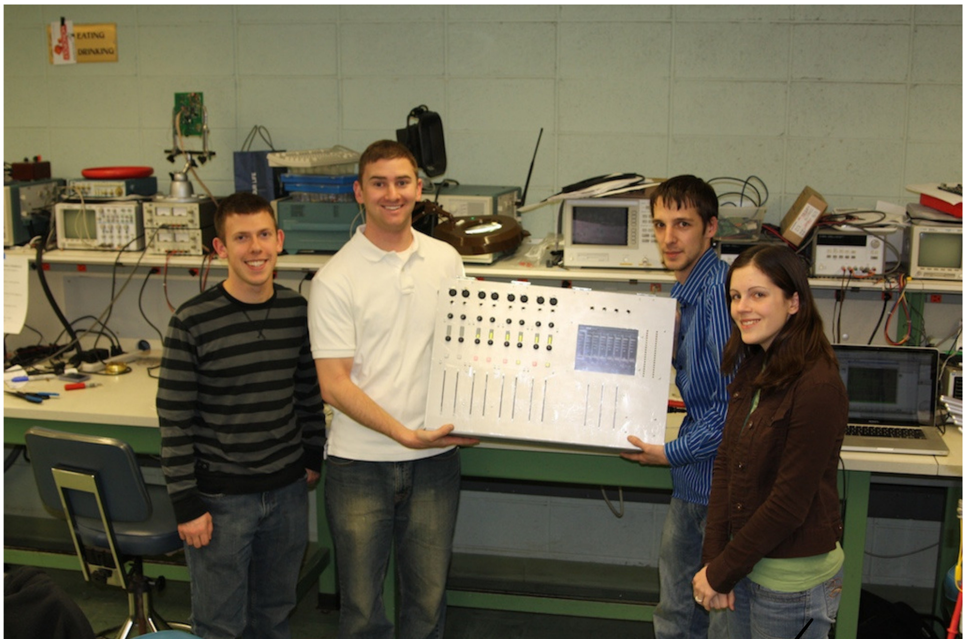
ECE 477 Final Report – Fall 2009
Team 2 – The "Drink Mixer"



Team Members:

#1: Levi J. Cowsert **Signature:** ____________ **Date:** 12-14-09

#2: Adam Johnson **Signature:** ____________ **Date:** 12-14-09

#3: Susanne Schmidt **Signature:** ____________ **Date:** 12-14-09

#4: David Estes **Signature:** ____________ **Date:** 12-14-09

CRITERION	SCORE	MPY	PTS
Technical content	0 1 2 3 4 5 6 7 8 9 10	3	
Design documentation	0 1 2 3 4 5 6 7 8 9 10	3	
Technical writing style	0 1 2 3 4 5 6 7 8 9 10	2	
Contributions	0 1 2 3 4 5 6 7 8 9 10	1	
Editing	0 1 2 3 4 5 6 7 8 9 10	1	
		TOTAL	

Comments:

TABLE OF CONTENTS

Abstract

The Drink Mixer fulfills all the audio mixing needs of the aspiring Digijock(ette). An eight channel audio mixer, it was envisioned to provide as many conveniences as possible to the user without sacrificing mixing abilities (including effects such as panning, reverb, and delay). Setting adjustments are displayed on an LCD screen. After adjusting gain and mix levels using the easy-to-use user interface and a USB keyboard, scene settings can be saved and recalled. The Drink Mixer is user-friendly, reliable, and capable of fulfilling (almost) all of the user's mixing dreams.

1.0 Project Overview and Block Diagram

The "Drink Mixer" is a digital audio mixer with individual input equalizer control as well as master output control. The goal of this project is to create a great sounding board with low noise and effects processing capability. The prototype will have eight (mono) input channels, right and left main mix output, and two auxiliary mix outputs. Each channel will have its own set of equalization, gain, and pan controls, as well as independent fader control for the main and auxiliary mixes. It will also be capable of adding effects as well as saving and loading scene settings. A photo of the prototype is shown as Figure 1. Also, Figure 2 shows the project's overall hardware structure.

