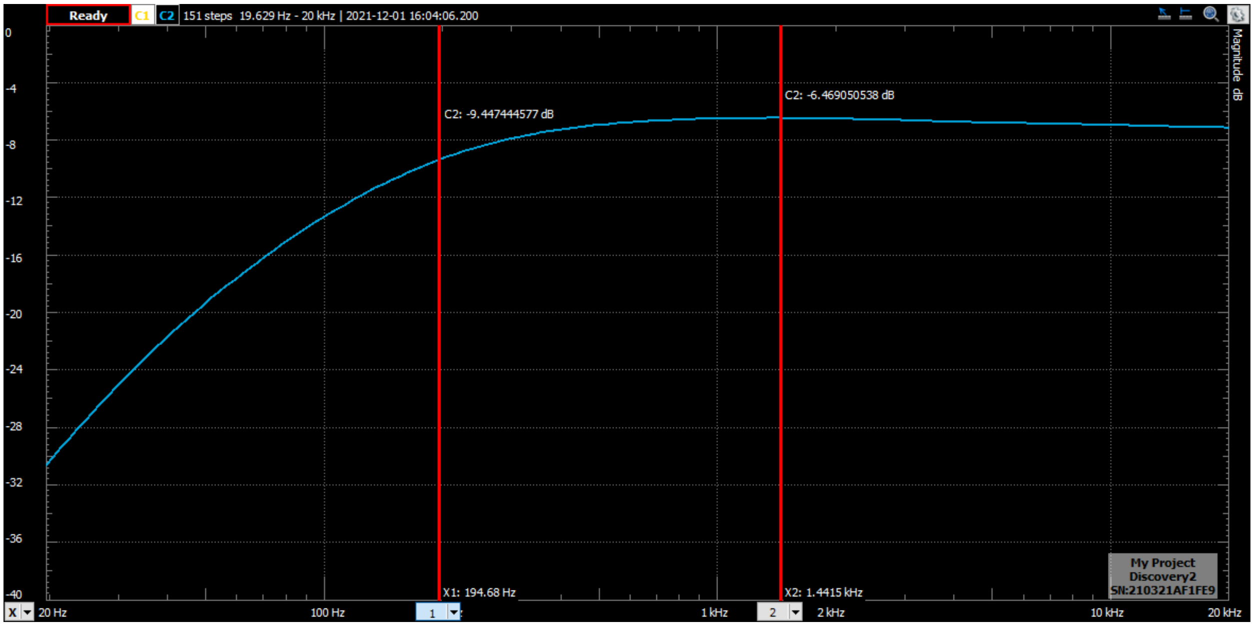

Figure 32: Frequency Response for HPF With A Load

Testing Plan:

Power Supply:

More research needs to be conducted before a power supply capable of outputting at least 200 W is purchased. Voltage regulators will also be necessary for stepping down the supply voltage to the appropriate level for each component. The detailed testing plan for the power supply is shown below.

1. Experiment #1: Voltage Regulator for Pi
   a. Goal: To ensure that the voltage regulator can power the Pi
   b. System components: Power Supply, Voltage Regulator, Pi
   c. Testing process:
      i. Connect the power supply to the voltage regulator input.
      ii. Connect a dummy load, such as a 10 kΩ resistor, to the voltage regulator output.
      iii. Set the voltage regulator to 5V.
      iv. Measure the average voltage and ripple with an oscilloscope.
      v. If the measured voltage is within the tolerance of the Pi, then this test is passed.
      vi. If not, then put in a larger capacitor across the regulator output and go to step iv.

   d. Evaluation: By the time this test is finished, the minimum capacitor required to properly power the Pi is found.
   e. Note: The capacitor may not be sufficiently large to eliminate audible noise from the regulator.

The power supplies will also need to be measured for noise. Noise will be dampened by inserting a sufficiently large capacitor at the voltage rails. The plan for minimizing noise from the power supply is shown below.

