



Mar. 13th, 2014

Andrew H. Rawicz
School of Engineering Science
Simon Fraser University
V5A 1S6

Re: ENSC 440 Design Specification --- SoundHub: Wireless Speaker Module

Dear Dr. Rawicz,

In regards to the course requirements of ENSC 305W/440W, enclosed to this letter is Arimus Audio's design specification for SoundHub: Wireless Speaker Module. We are designing and implementing a home wireless audio solution that enables music streaming over a Wi-Fi network to multiple speakers.

Our design specifications refer to a previous document "*Functional Specification for SoundHub: Wireless Speaker Module*", and discuss the design details to achieve the functional requirements [1]. This documentation provides the design justifications of SoundHub for its various stages of development, but mainly focuses on the proof of concept phase due to page limitations. In addition, test plan documentation is appended to the design specification and will be used as guidance when examining the model functionality.

Arimus Audio is a well balanced team comprised of five senior engineering students: Sherman Siu, Scott Malfesi, George Chang, Dongkai Miao, and David Yin. Their concentrations include an aggregation of computer engineering, electronics engineering, and engineering physics. We will be more than happy to discuss any additional questions or comments you may have regarding the functional specification. Please contact our CEO Sherman Siu at arimus.audio@gmail.com.

Sincerely,

A handwritten signature in blue ink, appearing to read 'Sherman Siu'.

Sherman Siu
CEO
Arimus Audio



Design Specification
WIRELESS SPEAKER MODULE

Project Team : Sherman Siu
Scott Malfesi
George Chang
Dongkai Miao
David Yin

Submitted to : Dr. Andrew Rawicz
Steve Whitmore
School of Engineering Science
Simon Fraser University

Issued date : March 13, 2014
Revision : 1.1

Abstract

This document details the design specifications for the entire SoundHub system as well as each of the individual components. The goal is to give the reader a detailed look at each portion of the proof-of-concept model from hardware to firmware/software and including specifications and justifications of the design approaches.

Overall the proof-of-concept model of SoundHub system includes three major components:

- Hardware development aimed to provide high quality audio transmission through the design and implementation of a customized Digital-to-Analog Converter (DAC) system
- Firmware development on the evaluation board to achieve audio streaming through a Wi-Fi network to multiple speaker systems
- Software development aimed to develop an Android application which is capable of controlling playback over a mobile device

Each of the three components has its own section describing the technical details which fulfill the functional requirements mentioned in the previous document “*Functional Specification for SoundHub: Wireless Speaker Module*” [1]. Justifications are provided for design approaches and the motives for choosing specific hardware components.

The final section of the document provides a set of preliminary test procedures to examine the model functionality of the proof-of-concept model. The test plan is divided into two parts: individual component testing and an evaluation of the integrated system. Detailed testing procedures for each part to be followed throughout the design and implementation are provided as an appendix in the end of the document.

Table of Contents

Abstract.....	ii
Table of Figures.....	vi
Glossary.....	vii
1.0 Introduction	1
1.1 Scope.....	1
1.2 Intended Audience.....	1
1.3 Project Background.....	1
1.4 Requirements Classification.....	2
2.0 System Specification and Justification	3
2.1 Use Case	3
2.2 Top Level Design	4
2.3 Functionality Justification	5
2.4 Safety and Sustainability.....	6
3.0 Electronic Design.....	7
3.1 SPDIF to I2S decoder circuit	8
3.1.1 Decoder system overview and theory of operation	8
3.1.2 Decoder Circuit Description	10
3.2 DAC circuit.....	12
3.2.1 DAC circuit system overview and theory of operation	12
3.2.2 DAC circuit description.....	14
3.2.2.1 I2S to Differential Stereo Current Output Stage	14
3.2.2.2 Amplification stage	15
3.3 Power Supply Circuit.....	16
3.4 System Status Circuit	17
3.5 Volume Knob.....	17
3.5.1 Theory of operation	17
3.5.2 Volume Knob circuit description.....	18
3.6 Line-in switch circuit	19
3.7 Parts Justification	19
4.0 Firmware Design	20



4.1 AllJoyn	20
4.1.1 Bus Attachment.....	21
4.1.2 Discovery & Advertising	21
4.1.3 Streaming.....	23
4.1.4 Synchronization.....	24
4.2 Audio Playback.....	24
4.3 Volume Adjustment	25
4.4 Status Lights	25
4.5 Joining the Network.....	26
4.6 RoomFlow	27
5.0 Software Design	28
5.1 Design Justification	28
5.2 Connection Protocol Utilization through Wi-Fi.....	28
5.2.1 Handle Sink Lost.....	29
5.3 Graphic User Interface	29
5.3.1 Playback Control	30
5.3.2 Volume Adjustments.....	30
5.3.3 Sink In Proximity.....	30
5.3.4 Refresh List of Sinks	30
6.0 Enclosure Design	31
6.1 Shape.....	31
6.1.1 Exterior Controls	31
6.1.2 Interior	32
7.0 Test Plan.....	33
7.1 Individual Component Test	33
7.1.1 Hardware	33
7.1.2 Firmware	33
7.1.3 Software.....	34
7.2 Integration Tests	35
8.0 Conclusions	36
9.0 References	37
Appendix A: Electronic Schematics.....	40



A.1 SPDIF Decoder Schematic.....	40
A.2 DAC Schematic.....	41
Appendix B: PCB Board Layout	42
B.1 SPDIF Decoder Layout.....	42
B.2 DAC Layout.....	43
Appendix C: Hardware Supplementary Section.....	44
C.1 DIR9001 pin layout.....	44
C.2 DIR9001 Pin Description	45
C.3 Pin Settings for Mode of Operation of DIR9001	46
C.4 PCM1794A Pin Layout.....	47
C.5 Pin Description for PCM1794A	48
C.6 Functional Block Diagram	49
C.7 Relevant Technical Specification	50
C.8 Pin settings for output format	50
C.9 Galvanic isolation.....	51
Appendix D: Firmware Flowcharts.....	52
D.1 AllJoyn Advertisement Process Flowchart.....	52
D.2 AllJoyn Discovery Process Flowchart.....	53
D.3 AllJoyn Single Source to Multi Sink.....	54
D.4 Flow Chart for Volume Thread	55
D.5 Flowchart for sending RSSI value in RoomFlow	56
D.6 Flowchart for receiving RSSI value in RoomFlow	57
D.7 Flowchart Handling of Sink Lost	58
Appendix E: Detailed Test Plans.....	59
E.1 Hardware Test Plans	59
E.2 Firmware Test Plans	64
E.3 Software Test cases.....	74
E.4. Integration Test Cases.....	80

Table of Figures

Figure 1: Artist’s SoundHub Rendering..... 2

Figure 2: Use Case Diagram for the SoundHub System 3

Figure 3: Block Diagram of the Wireless Speaker Adapter 5

Figure 4: Block Diagram of the Custom Audio Processing Hardware 7

Figure 5: Hardware System Block Diagram..... 8

Figure 6: Wandboard: TOSLINK Jack 9

Figure 7: Decoder Block Diagram..... 9

Figure 8: Composition of a 24 bit I2S Signal..... 10

Figure 9: Decoder Circuit Schematic..... 10

Figure 10: DAC Block Diagram 12

Figure 11: Unbalanced System 13

Figure 12: Balanced System 13

Figure 13: DAC System Schematic 14

Figure 14: DAC Amplifier Stage Schematic 15

Figure 15: Power Supply Schematic..... 16

Figure 16: Status LED Schematic..... 17

Figure 17: Rotary Encoder Schematic 18

Figure 18a: Bode Plot, 18b: Transient Response of the RC Low Pass Filter for the Rotary Encoder 18

Figure 19: 3PDT Switch Design and Function 19

Figure 20: AllJoyn Framework Stack 20

Figure 21: Sample Bus Attachment..... 21

Figure 22: AllJoyn Discover with Common Well-known Name 21

Figure 23: AllJoyn Discovery with Unique Well-known Name..... 22

Figure 24: AllJoyn Sink State Machines..... 23

Figure 25: Definition of Quadrature Encoding given by the Rotary Encoder 25

Figure 26: "AllJoyn" Steps for Device Communication 28

Figure 27: Artist's Rendering of the Main Playback Screen. Image based on Apollo App [28] 29

Figure 28: Front and Back views of the SoundHub..... 31

Figure 29: Volume Knob [29] 32

Glossary

3PDT	3-Pole Double Throw, a switch type with 9 pins
AC	Alternating Current
Alljoyn	Open source framework which enable wireless streaming through Wi-Fi network
Android	A science fiction robot with human appearance; also a mobile operating system developed and open sourced by Google
AP	Access Point
ARM	A British Company that specializes in developing computer instruction architecture sets
CPU	Central Processing Unit
DAC	Digital Analog Converter
FCC	Federal Communications Commission
FIFO	First In First Out
GPIO	General Purpose Input/Output
I2S	Integrated Interchip Sound, a standard serial bus used for digital audio communication
IC	Integrated circuitry
IEEE	Institute of Electrical and Electronics Engineers
iOS	A Cisco router operating system; also an embedded operating system created by Apple to run on their phones and some music players
LED	light Emitting Diode
Musydra	An open source android music streamer application with built in Alljoyn framework
OSI	Open Systems Interconnection
PCB	Printed Circuit Board
PCM	Pulse Code Modulation
RC	Resister Capacitor, This is referred to the resister capacitor based low pass filer design
RCA	Radio Corporation of America, a type of coaxial connection used to transfer audio and video signals.
RoHS	Restriction of Hazardous Substances Directive
RSSI	Received Signal Strength Index
Sink	An endpoint that receives audio sent from an application using the “Alljoyn” framework
SMT	Surface Mount Technology



SNR	Signal to Noise Ratio
Source	An endpoint that sends audio signals to the network destined to a point of interest. In this document, this is a mobile device or computer
SPDIF	Sony/Philips Digital Interface Format
THD+N	Total Harmonic Distortion
TOSLINK	Toshiba Link, A standard optical cable typically used to transfer audio signals
UDP	User Datagram Protocol
Wandboard	An evaluation hardware chosen for the purposes of developing the SoundHub as a proof-of-concept model
WI-FI	A popular technology that allows an electronic device to exchange data or connect to the internet wirelessly using radio waves
WPS	Wi-Fi protected setup

1.0 Introduction

1.1 Scope

This documentation describes the technical details of the wireless speaker adapter system named SoundHub, including the design approaches and possible modifications of the proof-of concept-model, the marketable revision, and the final consumer product. These technical details will be used as reference throughout the design and implementation phase and will be referred to in future documents.

This design specification will outline the following details:

- The SoundHub product and its features
- Overview of the system specification and justification
- Technical details of system design to support the functional requirements
- A set of preliminary test plans to examine the model functionality

1.2 Intended Audience

The intended users of this document include all members of team Arimus Audio and potential stakeholders of the project. The team leads can use this document as a guide to measure the overall progress while the developers can refer this when implementing the design. Stakeholders can use this document to gain high level view on how the device will be implemented.

1.3 Project Background

The SoundHub is the next revolutionary step in wireless speaker technology. Designed to be sleek and discreet, it allows existing wired speakers to attain wireless freedom by streaming music from other devices to it. Furthermore by utilizing modern Wi-Fi protocols, multiple speakers connected to separate SoundHub devices can be streamed to at the same time. With SoundHub, rooms or entire homes can have access to streamed audio content, with full control through your handheld devices or computers.

The goal is to produce a slim and discreet speaker attachment that will allow music streaming through Wi-Fi, while maintaining a price point much lower than existing competition. This product will allow users to gain the benefits of wireless streaming without having to upgrade their entire audio system. Figure 1 shows an artist's rendering of how the SoundHub will look along with an audio system.

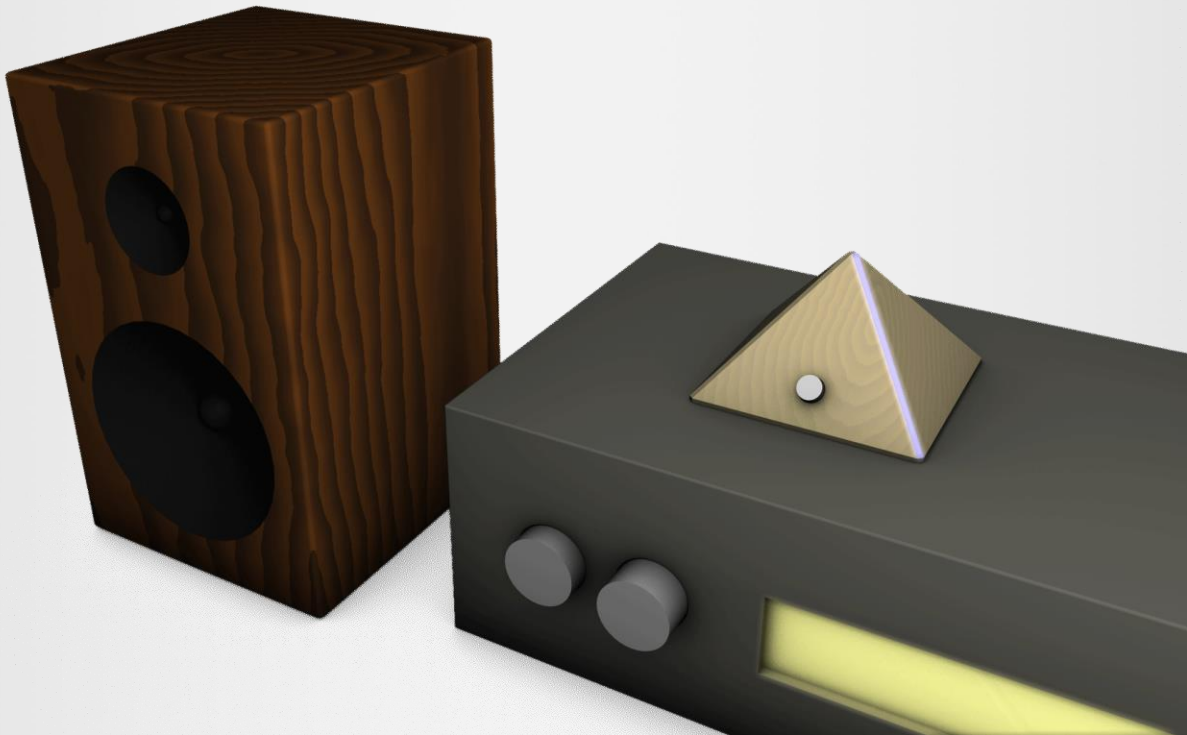


Figure 1: Artist's SoundHub Rendering

With user-friendliness in mind, the SoundHub requires minimal setup and has intuitive user controls. The setup requires the SoundHub to be connected to the user's speakers and will use a Wi-Fi network. Once connected to the network the mobile device will be able to find and wirelessly stream music to the SoundHub.

1.4 Requirements Classification

The requirements referenced throughout this document are taken from the functional specification document [1]. The following convention has been used to represent the functional requirement:

[Req x.y.z P]

where x.y.z indicates the requirement section and number and P number represents priority level. The priority level is divided into three levels

P1: Requirement that is high priority and is essential to the proof-of-concept model

P2: Requirement that is a moderate priority and is aimed for the marketable revision

P3: Requirement that is low priority and is applicable to the final consumer product

2.0 System Specification and Justification

2.1 Use Case

Our product consists of two systems for the user to interact with the application on their own mobile devices and the SoundHub device itself. Figure 2 shows a use case diagram which defines how a user may interact with the system. A use case diagram shows the functionality from the perspective of the user. Associations between the user and the use case are shown by solid lines with arrowheads indicating the initial invocation. Dotted lines indicate extended associations, in our case these are usually between the two systems which are separated by bounding boxes.

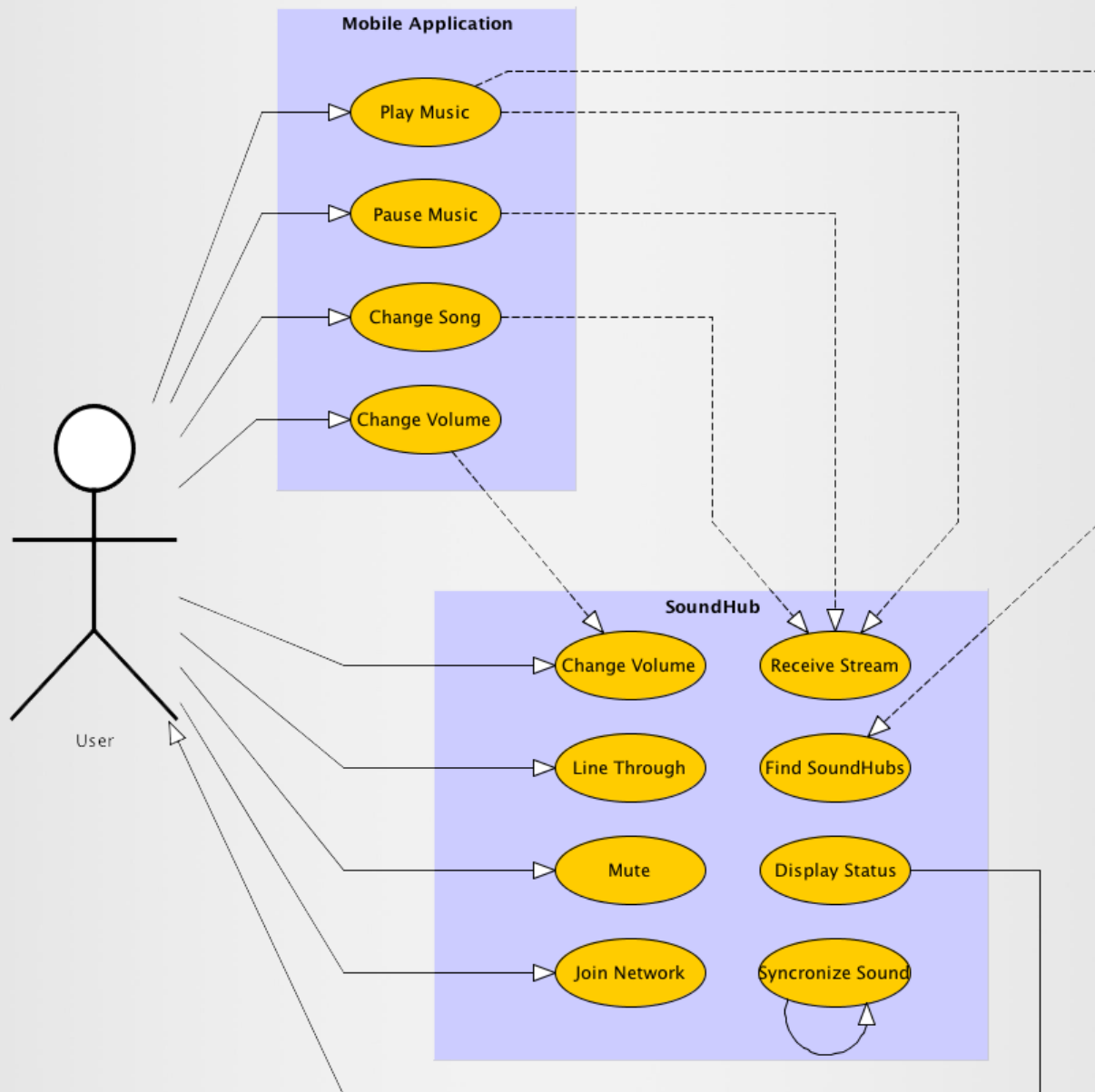


Figure 2: Use Case Diagram for the SoundHub System

The mobile application is the interface between the user and the SoundHub. Once the mobile and the SoundHub devices are connected to the Wi-Fi network, the application identifies the SoundHub device and is able to remotely send commands through Wi-Fi network such as play/pause, songs selection, and volume control. The user can also directly interact with the SoundHub device. A volume knob is provided on the front panel, so the volume adjustment can be achieved through physical interaction. On the back panel of the device, two 3.5mm jacks are provided for speaker connections and/or another wired input device, along with a toggle switch to select output from line-in, the wireless receiver and mute. A status light-emitting diode (LED) array is provided on the bevelled edge of the of the pyramidal which indicates the current status of the SoundHub device.

2.2 Top Level Design

The system design consists of three major sections:

- Hardware development aimed to provide high quality audio transmission through the design and implementation of a customized DAC system
- Firmware development on the evaluation board to achieve audio streaming through a Wi-Fi network to multiple speaker systems
- Software development aimed to develop an Android application which is capable of controlling playback over a mobile device

The hardware component of the SoundHub consists of an ARM Cortex-A9 central processing unit (CPU) on the development board named Wandboard [2], which is running an embedded Linux kernel and custom firmware. The hardware peripherals include a Wi-Fi module, a volume knob, status lights and customized circuitry to transform the streaming data into a signal which can be played through speakers. Within the circuitry a Sony/Philips Digital Interface Format (SPDIF) decoder is used to decode the digital audio stream. Following this, a DAC is used to convert the decoded digital signal into an analog signal. This analog signal is ready to be outputted through the 3.5mm line-out jack after a signal amplification stage. In addition a line-in feature is added which allows wired connection between a source device and the SoundHub. This way the users are able to promptly switch between wireless streaming and their original wired configuration.