Figure 1: "Drink Mixer" Prototype

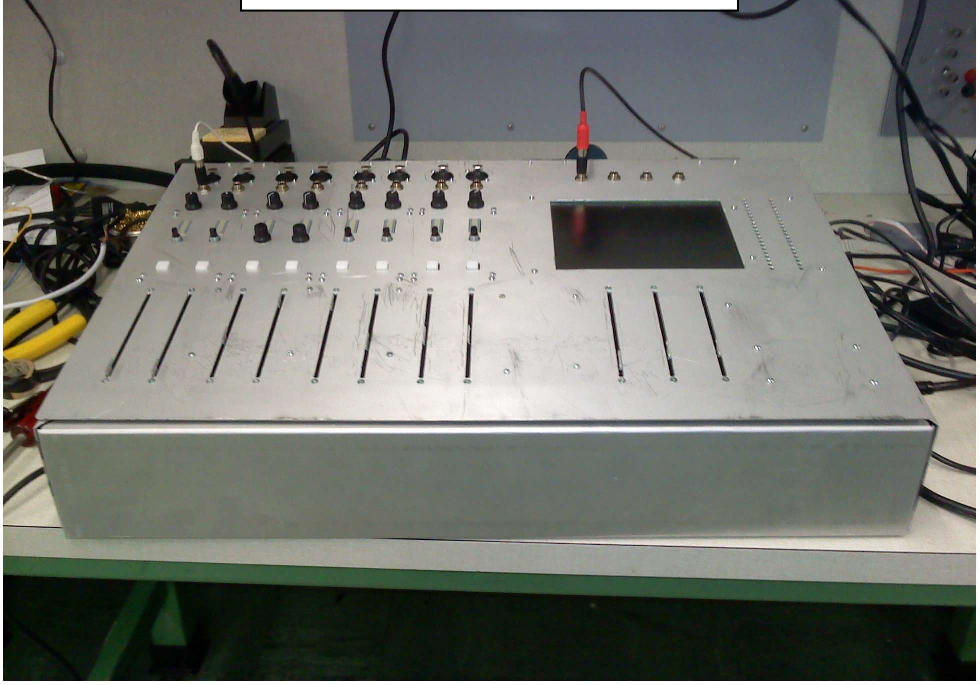
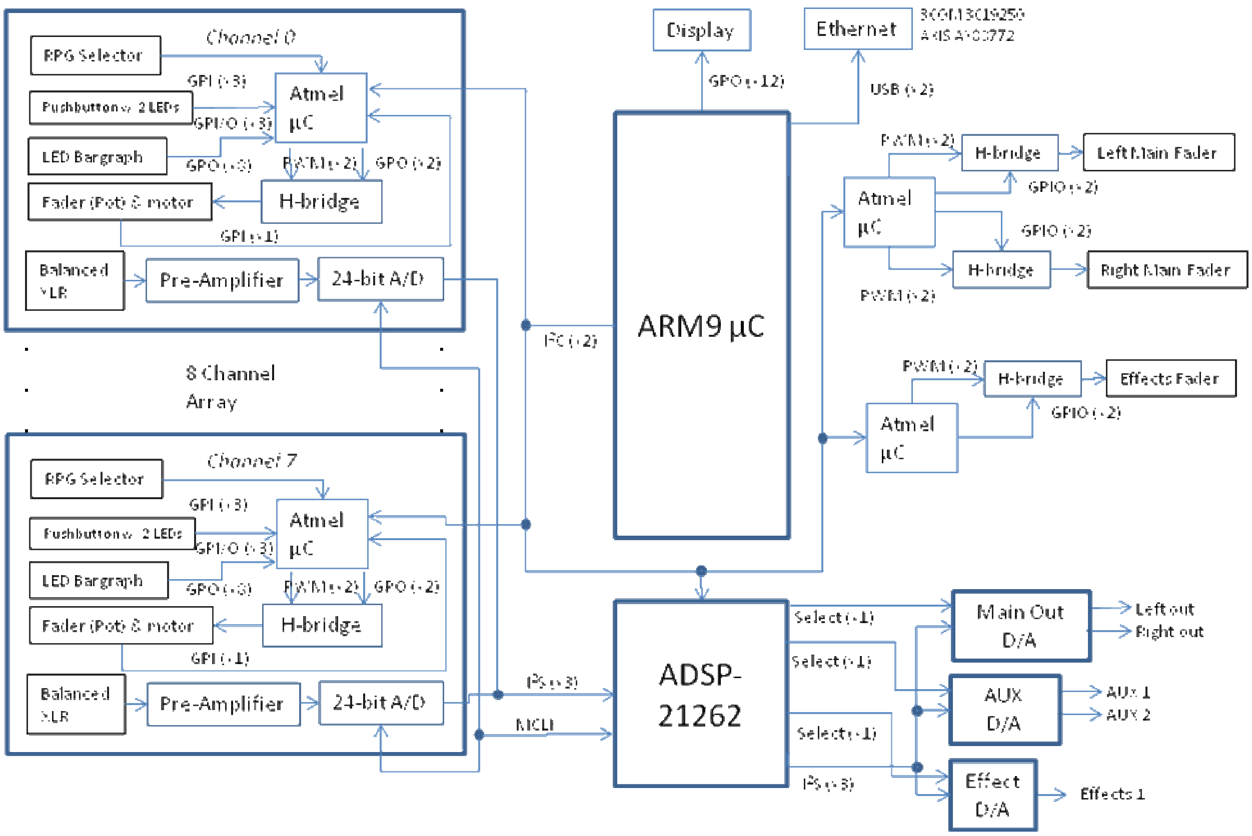


Figure 2: The “Drink Mixer” Block Diagram



2.0 Team Success Criteria and Fulfillment

2.1 An ability to digitally mix audio and adjust individual levels

This success criterion was not fulfilled. The team was unable to get audio working on the DSP selected for the project. This was partially due to a lack of time, since the majority of the team's time was spent on fulfilling other success criteria, and partially due to technical challenges. Technical challenges faced include that the DSP was one-time-programmable, and therefore needed to be boot-loaded from an embedded Linux processor. This interface and bootloading scheme needed to be developed, and slowed the team down.

2.2 An ability to adjust individual equalizer settings for the input channel.

This success criterion was fulfilled. Users are able to change settings for each input channel using the controls for each channel and a USB keyboard, and these settings are updated in the main processor.

2.3 An ability to display channel settings on an LCD display.

This success criterion was fulfilled. Users are able to adjust controls, and the LCD display will show the current setting in real time. Users can also use keyboard shortcuts to navigate between settings for main and auxiliary mixes.

2.4 An ability to save and load scene settings (from flash or EEPROM)

This success criterion was fulfilled. Users are able to adjust controls, then use a keyboard to navigate menus and save the settings. Users can then adjust the controls to different settings, and return to their saved settings by using the load function.

2.5 An ability to display amplitude of output signal.

This success criterion was not fulfilled. The necessary hardware and software for this success criterion was created, but without the ability "to digitally mix audio and adjust individual levels" it is impossible to demonstrate this success criterion.

3.0 Constraint Analysis and Component Selection

3.1.0 Introduction

Some of the constraints with this project involve a high bandwidth constraint for mixing 8 channels of audio together simultaneously. This puts a limit on what kinds of DSP processors we can use as well as what kind of A/D and D/A converters we can use. With motorized faders, custom designed pre-amplifiers, and several digital components there are a lot of design issues to consider and adjust for throughout project development. In the project we will be interfacing a micro-controller for each channel to perform small user interface tasks. These micro-controllers will be interfaced with a main processor devoted to coordinating tasks with all the other 8 processors, as well as adjusting settings on the DSP controller. While it is possible to accomplish a lot of these tasks on one heftier processor, we believe that by separating the user interface tasks from the sound processing, we can increase the reliability of audio conversion and reduce any types of possible hiccups in the system.

Project Success Criteria Include:

- An ability to digitally mix audio and adjust individual levels
- An ability to adjust individual equalizer settings for the input channels
- An ability to add an effect to a channel (i.e. delay / reverb)
- An ability to save and load scene settings (from flash or EEPROM)
- An ability to display level of output signal

3.2.0 Design Constraint Analysis

In this report several design constraints will be considered. These constraints include channel processors, main processors, and digital signal processors. It will also cover information regarding selection of the best A/D Converters as well as D/A Converters. Due to the use of multiple micro-controllers in the project each section will be split up by subsections to discuss the individual requirements for each set of micro-controllers.