2. Experiment #2: Noise from Voltage Regulator
   a. Goal: To ensure that the voltage regulator produces negligible audible noise
   b. System components: Power Supply, Voltage Regulator, Amplifiers, Speaker, Pi
   c. Testing process:
      i. Construct the circuit from experiment #1 and replace the dummy load with the Pi.
      ii. Connect the amplifier to the circuit.
      iii. Connect speakers to the amplifier.
      iv. Ground the amplifier audio inputs.
      v. Listen for humming, buzzing, or hissing from the speakers, which represents noise from the power supply.
      vi. If the noise is not audible, then this test is passed.
      vii. If not, then put a larger capacitor across the power supply rails and go to step v.
   d. Evaluation: By the time this test is finished, the minimum capacitor required to silence extraneous noise is found

DAC:

The PCM5122 has yet to be tested because it is a surface-mount component, so it cannot be plugged into a breadboard. A breadboard adapter PCB was created and the chip was soldered to it. However, further research must be done on the control registers before the DAC can be configured in stereo and be connected to the Raspberry Pi.

Software:

Mopidy has extensions for all of the features requested. Those extensions will soon be tested. The source for the software must also be studied for documentation as well.

A NAS must be set up to test the NAS functionality of Mopidy. The NAS will operate as a SMB server on a laptop and is expected to connect to the Pi through an ethernet cable..

The software for controlling the hardware connected to the Pi, such as the rotary encoders, the digital potentiometers, and the DAC, must be written. Special attention must be paid to GPIO contention, because certain interfaces like I^2S, I^2C, and SPI are only available on specific pins.

Operating Temperature:

Special attention must be paid to the operating temperature of the device due to the large power requirements for the system. Moreover, the amplifiers will dissipate the most heat; the heat sinks on the amplifiers need to be attached properly to avoid the amplifier from melting.

Subwoofer Signal:

The subwoofer music signal must be derived from the stereo output of the DAC. A challenge will be combining the stereo signals and avoiding crosstalk between the channels. A voltage adder will likely be used to do the combination.

1. Experiment #3: Crosstalk Between Audio Channels
    a. Goal: Measure the amount of crosstalk between the left and right audio channels
    b. System components: Voltage Adder, Amplifiers
    c. Testing process:
        i. Ground the left input into the voltage adder
        ii. Connect a function generator to the right input into the voltage adder. Set the function generator to 1kHz
        iii. Measure and record the output from the left stereo amplifier
            1. Listen for any noise from the left speaker
        iv. Ground the right input into the voltage adder
        v. Connect a function generator to the left input into the voltage adder. Set the function generator to 1kHz
        vi. Measure and record the right stereo amplifier
            1. Listen for any noise from the right speaker
        vii. This test is passed if there is no audible sound in step iii. and step vi.
        viii. If not, then the voltage adder design must be reevaluated and this test must be redone for the new design

Problems Encountered:

One issue encountered when testing the LCD touchscreen was a low voltage warning error from the Pi. This could be due to low power generated from the power supply. However, because the LCD touchscreen is compatible with the Pi, it is not a serious problem.

Another issue was that certain components needed for the project were out of stock. For example, the first choice for the amplifier chip was the TPA3156D2DAD. The chip went out of stock almost a week after it was picked as a final component. Similarly, the PCM5102 DAC went out of stock as well. These components needed to be opted out for the second option amplifier and the second option DAC. Fortunately, both alternatives meet the requirements for the project.

Another problem was experienced during the software testing part of the process. The exact chips used in the system were not found in Pspice or TI. Instead, similar models from the same family were used for simulation.

Future Tasks:

Development, implementation of the design specifications, and testing will occur after prototyping the design. A list of the major tasks and their subcategories for the coming months are listed below. This list is preliminary and is subject to change.

1. **Hardware Development (4 weeks)**
   a. PCB Design
   b. PCB Assembly
   c. Analog Volume Control Setup and Testing
2. **Software Development (2 weeks)**
   a. Open-source Interfacing
   b. Request Module
   c. Display Module
   d. Music Module
3. **System Integration (6 weeks)**
   a. Request Functionality
   b. Display Functionality
   c. Music Functionality
   d. Digital Volume Control
4. **Testing**

   (5 weeks)
   a. Experiment #1
   b. Experiment #2
   c. Experiment #3
   d. Experiment #4
5. **Reporting**
   a. Progress Report
   b. In-Progress Presentation

c. Final Report and Presentation

6. **Milestones/Demos**
    a. Demo #1 Hardware
    b. Demo #2 Software
    c. Demo #3 Integrated System

The allocation of responsibilities is subject to change depending on member interests and capabilities for a given task assignment. The team of six will be split into two groups. Ben and Danny will be tasked with Software Development, and Zara will help with implementing the SMB/CIFS protocol for the NAS. The second group will focus on Hardware Development and will consist of James, Zayd, Zuha, and Zara. These teams will accomplish both the hardware and software development simultaneously and then work together on system integration.