A high level block diagram of wireless speaker adapter is shown below in Figure 3:

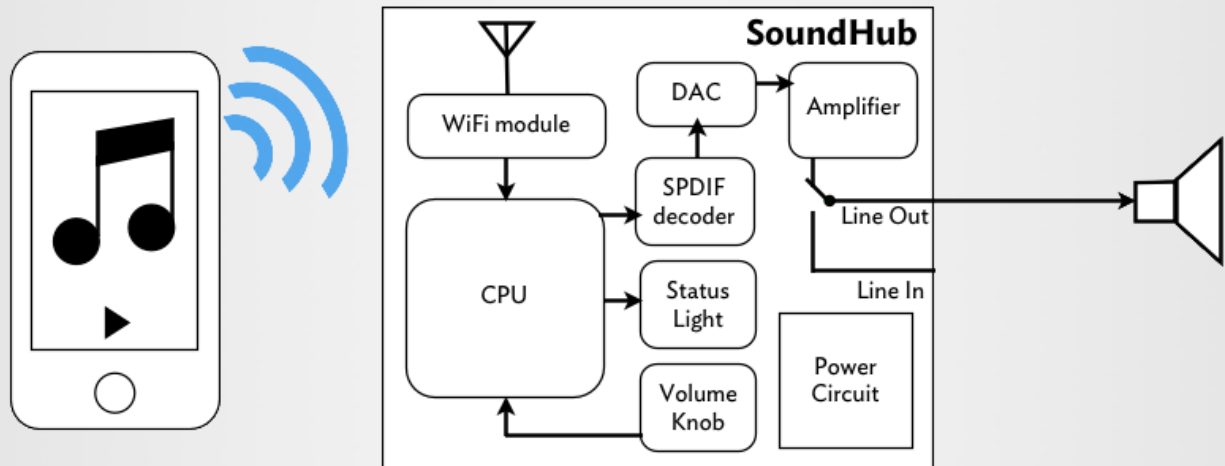


Figure 3: Block Diagram of the Wireless Speaker Adapter

For the firmware and software components of the system, an open source framework named “AllJoyn” [3] was adopted and modified to achieve the core functionalities of the SoundHub device. This framework was chosen because of the wide range of features and support in terms of both hardware and software with different language bindings and also varies development platform support. The mobile devices are able to discover the SoundHub when on the same Wi-Fi network using the auto-generated unique identifier. Once a valid connection between the SoundHubs and the mobile device is created in the discovery phase, exchanges requests to stream or adjust parameters can be made. For streaming music, the mobile application acts as the “Source” where the music files are streamed and the SoundHubs play the “Sink” role to receive the signal and pass them to the hardware component for decoding and playing.

2.3 Functionality Justification

Functionality was chosen based on the user demand for a wireless speaker system that is capable of streaming music to multiple devices to create a whole house audio experience. The goal for the SoundHub is to be a low cost, sustainable, quality music device. With this in mind, the functionality and design decisions were made to balance each of the three objectives.

The choice of adopting open source materials allows interested individuals in the open source community to develop more functions and features into the SoundHub. Also, by implementing a customized DAC system, the SoundHub can be future proofed by supporting higher specifications than consumer grade audio formats. This allows the SoundHub to be kept by a user for longer periods without having to pre-emptively replace it. To be even more sustainable the SoundHub incorporates



exclusively recyclable materials for its enclosure.

All the functionality has been tailored to make SoundHub a household electronic that will last. By making it versatile and able to be incorporated into any existing sound system, it discourages wasteful purchases of entirely new systems simply for the wireless capabilities. Its simple design and ease of use will make it an essential addition to home audio systems.

2.4 Safety and Sustainability

At Arimus Audio we are focused on making sure that our product is both sustainable and safe for users. We designed our product with the idealized goal of a cradle to cradle design in mind. This is built into the very purpose of our product which can be combined with a home's pre-existing audio system to allow users to forego purchases of entirely new speakers and promote speaker re-use. However, given the current nature of consumer electronics, this was not possible; still we focused on making sure our product to have as little negative impacts on the environment and peoples' lives as possible.

The enclosure is to be made primarily of wood to maximize recycled material and reduce the environmental impact. Electronic components are to be easily separable from the enclosure to allow simple recycling procedures. In addition, Restriction of Hazardous Substances (RoHS) [4] compliant materials have been selected to ensure there are no heavy metals such as lead found. By following this requirement throughout the project, this sets the standard for a lead-free device in future prototyping and production phases.

In regards the safety, the enclosure is to be designed without harmful sharp edges and corners while retaining an overall pyramidal shape. Components inside the enclosure are secured with standoffs against the pyramid shell to prevent movement and damage to parts. Screws will be placed on the opening plate preventing unintended access to electronic components. The SoundHub will also follow the guidelines of Federal Communications Commission (FCC) section 15 Class B Digital Device [5]. Section 15 Class B digital Device indicates that this, "digital device that is marketed for use in a residential environment notwithstanding use in commercial, business and industrial environments." [5]. Some examples of such devices includes, but are not limited to, personal computers, calculators, and similar electronics devices that are marketed for use by the general public. We intend on supporting wireless standards such as Institute of Electrical and Electronics Engineers (IEEE) 802.11n-2009 [6] and also more recent IEEE standards to promote the longevity of SoundHub as we develop the product into its further stages.

Future SoundHub roadmaps will be more focused on its safety and sustainability. Plans for certifications and assessment various safety and environmental standards are being discussed. Strategies for the SoundHub's components and enclosure's redesign to target energy efficiency and cost reduction are also going to be formulated during the prototyping development cycle.

3.0 Electronic Design

The following section presents the detailed design of the customized hardware DAC system. The main objective of implementing this customized DAC system is to provide a superior listening experience over the all-purpose audio codec on the Wandboard. The functionality of this DAC system includes receiving SPDIF signals sent from the CPU of the Wandboard which are decoded into Integrated Interchip Sound (I2S) format with corresponding system clock, and then the DAC circuitry converts digital I2S data into analog stereo audio signal to the 3.5mm audio jack. A high level block diagram is shown as Figure 4 below.

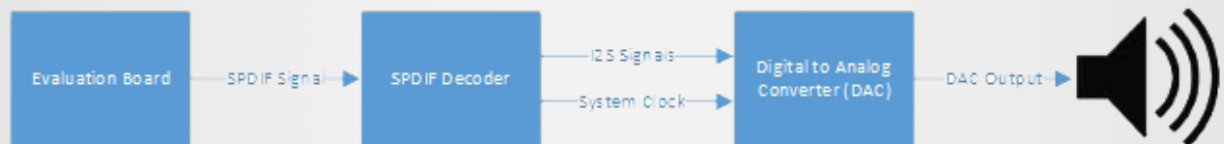


Figure 4: Block Diagram of the Custom Audio Processing Hardware

The realization of this DAC system will be constructed on a custom order printed circuit board (PCB) using mostly surface mount technology (SMT) components (eg. ICs, capacitors, inductors, and resistors), and few through hole components (eg. LEDs, voltage regulators, and audio jacks). The schematic and layout of the circuit is created using CadSoft Eagle PCB design software [7].

The customized DAC system consists of two main modules to manipulate the audio signal, two support modules, power and status indication, and two external modules, volume knob and line-in switch, for audio control via input from the user.

1. SPDIF to I2S decoder circuit*
2. I2S to stereo audio DAC circuit
3. Power supply circuit (providing +3.3V, +5V and $\pm 12V$ voltage)
4. System status circuit
5. Volume knob circuit
6. Line-in switch circuit

Figure 5 shown below is the hardware system block diagram of the SoundHub system:

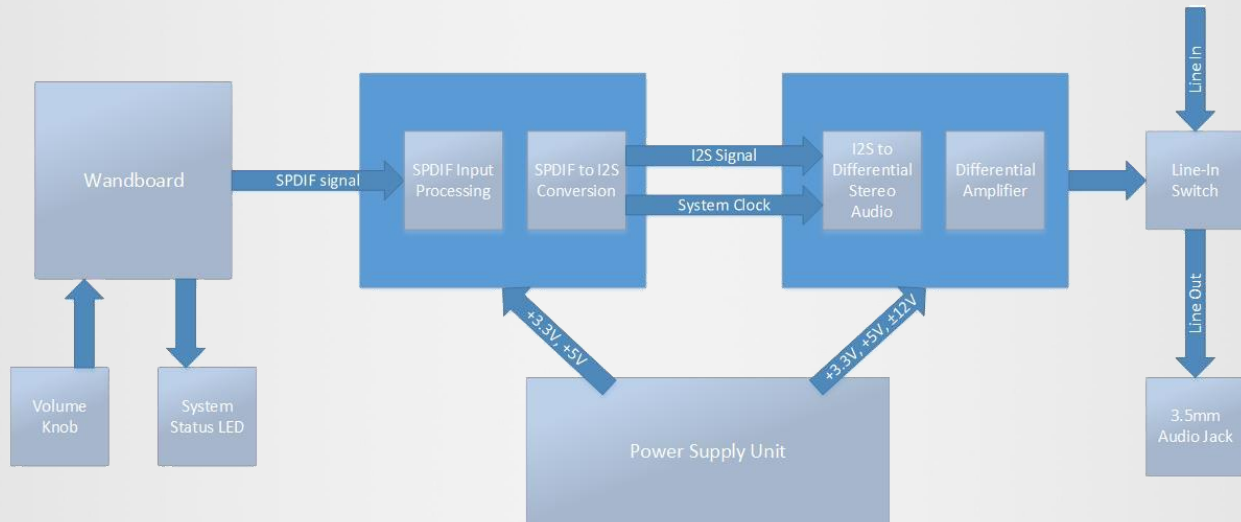


Figure 5: Hardware System Block Diagram

Detailed explanations of each module will be provided along with the relevant technical descriptions of each the components in the following sub-sections. The recommended designs from the datasheets of the decoder DIR9001 [8], and DAC PCM1794A [9] along with Pavouk’s schematic [10] were referenced when designing the customized DAC system. For the complete system circuitry of our design, please refer to the schematics and layout files in Appendix A and B.

*Note that the design specification outlines the proof-of-concept model. In this model, the audio data transmits via the SPDIF protocol. In the future prototype we DAC system is built together with the CPU, the entire decoder stage can be removed as the I2S signals can be retrieved directly from the CPU. This approach is currently unavailable in the proof-of-concept model due to the physical limitations of the development board.

3.1 SPDIF to I2S decoder circuit

3.1.1 Decoder system overview and theory of operation

In the proof-of-concept model, the digital audio data is streamed to the development board and the CPU will decode the received packets and encode the music data into SPDIF format, outputting through the TOSLINK port.

Figure 6 shows the TOSLINK (Toshiba Link) connection on the development board.



Figure 6: Wandboard: TOSLINK Jack

The SPDIF signal contains both the audio serial data signals and its corresponding clocks. Since the DAC stage will require a clock source with low phase jitter and noise for optimal performance, the audio serial data and clocks should be separated. Satisfying requirement [Req 3.2.3 - P1], a decoder circuit is built to convert SPDIF signals into I2S signals. The block diagram of the decoder circuit is shown as Figure 7 below.

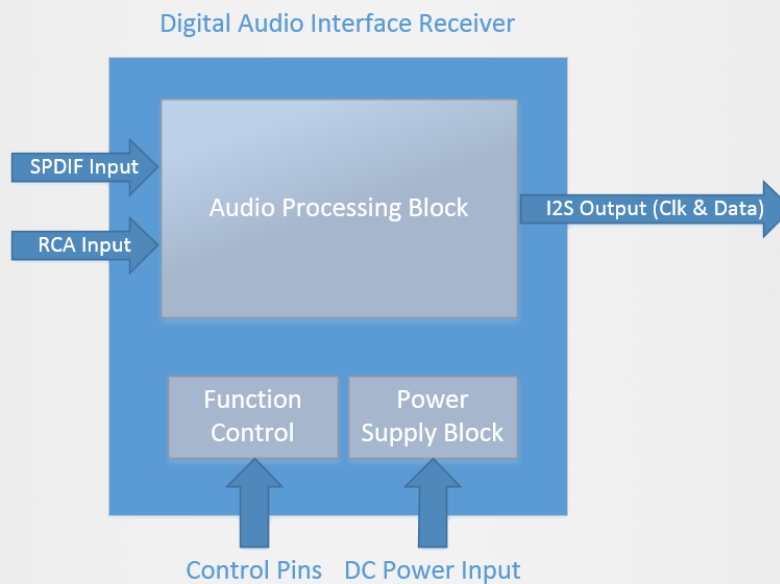


Figure 7: Decoder Block Diagram

The I2S protocol is an electrical serial bus interface standard used for communicating pulse-code modulated (PCM) audio data between integrated circuits in an electronic device [11]. The I2S protocol utilize three buses in transmission: the first bus for left and right channel selection, the second bus for bit clock and the third bus for audio serial data. Figure 8 shows the composition of a 24 bit I2S signal.

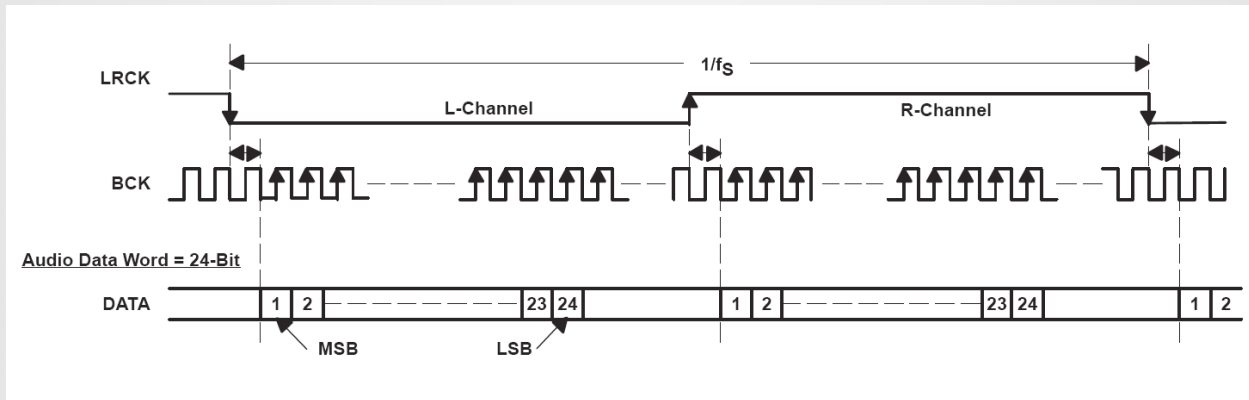


Figure 8: Composition of a 24 bit I2S Signal

3.1.2 Decoder Circuit Description

Figure 9 shows the entire decoder circuit.

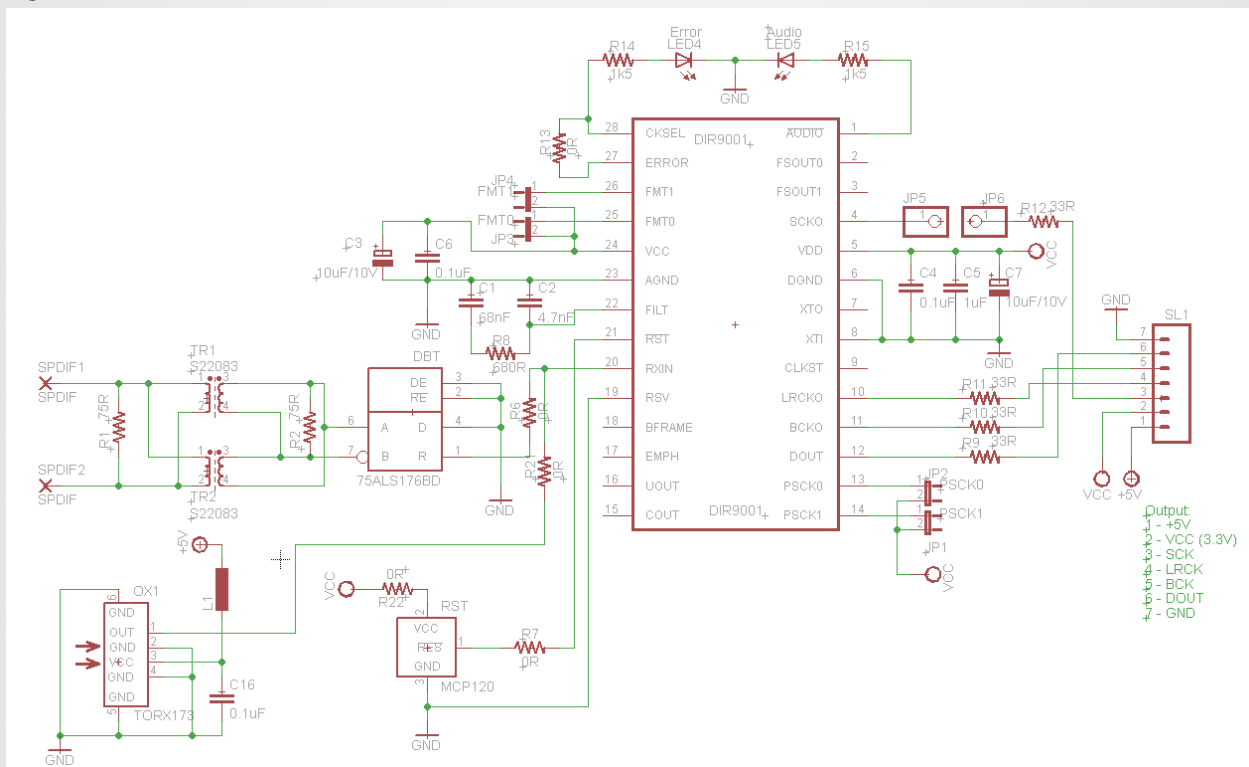


Figure 9: Decoder Circuit Schematic

The decoder chip used in this stage, the DIR9001, is renowned for its high performance in audio applications such as mobile audio devices and high-end PC sound cards. The DIR9001 chip offers sample

word length of up to 24 bits at sampling frequency of 96kHz [Req 3.2.10-P2] The decoder chip will convert SPDIF signals into I2S signal and its corresponding system clock. Please refer to Appendix C.1 for the decoder pin layout table.

Note the input pin RXIN accepts both optical TOSLINK connection and electrical Radio Corporation of America (RCA) connection. However for the RCA connection, the DIR9001 needs an external differential to single converter. In addition, a transformer is required to establish galvanic isolation between the input source and DAC system. (For specific information about galvanic isolation, please refer to Appendix C9.) Thus, S22083 [12] transformer and SN75178B differential bus transceiver [13] are selected for SPDIF input signal processing and the output from SN75178B feeds into the DIR9001 decoder chip. With revision A, the zero ohm resistor R6 is disconnected, leaving only the TOSLINK signal connected to the DIR9001 input pin. The supervisory circuit IC MCP130 [14] is needed in this circuit for controlling the reset pin of the DIR9001 decoder. Error LED4 will light up when SPDIF signal is not present or when an error with the input signal detected. Audio LED5 will light up when the input SPDIF signal does not contain raw PCM audio data.

Table 1: Control pins settings of DIR9001

Pins	Logic level
CKSEL*	Low
ERROR	Low
PSCK0	High
PSCK1	High
FMT0	High
FMT1	High

CKSEL pin and ERROR pin are set to logic low in order to recover clocks from SPDIF input. FMT0 and FMT1 are set to logic high to I2S output format. PSCK0 and PSCK1 are set to logic high for faster system clock.

*Note that the CKSEL pin can also be tied to the ERROR pin. This sets the DIR9001 to AUTO mode which the chip automatically selects the clock source based on the output of the ERROR pin. For specific pins for controlling the mode of operation refer to Appendix C. 3.

After this stage, the retrieved I2S signal and its system clock will be transmitted to the DAC circuit for digital to analog audio conversion.

3.2 DAC circuit

3.2.1 DAC circuit system overview and theory of operation

After the decoder stage, the decoded I2S signals with its corresponding system clock signal are produced. In the DAC circuit stage, the DAC circuit will convert the received I2S signal into analog stereo audio format to end user [Req 3.2.6-P1]. The corresponding system clock is required for operating digital interpolation filters and data segmentation. After digital to analog conversion, the generated stereo audio signal will be amplified through operational amplifier stage for optimal audio experience. A simple block diagram illustrating this stage is shown below as Figure 10.