3.2.1 Computational Requirements

Individual Channel Controller
The primary tasks of these micro-controllers do not entail large amounts of computational requirements. The primary purpose of each chip is peripheral management including rotary pulse switches, motor driven faders, and buttons.

Digital Signal Processor

The digital signal processor will have to be capable of processing 8 channels of audio input at 24bits and 44kHz. These calculations could be enormous depending on FIR filter detail as well as post processing effects. Based on filter designs and testing in MATLAB we have determined that the optimal filter detail will be 64^{th} order. This means that 8*64 * 44100 = 22.6 million floating-point calculations per second will have to be performed on the digital signal processor at a bare minimum.

Primary Interface Micro-controller

The primary user interface micro-controller will require fewer calculations, most of which will be generating new filters and sending that information to the DSP.

3.2.2 Interface Requirements

Individual Channel Controller

The individual channel micro-controllers will be doing most of the user interface management for the main channel interfaces. This particular sound board design is different in that rather than having 6 potentiometers to maintain there is one rotary pulse generator and a series of push buttons to select which property is being edited. It was discussed that one button could be used to toggle through the different actions but it would present usability issues with the operator and reduce the speed at which a property could be adjusted. This micro-controller will also have to be able to interface with the primary micro-controller to notify it of value changes as well as change channel values if the primary controller instructs it to do so. This becomes useful when using scene control or switching output modes for fader control. In order to accomplish all these tasks this micro-controller will have to interface via I^2C with the primary controller and SPI for the LCD display control.

Primary Interface Micro-controller

The primary micro-controller will be required to interface over I^2C to all other micro-controllers for control management. The SPI bus will be used to interface with the SHARC DSP processor which will act as an SPI slave. The SPI Channel will also be used to interface with the supported Ethernet controller. This Ethernet controller is not in the PSSCs and is strictly being built onto the board for future upgradability in the area of remote control of system. The display uses the ARM9's built in display controller requiring 12 pins.

Digital Signal Processor

The DSP will be required to process up to 8 24-bit audio streams at 44kHz sample rate and simultaneously output to up to 4 24-bit output channels. Support for the I^2C data bus for communicating with the primary controller would be ideal, but not necessary. Other options include SPI and emulated I^2C bus.

3.2.3 On-Chip Peripheral Requirements

Individual Channel Controller

- 1 8-bit A/D Converter (Fader Readout Control)
- 2 PWM Generators (Fader Motor Drive Control) (1 per direction)
- I^2C Bus (For Communication with primary microcontroller)
- 1 Pushbutton Input (to turn each channel on or off)
- 1 Rotary Pulse Inputs (used to adjust channel level in dB)

Primary Interface Microcontroller

- 2-pins I^2C Bus
- 3-pins SPI bus x2 (Chip supports 2 SPI channels)
- 2 SPI-Select Pins GPIO (1 Ethernet, 2 DSP)
- 12-pins LCD 7.1" Panasonic TFT Display Interface
- 2-pins A/D Input for Touch Screen
- 2-pins RS232 Serial Interface
- 2-pins USB Master Controller (future expansion and firmware updates)
- 2-pins USB Slave Controller (future expansion)

Digital Signal Processor

- 3-pins Serial TDM Input
- 3-pins Serial I^2S Output
- 3-pins SPI Interface with Microcontroller and A/D Converters for configuring.
- 10-pins LED Bar Graph of Amplitude Output of Signal (PSSC 5)

3.2.4 Off-Chip Peripheral Requirements

Individual Channel Controller

- H-Bridge for Motor Drive Control on Fader
- Rotary Pulse Encoder
- LED Bar Graph
- Pushbutton with two integrated LEDs

Primary Interface Micro-controller

- LCD Color 8-bit TFT Display [1]
- Ethernet Controller
- A/D input with Touch Screen interface.

Digital Signal Processor

- 8 24-bit A/D Converters I^2S / TDM Mode
- 2-4 24-bit D/A Converters I^2S / TDM Mode

3.2.5 Power Constraints

Most power related constraints involve the need for an extremely low noise power source as well as noise suppression capabilities. This noise suppression will reduce incoming noise in the preamplifiers from potential back EMF from the fader motors as well as various switching noises produced by the digital controllers. The “Drink Mixer” will require 4.68W at +/-15V for the preamplifiers, 105.6W at 9V for the fader motors, and 18W at 5V for everything else. The 5V power rail will also be stepped down to 3.3V and 1.2V to power the DSP. All of this will be supplied from a standard 120VAC household jack.

3.2.6 Packaging Constraints

The mixer console needs to be of an appropriate weight such that it is portable and can be transported safely by one person. To do this, the final package needs to weigh 50 lbs or less. The console also needs to be made of material tough enough to withstand use by Liberal Arts majors. This is because Liberal Arts majors are the people that tend to run mixer consoles. The console shall be laid out such that it is functional with the input channels and faders on the left side, and the master controls, effects, and displays on the right side.

3.2.7 Cost Constraints

The commercially available product that is most similar to the “Drink Mixer” is the KORG Zero 8. Is has a list price of $2,450 [2]. Because we are only accounting for the development cost of parts, and not labor, this is a price that we should be easily able to beat. We are shooting for a total parts cost around $1000.

3.3.0 Component Selection Rationale

Individual Channel Controller

The mixer's individual channel microcontroller must run at 16 MHz, have an A/D converter, a PWM channel, at least 10 IO pins (more would enable hardware self-checks), and an I^2C bus. Two possible candidates are Atmel's ATmega32A [3] and Microchip's PIC16F1934 [4]. Both have multiple 10-bit A/D converters, PWM channels, and over 30 I/O pins (when not used for other purposes). The Atmel has slightly more channels for each of these peripherals. Both chips will also run on 5.5 Volts, and are available as both dual-in-line and quad-flat-pack. While the Atmel can run at 16 MHz when used with an external crystal, the PIC has an internal oscillator which is factory calibrated and can run at 32 MHz with 1% error. However, the most important difference between the chips is the Atmel's use of its TWI (Two Wire Interface), which is a proprietary version of I^2C. However, I^2C libraries exist for Atmel processors, and Chuck Barnett, the lab attendant for ECE477, has past experience with this functionality. Based upon his encouragement, and the confidence that comes from extensive documentation available for the ATmega chipset and compiler, the ATmega32A was chosen for the individual channel microcontroller.