Each experiment that will be tested will be overseen by at least two members of the group: one person will be the lead on the test and the other person will provide support. The hardware based experiments will be supervised by those on the hardware team, and software based experiments will be supervised by those who developed the software. However, additional input, comments, or suggestions on software will be accepted from the hardware team. Similarly, any feedback from the software team regarding hardware will also be accepted.

Reporting will be done based on team assignments and accomplishments, and each member is tasked with keeping track of their own personal progress. Weekly team meetings will be held to provide updates and discuss any issues or concerns that the team may have. The information from these meetings will be combined to create an overall progress report and in-progress presentation.

The first demonstration will be completed by the hardware team and will test the functionality of the amplifiers and the DACs through music output from the system. The second demonstration will be performed by the software team and will assess the performance of the LCD touchscreen and the software interface for the user. Lastly, the third demonstration will evaluate the overall completed system and will require the team to work together to complete the final design.

Schedule and Milestones:

Table 5 displays a tentative schedule that contains the different phases of the development of the system. This table contains the schedule for the progress done in ECE 492, as well as future tasks for ECE 493. The dates of the future tasks are subject to change dependent on project progression and other factors that affect the design.

TABLE 5: SCHEDULE AND MILESTONES FOR SENIOR DESIGN

Kapstone Raspberry Pi Radio
Team Members: Benjamin Ong, Danny Truong, Jamie Norvell, Zara Shoukat, Zayd Fazelyar,

Zuha Abdul Wakeel			
Task	**Progress**	**Start**	**End**
1. Proposal and Planning			
1.1 Research	100%	9/5/2021	10/22/2021
1.2 Project Requirements Specification	100%	9/5/2021	9/17/2021
1.3 Conceptual Design	100%	9/17/2021	9/24/2021
1.4 External Systems Diagram	100%	9/18/2021	9/24/2021
1.5 Sample Input/Output Diagram	100%	9/20/2021	10/1/2021
1.6 Functional Decomposition	100%	9/21/2021	10/1/2021
1.7 Physical and System Architecture	100%	9/22/2021	10/1/2021
1.8 Proposal	100%	10/1/2021	10/07/2021
1.9 Proposal Presentation	100%	10/08/2021	10/08/2021
1.10 Finalization of Components	100%	10/09/2021	10/15/2021
1.11 State Diagram and Flow Charts	100%	10/11/2021	10/22/2021
1.12 Background Foundations	100%	10/21/2021	11/5/2021
1.13 Design Review Presentation	100%	10/25/2021	11/5/2021
1.14 Final Design Document	75%	11/1/2021	11/12/2021
2. Early Prototyping and Prototyping			
2.1 Software Installation	100%	10/2/2021	10/08/2021
2.2 Purchasing and Testing Components	100%	10/4/2021	10/29/2021
2.3 Circuit Schematic Development	75%	10/18/2021	10/29/2021
2.4 Prototype Circuit Assembly	50%	11/05/2021	11/14/2021
2.5 Prototype Software Modules	50%	11/05/2021	11/14/2021
2.6 Prototype System Integration	0%	11/15/2021	11/19/2021
2.7 Prototype testing	0%	11/20/2021	11/22/2021

3. Hardware Development			
3.1 PCB Design	0%	01/17/2022	01/28/2022
3.2 PCB Assembly	0%	01/31/2022	02/05/2022
3.3 Analog Volume Control Setup and Test	0%	02/06/2022	02/12/2022
4. Software Development			
4.1 Request Module	0%	01/17/2022	01/28/2022
4.2 Display Module	0%	01/28/2022	02/05/2022
4.3 Music Module	0%	02/06/2022	02/13/2022
5. System Integration			
5.1 Request Functionality	0%	02/14/2022	02/18/2022
5.2 Display Functionality	0%	02/19/2022	02/23/2022
5.3 Music Functionality	0%	02/24/2022	02/28/2022
5.4 Digital Volume Control	0%	03/01/2022	03/05/2022
6. Testing			
6.1 Experiment 1	0%	03/06/2022	03/17/2022
6.2 Experiment 2	0%	03/17/2022	03/26/2022
6.3 Experiment 3	0%	03/28/2022	04/04/2022
6.4 Experiment 4	0%	04/08/2022	04/13/2022
6.5 Experiment 5	0%	04/19/2022	04/22/2022
7. Reporting			
7.1 Progress Report	0%	02/07/2022	02/13/2022
7.2 In-Progress Report	0%	03/21/2022	03/27/2022
7.3 Final Progress Report and Presentation	0%	04/25/2022	05/01/2022
8. Demos			
8.1 Hardware Demo	0%	02/13/2022	02/14/2022