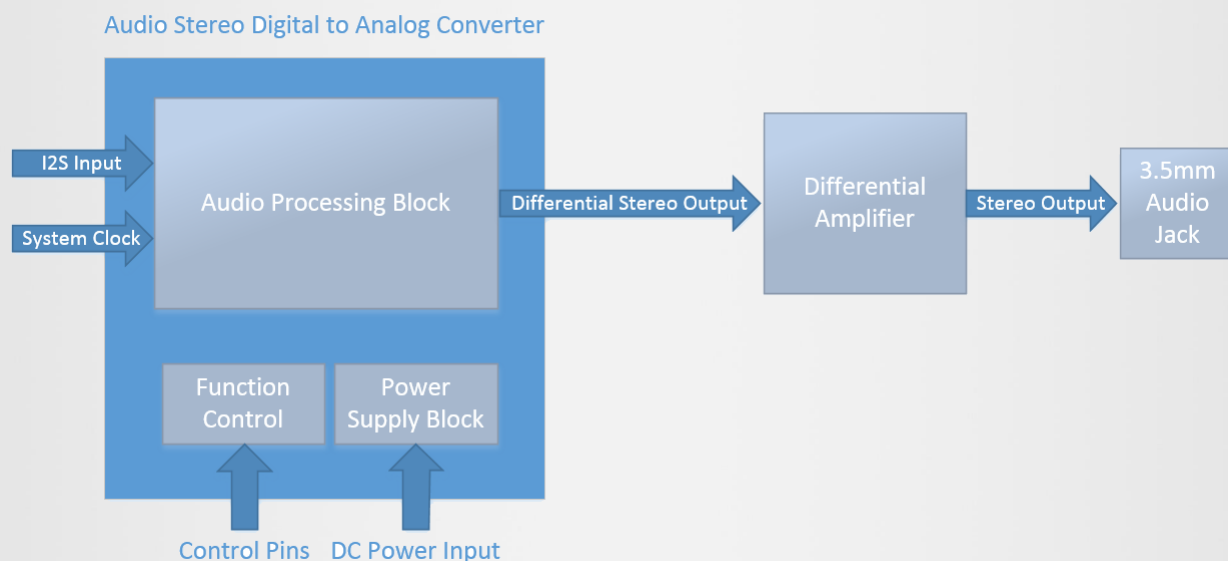


Figure 10: DAC Block Diagram

Since noise and distortion generated in the amplification stage are not negligible, the design of the amplifier circuit is critical to achieving high Signal to Noise Ratio (SNR) of more than 115dB [Req 3.2.4 - P1] and low Total Harmonic Distortion (THD+N) of less than -95dB [Req 3.2.5 - P1]. Therefore, a high performance differential amplifier is selected for the amplifier stage.

The following Figure 11 demonstrates an unbalanced system.

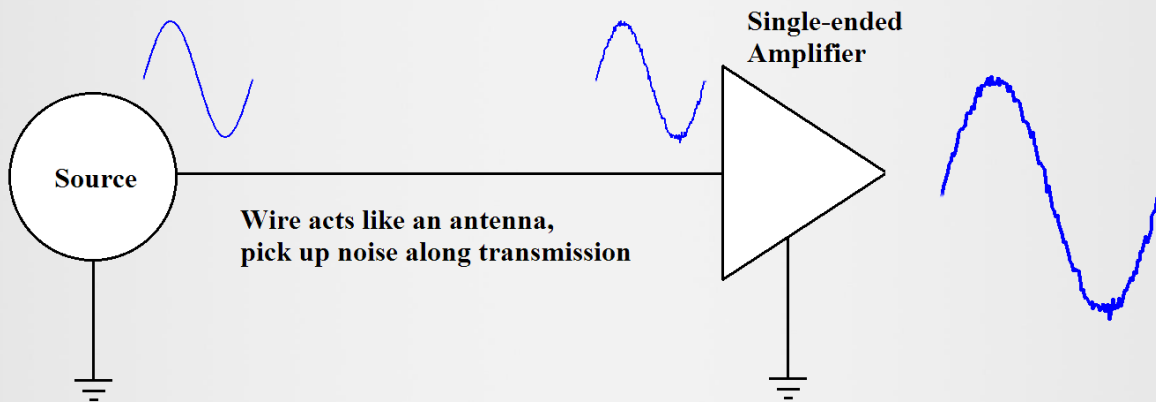


Figure 11: Unbalanced System

An unbalanced signal is the scenario when only one wire carries the signal. Even if the signal from a source is clean, the wire acts as an antenna and background noise will be added along the wire. Thus, when the signal reaches the amplifier, the noise coupled with the signal will also be amplified, producing a poor output signal.

The following Figure 12 demonstrates a balanced system.

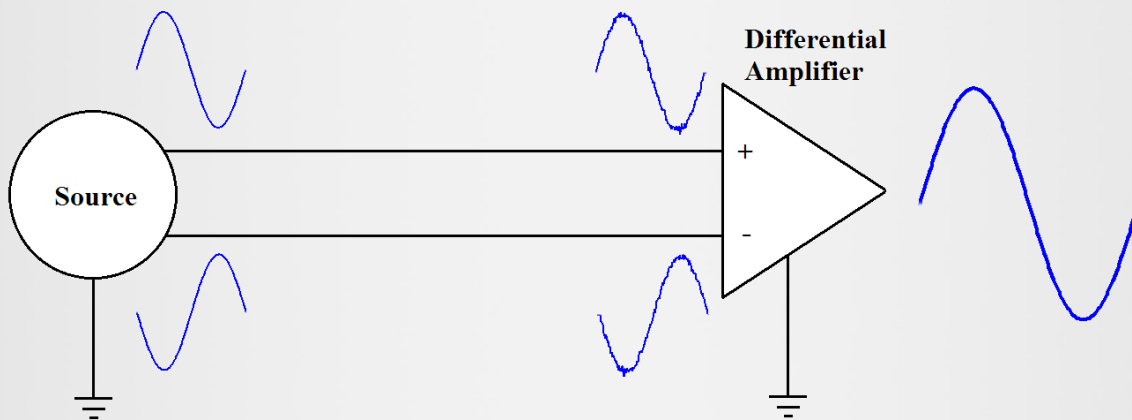


Figure 12: Balanced System

A balanced signal means there are two wires carrying the signal with a 180 degree of phase difference. As the signals are transmitted through wires, they pick up the same background noise as the unbalanced signal. Although these two signals are 180 degrees out of phase to each other, the coupled noise are in

phase. When they are processed by the differential amplifier, the noise will be subtracted out with the following equation shown below.

$$[(A + B) - (-A + B)] = 2A$$

3.2.2 DAC circuit description

The DAC circuit consists of two sub-stages: I2S to differential stereo output stage and the amplification stage. The Figure 13 below shows the entire DAC circuit.

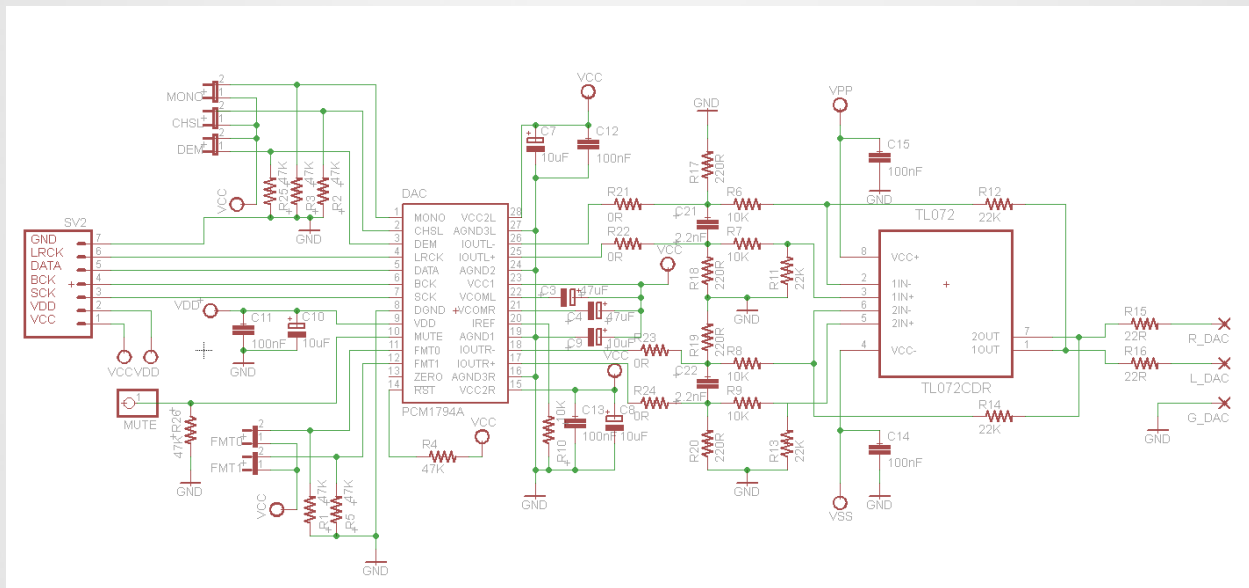


Figure 13: DAC System Schematic

3.2.2.1 I2S to Differential Stereo Current Output Stage

Refer to Appendix C.7 for relevant specification information on the PCM1794A DAC IC.

The DAC chip used in this stage is PCM1794A, a high performance DAC supporting up to 24 bit 192kHz audio signal that converts I2S signals into differential stereo audio format. In the proof-of-concept model the DAC chip is operating at 24 bit 96kHz as the signal output from the DIR9001 decoder is in this format. For future prototypes where the decoder is no longer required, the DAC chip will be set to operate at 24bit 192kHz for optimal performance. For a detailed functional block diagram of PCM1794A, please refer to Appendix C.6.

Table 2: Pin Settings for the PCM1794A DAC

Pin Name	Logic Level
Mono	L
CHSL	L
FMT0	L
FMT1	L

In order to convert the I2S signal to stereo differential out, mode of operation needs to be set by the control pins, either 5V high, or ground as low. All four output format control pins are set to L for I2S stereo output. For specific pin settings please refer to Appendix C.8. The DEM pin enables de-emphasis such that the high frequency signal is attenuated when the pin is set to logic high. The MUTE pin enables hardware mute such that outputs are transitioned to the bipolar zero level in -0.5 dB. The MUTE pin is connected to General Purpose Input/Output (GPIO) from the development board so that the DAC chip can be electrically muted by the user from software control. Note that in the DAC schematic, many control pins are tied to VCC with a jumper and a pull down resistor to ground. When these jumpers are not installed, control pins are set to logic low by the pull-down resistor.

3.2.2.2 Amplification stage

The Figure 15 below details the amplifier stage of the DAC system.

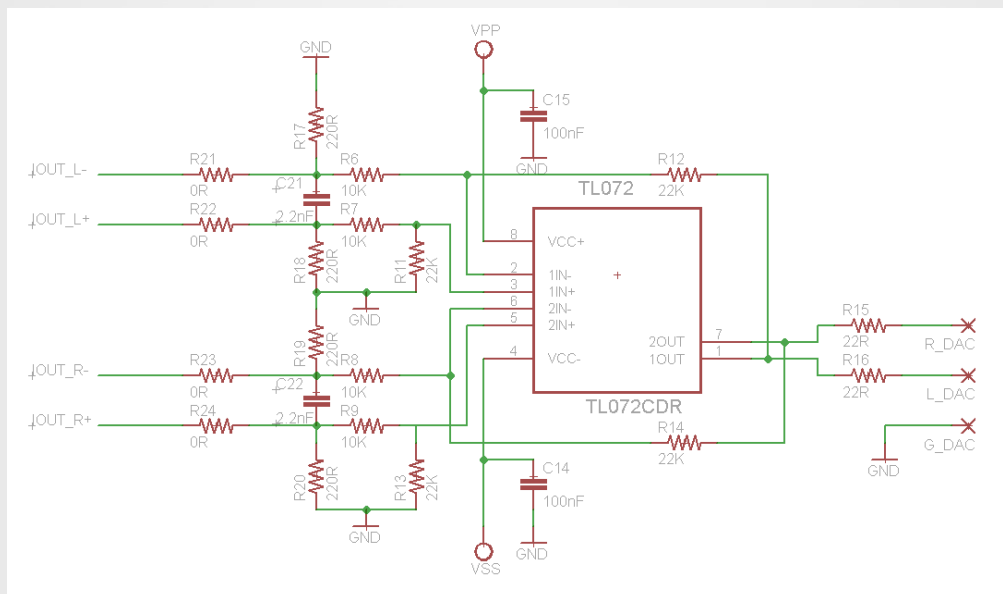


Figure 14: DAC Amplifier Stage Schematic

After the DAC chip converts I2S signal to current differential output for left and right audio channel, the amplifier stage takes the differential input and converts it into single stereo audio format which is connected to a line-out 3.5mm audio jack. C21 and C22 are used for filtering the high frequency components out of the audio band. As explained in the theory of operation section, by employing differential amplification design, higher SNR can be achieved. Passive SMT components will allow flexibility when tuning values to improve performance in future revisions.

Note that the TL072 [15] differential amplifier is not the optimal chip for this specific type of application. The differential amplifier LT1028 recommended by Texas Instrument’s reference design will provide an even higher SNR, but at a higher input voltage level of $\pm 15V$. Considering the current time constraints, building an Alternating to Direct Current (AC-DC) power supply is not within the scope of this project. Currently the DAC system is supplied by an ATX PSU which does not include a $\pm 15V$ rail. Therefore, the TL072 is chosen as an alternative as it has been tested to work at $\pm 12V$.

3.3 Power Supply Circuit

Figure 15 below shows the schematic of the power circuit.

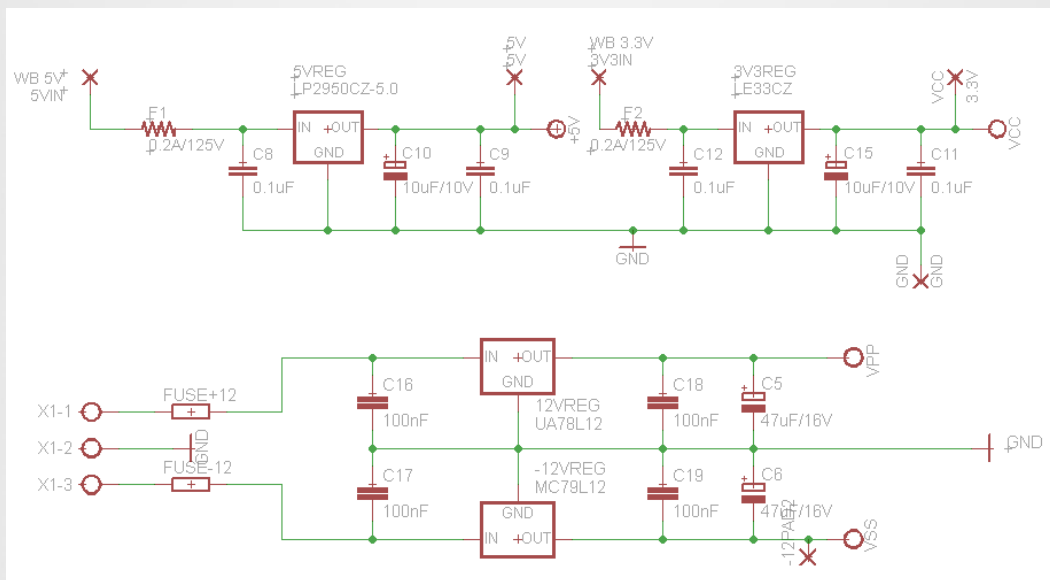


Figure 15: Power Supply Schematic

In this proof-of-concept model, the modules of the DAC circuitry are powered by an external ATX PSU. An ATX PSU conveniently provides all the voltage lines required: +3.3V, +5V, +12V, -12V, and also ensures [Req 3.2.1 - P1], [Req 3.2.2 - P1], [Req 3.2.9 - P2] are met. Voltage regulators: +5V LP2950 [16], +3.3V LE33CZ [17], +12V UA78L12 [18], and -12V MC79L12 [19] are additionally used to ensure stabilized DC voltage outputs. Tantalum and thin film capacitors are used in parallel for power supply rail

decoupling, effectively reducing or eliminating noise that may affect the rest of the circuitry. Note that in the PCB layout design, thicker traces (0.032 mils) are used for the power lines to allow for greater heat dissipation.

3.4 System Status Circuit

The only source of user feedback in the SoundHub is the LED status lights as required in [Req 3.2.16 - P2]. The circuit for shown in Figure 12 shows the 3 LEDs powered through the GPIO lines with 1.5kΩ resistors to ensure the current through the LEDs is within spec. LED parts are still currently being sourced, but an alternative value for the resistors can be easily substituted in both documentation and PCB.

Figure 16 below shows the schematic for the status lights of the system.

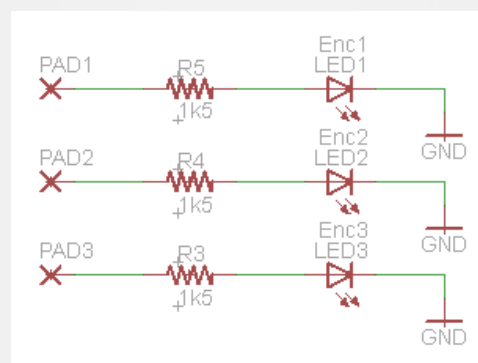


Figure 16: Status LED Schematic

To direct the light out of the enclosure we are using a light pipe made from acrylic [20]. The shape of the light pipe spreads the light out along the bevel of the enclosure and uses internal reflection to ensure that most of the light reaches the outside. Additional information on the algorithms used to control LED status lights can be found in Section 4.4.

3.5 Volume Knob

3.5.1 Theory of operation

The volume of an individual SoundHub can be adjusted via the mobile application or by volume knob on the enclosure as required by [Req 3.2.18 - P2]. A volume knob as opposed to push buttons was decided due its ability to easily enable fine and coarse adjustments. The knob is attached to a rotary encoder to encode the rotation signal into a digital signal. We choose to use a rotary encoder which has a precision of 24 pulses per revolution and gives a two wire quadrature signal [21].

3.5.2 Volume Knob circuit description

The circuit to implement the rotary encoder EVE-KC2F2024B [21] is shown in Figure 17.

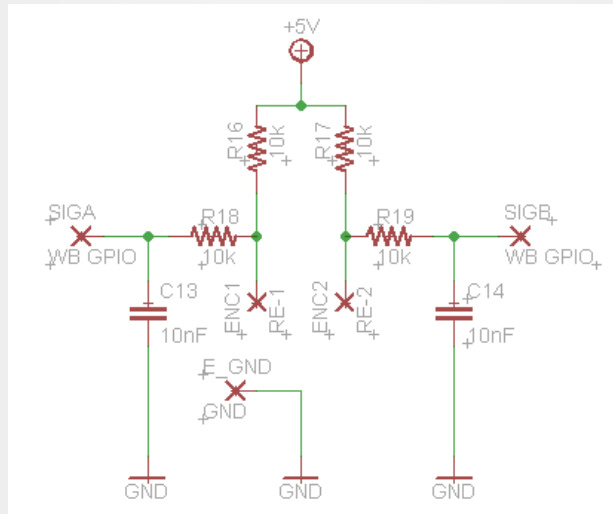


Figure 17: Rotary Encoder Schematic

Resistors R16 and R17 are to ensure that the power and ground are not shorted by the rotary encoder, and the combination of R16, R18 and C13 and R17, R19 and C14 create a basic RC low pass filter. This filter works to debounce the encoder so that only one interrupt will be fired per pulse. The cut off frequency of the filter can be calculated from the resistance of 20kΩ and capacitance of 0.1μF to be 80Hz. The bode plot and transient response of the filter are shown in Figure 14(a) and 14(b).

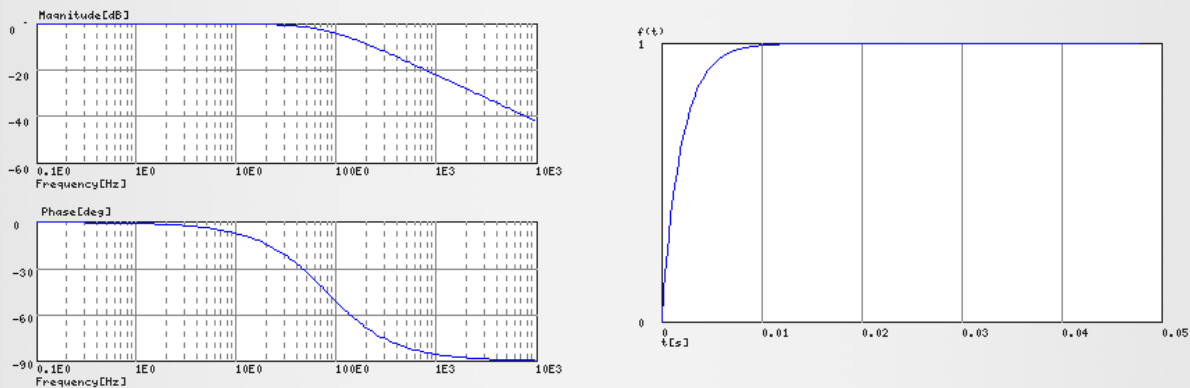





Figure 18a: Bode Plot, 18b: Transient Response of the RC Low Pass Filter for the Rotary Encoder

3.6 Line-in switch circuit

The customized DAC has both line-in and line-out 3.5mm jacks as specified by [Req 3.2.6 - P1] and [Req 3.2.7 - P1]. The line-out 3.5mm jack is connected to either the line-in (essentially creating a loop-back circuit), or the outputs of the customized DAC circuitry. Each set of signals include the left channel, right channel, and common ground. This line-in functionality is achieved via a 3 Pole Double Throw (3PDT) switch 1003P1TB1M1QEH [23]. See below for a diagram of the switch’s functionality.