Primary Interface Micro-controller
Tin Can Tools' ARM9 based Hammer [5] was compared against Atmel's AT32AP7000 [6] for the primary microcontroller. Both devices satisfy the on-chip peripheral requirements listed in section 2.3 and support embedded Linux. Although the AT32AP7000 has DSP instructions and a 16-bit stereo audio DAC, no one on the team has any experience in programming the device. Not only do we already have a Hammer in our possession, but we also have experience with programming it. Also the AT32AP7000 has 160 GPIO pins, an unreasonable amount compared to the 30 on the Hammer.

Digital Signal Processor
In the selection for the digital signal processor we chose to compare the Analog Devices ADSP-21262[7] and the Texas Instruments Tms320Dm355Zce270 [8]. The Texas Instruments device operates at 270MHz while the Analog Devices operates at 200MHz. The Texas Instruments is actually built on top of an ARM9 modified core but has very limited on board ram of 32KB. Whereas the Analog Devices is built off a SHARC RISC core and has 2Mb of on board ram. This memory limitation alone will be sufficient for the amount of DSP we will be performing and give us headroom for possible effects. It is also important to note that we have access to the necessary development kits for the SHARC processor and not for the Texas Instruments DSP.

3.4.0 Summary

The "Drink Mixer" will use a main processor interfaced with a microcontroller for each channel and a DSP. The main processor will be devoted to fetching and processing data from the individual channel processors (over I^2C), and adjusting settings on the DSP (using SPI). The individual channel processors will perform user interface tasks specific to each channel, and the DSP will perform audio mixing and effects processing. For the main processor, the ARM9 based Hammer was chosen. The ATmega32A became the individual channel processor, and the DSP will be an SHARC ADSP-21262. With these tools, we will be able to create an audio board with low noise, motorized faders, capability to process effects, and good-sounding output.

4.0 Patent Liability Analysis

4.1.0 Introduction

The “Drink Mixer” is a digital audio mixer. Its features include individual input equalizer control and master output control. Naturally, other entities may have had similar design goals in the past, and if so, there may be patents on the design we are creating. The goal of this section of the report is to identify any patent liability issues the Drink Mixer may encounter should it enter production and market.

4.2.0 Results of Patent and Product Search

There are many patents for audio mixing devices. Most fall within one of two types. The first type consists of patents on a method or device for audio mixing. Usually these patents describe how signals are processed, the number and type of inputs and outputs, and include a detailed block diagram. The second type consists of patents on the “ornamental design” of a mixing device. These patents generally contain many figures of the mixer’s packaging and user interface layout. The following is an analysis of several patents which are representative of the variety of audio mixing patents in the U. S.

Patent #1: Integrated Audio Mixer, Issued Nov. 28, 2000
This patent was issued for “an integrated, multi-input audio mixer…digitizing the analog input signals, digitally processing and mixing the digitized input signals and producing both digital and analog representations of the mixed inputs” [9]. The patent is for a particular type of signal processing and mixing used in a device. Its pages are mostly filled with block diagrams indicating the particular niche of mixing that the patent protects. For this particular patent, the inventors are protecting the ability to do certain computations on analog signals and then converting them to digital versions to do other transformations. Specifically, “unlike the prior art, which requires that all analog inputs be applied to individual full…analog-to-digital converters, the present invention applies each analog input…only to the first sub-component of a traditional…analog-to-digital converter”[9]. The “Drink Mixer” does not implement a signal processing scheme identical to that presented in the patent. Also, many diagrams of “prior art” are referenced by the patent. The prior art describes a general form of additive signal mixing more akin to the signal processing of our device.

Patent #2: Audio Mixer, Issued Apr. 20, 1999
This patent describes an audio mixer which has an effects processor whose output is separate from its “dry output” (mixed audio without effects) [10]. In the patented design, the audio mixer has “separate dry mix, effects returns mix and main mix mixing buses wherein each mixing bus provides a separate mix output,”[10]. While this design is interesting and could have potentially

posed a problem for us, the “Drink Mixer” is similar in design to the prior art which is referenced in the patent, and not the patented signal processing scheme.

Patent #3: Digital Audio Mixer, Issued Feb. 5, 2002
This patent is an example of a patent on “the ornamental design for a digital audio mixer as shown and described” [11]. The document consists primarily of figures detailing the device’s physical appearance and control layout. The patent is not for the processes that the device carries out, but rather for the unique physical design that the user interacts with. There are many examples of similar patents, each with a somewhat different user interface and package shape or style.

4.3.0 Analysis of Patent Liability

Because our design is an audio mixer, many examples of similar devices exist in the market. There are numerous examples of prior art, and so the basic concept of our device, a machine that performs additive operations on signals, is not patentable. Patents (like the Integrated Audio Mixer) do not include the basic concepts that the “Drink Mixer” employs, and so there is no infringement. Also, the “Drink Mixer” does not infringe upon patents for ornamental designs of mixers, unless there exists somewhere a patented audio mixer with packaging and user interface identical to our project.

4.4.0 Action Recommended

The Drink Mixer’s design is such that it does not infringe upon any patents known to the design team. Therefore, no legal action is necessary. This is because an audio mixer is based mostly upon prior art, which is not patentable, or, if it ever had been, is long expired. If we change our design to incorporate a novel and interesting feature (something more “inventive” than adding audio signals together and performing a few basic effects on them), then further research will be necessary to determine if the new features infringe upon any existing patents. For now, we seem to be legal.

4.5.0 Summary

The “Drink Mixer” is free of patent infringement, both literally and under the doctrine of equivalents. This is because the concept of mixing audio is rather old, and cannot be patented as it is not novel or interesting (in a legal way at least). This was demonstrated by citing patents of several types and discussing the reasons that the “Drink Mixer” does not infringe upon them. Patents on “ornamental design” are specific to a certain manufactured product’s packaging and control layout and do not pose a threat of infringement. Patents describing audio mixing systems reference the “Drink Mixer’s” functions as prior art, and so cannot claim infringement.