8.2 Software Demo	0%	02/14/2022	02/15/2022
8.3 Integrated System Demo	0%	03/05/2022	03/06/2022

References

[1] N. Pino and O. Tambini, "The best wireless speaker 2021: our top picks from Sonos, Bose, Google, and more," Future Publishing Limited Quay House, 8 November 2021. [Online]. Available: https://www.techradar.com/in/news/best-speaker. [Accessed 9 November 2021].

[2] Adafruit Industries, "Pimoroni Pirate Radio - Pi Zero W with Headers Project Kit," Adafruit Industries, [Online]. Available: https://www.adafruit.com/product/3477. [Accessed 9 November 2021].

[3] Mozilla Foundation, "An overview of HTTP," Mozilla Foundation, 3 October 2021. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview. [Accessed 9 November 2021].

[4] Mozilla Foundation, "Port," Mozilla Foundation, 7 October 2021. [Online]. Available: https://developer.mozilla.org/en-US/docs/Glossary/Port. [Accessed 9 November 2021].

[5] The Samba Team, "About Samba," [Online]. Available: https://www.samba.org/samba/. [Accessed 10 November 2021].

[6] Micro Focus, "Understanding CIFS," Micro Focus, 2018. [Online]. Available: https://www.novell.com/documentation/open-enterprise-server-2018/file_cifs_lx/data/bld9p53.html. [Accessed 9 November 2021].

[7] G. N. Gibson, "6.1 Gain and Feedback," University of Connecticut, [Online]. Available: https://www.phys.uconn.edu/~gibson/Notes/Section6_1/Sec6_1.htm. [Accessed 9 November 2021].

[8] C. R. Nave, "Amplifiers," Georga State University, 2017. [Online]. Available: http://hyperphysics.phy-astr.gsu.edu/hbase/Audio/amp.html. [Accessed 9 November 2021].

[9] E. Gaalaas, "Class D Audio Amplifiers: What, Why, and How," *Analog Dialogue,* vol. 40, June 2006.

[10] D. Hawley, "How To Set Crossover Frequency For A Car Audio System," J. D. Power, 10 February 2021. [Online]. Available: https://www.jdpower.com/cars/shopping-guides/how-to-set-crossover-frequency-for-a-car-audio-system. [Accessed 9 November 2021].

[11] SparkFun Electronics, "Raspberry Pi LCD - 7" Touchscreen," SparkFun Electronics, [Online]. Available: https://www.sparkfun.com/products/13733. [Accessed 11 November 2021].

[12] S. Sirois, "How Do Touch Screens Work on Laptops and Tablets?," HP Inc., 12 December 2018. [Online]. Available: https://www.hp.com/us-en/shop/tech-takes/how-do-touch-screens-work. [Accessed 10 November 2021].

[13] Nelson-Miller, Inc., "Resistive vs Capacitive Touchscreens: Which Is More Common?," Nelson-Miller, Inc., 29 August 2019. [Online]. Available: https://www.nelson-miller.com/resistive-vs-capacitive-touchscreens-which-is-more-common/. [Accessed 10 November 2021].

[14] koppa, "Capacitive Touch Technology," GeeksforGeeks, 27 November 2019. [Online]. Available: https://www.geeksforgeeks.org/capacitive-touch-technology/. [Accessed 9 November 2021].

[15] Electrical Technology, "Digital to Analog Converter (DAC) – Types, Working & Applications," Electrical Technology, 2020. [Online]. Available: https://www.electricaltechnology.org/2020/04/digital-to-analog-converter-dac.html. [Accessed 10 November 2021].

[16] Headphonesty, "Understanding Sample Rate, Bit Depth, and Bit Rate," Headphonesty, 2 September 2021. [Online]. Available: https://www.headphonesty.com/2019/07/sample-rate-bit-depth-bit-rate/. [Accessed 10 November 2021].

[17] Texas Instruments Incorporated, "PCM5122," Texas Instruments Incorporated, 2021. [Online]. Available: https://www.ti.com/product/PCM5122. [Accessed 9 November 2021].