SWITCH FUNCTION		
POS. 1	POS. 2	POS. 3
		
ON	NONE	ON
2-3, 5-6, 8-9	N/A	2-1, 5-4, 8-7

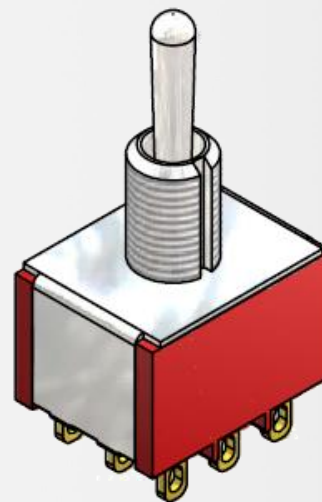


Figure 19: 3PDT Switch Design and Function

If the 3 PDT switch is in position 1, a connection forms between the line-in and line-out jacks. If in position 3, the line-out and DAC outputs are connected together. Position two connects neither of the two to the line-out, and functions as a mute for the system. Multi-paired shielded cable is used to connect the pads on the PCB to the 3PDT switch to prevent any external noise from effecting onto the output signals.

3.7 Parts Justification

Parts for this proof-of-concept model have been carefully chosen to be suited for audio applications. Parts are also chosen to be RoHS compliant as stated in [Req 3.0.5 - P2]. Components are chosen to meet a wide operational temperature and storage temperature range [Req 3.2.11 - P2], [Req 3.2.12 - P2]. Though this current model is only required to receive an audio stream of 16 bits at 44.1kHz [Req 3.1.7 - P1], a high performance decoder and DAC are used to ensure 24 bit audio at 96kHz is achievable for when this is upgraded in the prototype and marketable revisions.

4.0 Firmware Design

The firmware section will describe the operations of code running on the Wandboard. The subsections break down the operation into the modular features including the AllJoyn framework, volume adjustment, status lights, joining a network and RoomFlow. The AllJoyn framework section is relevant for both the firmware and software sections as it has to be implemented on both sides but will only be explained in this section.

4.1 AllJoyn

AllJoyn is an open source framework and a series of core services provided to promote the “Internet of Things” [3]. AllJoyn was developed by Qualcomm and announced in the 2011 Mobile World Congress with the goal of achieving cross platform interoperability. This framework and Software Development Kit (SDK) was chosen due to its wide range of support in terms of hardware, software, and also various development platforms. The Audio Service Framework from AllJoyn was then the ideal core feature to be adopted by Arimus Audio. The reasoning behind this decision was due to the framework’s ability to perform cross platform peer to peer communication with ease and also providing support for various programming languages. On top of that, AllJoyn also handles the lower levels of the Open Systems Interconnection (OSI) model stacks and connectivity, eliminating the need to code for numerous protocols used in networking. With the open source nature of AllJoyn acting as the backbone for SoundHub, developers and enthusiasts can customize their own features and functions, further promoting the longevity of SoundHub. As shown in Figure 20, any related applications sits on top of the AllJoyn framework. Below the AllJoyn framework are the AllJoyn Routing nodes and AllJoyn Client Libraries. With the Routing nodes and libraries, developers can build of their applications on top to include various network configurations and connectivity.

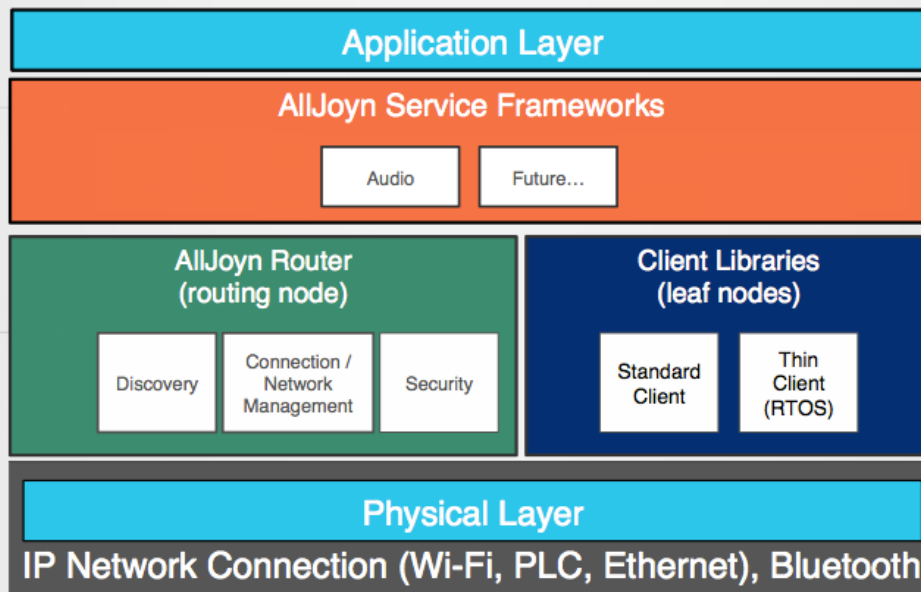


Figure 20: AllJoyn Framework Stack

4.1.1 Bus Attachment

The AllJoyn bus attachment is the first connection made between applications. It is the first step in connecting and setting up the AllJoyn connections. The BusAttachment call will initiate the first steps in attempting to reach other application that also has initiated a BusAttachment call. The bus attachment will handle all the lower layer stacks of the OSI model like the Physical and Data Link layers to setup the required network and connections. With this BusAttachment, applications are then able to discover, advertise, and connect with each other. The BusAttachment call will automatically generate a unique name that gets assigned to the application that are used for advertising and identification. Figure 21 demonstrates a sample bus attachment scenario.



Figure 21: Sample Bus Attachment

4.1.2 Discovery & Advertising

In order for AllJoyn applications to discover each other, an unique identifier is required per application for discovery to work. This is based on a user defined, "well-known name," prefix. This prefix is the ID for each individual application to avoid conflicts during the discovery and session establishment phase. An example as shown in Figure 21, App3, being in a discovery state knows the existence of App1 and App2, but App3 is confused about which application it needs to connect to, due to identical well-known name.

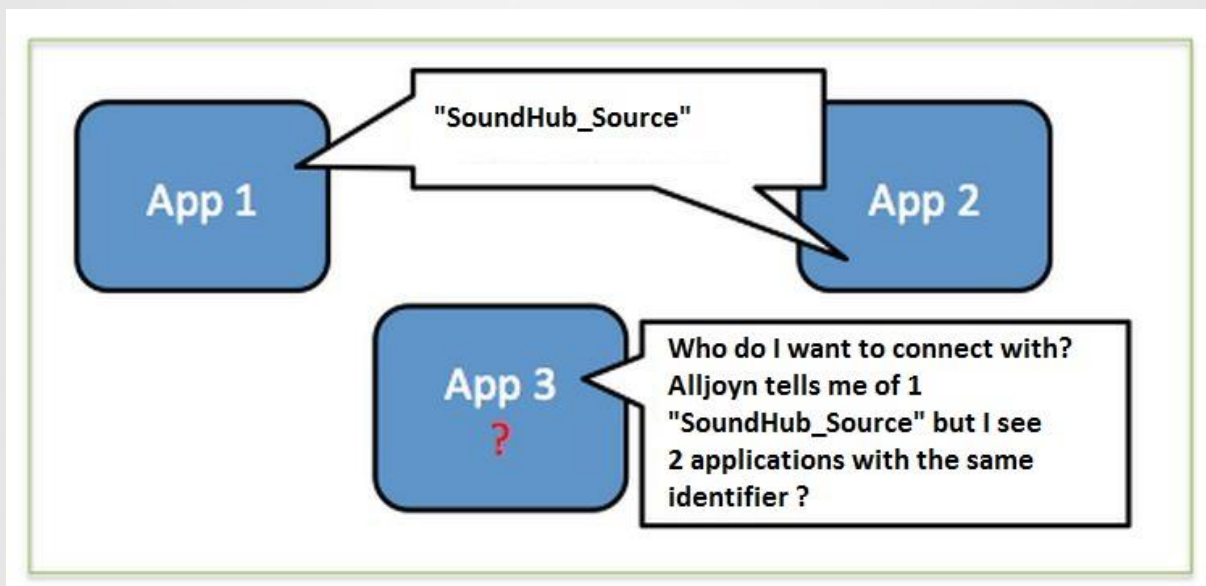


Figure 22: AllJoyn Discover with Common Well-known Name

Therefore if each individual application can be uniquely identified, applications can search for each other with ease as shown in Figure.22.

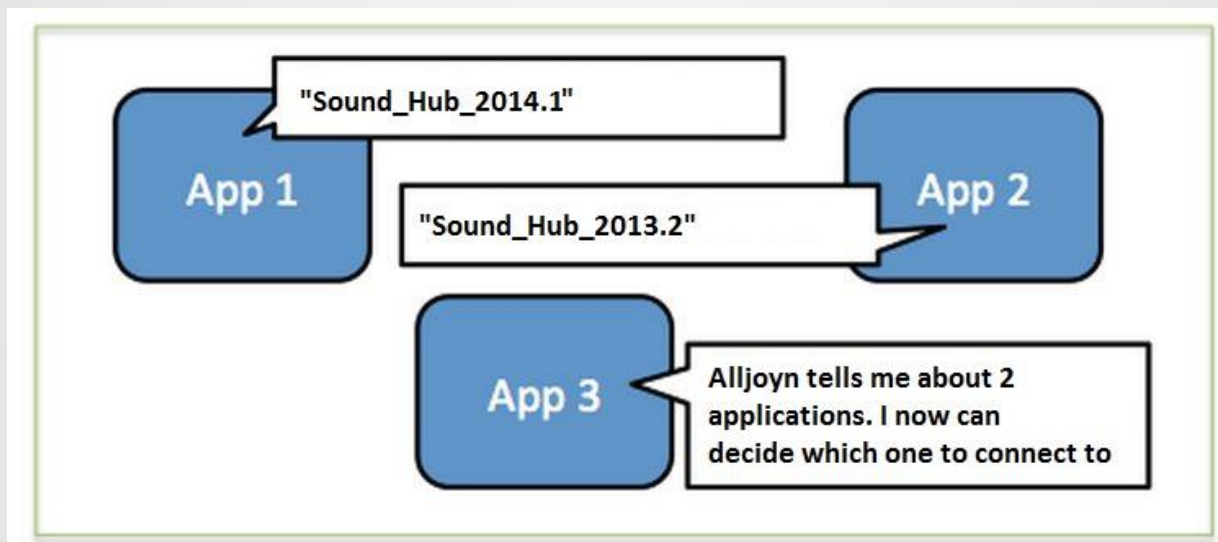


Figure 23: AllJoyn Discovery with Unique Well-known Name

As mentioned in section 4.1.1, a call to BusAttachment will auto-generate a unique name that is used to advertise the application. In relation to the unique name, a well-known name is a user chosen alias or representative of the unique name. Since the unique name contains no useful information about the application, a well-known name can therefore help by providing meaning to an application. This is similar to how an URL is associated to an IP, and that the URL would provide more meaning to the domain or website than the associated IP.

To start Advertising the existence of a session, invoke a call to AdvertiseName to notify nearby applications using the well-known name for discovery. The Advertisement of this well-known name will be published to the AllJoyn Routing node to be broadcasted over the User Datagram Protocol (UDP) port 9956 as a standard. Refer to Appendix D.1 Alljoyn Advertisement Process Flowchart for the detailed flowchart of the AllJoyn advertisement process.

To start the Discovery of a session, invoke a call to FindAdvertiseName <prefix of interest> via the BusListener object to search for any broadcasted message over UDP port 9956 with the specified “prefix name of interest.” By registering the BusLisenter object, we are able to collect the relevant information of any nearby application like its well-known name, transport format and the prefix. When a prefix match has been identified, the BusListener callbacks with FoundAdvertiseName. From this point, a decision can be made to call JoinSession to connect or perform other operations with the information. Refer to Appendix D.2 for the detailed flowchart of the AllJoyn discovery process. With AllJoyn’s discovery and advertise mechanism, [Req 3.1.6 - P1] will be satisfied.

4.1.3 Streaming

Once a valid well known name has been discovered and connected via the JoinSession call through the discovery phase, exchanges of requests can be made between the applications. For streaming music the terms “Sink” and “Source” will be used to denote the difference between the music streaming provider, the “Source,” and the music stream receivers, the “Sinks”. The current implementation of the SoundHub will only allow one instance of a Source, therefore all the Sinks with the same well known name will be connected to the only one Source in a bidirectional “one to many topology”.

The AllJoyn Sink adopts a First In First Out (FIFO) styled control. As shown in Figure 23 below

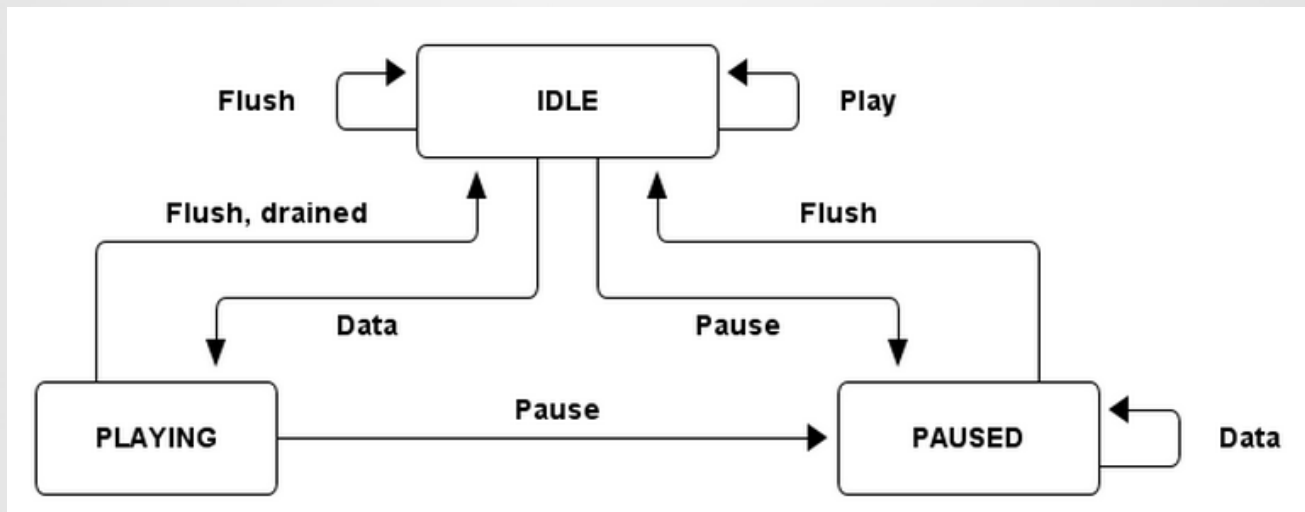


Figure 24: AllJoyn Sink State Machines

the sink’s state machine mechanism allows the exchange of audio data as a request call is made for the application’s FIFO size. The Source then will send the audio data corresponding to the FIFO size of the sink for it to play. Once the “low water mark” has been reached, the Sink can then send the FifoPositionChanged signal, indicating the amount of memory left in the buffer, to the Source to send more data. The FifoPositionChanged is the difference between the FifoSize and FifoPosition. FifoSize is the total size of an empty buffer, and the FifoPosition is the amount of memory left in a FIFO buffer after playing out the data. This allows the Source to send precisely the amount of audio data to the Sink without dropping any excess data. A flush call can also be made by the Sink to drop all of the data within its buffers for incoming data, renewing the FifoPosition. Refer to Appendix D.3 Alljoyn Single Source to Multi-Sink for a detailed flow chart of a single Source to multi Sink scenario.

With the FIFO mechanism, requirement [Req 3.1.9-P2] will be satisfied by ensuring that the Sink buffer will always be filled with new audio data from the Source and that the audio data sent will never exceed the buffer size of the Sink via the FifoPositionChanged signal. These will guarantee a smooth audio playback without significant loss or lack of packets under good signal and network conditions.

4.1.4 Synchronization

Audio synchronization is a major part in designing an audio system that supports one-to-many streaming. The main cause of playback delays between multiple Sinks and a Source is the latency of the network. The distance between each Sink and the Source may also vary, increasing the severity of delay in relationship to the distance from the signal. Therefore, a reliable mechanism is required to ensure that the audio packets being transmitted through a wireless medium are being synchronized across to all devices. To achieve this, a dedicated timing mechanism was implemented.

Access timing within AllJoyn is achieved by implementing the Stream.Clock interface. The Stream.Clock interface will provide the timing necessary to achieve synchronization. The Stream.Clock utilizes the Unix Epoch time as the master clock for time stamping packets. The Unix Epoch is the number in seconds since Thursday, January 1st 1970 Coordinated Universal Time (UTC), which is the standard time in most Unix based systems [24]. The outbound packets from the source will be timestamped using the Unix Epoch into the future to also compensate for the network and processing delays to the Sink. The Sink will use the compensated timestamp from the inbound packets, and compare it to the Unix Epoch for a time difference. Since all the existing Sink sessions has the same Unix Epoch time value, the delay timing of an same audio packets can be calculated and synchronized. Any data that are not rendered in time by the the Sink is dropped. This method was chosen based on the reliability of an accurate standard clock and the ease of computation to minimize strains on hardware. A more complex synchronization and time adjusting algorithm will be implemented in the production version of the SoundHub.

To adjust for the timing playback for each Sink in the network, a call to AllJoyn's AdjustTime will calculate the time difference, either positive or negative will be made to the timing of the playback. A sample procedure is as shown below for the clock synchronization process.

1. Record current Unix Epoch time (t_0) in nanoseconds
2. Call SetTime(t_0) and wait for the method reply.
3. Record the packet's timestamp as (t_1) in nanoseconds
4. Call AdjustTime($(t_1 - t_0) / 2$).

The maximum error of the clock skew is the value passed to AdjustTime, $(t_1 - t_0) / 2$.

4.2 Audio Playback

Audio playback is handled through using the advanced Linux sound architecture (ALSA) driver. This driver abstracts the audio hardware, handles sound mixing and some decoding [25]. As our hardware is a PCM device, a pipe is set up to the ALSA driver over which the PCM data is sent. As requirement [Req 3.1.6 - P1] describes, for the proof-of-concept model we support streams of 2 channel 16 bit audio sampled at 44.1kHz. This baseline was set because this is the standard for CD quality music. Currently the decoding of lossy audio formats such as mp3 and wma are handled on the mobile device prior to streaming. There are plans of expanding this requirement to support higher bit rates and decoding of other audio formats on the SoundHub in the marketable revision.

4.3 Volume Adjustment

The digital signals from the rotary encoder are attached to the general purpose input / output (GPIO) pins of our board. The rotary encoder outputs a two wire quadrature signal as defined by the data sheet can be seen in Figure 24 [21].

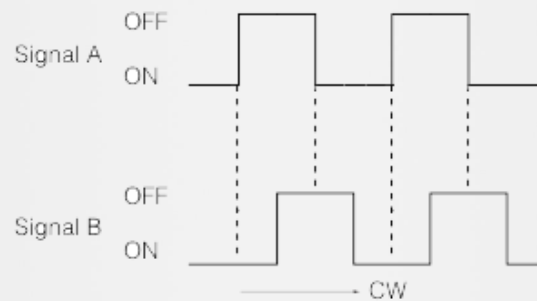


Figure 25: Definition of Quadrature Encoding given by the Rotary Encoder

The quadrature encoding does not give an absolute position of the volume knob but will tell the relative movement which is ideal for a knob which adjusts a variable controllable by another source. The algorithm for reading these signals is to set up an interrupt on the rising edge of signal A, when this interrupt happens read signal B. If signal B is high at the interrupt then the direction is counter clockwise and the volume should be decremented. A flowchart of this algorithm is shown in appendix D.4. In practice there will be a pool of threads which are implementing this algorithm, and a mutex lock which ensures that only one will respond to the interrupt at a time. This way another thread will be waiting for the next pulse while the system audio is being adjusted which may take some time.

4.4 Status Lights

Without the LED status lights described in [Req 3.2.6 - P1] and [Req 3.2.16 - P2], the user would have no way of knowing what state the SoundHub is currently in. To show all of the states using three different colours of LED's: green, blue and red. Using different patterns of these colours we are able to indicate the status without to resort to distracting blinking patterns when user attention is not required. The possible states shown are described in table 1 along with the corresponding colour of LED's.

Table 3: Table of SoundHub statuses and corresponding LED pattern

LED Pattern	Status Description
No lights	Not connected to power
Blinking Red	Started up but is not connected to a wireless network
Blue	Connected to the network but not streaming
Green	Currently Streaming
Red	Muted
Green and Red	Currently Streaming and muted
Green and Blue	Other SoundHubs in the network are streaming but not this one (due to RoomFlow)

In order to get timely notifications of network changes and streaming status we have registered callbacks with the AllJoyn BusAttachment and StreamObject to update the LEDs.

4.5 Joining the Network

Network support is built into the linux operating system, which was one of the main reasons for building on top of an operating system. Each wireless access point (AP) credentials are automatically stored in the `/etc/network/interfaces` file. This way upon startup a known AP will be automatically joined as required in [Req 3.1.4 - P2]. Similarly support for wifi protected setup (WPS) [22] is included in the `wpa_supplicant` package [26]. To use it you need to run the command `wpa_cli wps_pbc`. To implement [Req 3.1.2 - P2] we have attached a push button to the GPIO pins on the board and set up an interrupt to call this command on the rising edge of this GPIO pin.