5.0 Reliability and Safety Analysis

5.1.0 Introduction

The Drink Mixer's audio processing and user interface functions require lots of processing power. To perform all of this, the Drink Mixer has a variety of microprocessors which communicate with each other. There is one monitoring the channel interfaces, another that is the brain of the operation, and another which actually does all of the digital mixing and calculations. These three are all critical components for reliability. Additionally, there is significant current draw on the 5V rail through a linear regulator. This will cause a large amount of heat, which will cause the regulator to fail faster, and is of concern. Another component susceptible to failure is the motorized fader, as it is a moving part that operates by a motor and is belt driven, however this will not be further analyzed due to its failure being mechanical in nature instead of electrical. Finally, there is danger of a sudden jump in volume causing damage to speakers or (more importantly) to ears.

5.2.0 Reliability Analysis

- ADSP-21262 SHARC – This is the processor used to actually perform the calculations and do the digital "mixing" of the different audio inputs. It has 144 pins on it, which increases its chances of failure. Also, since it is a processor, it is a critical component, and we need to know how long we can rely on it before failure. If this component fails, the entire product is useless until replaced.

Parameter name	Description	Value	Comments
C_1	Die complexity	0.56	32-bit/40-bit floating, MOS
π_T	Temperature coeff.	1.5	From datasheet, T_J is ambient temperature + 65°C. Using 35°C, T_J is 100°C
C_2	Package Failure Rate	0.078	Nonhermetic 144-pin
π_E	Environment Factor	2.0	G_F = "Fixed Ground"
π_Q	Quality Factors	10.0	COTS Equipment
π_L	Learning Factor	2.0	Production Availability 3/19/10, thus not yet in production
Entire design:			
λ_p	Predicted number of failures per 10^6 hours	19.92	
MTTF	Mean Time to Failure	50,200.80 hours = 5.73 years	

- ARM9 – This is the processor that is fondly referred to as the “brain” of the operation. It takes all of the input information from the ATMega32As, and tells the SHARC what to do with the audio signals, and is comprised of 40 pins. It also directly controls the LCD screen, and processes the user input on the touch screen. Just as the SHARC, if this component fails, the entire product is useless until replaced.

Parameter name	Description	Value	Comments
C1	Die complexity	0.24	32-bit, assumed to be Bipolar as the ARM9 is built for its reliability.
π_T	Temperature coeff.	1.5	Using $T_J = 100^oC$
C_2	Package Failure Rate	0.019	Nonhermetic 40-pin
π_E	Environment Factor	2.0	G_F = “Fixed Ground”
π_Q	Quality Factors	10.0	COTS Equipment
π_L	Learning Factor	1.0	In production >2.0 years
Entire design:			
λ_p	Predicted number of failures per 10^6 hours	3.98	
MTTF	Mean Time To Failure	251,256.28 hours = 26.68 years	

- ATmega32A – This is the processor used for the individual channel interface, comprised of 44 pins. It monitors the user inputs on each channel, and relays information to the Hammer for processing. There are also 10 of these within the package, one for each of the faders, with the main L and R faders being controlled by the same processor. It is not as much of a critical component as the DSP or Hammer, but if it fails, then that particular input channel is no longer usable.

Parameter name	Description	Value	Comments
C1	Die complexity	0.14	8-bit, assumed to be MOS as it is the conservative number.
π_T	Temperature coeff.	1.5	Using $T_J = 100^oC$
C_2	Package Failure Rate	0.022	Nonhermetic 44-pin
π_E	Environment Factor	2.0	G_F = “Fixed Ground”
π_Q	Quality Factors	10.0	COTS Equipment
π_L	Learning Factor	1.0	Assumed in production >2.0 years

Entire design:			
λ_p	Predicted number of failures per 10^6 hours	2.54	
MTTF	Mean Time To Failure	393,700.79 hours = 44.94 years	

- 5V Linear Regulator – It is anticipated that this will be the hottest part within the package. Although it will be connected to a very large heat sink, heat will still be a contributor to device failure. There are actually two of these regulators, one producing 5V output for 5V devices, and another acting as an intermediate step to power the 3.3V and 1.2V regulators. If the 5V rail is not active, then all devices on the channel boards, the Hammer, and the Display will all be unable to function. If the intermediate regulator fails, the SHARC will be unable to function properly.

Parameter name	Description	Value	Comments
C1	Die complexity	.01	Contains 15 bipolar transistors
π_T	Temperature coeff.	16	Using $T_J = 100^oC$
C_2	Package Failure Rate	0.0012	Nonhermetic 3-pin
π_E	Environment Factor	2.0	G_F = "Fixed Ground"
π_Q	Quality Factors	10.0	COTS Equipment
π_L	Learning Factor	1.0	Assumed in production >2.0 years
Entire design:			
λ_p	Predicted number of failures per 10^6 hours	1.624	
MTTF	Mean Time To Failure	615763.55 hours = 70.29 years	

- According to this analysis, the weakest link in the design is the ADSP-21262 SHARC Processor. The failure rate of this processor could be cut in half if it had been in production for more than 2 years. However, the current revision of it has not even gone into production at the time this document was written. The earlier revisions of the chip have been in production for a while, but the samples that we obtained are of the new version, and as such were calculated as being in production for less than 0.1 years. Also, some of the products may have been tested to some sort of MIL-SPEC that was not easily found. I simply assumed a value of 10 because they are all COTS equipment. The type of transistors within the ATMega32A and the SHARC are also unknown, and were assumed to be MOS as this was the conservative number. The final consideration is that T_J was calculated by assuming an ambient temperature of 35°C. This takes into account any heat created by the surrounding air. However, the typical ambient temperature is 25°C, which would decrease the value of π_T. Since the analysis was performed on the critical processors, there are not any design elements that could be modified or improved upon to improve the reliability of the design other than using different processors. Doing this would cause a complete redesign of the entire project, and as such is not a viable option.

5.3.0 Failure Mode, Effects, and Criticality Analysis (FMECA)

The schematics for the Drink Mixer's multiple boards have been broke down into functional blocks, and can be found in Appendix A. Part 1 is the "brain" Hammer ARM9, part 2 is the DSP, part 3 is the ATMega32As, part 4 is the channel interface peripherals, part 5 is the power supply, part 6 is A/D and D/A, and part 7 is the audio preamp. Many of the possibly failure conditions for each functional block and the possible causes of each are included in Appendix B.