[18] D. Purves, G. Augustine and D. Fitzpatrick, "The Audible Spectrum," in *Neuroscience. 2nd edition.*, Sunderland, Massachussets: Sinauer Associates, 2021.

[19] H. Zhang, "AN-140: Basic Concepts of Linear Regulator and Switching Mode Power Supplies," Analog Devices, Inc., October 2013. [Online]. Available: https://www.analog.com/en/app-notes/an-140.html. [Accessed 10 November 2021].

[20] ROHM Co., Ltd., "Difference Between Linear Regulator and Switching Regulator," ROHM Co., Ltd., 2021. [Online]. Available: https://www.rohm.com/electronics-basics/dc-dc-converters/linear-vs-switching-regulators. [Accessed 10 November 2021].

[21] Adafruit Industries, "Adafruit I2S Stereo Decoder - UDA1334A Breakout," Adafruit Industries, [Online]. Available: https://www.adafruit.com/product/3678. [Accessed 13 November 2021].

[22] K. Rembor and lady ada, "Adafruit I2S Stereo Decoder - UDA1334A," Adafruit Industries, 21 November 2017. [Online]. Available: https://learn.adafruit.com/adafruit-i2s-stereo-decoder-uda1334a. [Accessed 13 November 2021].

[23] Texas Instruments Incorporated, "LM386," Texas Instruments Incorporated, 2021. [Online]. Available: https://www.ti.com/product/LM386. [Accessed 13 November 2021].

[24] J. Barnsley, "Mopidy-Iris," Stein Magnus Jodal, [Online]. Available: https://mopidy.com/ext/iris/. [Accessed 13 November 2021].