4.6 RoomFlow

One differentiating features of the SoundHub is its RoomFlow feature which is planned to be implemented for the marketable revision. This is where the music will follow the user as they move around and only play out of the closest SoundHubs as described in [Req 3.1.12 - P3]. We are detecting proximity using the received signal strength indication (RSSI) of the packets coming from the mobile device. This value is a measure of the signal strength energy and can take on values between 0 and 127 [6]. While the antenna geometry will be a factor the signal strength should fall off at approximately $1/r^2$ where r is the distance from the source. This makes the RSSI a decent measure of proximity when only caring about the order of which SoundHubs are closest to the source.

Unfortunately getting the RSSI value of the packets coming out of the mobile device is non-trivial on a Wi-Fi network. This is because these packets are directed to the router, and under normal operating conditions the Wi-Fi chip will only report network traffic coming from the router directed to the SoundHub. To get around this we are putting the Wi-Fi chip into monitor mode; in this mode the Wi-Fi chip will report all traffic. As we only care about traffic coming from the mobile device we can filter out all of the rest of the traffic by sender's IP address.

Every second the SoundHub will broadcast average of the last 20 received RSSI values. This will filter out any noise or sudden changes in the signal. The SoundHubs are all able to share their RSSI values by connecting over the AllJoyn Bus and sending out a multicast signal. The flowchart for the algorithm to send RSSI updates can be found in appendix D.5. Each SoundHub will be keeping an internal list of all of the other SoundHubs in the network and its last received RSSI value. When a new RSSI value is received this list will be updated and resorted. We are implementing a two threshold hysteresis algorithm for deciding when a SoundHub will play. If a SoundHub's RSSI value is within the higher threshold of the top value in the list then that SoundHub will play. Once a SoundHub is playing then it must be out of the lower threshold before it will stop playing. This is so a SoundHub will not continuously switch on and off when the RSSI value is on the edge. The other AllJoyn Bus will also notify the other SoundHubs in the network when one of the SoundHubs leaves the network through the SessionLost callback. This way the SoundHubs will not get stuck thinking that some other SoundHub is playing. The flowchart for receiving RSSI updates can be seen in appendix D.6.

5.0 Software Design

5.1 Design Justification

As mentioned in the previous sections, the open source Framework “AllJoyn” was chosen because of the wide range of support with different language bindings and also with various development platform support. In fact the Framework was adopted by many applications on the Android platform [27]. Many of these applications are also open source which allow further adjustments and modifications. With this idea in mind, the software developers decided to adopt an existing open source music player application named “Musydra” which has “AllJoyn” built in and make changes to the source to achieve unique features such as SoundHub detection and volume adjustments through Wi-Fi network. This decision fulfilled requirement [Req 3.3.1 - P2] from the functional specifications.

5.2 Connection Protocol Utilization through Wi-Fi

Streaming media is fundamental to the operation of the SoundHub and cannot suffer any compromises. Prior to establishing Wi-Fi as the wireless protocol for streaming, alternatives such as Bluetooth were considered. The limitations of the Bluetooth protocol are significant when streaming high quality music where large bandwidth and speed are required. Wi-Fi not only meet these requirements but also exceeds them noticeably.

Similar to the firmware component, the android device is assigned with a unique identifier through bus attachment call. This unique identifier distinguishes individual SoundHubs and the “Source” android device, which circumvents the need for an overly complicated device searching algorithm [Req 3.3.2 - P2]. As presented in Figure below, a simplified diagram indicates how each “AllJoyn” enabled devices communicates with one another over the Wi-Fi network. The detailed flowchart of device detection can be referred to Appendix D. Figure 25 below shows the steps required for device communication.

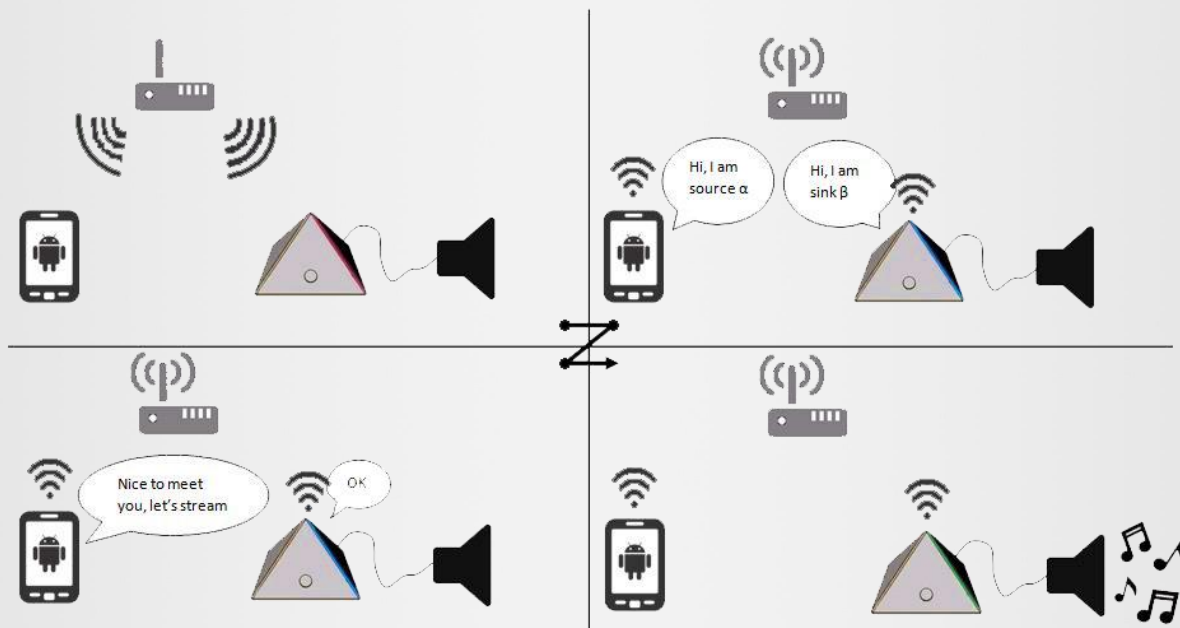


Figure 26: "AllJoyn" Steps for Device Communication

5.2.1 Handle Sink Lost

Many problems may cause losing sinks during the streaming. For example powering off and leaving the network are the major causes of sink lost. When a sink lost happens, source responses will vary depending on the number of connected sinks. If the source is streaming to multiple sinks, the audio should continue on the other connected Sinks and a popup should appear to inform the user that a Sink has disappeared including the name of the Sink. If audio is being played on a single Sink and SinkLost or SinkRemoved occurs, a pop-up should appear that gives the user the option to continue playing locally or refresh to find other sinks. In case that a SinkLost callback occurs and audio is not being played, the line item representing the lost Sink will be removed without alerting the user. A flowchart in Appendix D.7 illustrates the sink lost handling algorithm.

5.3 Graphic User Interface

The mobile application is the primary interface between the system and user. Control functions such as playback controls and volume adjustments are displayed on the main screen of the application. In addition, refresh and select proximity sinks are also considered in the design for user convenience. An example of how the GUI may look is shown below in Figure 26. Notice that the sinks refresh and select functions are not shown on the main playback screen.

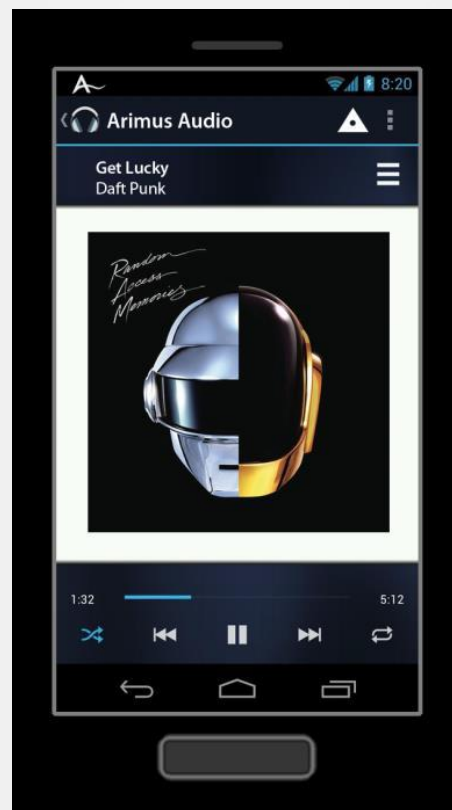


Figure 27: Artist's Rendering of the Main Playback Screen. Image based on Apollo App [28]

For the proof-of-concept model we are only implementing an Android version of the app as this is all that is necessary to show that our system works. Therefore some of the features will not be available within the proof-of-concept phase. However for the production model we will provide the full functionalities as described below and also have an iOS version available so that users have their choice of device.

5.3.1 Playback Control

As adopted from open source application “Musydra”, the basic playback functionality are provided such as play/ pause, previous/next song, and repeat and shuffle songs [Req 3.3.3 - P2]. No further modifications will be made on these functionalities.

5.3.2 Volume Adjustments

Mentioned in Section 4.3, the volume of streaming music should be adjustable by pressing the volume up/down buttons on the side of the android device [Req 3.3.4 - P2]. In the proof-of-concept stage, volume adjustments are done to all sinks at once so that each sink will follow the present volume tuning. In future stages, independent volume control can be achieved by further utilize the “AllJoyn” Framework [Req 3.3.5 - P3].

5.3.3 Sink In Proximity

This application includes a button that a user can select to list nearby Sinks. These Sinks can be selected/deselected as desired by the user so that only selected one could stream the music. The list is populated via the SinkFound callback and should only appear if there is a Sink in proximity.

5.3.4 Refresh List of Sinks

There are times, due to data propagation delays, when a Sink is not displayed right away. A button is provided to refresh the list which is an easy way to start searching for Sinks again. When the Refresh button is pressed, the list that is displayed should be cleared and, as the SinkSearcher responds with SinkFound callbacks, the list is populated again.

6.0 Enclosure Design

The SoundHub is planned to have an enclosure to prevent unintentional damage and provide a friendlier and sleeker appearance as required in [Req 3.2.18 - P2]. The enclosure is planned to be included as a part of the prototype design. This is largely because the enclosure is not necessary to prove the concept of our product and the final design depends largely on the final design of the PCB.

6.1 Shape

To differentiate the product from all of the rectangular boxed audio equipment currently on the market we have chosen a pyramidal shape. We have planned to put a bevel along each of the edges of the pyramid. This bevel helps us meet the safety requirement [Req 3.0.11 - P2] of avoiding sharp vertices and edges. Along in one of bevelled surfaces we are planning to inlay the light pipes to the status LEDs as required in [Req 3.2.16 - P2] and described in Sections 3.4 and 4.4.

6.1.1 Exterior Controls

The front and back views of the design with major features labeled is shown in Figure 27.

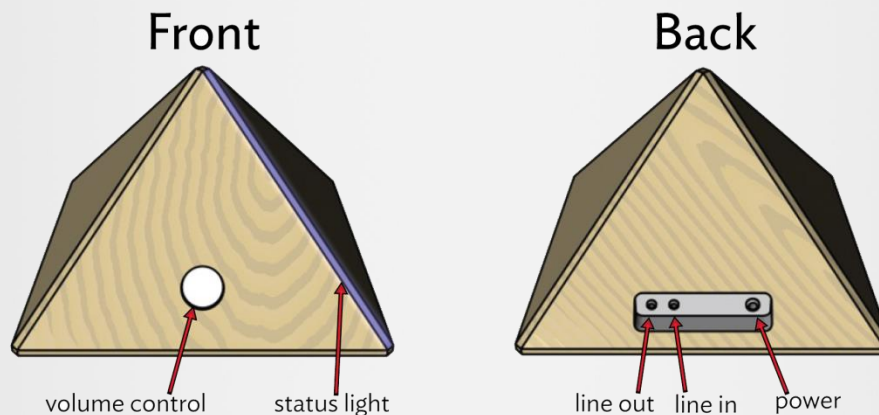


Figure 28: Front and Back views of the SoundHub

To keep with the overall simplicity of the design, only the volume knob and status lights are visible on the front panel of the enclosure. This is in line with requirements for the volume knob [Req 3.2.18 - P2] and for the status lights [Req 3.2.16 - P2]. We chose these controls to be on the front because they are the most frequently used. The volume knob that was selected to go on the rotary encoder is the OEDL-63-1-7 [29]. This is a metallic knob with a glossy finish and diamond knurl cut into the side for grip. The knob is planned to be inset into the front panel but still leaving enough to easily turn. A picture of the volume knob is shown in Figure 28.



Figure 29: Volume Knob [29]

As seen in Figure 28, the rest of the ports and controls are accessible on the back of the enclosure as required in [Req 3.2.17 - P2]. As it may be difficult to insert cables into ports which are on the slant of the pyramid, we have cut out a panel on the back which will be vertical. This goes towards addressing requirement [Req 3.2.15 - P2] and will also make it easier for us to mount our controls and ports. In the marketable revision all our ports and controls will be clearly labeled with all functional positions clearly marked where applicable to avoid user confusion. This will address requirement [Req 3.2.20 - P3].

6.1.2 Interior

Our pyramid shape is not only unique and aesthetically pleasing, but also has the added benefit of including extra room to hide an unsightly Wi-Fi antenna while not taking up too much unnecessary volume. Inside the enclosure all of the components will be secured to each other and the base of the enclosure using standoffs. This is required as a part of requirement [Req 3.0.12 - P2]. Screws will be placed on the opening plate to prevent unintended access to electronic components. Since device is made to be running all the time and the power draw of the final amplification stage can be quite large, heat dispersion will likely be an issue. As such we are designing subtle openings along the base of the side panels to allow for heat dispersion and air flow as required in [Req 3.2.18 -P2]. Once completed the electronics should not take up too much volume so the size of the enclosure will be quite small as required in [Req 3.2.22 - P2]. This will make for a more discrete product and allow your music to be the center of attention.

7.0 Test Plan

Team Arimus Audio has developed a set of preliminary test procedures to ensure proper system functionality. The test plan is divided into two parts: individual component testing and an evaluation of the integrated system. Detailed testing procedures for each part to be followed throughout the design and implementation are outlined in the following sections.

7.1 Individual Component Test

7.1.1 Hardware

The hardware test plan is divided into several sections focused on the output of each major component. Especially for the proof-of-concept stage, the test plan is focused on the custom audio processing hardware. There are two major audio processing blocks: the decoder block and DAC block. The decoder block first receives the digital SPDIF signal from the CPU, then converts the signal into I2S format [1] and feeds it into the DAC block. The DAC block produces two set of differential signals from the I2S inputs, and post-processing amplifier circuit will output the stereo signal with high audio quality.

This section will outline the hardware test procedure for the customized DAC system. Prior to entire system integration test, validation on DAC system will be conducted immediately once it is fabricated. Unit tests will be conducted to ensure they pass criteria specified below: supporting document: PCM1794A datasheet [9].

1. The hardware DAC system shall produce audio stereo output when feed with SPDIF input signal
2. The DAC system shall produce accurate audio stream with signal to noise ratio of at least 115dB.
3. The 3.5 mm line-in switch block should function properly without noise generation
4. Quadrature signal is produced when volume knob is turned indicating the direction of rotation

7.1.2 Firmware

The firmware test plan section will be split up to three major categories, networks, system, and audio. The network section will detail the specific test plan procedures used to handle all the networking and connectivity issues with SoundHub. The system section will cover the general software controlled devices and user interaction with SoundHub. The audio test plan section will detail the streaming and audio playback functionalities. All of these sections will be tested individually. The software code that will be developed to run on the SoundHub will also be modularly tested. By testing each section of software independently we can easily identify the source of the bug and make necessary corrections before the marriage of hardware and software. The agile development process will be adopted throughout the project to help minimize the overall errors and increase productivity.

Networks:

- Connectivity between the wireless AP with a SoundHub
- Utilizing WPS as an alternative method to connect the SoundHub to the wireless AP
- Connection to the wireless AP will remain active within the signal range
- Identify the network connection between the source and the sink

System:

- Time to boot up and start up applications after power is connected takes less time than 2 minutes
- Varying the volume knob will change the system volume
- Status is indicated by LED

Audio:

- Able to transfer the music packets from a single source to a single sink on a single machine
- Different support for audio format and sampling frequencies
- One music source device to one speaker system streaming through Wi-Fi network
- One music source device to multiple speaker systems streaming through Wi-Fi network
- Multiple music source devices to multiple speaker systems streaming
- Handling of a single sources to a single speaker system and later to multiple speaker systems
- Able to perform playback functionality (play, pause, music seek, up and down volume controls) within the required minimum lag duration.
- Multiple SoundHub Connected to multiple speaker systems streaming with RoomFlow function
- Active handling of Wi-Fi connectivity when dropped or lost

7.1.3 Software

The software application is the primary interface between the user and the system. Therefore the GUI of the application must be straightforward and fulfill the requirements listed in Section 3.3 of the functional specifications. The features of the application will be tested as an individual component without considering the SoundHub system

- Successfully run on android devices without frequent crashing
- Correct layout of GUI and each button corresponds to its functionality
- Automatic identifies the SoundHub device once the mobile device and the SoundHub is on the same Wi-Fi network
- Stream to a separate device other than the SoundHub known to implement the sink protocol

Please refer to the Appendix.E for a list of detailed test cases

7.2 Integration Tests

The integrated system can be evaluated in different categories once the individual components are examined to be working properly and the integration has been completed. In the proof-of-concept stage, the highest priorities are the audio quality and the latency response during the streaming. Therefore the following tests will be done and future adjustments will be made accordingly.

Perceived audio quality:

- The wireless streamed music through the system should be of noticeably better quality compared to non-processed music through line in from the same source. This subjective measurement will be conducted and judged by experienced individual.
- The wireless streamed music should retain its quality over different frequency range, sample music such as heavy bass and soprano will be used to test

Latency Test:

- The wireless streamed music should respond to user's command through the mobile application within five seconds. The latency will be evaluated based on different circumstances such as the relative signal strength of the source and the sink, and the actual command (tuning volume, play/pause, queuing new songs). The results will be handed over to the development team for future improvements.

Power Consumption Test:

- The system should not exceed the power rating on the AC converter

Stress Test:

- Stream music to the SoundHub for a period of twelve hours and determine the rate of dropped music during that duration

8.0 Conclusions

This documentation has laid out the design specifications and provides design approaches and technical details which correspond to the functional specifications of the SoundHub wireless speaker adapter system. The system consists of three major sections:

- Hardware development aimed to provide high quality audio transmission through the design and implement of a customized DAC system
- Firmware development on the evaluation board to achieve audio streaming through a Wi-Fi network to multiple speaker systems
- Software development aimed to develop an android application which is capable of controlling playback over a mobile device

Each of the components above had its own section explaining the specifications and justifications during the design phase. The requirement codes from functional specifications are mentioned whenever corresponding text are presented.

The final section of the document provides a set of preliminary test procedures to examine the model functionality of the proof-of-concept model. The test plan is divided into two parts: individual component testing and an evaluation of the integrated system. Detailed testing procedures for each part to be followed throughout the design and implementation are provided as an appendix in the end of the document.