To determine the reliability of the entire system, three different degrees of criticality have been defined. Any failure that concerns the safety of the operator or any bystanders is defined as being a high criticality failure with an industry standard rate of $\lambda_p < 10^{-9}$. This type of failure could result in direct injury to the user, and may also cause further damage to the product and its components. Some examples of this may be power supply failures, shorts causing fires, or audio levels being turned up so far that they cause hearing damage to those near the output speakers. The other types of failure that have been defined are medium and low criticality failures. These are not determined by a value of λ_p, but rather by the consequence of the error. Failures of this type will not cause injury, but may cause parts of the device to not work or cause a nuisance. Examples of a medium criticality failure would include the touch screen malfunctioning, or the DSP or ARM9 not working properly. A low criticality failure would be something similar to a single RPG not working properly, or the LED bar graph indicator on the channel not working properly, or the On/Off button not working. These simply serve as an inconvenience to the user, and may cause dissatisfaction to the user. For a more complete list of failures, please refer to Appendix B: FMECA Worksheet.

5.4.0 Summary

The purpose of this document is to provide an analysis of the reliability and safety of the drink mixer. Each of the different microprocessors contained within the product were critically analyzed, and their mean time to failure was calculated. It was also determined that the weakest link in the product is the ADSP-21262 SHARC Processor, with the highest failure rate. The schematic has been broken up into several different functional blocks, and each of these blocks analyzed for critical failures. Each of these critical failures has then been looked at and a probable cause determined, along with its severity and consequences. It has also been determined that there are incredibly few possibilities of an error or malfunction that could cause harm to the user or bystanders. As a result of this, almost any error that would occur is simply a nuisance or functionality error.

6.0 Ethical and Environmental Impact Analysis

6.1.0 Introduction

In regards to ethical considerations, care must be taken to ensure that the Drink Mixer functions as it is supposed to. Also the user must be warned of any action that could result in injury to themselves or the device. In regards to environmental concerns, the current prototype contains materials that could be harmful to the environment. However, any future devices can be greatly improved upon and as long as the user follows disposal instructions, no harm should be done.

6.2.0 Ethical Impact Analysis

As with all manufactured products, it would be unethical to release a product that does not function as advertised or causes harm to the user. Once the hardware of the device has proved functional and reliable, testing must be done to ensure that all of the software works as it should. Each channel must be tested with combinations of effects to ensure that all effects can work in any combination. There must also be tests done with all of the channels in use with a variety of effects. There is a chance that there will not be enough memory if certain effects, such as delay, are used on every channel. If such a case exists and cannot be fixed, the user must be warned that the device may not function properly under these cases. Software will be written to determine whether or not there is enough memory to process all of the user's effects on all of the channels. If there is not, then the user will be advised to change their chosen settings. The user manual will include more details if the user would like to understand the specifics on why their request cannot be processed.

There is a chance that the user may decide to prevent the fader from moving to the location specified by the Hammer. This would result in excess wear to the fader motor and the fader could cease to function. Software has been implemented to prevent this from happening. If the fader motor is unable to move the fader to the specified position after a few seconds, the motor will stop and the Hammer will be notified of the current fader position. There has been mention of user harm if the H bridges were to burn up and cause the entire device to burst into flames from the inside out. This can happen if two PWM channels are actively trying to move the fader in opposite directions. However, software changes have ensured that both PWM channels cannot be turned on at the same time. Thus, the H bridges and any nearby flammable components will be safe.

If the device were to be damaged by water or some other liquid, various components may short circuit and the entire device could cease to work. There is also a potential that the user could be shocked if this were to happen. The user must be cautioned in the user manual that this device is meant to be used in an indoor environment and should be kept away from liquids. The user could also be shocked if they were to open the device and begin messing around with the components inside. There will be a warning in the user manual and may also be a warning label specifying that there are no user serviceable parts inside, so the device should not be opened.

The packaging of the device is made from aluminum and may have sharp edges. If these edges cannot be filed down, it is possible that the user could cut themselves on the sharp edges. If this is the case then plastic brackets or smoothed down metal will be added to create a smoother and more user friendly edge

There will be a warning label on the power supply. The user must be warned not to open up the power supply, obscure the vent openings, or stick things into the vents. In the user manual, the user must also be notified that the cord leading from the power supply to the device should be placed in such a way to discourage tripping over the cord or in any way removing the cord. They will be warned not to use the device if the cord is frayed or compromised in any fashion.

6.3.0 Environmental Impact Analysis

Frank Splitt has put a great deal of emphasis on engineering in respect to environmental solutions, to the extent that he believes “environmental factors need to be considered at the beginning of every engineering problem [18].” It may be a credit to the revised ABET standards, the fact that our generation is more environmentally minded, or simply just the nature of our project that the Drink Mixer is an environmentally friendly device.

The current prototype of the Drink Mixer contains a fair amount of lead due to the eight PCB’s that are contained within it. Lead is also contained in the solder used to attach components to the boards. Lead is extremely hazardous to the health of humans, plants, and animals; contact in humans occurs most often through inhalation or ingestion of lead [19]. With this prototype, there need only be warnings not to open the packaging, but for future devices the PCB’s can be made with a lead-free solder finish at no additional charge [20]. Lead free solder can also be used to attach the components to the PCBs. These precautions will prevent lead from entering into the systems of any users or from polluting the environment in any way. The reliability of the lead free solder may be a possible ethical issue, but reliability is sure to improve with time and should not be an issue with this project [21].

During normal use the Drink Mixer will not adversely affect the environment. The user will be advised to turn the device off when not in use in order to conserve electricity, but there is no reasonable chance of harm while the device is running.

The packaging is made entirely of aluminum and can be easily recycled by the user. Aluminum does not break down through recycling and most places that recycle aluminum cans will also recycle scrap aluminum [22].

The touch screen LCD contains mercury within the CCFT [23]. Mercury is extremely poisonous, hazardous to the environment as well as the user. The user will be notified in the user manual that the screen will need to be disposed of properly through household hazardous waste collection centers [24]. Locations of collection centers for each state can be found on the website for the Environmental Protection Agency.

6.4.0 Summary

The Drink Mixer has the ability to become an ethical and environmentally friendly product. This will be dependent on how vigilant we are in testing the device to ensure that there are no issues with the software. It will also rely on how faithful the user is to usage and disposal requirements. If the device is used and disposed of as specified, there should be no harm caused to the user or the environment.