Appendix C: Setup Script

```
#!/bin/bash
# setup script
# run as super user
# make sure you are in the same directory as THIS script before running
it!

xset s noblank
xset s off
xset -dpms
# prevent Pi from sleeping in the middle of the install process
# because that will mess things up
# instructions from here https://pimylifeup.com/raspberry-pi-kiosk/

echo "starting install of PaPiYa Radio software!"

MY_HOME="/home/pi"
# because running this as super user will make it forget where ~ points to
# so I need to hardcode it here...

echo "updating packages!"
apt update
apt upgrade -y
# update any existing packages

echo "installing libasound!"
apt install libasound2-dev -y
# required for installing on 64-bit Pi OS
# otherwise the ALSA plugin for Mopidy won't install
# see here https://github.com/nihal111/J.A.R.V.I.S/issues/17

echo "installing Samba support!"
apt install samba-common smbclient samba-common-bin smbclient cifs-utils
-y
# install SMB server support (for our NAS)
# following
https://raspberrypi.stackexchange.com/questions/40974/access-network-samba
-share-from-pi-client

echo "installing pip3!"
apt install python3-pip -y
# install pip3 for installing Python packages

echo "installing xdotool!"
apt install xdotool -y
# for programmatically sending keystrokes
```

```
echo "installing unclutter!"
apt install unclutter -y
# for hiding mouse cursor

echo "setting up DAC!"
# setup DAC
# see here
https://www.hifiberry.com/docs/software/configuring-linux-3-18-x/
# see here https://www.hifiberry.com/docs/data-sheets/datasheet-dac2-pro/
# I'm trying to trick the Pi into thinking a Hifiberry board is attached
# because that way the Hifiberry drivers which are included in the kernel
will just... work
cp /boot/config.txt /boot/config.txt.bak
# make backups of the config files we are about to edit
sed -i 's/dtparam=audio=on/#dtparam=audio=on/g' /boot/config.txt
sed -i 's/dtoverlay=vc4-fkms-v3d/dtoverlay=vc4-fkms-v3d,audio=off/g'
/boot/config.txt
sed -i 's/dtoverlay=vc4-kms-v3d/dtoverlay=vc4-kms-v3d,audio=off/g'
/boot/config.txt
/bin/bash -c 'echo "force_eeprom_read=0" >> /boot/config.txt'
/bin/bash -c 'echo "dtoverlay=hifiberry-dacplus" >> /boot/config.txt'
# make those edits

cp misc/asound.conf /etc/asound.conf
# ALSA sound config file

echo "installing Mopidy!"
# see here https://docs.mopidy.com/en/latest/installation/

mkdir -p /usr/local/share/keyrings
wget -q -O /usr/local/share/keyrings/mopidy-archive-keyring.gpg
https://apt.mopidy.com/mopidy.gpg
wget -q -O /etc/apt/sources.list.d/mopidy.list
https://apt.mopidy.com/buster.list
apt update
apt install mopidy -y
ln -s /etc/asound.conf $MY_HOME/.asoundrc
adduser mopidy video
setfacl -R -m u:mopidy:rwx /media
# gives the mopidy user access to USB drives
# see here
https://askubuntu.com/questions/487527/give-specific-user-permission-to-wr
ite-to-a-folder-using-w-notation
mkdir /etc/mopidy
cp misc/mopidy.conf /etc/mopidy/mopidy.conf
```

```
# copying a premade mopidy.conf file instead of letting Mopidy generate
its own
# because there's a few lines I need to change and I dunno a good way to
let Mopidy run for the first time
# then shut it down
# then edit its config file automatically
# so have a premade one
cp /etc/pulse/default.pa /etc/pulse/default.pa.bak
sed -i 's/#load-module module-native-protocol-tcp/load-module
module-native-protocol-tcp auth-ip-acl=127.0.0.1/g' /etc/pulse/default.pa
# make Pulseaudio server listen on localhost
# see here https://docs.mopidy.com/en/latest/running/service/
systemctl enable mopidy
# enable mopidy to run as a service on startup

echo "installing Mopidy extensions!"
python3 -m pip install Mopidy-Iris
sh -c 'echo "mopidy ALL=NOPASSWD:
/usr/local/lib/python3.9/dist-packages/mopidy_iris/system.sh" >>
/etc/sudoers'
# WARNING! the above path to
/usr/local/lib/python3.9/dist-packages/mopidy_iris/system.sh might change
depending on your python install
# see here https://github.com/jaedb/Iris/wiki/Getting-started#installing
python3 -m pip install Mopidy-Local
# because the built in media scanning function is kind of awful?
python3 -m pip install Mopidy-Youtube
python3 -m pip install --upgrade youtube-dl
python3 -m pip install --upgrade ytmusicapi
# YouTube playback
python3 -m pip install Mopidy-ALSAMixer
# alsamixer extension, for hardware volume control
python3 -m pip install Mopidy-Raspberry-GPIO
usermod -a -G gpio mopidy
# GPIO control for Mopidy
python3 -m pip install Mopidy-TuneIn
# TuneIn internet radio
# TODO install other necessary extensions

ln -s /media/pi/ $MY_HOME/Music/USB
# create symlink to USB mount point
# so I can run
# sudo mopidyctl local scan
# OR have the Iris interface do it
# and check out the stuff in the drive
```

```
# see here
https://discourse.mopidy.com/t/can-i-have-multiple-sources-for-media-dir-i
n-mopidy-conf/4334

echo "installing other startup items"

/bin/bash -c 'echo "@reboot root pigpiod -t 0" >> /etc/crontab'
# pigpio daemon
# inserting the flags -t 0 to tell pigpio to use the PWM clock instead of
PCM clock
# because we need the PCM clock for audio
# see here https://forums.raspberrypi.com/viewtopic.php?t=213458
# see here https://github.com/joan2937/pigpio/issues/87
# see here
https://raspberrypi.stackexchange.com/questions/116669/slow-audio-playback
-with-pcm5102a-i2s-dac
# setting this thing up with systemd would be more elegant and there IS a
way to give it the required flags
# but this way looks more transparent

/bin/bash -c 'echo "@reboot root sleep 10; python3
$(pwd)/startup/uni_control.py" >> /etc/crontab'
# volume control program

chmod +x startup/kiosk.sh
/bin/bash -c 'echo "@$(pwd)/startup/kiosk.sh" >>
/etc/xdg/lxsession/LXDE-pi/autostart'
# set kiosk to run at boot
# see startup/kiosk.sh for more details

cp startup/shutdown.py /usr/local/bin
chmod +x /usr/local/bin/shutdown.py
cp startup/shutdown.sh /etc/init.d/
chmod +x /etc/init.d/shutdown.sh
update-rc.d shutdown.sh defaults
/etc/init.d/shutdown.sh start
# shutdown button
# see here
https://howchoo.com/g/mwnlytk3zmm/how-to-add-a-power-button-to-your-raspbe
rry-pi

echo "rebooting!"
reboot
```