9.0 References

- [1] Arimus Audio. (2013, February 12) Functional Specification: Wireless Speaker Module [Online] <http://www2.ensc.sfu.ca/~whitmore/courses/ensc305/projects/2014/11func.pdf>
- [2] Wandboard Org. (2012, August 16) Wandboard Specifications [Online] <http://www.wandboard.org/index.php/details>
- [3] Qualcomm Innovation Center, Inc. (2012, February 16) About AllJoyn. [Online] <https://www.AllJoyn.org/about>
- [4] Restriction of Hazardous Substances Guide (2013, Feb. 13) RoHS Guide Compliance. [Online] <http://www.rohsguide.com/>
- [5] Federal Communications Commission. (2010, October) Rules and Regulation. [Online] <http://transition.fcc.gov/oet/info/rules/>
- [6] IEEE Standards Association (2012, March 29) Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications [Online] <http://standards.ieee.org/getieee802/download/802.11-2012.pdf>
- [7] CadSoft (2005, August 5) EAGLE PCB Design Software [Online] <http://www.cadsoftusa.com/>
- [8] Texas Instruments (2011, January 19) DIR9001 96-kHz 24-bit Digital Audio Interface Receiver [Online] <http://www.ti.com/lit/ds/symlink/dir9001.pdf>
- [9] Texas Instruments (2010, April 25) PCM1794A 24-bit 192kHz Audio Stereo Digital-to-Analog Converter [Online] <http://www.ti.com/lit/ds/symlink/pcm1794a.pdf>
- [10] PAVOUK. (2012, January 2) "Modular Audio DAC" [Online] http://pavouk.org/en_index.html
- [11] Philip semiconductors. (2006, July 4) I2S Specification [Online] https://web.archive.org/web/20060702004954/http://www.semiconductors.philips.com/acrobat_download/various/I2SBUS.pdf
- [12] Newava Technology. (2006, April 3) S22083 Pulse Transformer [Online] <http://www.newava.com/pdf/S22083.pdf>
- [13] Texas Instruments. (2009, January 3) SN75178B Differential Bus Repeaters [Online] <http://www.ti.com/lit/ds/symlink/sn75178b.pdf>
- [14] Microchip. (2008, December 25) MCP130 Microcontroller Supervisory Circuit with Open Drain Output [Online] <http://ww1.microchip.com/downloads/en/DeviceDoc/11184d.pdf>

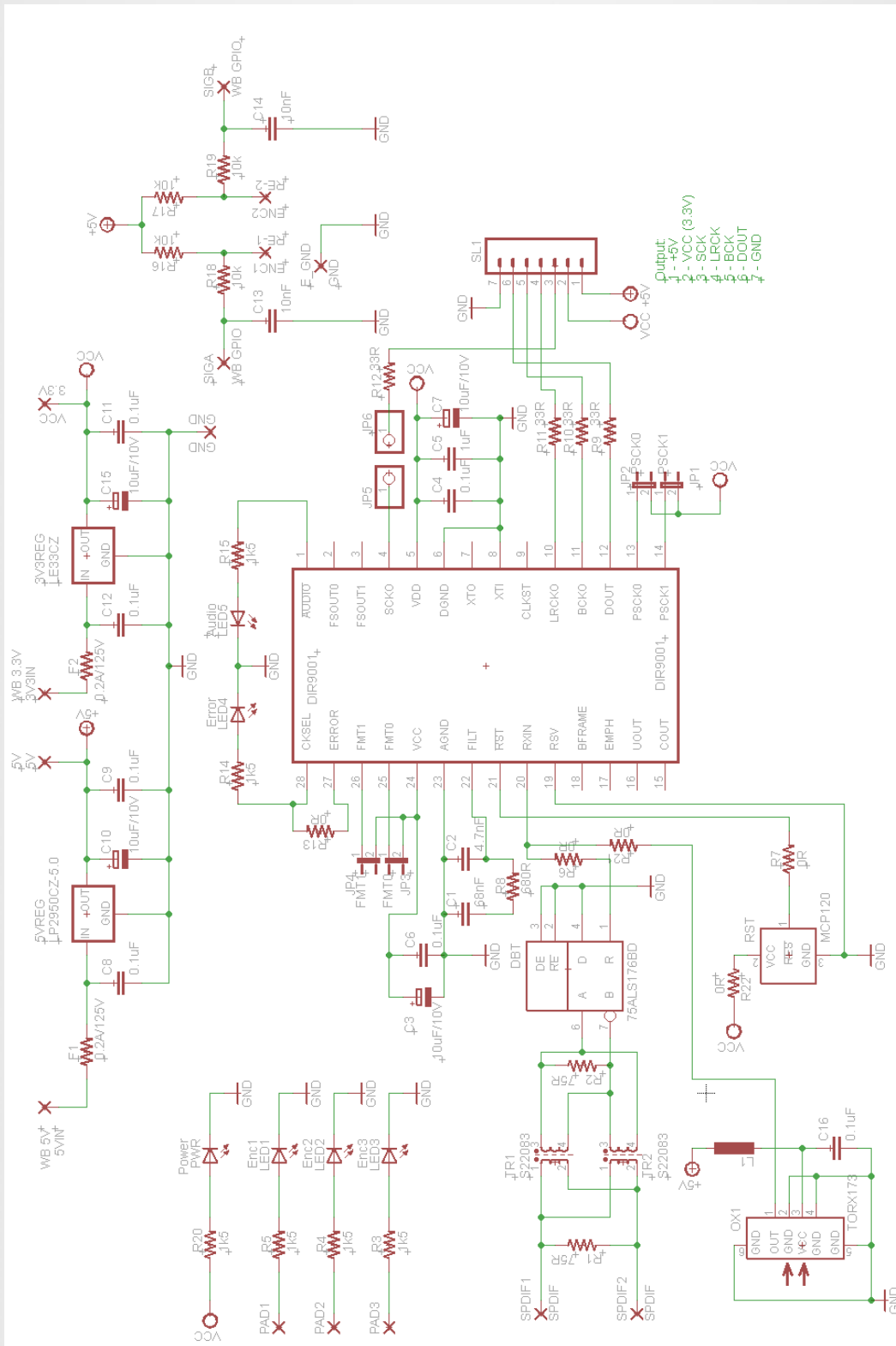
- [15] Texas Instruments. (2011, March 2) TL072 J-FET Low Noise General Purpose Operational Amplifier [Online] <http://www.ti.com/lit/ds/symlink/tl072.pdf>
- [16] Texas Instruments. (2011, January 4) LP2950N 5V Adjustable Micro-power Voltage Regulator [Online] <http://www.ti.com/lit/ds/symlink/lp2950-n.pdf>
- [17] STMicroelectronics. (2003, April 5) LE33CZ 3.3V Very Low Drop Voltage Regulator [Online] <http://www.st.com/web/en/resource/technical/document/datasheet/CD00000545.pdf>
- [18] Texas Instruments. (2008, July 8) UA78L12 +12V Fixed Positive Voltage Regulator [Online] <http://www.ti.com/lit/ds/symlink/ua7812.pdf>
- [19] Texas Instruments. (2012, March 8) MC79L12 -12V Negative Voltage Regulator [Online] <http://www.ti.com/lit/ds/symlink/mc79l12.pdf>
- [20] Visual Communications Company Inc. (2010, April 23) LPC197CTP – Acrylic Light Tube [Online] http://vcclite.com/_pdf/lpc-4mm-round1.pdf
- [21] Panasonic. (2012, Oct. 10) 12mm Square GS Encoders [Online] <http://industrial.panasonic.com/www-data/pdf/ATC0000/ATC0000CE4.pdf>
- [22] Wi-Fi Alliance. (2006, December 3) Wi-Fi Protected Setup Specification. [Online] http://gpl.back2roots.org/source/puma5/netgear/CG3200-1TDNDS_GPL/ap/apps/wpa2/original/Wi-Fi%20Protected%20Setup%20Specification%201.0h.pdf
- [23] E-Switch. (2006, August 24) 1003P1TB1M1QEH – 3PDT Switch [Online] <http://spec.e-switch.com/ST/ST131003.pdf>
- [24] Epoch Converter. (2014 February 3) Epoch & Unix Timestamp Conversion Tools. [Online] <http://www.epochconverter.com/>
- [25] J. Kysela et al. (2014, March 12) Advanced Linux Sound Architecture Project [Online] http://www.alsa-project.org/main/index.php/Main_Page
- [27] Qualcomm Innovation Center, Inc. (2012, January 27) [Online] <https://www.AllJoyn.org/app-developers/featured-apps>
- [26] J. Malinen. (2013, Jan 12) Linux WPA/WPA2/IEEE 802.1X Supplicant. [Online] http://hostap.epitest.fi/wpa_supplicant/
- [28] Pantazi Costin. (2014, January 23) Apollo Reborn music player. [Online] <https://play.google.com/store/apps/details?id=ro.pca.apolloreborn>
- [29] K Fowler. (2005, March 5) Kilo International OED Catalog [Online] <http://media.digikey.com/pdf/Data%20Sheets/Kilo%20International%20PDFs/OED.pdf>



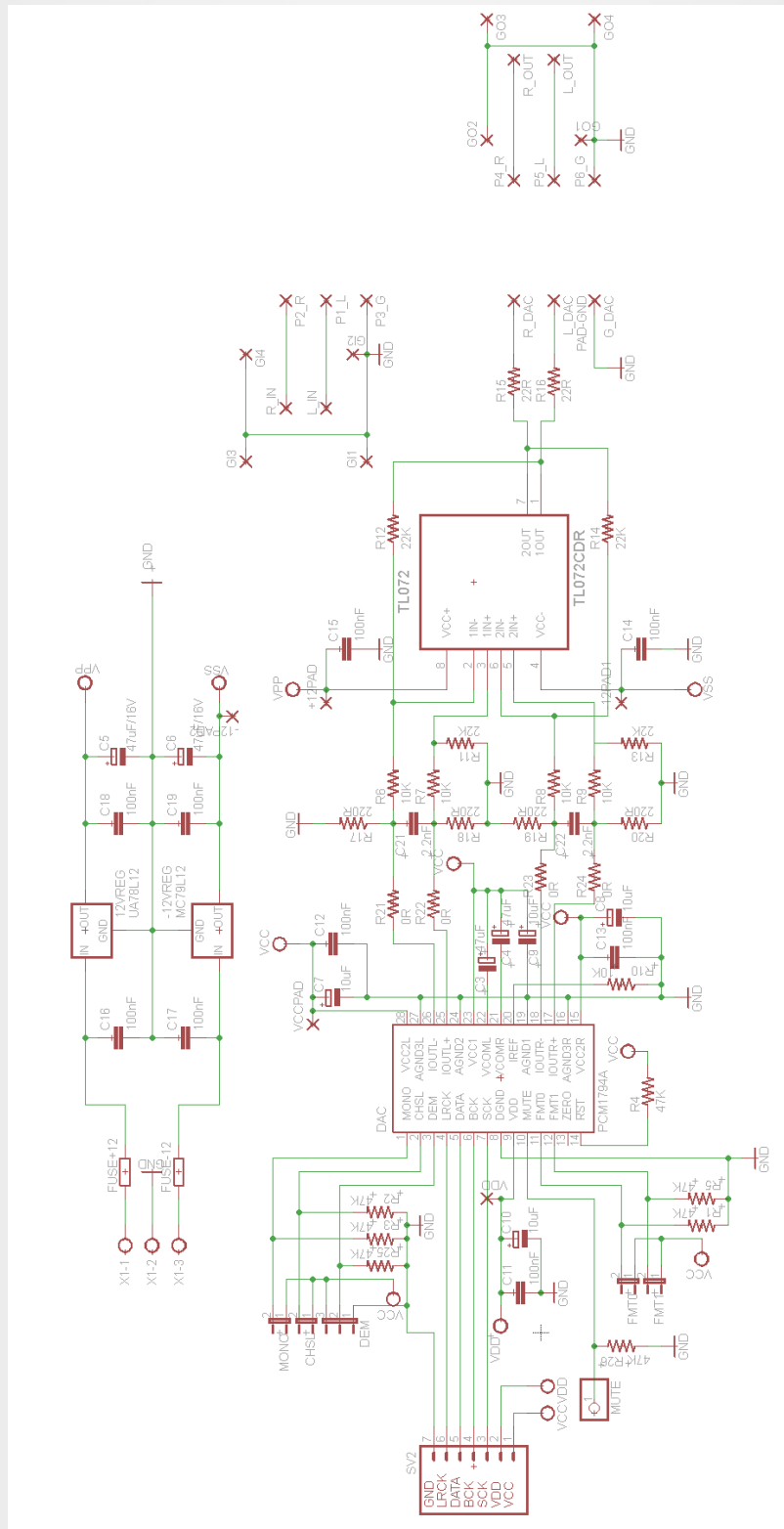
- [30] Qualcomm Innovation Center, Inc . (2013, December 8) Allseen Alliance. [Online]
<https://allseenalliance.org/docs-and-downloads/documentation/alljoyn-audio-service-framework-10-interface-specification>

Appendix A: Electronic Schematics

A.1 SPDIF Decoder Schematic

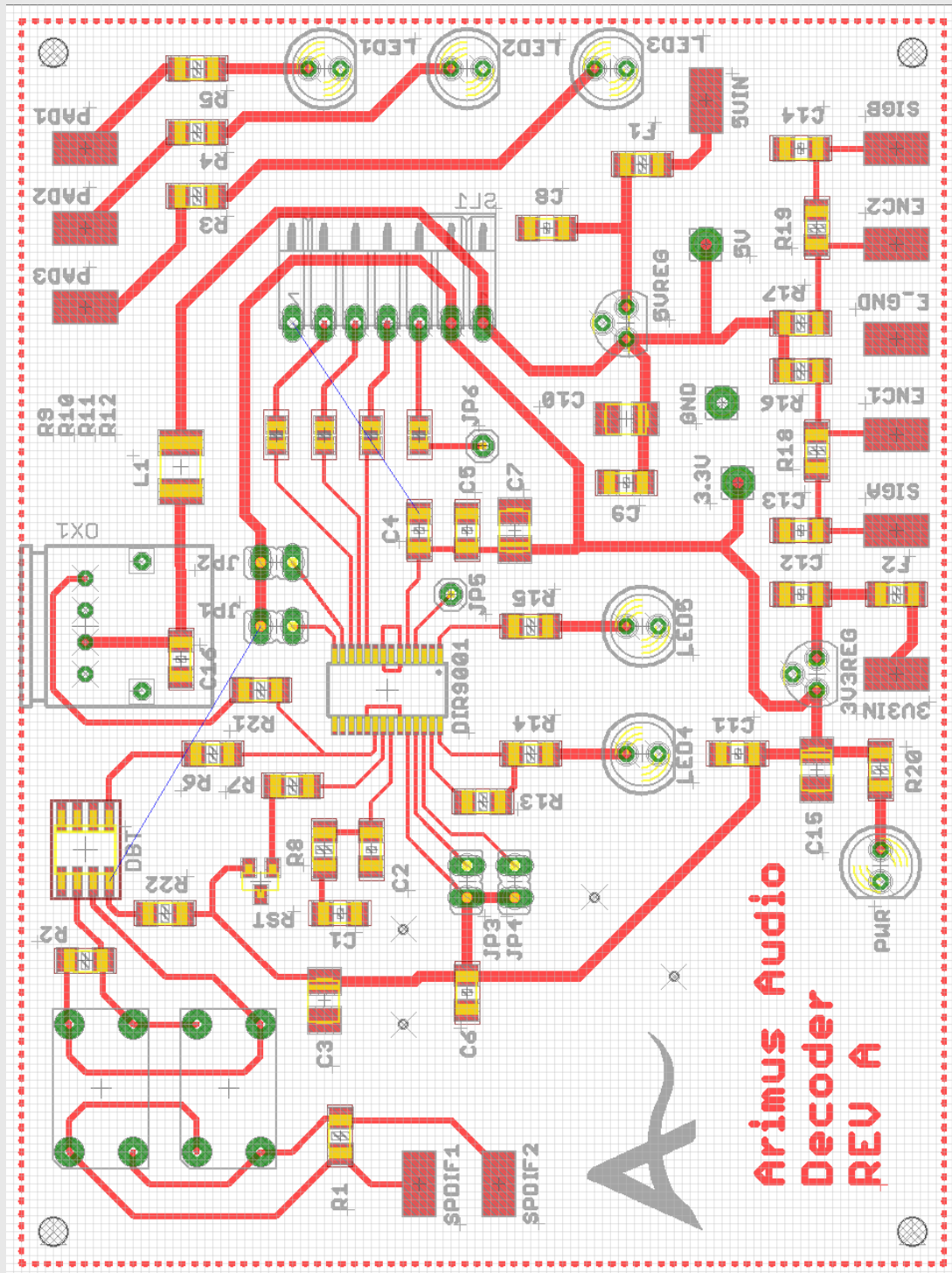


A.2 DAC Schematic



Appendix B: PCB Board Layout

B.1 SPDIF Decoder Layout



C.1 DIR9001 pin layout



C.2 DIR9001 Pin Description

TERMINAL		I/O	PULL UP/DOWN	REMARKS	DESCRIPTION
NAME	NO.				
AGND	23	–			Analog ground
AUDIO	1	OUT		CMOS	Channel-status data information of non-audio sample word, active-low
BCKO	11	OUT		CMOS	Audio data bit clock output
BFRAME	18	OUT		CMOS	Indication of top block of biphase input signal
CKSEL	28	IN	Pulldown	5-V tolerant TTL	Selection of system clock source, Low: PLL (VCO) clock, High: XTI clock ⁽¹⁾
CLKST	9	OUT		CMOS	Clock change/transition signal output
COUT	15	OUT		CMOS	Channel-status data serial output synchronized with LRCKO
DGND	6	–			Digital ground
DOUT	12	OUT		CMOS	16-bit/24-bit decoded serial digital audio data output
EMPH	17	OUT		CMOS	Channel-status data information of pre-emphasis (50 μ s/15 μ s)
ERROR	27	OUT		CMOS	Indication of internal PLL or data parity error
FILT	22	–			External filter connection terminal; must connect recommended filter.
FMT0	25	IN	Pulldown	5-V tolerant TTL	Decoded serial digital audio data output format selection 0 ⁽¹⁾
FMT1	26	IN	Pulldown	5-V tolerant TTL	Decoded serial digital audio data output format selection 1 ⁽¹⁾
FSOUT0	2	OUT		CMOS	Actual sampling frequency calculated result output 0
FSOUT1	3	OUT		CMOS	Actual sampling frequency calculated result output 1
LRCKO	10	OUT		CMOS	Audio data latch enable output
PSCK0	13	IN	Pulldown	5-V tolerant TTL	PLL source SCKO output frequency selection 0 ⁽¹⁾
PSCK1	14	IN	Pulldown	5-V tolerant TTL	PLL source SCKO output frequency selection 1 ⁽¹⁾
RST	21	IN	Pullup	5-V tolerant TTL	Reset control input, active-low ⁽²⁾
RSV	19	IN	Pulldown		Reserved, must be connected to DGND ⁽¹⁾
RXIN	20	IN		5-V tolerant TTL	Biphase digital data input ⁽³⁾
SCKO	4	OUT		CMOS	System clock output
UOUT	16	OUT		CMOS	User data serial output synchronized with LRCKO
V _{CC}	24	–			Analog power supply, 3.3-V
V _{DD}	5	–			Digital power supply, 3.3-V
XTI	8	IN		CMOS Schmitt-trigger	Oscillation amplifier input, or external XTI source clock input
XTO	7	OUT		CMOS	Oscillation amplifier output

(1) TTL Schmitt-trigger input with internal pulldown (51 k Ω typical), 5-V tolerant

(2) TTL Schmitt-trigger input with internal pullup (51 k Ω typical), 5-V tolerant

(3) TTL Schmitt-trigger input, 5-V tolerant.

C.3 Pin Settings for Mode of Operation of DIR9001

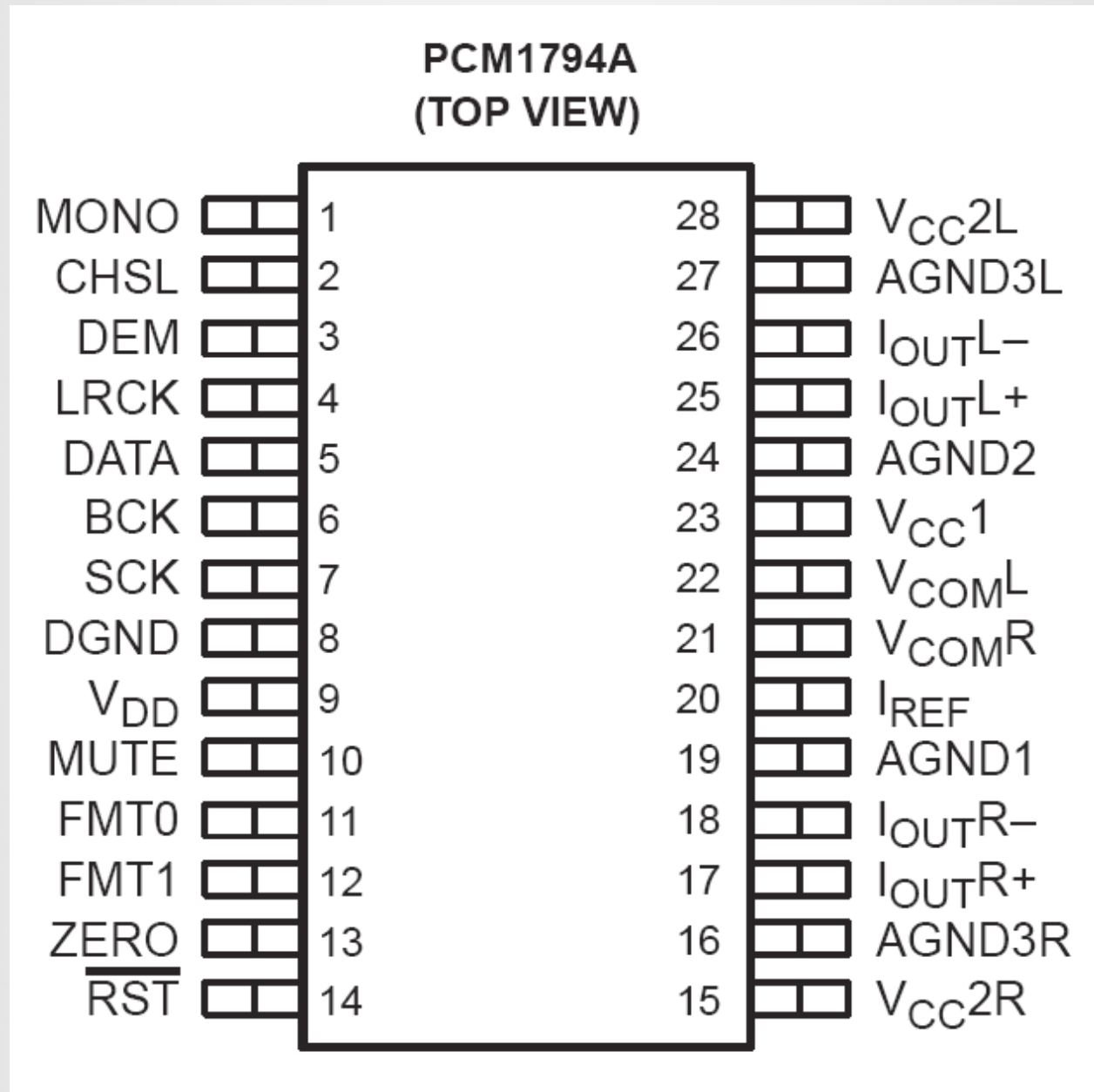
OPERATION MODE	CKSEL PIN SETTING	ERROR PIN STATUS	SCKO, BCKO, LRCKO CLOCK SOURCE	DOUT DATA	AUDIO EMPH	FSOUT [1:0]	BFRAME	COU UOUT
PLL	L	H	PLL (VCO) free-running clock ⁽¹⁾	MUTE (Low)	LOW	HL	LOW	LOW
		L	PLL recovered clock	Decoded data	OUT	OUT	OUT	OUT
XTI	H	H	XTI clock	MUTE (Low)	LOW	HL	LOW	LOW
		L	XTI clock	MUTE (Low)	OUT	OUT	OUT	LOW
AUTO	Connected to ERROR pin	H	XTI clock	MUTE (Low)	LOW	HL	LOW	LOW
		L	PLL recovered clock	Decoded data	OUT	OUT	OUT	OUT

(1) The VCO free-running frequency is not a constant frequency, because the VCO oscillation frequency is dependent on supply voltage, temperature, and process variations.