7.0 Packaging Design Considerations

7.1.0 Introduction

The Drink Mixer‘s product packaging is designed to be functional and portable, yet still maintain an aesthetic appeal. In this section of the report, several products with similar packaging requirements are compared with the Drink Mixer. By comparing the features of these products to those of the Drink Mixer, an appropriate packaging design is formulated.

7.2.0 Commercial Product Packaging

There are many commercially available products that are similar to the Drink Mixer, but the two below have some particular characteristics that we either wanted to improve upon, or that gave us motivation for some of our ideas. The two products are the Soundcraft Si Series, of which the Si2 [25] is shown below, and the Korg Zero8 [2]. Many of our inspirations came from Soundcraft, while many of the things that we wanted to eliminate came from the Korg. Ultimately, we wanted to recreate the functionality of the Soundcraft, but in the packaging size and shape of the Korg.

7.2.1 Product #1

The first product to compare to is the Soundcraft Si Series Mixers. The first thing to notice is the shape of the mixer. Instead of being a flat surface, there is a flat surface for the fader, and an upright surface for the display and EQ control knobs. We are utilizing a flat surface overall because it is easier to move quickly from faders to control knobs on a flat surface than going between different surfaces. Another thing to notice is the number of channels. Due to the simplicity needed in order to complete the project within 1 semester, we are only doing 8 channels. The commercial products offered by Soundcraft have varieties from 16 – 64 channels and beyond. The Si series also has input channels on both the left and right sides, with the master controls in the center. Our design will have the inputs on the left side, with the masters on the right side.

The touch screen display on the Si Series is fairly small, as seen in the picture. This is mostly because of the purpose of the touch screen on the Soundcraft. The touch screen is not used for much more than typing names for storage, and navigation of the display itself. There are many push buttons on the commercial mixer to do functions which we have incorporated into the touch screen in our design of the Drink Mixer. While we are utilizing a touch screen as well, ours is much bigger and used more functionally. Instead of having a panel full of buttons and a small screen, we will have a large screen with the buttons incorporated into the screen. One feature of the Si series console that we are trying to somewhat duplicate is the individual display on each channel. The Soundcraft board uses a full color display to show which mode the RPG is in, and the exact level. While we are not displaying all of this information directly on the channel, we are displaying an approximate level of the current parameter being adjusted on an LED bar graph. This will allow for quick adjustments without having to look back

at the main screen unless a precision adjustment is needed. Another important thing to note is the location of the XLR and ¼” input and output jacks. On the Soundcraft they are on the back, while on the Drink Mixer they are on the top. The decision to put audio jacks on the top of the case was based on improving access to the Drink Mixer’s interior, while being able to integrate each individual channel separately on its own PCB. The final thing to note about the packaging here is that the commercial product is made of a combination of sheet metal surfaces and molded plastic ends and corners. Due to the highly custom nature and high cost of molded plastic parts, our design will not incorporate this.

7.2.2 Product #2

The second product to compare is the Korg Zero8 8 channel digital mixing console. The first thing to note is that this mixer has 8 channels, which we are duplicating in our design. It is also a single flat surface with the individual channels on the left and the masters on the right, which we are also mimicking. The Korg also has a small touch screen like the Soundcraft, and we will be replacing the small one and the buttons surrounding it with a large touch screen with incorporated buttons. Probably the biggest difference between the Korg and the Drink Mixer is individual channel controls. The Korg has individual knobs for each level of EQ and each auxiliary “monitor” mix on each input channel. The Drink Mixer is incorporating EQ and pan control into a single RPG on each channel. The auxiliary mixes and main mixes are incorporated into a single fader. The function of fader and RPG will be selected via the touch screen. The individual faders will then all move to the preset locations for their selected mix. The input and output jacks on the Korg are also on the back like the Soundcraft, whereas we will be putting them on the top of the console on the Drink Mixer. The final thing to note about the Korg is the size and shape of the faders and the plastic knobs.

The Korg uses 60mm faders with short, stubby knobs. We will be using 100mm faders with tall, skinny knobs. Also, the Korg does not have master faders, but instead has a single master volume knob on the top right of the console. We feel that it is imperative that a good mixing console have master faders, which we have incorporated on the Drink Mixer.

7.3.0 Project Packaging Specifications

The packaging of the Drink Mixer is fairly straightforward. The primary piece of the packaging is the case. It will be constructed of folded sheet metal, either aluminum or steel. The top of the case will be hinged at the front, and screwed together at the back. This will allow for easy access to the interior of the case should it be necessary to modify anything.

There are eight individual channels on the left side of the board, all identical. At the top, there is an XLR balanced input jack and an unbalanced ¼" input jack, followed by a red LED clip light and a rotary potentiometer to control the level of the preamp. Beneath this is a 10 segment LED bar graph and an RPG. The LED bar graph displays the approximate level of the parameter being adjusted by the RPG. Next is the channel on/off button with integrated G/R LED followed by the solo button and red "solo active" LED. Finally, there is a motorized K-fader which is used to adjust the mix volume of the channel in either the main mix, or one of the auxiliary "monitor" mixes.

On the right half of the board, there are several key functional components. Along the top are balanced XLR outputs for the main stereo mix and the mono auxiliary mixes. Beneath that is a 7.8" color touch screen display. This display will show which mode the RPGs are in for each channel and the exact value for the mode, as well as show the current mode of the faders. The display will be the user interface for storing and loading presets, as well as selecting effects. To the right of the screen is an LED sound level indicator for the main mix, as well as a ¼" headphone jack. Beneath the display are the effects and master left and right faders.

There is a master power switch on the rear of the console, along with a polarized plug for transformed AC power input. This will then be converted to several different DC voltages internally.

7.4.0 PCB Footprint Layout

The drink mixer will use multiple PCBs linked together to make the final product. Every two input channels will share a PCB containing two ATMELS, motorized K-Faders, LED bar graphs, RPGs, and On/Off Button (with integrated LEDs). There will also be two PCBs containing one stereo ATD for every two channels, the preamp circuitry for every channel, the AC to DC circuitry, and the six-channel DTA. This will be done by using a stereo ATD and passing one channel as the right and the second channel as the left. The last user interface PCB will be mounted to the back side of the touch screen and will contain 40 LEDs for the main mix level indication, as well as the ARM9, the SHARC, and two ATMELs to control the master and effects faders. Finally, there will be a power supply PCB in the bottom of the case.