PSCK[1:0] SETTING		OUTPUT CLOCK FROM PLL SOURCE		
PSCK1	PSCK0	SCKO	BCKO	LRCKO
L	L	128 f_S	64 f_S	f_S
L	H	256 f_S	64 f_S	f_S
H	L	384 f_S	64 f_S	f_S
H	H	512 f_S	64 f_S	f_S

FMT[1:0] SETTINGS		DOUT SERIAL AUDIO DATA OUTPUT FORMAT
FMT1	FMT0	
L	L	16-bit, MSB-first, right-justified
L	H	24-bit, MSB-first, right-justified
H	L	24-bit MSB-first, left-justified
H	H	24-bit, MSB-first, I ² S

C.4 PCM1794A Pin Layout



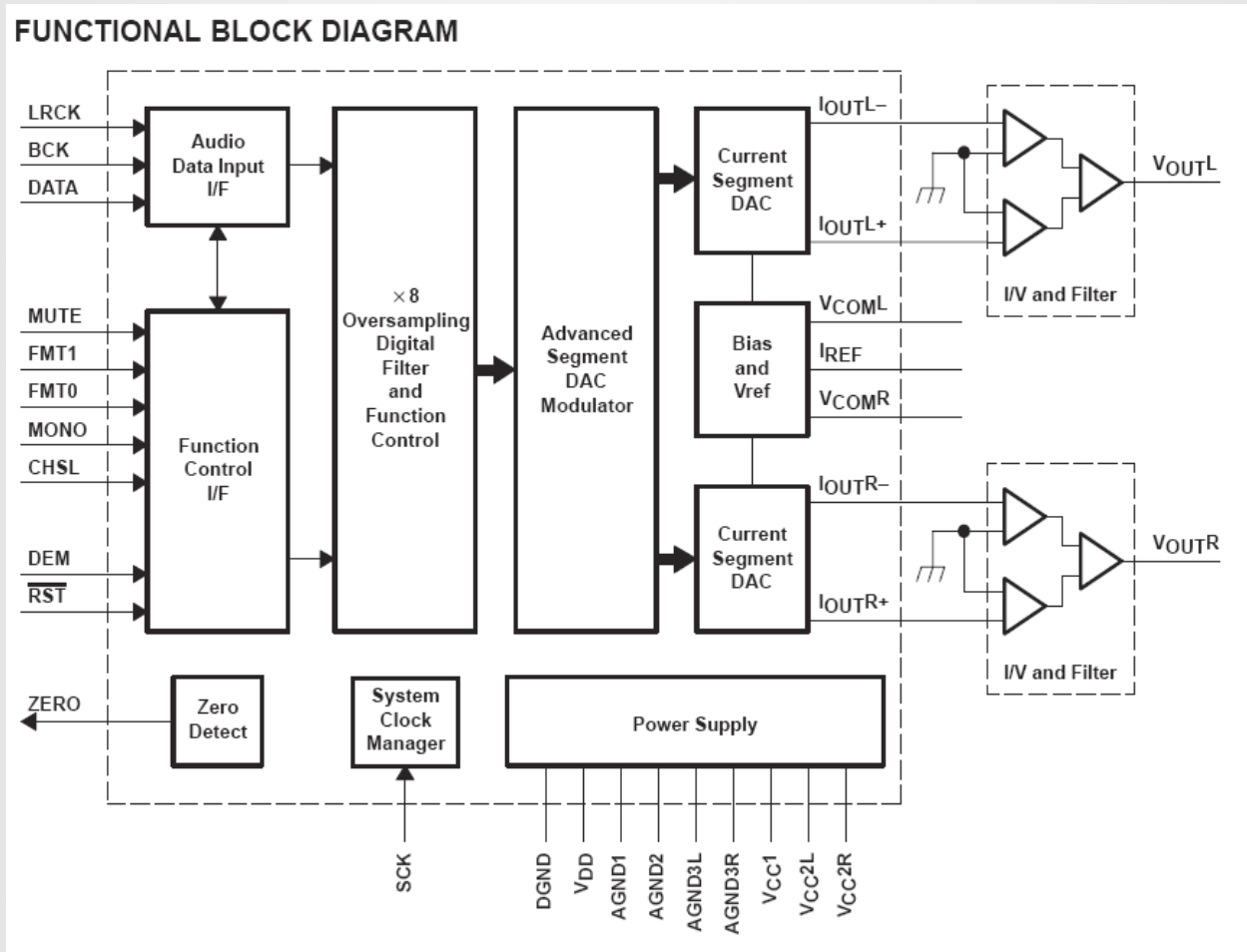
C.5 Pin Description for PCM1794A

Terminal Functions

TERMINAL		I/O	DESCRIPTIONS
NAME	PIN		
AGND1	19	–	Analog ground (internal bias)
AGND2	24	–	Analog ground (internal bias)
AGND3L	27	–	Analog ground (L-channel DACFF)
AGND3R	16	–	Analog ground (R-channel DACFF)
BCK	6	I	Bit clock input ⁽¹⁾
CHSL	2	I	L-, R-channel select ⁽¹⁾
DATA	5	I	Serial audio data input ⁽¹⁾
DEM	3	I	De-emphasis enable ⁽¹⁾
DGND	8	–	Digital ground
FMT0	11	I	Audio data format select ⁽¹⁾
FMT1	12	I	Audio data format select ⁽¹⁾
IOUTL+	25	O	L-channel analog current output +
IOUTL–	26	O	L-channel analog current output –
IOUTR+	17	O	R-channel analog current output +
IOUTR–	18	O	R-channel analog current output –
IREF	20	–	Output current reference bias pin
LRCK	4	I	Left and right clock (f_S) input ⁽¹⁾
MONO	1	I	Monaural mode enable ⁽¹⁾
MUTE	10	I	Mute control ⁽¹⁾
$\overline{\text{RST}}$	14	I	Reset ⁽¹⁾
SCK	7	I	System clock input ⁽¹⁾
VCC1	23	–	Analog power supply, 5 V
VCC2L	28	–	Analog power supply (L-channel DACFF), 5 V
VCC2R	15	–	Analog power supply (R-channel DACFF), 5 V
VCOML	22	–	L-channel internal bias decoupling pin
VCOMR	21	–	R-channel internal bias decoupling pin
VDD	9	–	Digital power supply, 3.3 V
ZERO	13	O	Zero flag

⁽¹⁾ Schmitt-trigger input, 5-V tolerant

C.6 Functional Block Diagram



C.7 Relevant Technical Specification

ELECTRICAL CHARACTERISTICS (Continued)

all specifications at $T_A = 25^\circ\text{C}$, $V_{CC1} = V_{CC2L} = V_{CC2R} = 5\text{ V}$, $V_{DD} = 3.3\text{ V}$, $f_S = 44.1\text{ kHz}$, system clock = $256 f_S$, and 24-bit data, unless otherwise noted

PARAMETER	TEST CONDITIONS	PCM1794ADB			UNIT
		MIN	TYP	MAX	
DYNAMIC PERFORMANCE (2-V RMS OUTPUT) (1)(2)					
THD+N at $V_{OUT} = 0\text{ dB}$	$f_S = 44.1\text{ kHz}$		0.0004%	0.0008%	
	$f_S = 96\text{ kHz}$		0.0008%		
	$f_S = 192\text{ kHz}$		0.0015%		
Dynamic range	EIAJ, A-weighted, $f_S = 44.1\text{ kHz}$	123	127		dB
	EIAJ, A-weighted, $f_S = 96\text{ kHz}$		127		
	EIAJ, A-weighted, $f_S = 192\text{ kHz}$		127		
Signal-to-noise ratio	EIAJ, A-weighted, $f_S = 44.1\text{ kHz}$	123	127		dB
	EIAJ, A-weighted, $f_S = 96\text{ kHz}$		127		
	EIAJ, A-weighted, $f_S = 192\text{ kHz}$		127		
Channel separation	$f_S = 44.1\text{ kHz}$	120	123		dB
	$f_S = 96\text{ kHz}$		122		
	$f_S = 192\text{ kHz}$		120		
Level linearity error	$V_{OUT} = -120\text{ dB}$		± 1		dB

C.8 Pin settings for output format

Table 2. Audio Data Format Select

MONO	CHSL	FMT1	FMT0	FORMAT	STEREO/MONO	DF ROLLOFF
0	0	0	0	I ² S	Stereo	Sharp
0	0	0	1	Left-justified format	Stereo	Sharp
0	0	1	0	Standard, 16-bit	Stereo	Sharp
0	0	1	1	Standard, 24-bit	Stereo	Sharp
0	1	0	0	I ² S	Stereo	Slow
0	1	0	1	Left-justified format	Stereo	Slow
0	1	1	0	Standard, 16-bit	Stereo	Slow
0	1	1	1	Digital filter bypass	Mono	–
1	0	0	0	I ² S	Mono, L-channel	Sharp
1	0	0	1	Left-justified format	Mono, L-channel	Sharp
1	0	1	0	Standard, 16-bit	Mono, L-channel	Sharp
1	0	1	1	Standard, 24-bit	Mono, L-channel	Sharp
1	1	0	0	I ² S	Mono, R-channel	Sharp
1	1	0	1	Left-justified format	Mono, R-channel	Sharp
1	1	1	0	Standard, 16-bit	Mono, R-channel	Sharp
1	1	1	1	Standard, 24-bit	Mono, R-channel	Sharp

C.9 Galvanic isolation

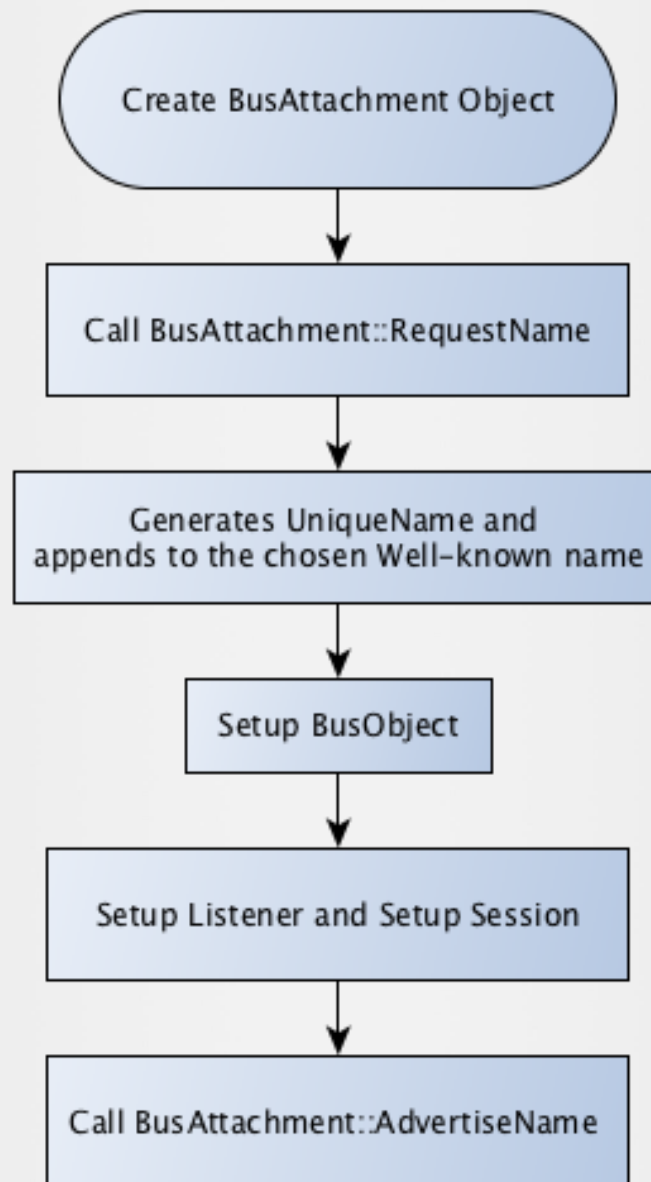
In audio application, when connecting two audio hardware together to a common ground, A ground loop may occur. A ground loop happens when two grounds connects together with one DC offset higher than the other. This potential difference will cause a current loop flowing through the system.

Depending on the circuit configurations of the audio hardware, such current could contaminate audio signal quality by inducing voltages at the output signal. As a result, user will hear “humming” noise from the audio output. This effect could still happen even if both of the hardware devices are plugged into a common outlet

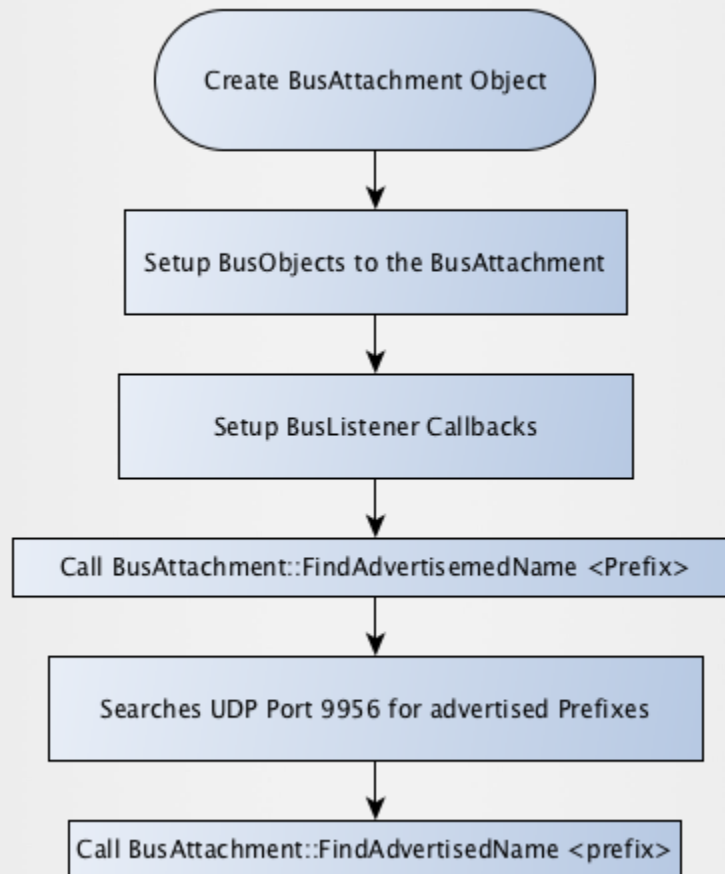
To eliminate this ground loop, we can put a transformer in between the circuit overlap to isolate the current. Such isolation is referred as galvanic isolation. This type of noise elimination technique is critical to audio electronics and a prerequisite for high end audio device.

Appendix D: Firmware Flowcharts

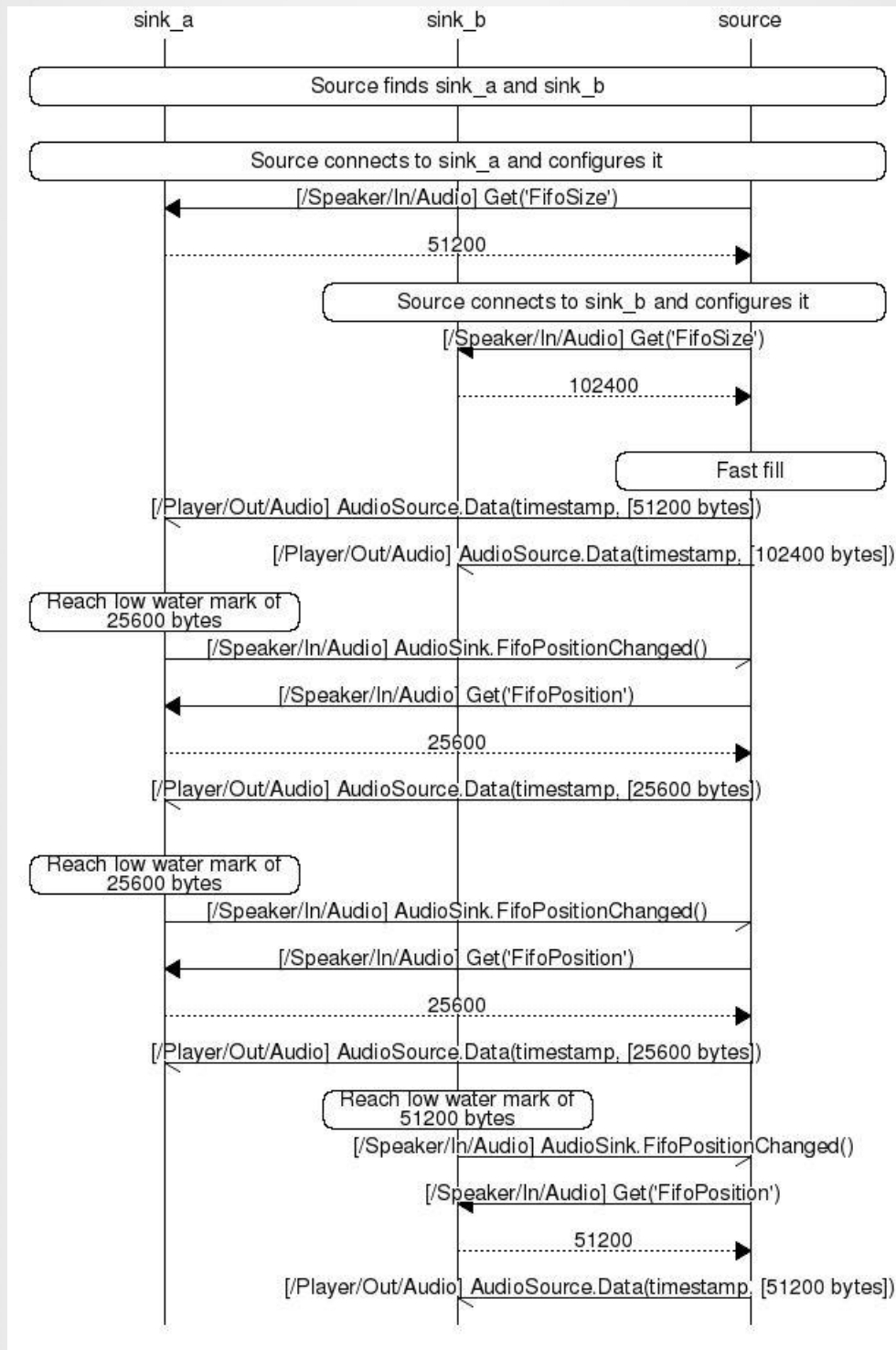
D.1 AllJoyn Advertisement Process Flowchart



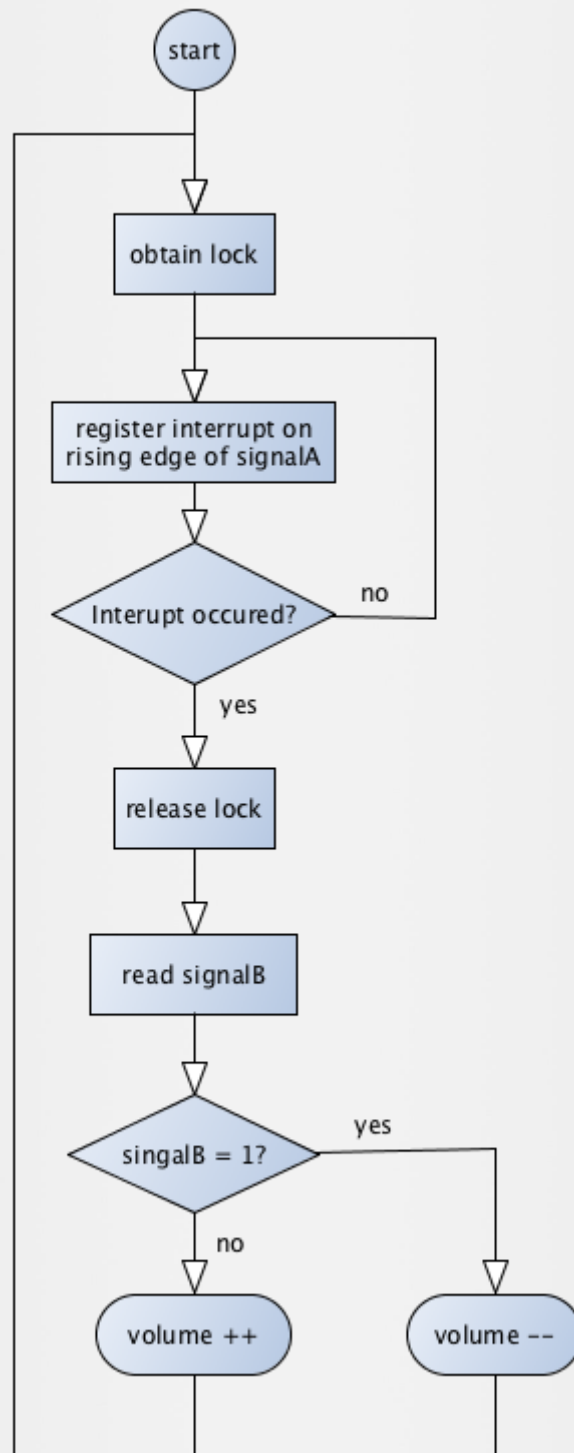
D.2 AllJoyn Discovery Process Flowchart



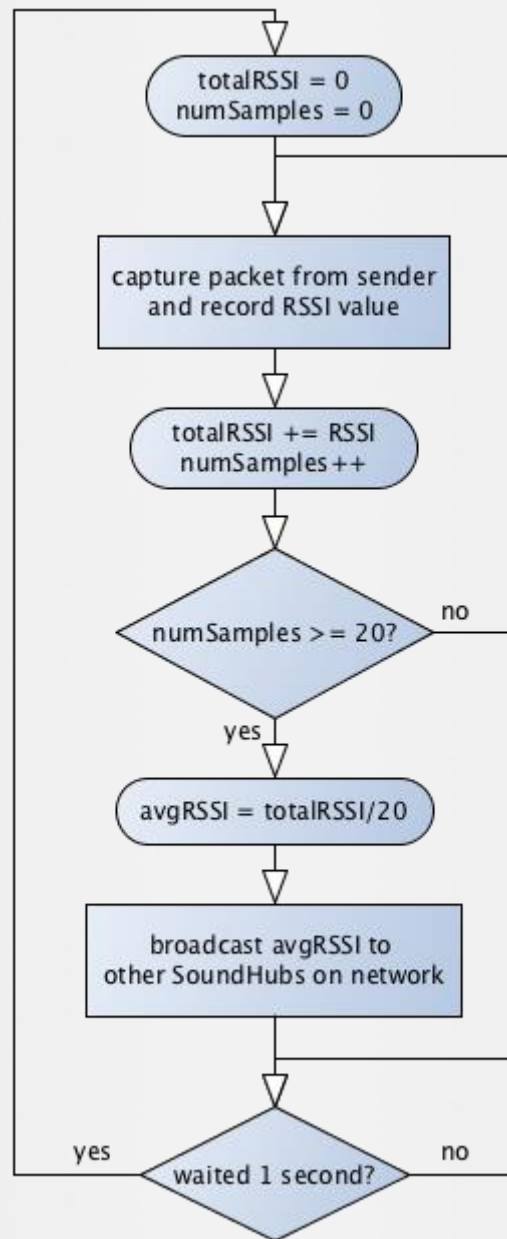
D.3 AllJoyn Single Source to Multi Sink [30]



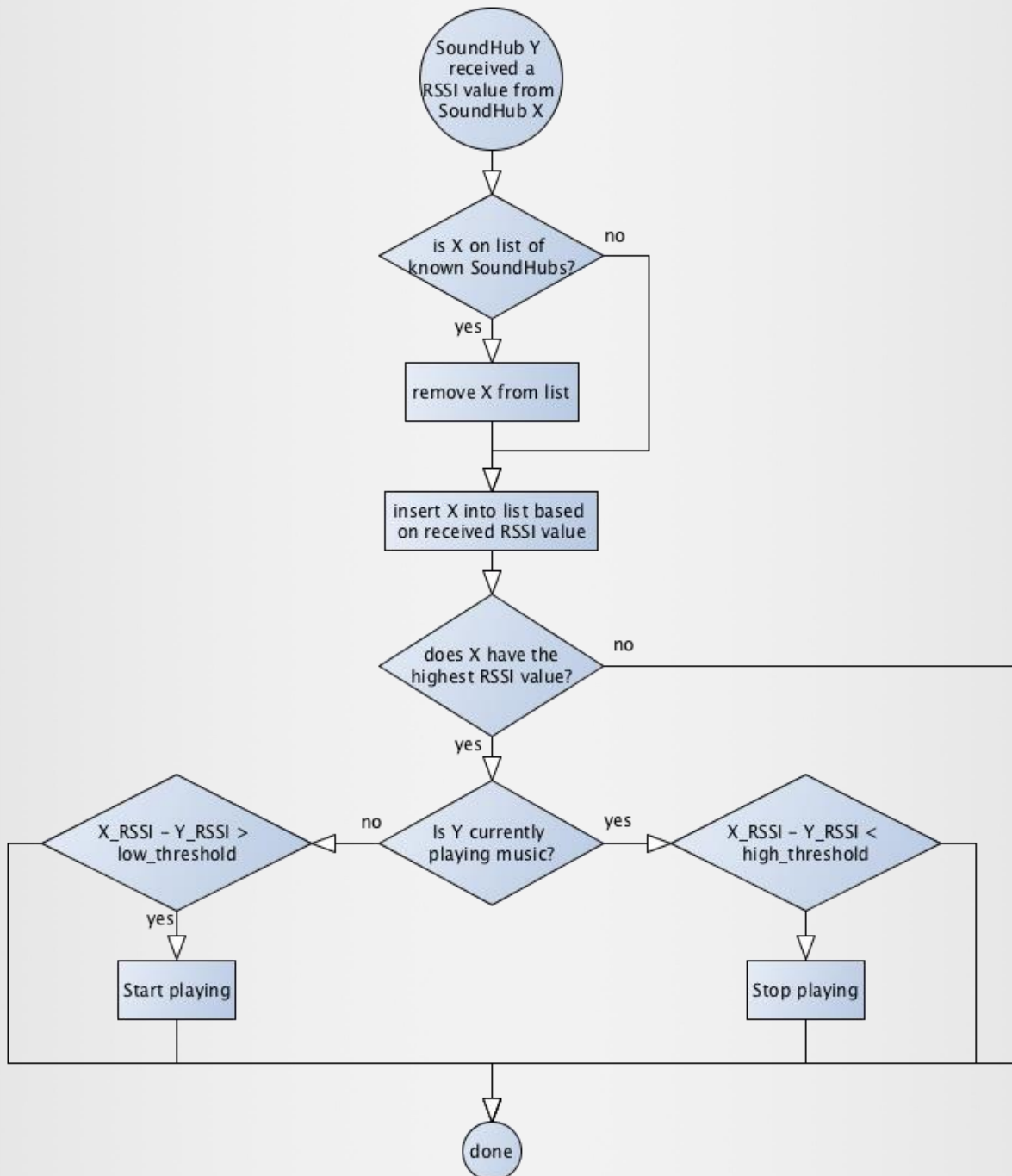
D.4 Flow Chart for Volume Thread



D.5 Flowchart for sending RSSI value in RoomFlow

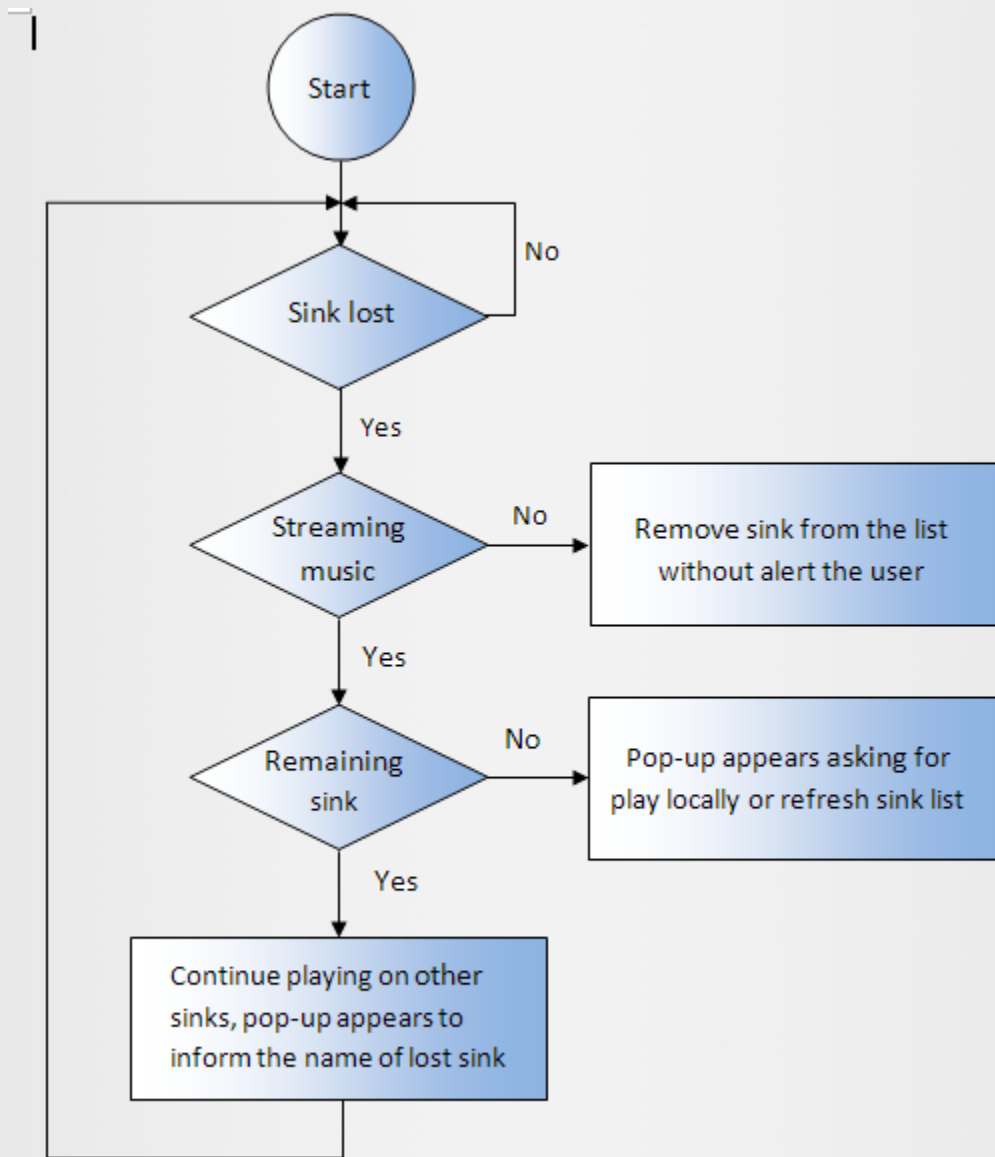


D.6 Flowchart for receiving RSSI value in RoomFlow



Note: this same flowchart is also used when SoundHub X and SoundHub Y are the same.

D.7 Flowchart Handling of Sink Lost





Appendix E: Detailed Test Plans

E.1 Hardware Test Plans

Arimus Audio's SoundHub Evaluation Sheet

Test Summary

Test ID: _____

Prepared By: _____

Review By: _____

Test Date: _____

Additional Notes:

Results

Approved

Conditional approved

Disapproved

Retest

Observations and Comments:

I certify that this test procedure has been performed according to the documentation, and that the results recorded are accurate and complete.

Tester Name: _____

Signature: _____

Date: _____

Witness Name: _____

Signature: _____

Date: _____

©Team Arimus Audio, 2014

This document is licensed to Team Arimus Audio and cannot be used, reproduced, published and/ or revealed without prior written authorization.

Archived Date:

Hardware test case 1: Overall audio function

Test Method:

The tester will feed the DAC system with SPDIF signal from any audio source. The line-out switch is set to position 3.

Pass Criteria:

User is able to hear music from the 3.5mm line-out jack of the DAC system.

Test Summary	
Output notified	Hearing music
Results	
Pass	Fail
Observations and Comments:	

Hardware test case 2: SNR

Test method:

The tester will use laboratory instrument to measure the magnitude of the output audio signal by probing to the output of the differential amplifier. The tester will measure the both output signal magnitude with SPDIF input and without input source. The signal to noise ratio will be calculated based the measured result.

Pass Criteria:

The audio signal coming out from the amplifier is clean such that the calculated signal to noise ratio is above 115dB.

Test Summary	
Expected SNR	Measured SNR
Results	
Pass	Fail
Observations and Comments:	

Hardware test case 3: Line-in switch

Test Method:

The tester will input audio into the 3.5mm line-in audio jack, set the 3PDT switch to position 1, and ensure the input and output signals are the same. The tester will set the 3PDT switch to position 3 and perform Hardware Test Case 1. The tester will set the 3PDT switch to position 2 and hear no audio from the 3.5mm line-out jack.

Pass Criteria:

Position 1: The noise level between two nodes is indistinguishable, and audio is playing from the line-out.

Position 3: See Hardware Test Case 1

Position 2: No audio should be playing from the 3.5mm line-out jack.

Test Summary		
Position 1 Output	Position 2 Output	Position 3 Output
Results		
Pass		Fail
Observations and Comments:		

Hardware test case 4: Volume knob

Test method:

The tester will use an oscilloscope to probe signals A and B from the rotary encoder. The volume knob will be turned clockwise and counter clockwise at constant rates. Special attention should be paid to the phase relation between signals A and B.

Pass Criteria:

When the volume knob is turned clockwise, signal B should be low on the rising edge of signal A. When the volume knob is turned counter clockwise, signal B should be high on the rising edge of signal A. There should not be excessive bouncing on the edges of either signal.

Test Summary	
Clockwise rotation	Counter clockwise rotation
Results	
Pass	Fail
Observations and Comments:	



E.2 Firmware Test Plans

Arimus Audio's SoundHub Evaluation Sheet

Test Summary			
Test ID: _____		Prepared By: _____	
Review By: _____		Test Date: _____	
Additional Notes:			
Results			
Approved	Conditional approved	Disapproved	Retest
Observations and Comments:			
I certify that this test procedure has been performed according to the documentation, and that the results recorded are accurate and complete.			
Tester Name: _____		Signature: _____	Date: _____
Witness Name: _____		Signature: _____	Date: _____
©Team Arimus Audio, 2014 This document is licensed to Team Arimus Audio and cannot be used, reproduced, published and/ or revealed without prior written authorization.			Archived Date:

Firmware test case 1: network connectivity

Test method:

Through a serial connection the tester will prep the `/etc/network/interfaces` file have the service set identification (SSID) and password a wireless AP. The tester will then use the `/etc/init.d/networking restart` command to reset the system. Finally the tester will see if the wireless AP has been joined using the `iwconfig` command.

Pass Criteria:

The `iwconfig` command return value should show wlan4 to be connected to the desired wireless AP.

Test Summary	
Results	
Pass	Fail
Observations and Comments:	

Firmware test case 2: WPS connectivity

Test method:

Using a wireless AP that the SoundHub has not connected to, the tester will press the WPS button on the wireless AP and then press the WPS button on the SoundHub. The tester will see if the wireless AP has been joined using the `iwconfig` command.

Pass Criteria:

The `iwconfig` command return value should show wlan4 to be connected to the desired wireless AP.

Test Summary	
Results	
Pass	Fail
Observations and Comments:	

Firmware test case 3: Signal range test

Test method:

Using a known wireless AP, the SoundHub will be started at the distances of 5m, 10m, 20m and 30m from the wireless AP. Once started the tester will see if the wireless AP has been joined using the `iwconfig` command.

Pass Criteria:

The `iwconfig` command return value should show wlan4 to be connected to the desired wireless AP.

Test Summary	
Results	
Pass	Fail
Observations and Comments:	

Firmware test case 4: Boot time

Test method:

The tester will use a stopwatch to measure time and will be using a known wireless AP. The stopwatch will be started as the SoundHub is plugged in and will be stopped when the status LED indicates that the wireless AP has been joined.

Pass Criteria: The stopwatch reads a time less than 5 minutes.

Test Summary	
Results	
Pass	Fail
Observations and Comments:	

Firmware test case 5: Volume knob

Test method:

The tester will be playing a constant note through the SoundHub. Next the tester will turn the volume knob clockwise and then counter clockwise.

Pass Criteria:

As the volume knob is turned clockwise, the volume of the note will increase. As the volume knob is turned counter clockwise the volume of the note will decrease.

Test Summary	
Results	
Pass	Fail
Observations and Comments:	

Firmware test case 6: Streaming one to one

Test method:

Using a mobile device and a SoundHub on the same network, the tester will stream a song from a mobile device to the SoundHub.

Pass Criteria:

The stream should be played with no noticeable differences between the SoundHub’s playback and playback locally.

Test Summary	
Results	
Pass	Fail
Observations and Comments:	

Firmware test case 7: Streaming one to many

Test method:

Using a mobile device and two SoundHubs on the same network, the tester will stream a song from a mobile device to the network of SoundHubs.

Pass Criteria:

Both SoundHubs will play the stream in sync with no noticeable differences between the SoundHubs' playback and playback locally.

Test Summary	
Results	
Pass	Fail
Observations and Comments:	

Firmware test case 8: Playback functionality

Test method:

Using a mobile device and a SoundHub on the same network, the tester will stream a song from a mobile device to the SoundHub. While playing the tester will use to mobile device to change the volume, seek ahead in the song, pause the song and restart the song.

Pass Criteria:

When the user changes the volume on the mobile device, this change is reflected in the volume coming from the SoundHub.

Test Summary	
Results	
Pass	Fail
Observations and Comments:	

Firmware test case 9: RoomFlow

Test method:

Using a mobile device and a two SoundHubs some distance apart but on the same network, the tester will turn on RoomFlow and stream a song from a mobile device to the SoundHub. While playing the user move very close to one of the SoundHubs and then move to the other.

Pass Criteria:

The music stream will only play out of the SoundHub which is closest to the mobile device..

Test Summary	
Results	
Pass	Fail
Observations and Comments:	



E.3 Software Test cases

Arimus Audio's SoundHub Evaluation Sheet

Test Summary

Test ID: _____ Prepared By: _____

Review By: _____ Test Date: _____

Additional Notes:

Results

Approved

Conditional approved

Disapproved

Retest

Observations and Comments:

I certify that this test procedure has been performed according to the documentation, and that the results recorded are accurate and complete.

Tester Name: _____ Signature: _____ Date: _____

Witness Name: _____ Signature: _____ Date: _____

©Team Arimus Audio, 2014

This document is licensed to Team Arimus Audio and cannot be used, reproduced, published and/ or revealed without prior written authorization.

Archived Date:

Software test case 1: Android Compatibility

Test method:

The tester will compiled the final version of codes on the Android SDK, and run the application through an Android device which is connected through USB port.

Pass Criteria:

Upon successful installation on the Android device, the application should be running without crashing and the GUI is correctly presented

Test Summary	
Able to compile and run on an Android Device	
Results	
Pass	Fail
Observations and Comments:	

Software test case 2: Graphic User Interface Buttons

Test method:

While the application is running and is in the default state, the tester will click each of the buttons appeared on the GUI.

Pass Criteria:

All buttons should be clickable without crashing the application.

Test Summary			
Pause/continue	Repeat song	Seek ahead	Next song
Results			
Pass		Fail	
Observations and Comments:			

Software test case 3: SoundHub Detection and Connection

Test method:

The tester turns on the Wi-Fi for both the Android device and SoundHub so they are connected to the identical Wi-Fi network.

Pass Criteria:

The Android device should be able to detect the existence of SoundHub through the application interface and connect to them

Test Summary	
SoundHub detection	SoundHub connection
Results	
Pass	Fail
Observations and Comments:	

Software test case 4: Playback Control

Test method:

Once the Android device and SoundHub are connected on the same Wi-Fi network, the tester will stream a song from the Android device to the SoundHub. While playing the tester will use to mobile device to pause/continue the song, seek ahead in the song, and restart the song.

Pass Criteria:

The SoundHub should be able to recognize the commands and perform the corresponding actions to pause/continue the song, seek ahead in the song, and restart the song

Test Summary			
Pause/continue	Repeat song	Seek ahead	Next song
Results			
Pass		Fail	
Observations and Comments:			

Software test case 5: Volume Adjustment

Test method:

The tester will be playing a constant note through the SoundHub. Then the volume up/ down buttons will be pressed in single clicks on the source device

Pass Criteria:

As the volume up or down buttons are pressed from the source device, the volume for all connected sinks will be changed accordingly.

Test Summary	
Volume Up key	Volume down key
Results	
Pass	Fail
Observations and Comments:	

E 4. Integration Test Cases

Integration test case 1: Music streaming quality

Test method:

The tester streams sample music such as heavy bass and soprano from a mobile device once all components of the system are integrated

Pass Criteria:

The wireless streamed music should retain its quality over different frequency range

Integration test case 2: Streaming latency

Test method:

The tester streams sample music from a mobile device, and clicks pause/continue while streaming.

Pass criteria:

The wireless streamed music should respond to user's command through the mobile application within five seconds

Integration test case 3: Power consumption

Test method:

A powermeter will be used to measure the power consumption of SoundHub during streaming

Pass criteria:

The system should not exceed the power rating on the AC converter

Integration test case 4: Streaming stress

Test method:

The tester streams a longlast sample music

Pass criteria:

Stream music to the SoundHub for a period of twelve hours and determine the rate of dropped music during that duration