7.5.0 Summary

In this report, the packaging design for the Drink Mixer was discussed. Several other products that had similar packaging were shown to compare their features to those designed into the "Drink Mixer" project. There are several drawings in Appendix B to show the packaging design in a more descriptive way.

8.0 Schematic Design Considerations

8.1.0 Introduction

A significant amount of hardware is required to accomplish the tasks of the Drink Mixer. Not only are there components with a critical need of noise reduction, there are also motorized fader components requiring drivers and high current draw. In addition to these issues there is the added complexity contributed by the vast array of LEDs and rotary pulse encoders. All these systems must be able to communicate in order to adjust the various settings of the audio mixer.

8.2.0 Theory of Operation

The following sections will describe all the major subsections of the audio mixer and how they interact with each other.

Master Board Subsection
The master board is considered the brain of the system. This is where all of the user interface information as well as incoming data is received and processed. For this reason the Hammer ARM9 is on this board as well as the SHARC ADSP-21262. These 2 chips connect out to the other subsections for their specific purposes. While the ARM9 is acquiring user interface data, the DSP is processing the incoming audio streams and sending them to the outputs. In order for the ARM9 to communicate to the DSP to notify it of processing changes the DSP runs in SPI slave mode at 15MHz and is told what to do by the ARM9 after it has finished processing any user interface changes. This separation of processes gives the benefit of maximizing the use of the features of the DSP and reducing any chances of user interface processing from slowing and possibly distorting audio output. The ARM9 will also be responsible for displaying user pertinent information on the LCD display and taking feedback from the keyboard or mouse.

Also on the main board are two ATMEGA32A micro-controllers. These are located here to control and monitor the master faders and effects faders for output control. These talk to the on board ARM9 via an I^2C interface, which runs at 100kHz. Jumper pins connect the I^2C interface to the four dual channel subsections described later.

As for audio data interaction, the DSP has two I^2S busses that are connected over jumpers to the analog processing subsection, which will be described later.

Power Supply Subsection
The power supply subsection contains all the circuitry to generate the power rails for the Drink Mixer. All the power supply related circuits will be placed on a separate PCB and power will be sent out to all the other subsections from terminals which connect to headers.

Dual Channel Interface Subsection

The dual channel interface subsection is used to handle motorized faders, LEDs, RPGs, and other user interface related parts. On each of these boards there are two ATmega32A micro-controllers. These micro-controllers each handle a channel and each one of them will be flashed with code that is unique to their ID number and I^2C address. Each channel interface consists of a LED bar graph for displaying the control's estimated levels as well as an RPG for adjusting parameters and the motor-driven fader. Each fader motor requires an H-bridge in order to be interfaced with the Atmel. There are a total of four of these subsections in the design and all communicate with the Main Board over an I^2C data bus. Through the I^2C bus, each ATmega32A will be able to get instructions about where to move faders to. The I^2C bus will also be used to update the ARM9 with user interface changes. This will be accomplished through the use of several registers: one for fader position, one for button presses (with a buffer of three button presses), one for the LED bar graph, one for turns of the RPG. These registers can be read and reset periodically, so that user interface events can be processed quickly.

Analog Processing Subsection

The Analog processing subsection is where all the analog audio inputs are processed and digitized as well as where the outputs are processed and converted to analog. In order for this to happen, each channel must have its own balanced preamplifier circuit. This circuit consists of a modified custom design [26] using JFET transistors as the inputs. The high impedance input and low noise are the two reasons for the JFET usage. Package mounted female XLR connectors will feed these input channels.

For every two channels there exists a dual input A/D converter. This means that there will be a total of four of these in this subsection. These are Analog Devices AD1871 analog to digital converters. All of these will be configured to run in TDM mode for the DSP as well as a daisy chained SPI. This means that the command sets can be sent into them like a daisy chain / shift register. This SPI interface is used to initialize and configure the A/D converters. This SPI bus will interface to the ARM9, which will take on the task of configuring the A/D converters at power on. These A/D converters then use an I^2S bus in TDM mode to send data back to the main board subsection.

Also on the board are the AD1852 24-bit D/A converters. These will be used to convert the digital output signals to an analog output. These in turn will be sent to basic pre-out amplifiers to balance the signal for master output. These receive their data and are initialized over the same interface as the A/D converters with the main subsection.

8.3.0 Hardware Design Narrative

Main Board Subsection ARM9
The Hammer ARM9 interfaces with the other processors on the "Drink Mixer" using both SPI and I^2C. It is also important to note that the video out support of the Hammer ARM9 will be fully utilized to drive an 8-bit color display.

As far as the I2C bus is concerned the SDA and SCL pins are sent to a bidirectional level shifting circuit. This is due to the 3.3V board operation and the ATMEGA processors running at 5V. This I^2C bus will be used to receive updated interface information as well as send updates to the individual channel interface microcontrollers.

The Hammer ARM9 we are currently using has two SPI ports. Both of these ports can potentially be used, but the primary SPI bus (SPI0) pins will be used to initialize the A/D converters, D/A converters, as well as communicate with the DSP. The other SPI module on the chip could be used in a future iteration of the "Drink Mixer" to interface with a SD Card.

In this design, the TX0 and RX0 pins will be utilized and routed to a level shifter for RS232 communication. This is the primary means for programming and debugging the Hammer.

Main Board Subsection DSP SHARC ADSP-21262
The Analog Devices SHARC digital signal processor has the primary task of processing data input and sending it back out to the D/A. This processor has four TDM mode serial interfaces. In the design only two out of these four will be needed and turned on accordingly. The other system worth mentioning is the slave mode pin that will have to be tied high so that the SHARC knows to operate its SPI interface in slave mode. This will allow the Hammer ARM9 to dish out commands to the DSP. The rest of the pins will be used for general purpose I/O to show the amplitude variations with 40 LEDs.

Individual Channel ATmega32A
The ATMEGA32A micro-controller, as mentioned earlier, will be used to control most of the user interface switches. This will involve the ATD0 pin enabled as an analog to digital converter to monitor the fader value. Also two PWMs will have to be used to operate the H-bridge for the motor. There is no H-bridge enable pin. The last important pins to mention are the TWI (I^2C) pins. These pins will have to be set up to operate in slave mode and switch to master mode upon an addressing call from the Hammer ARM9. While in master mode the ATmega32A will send requested information back to the Hammer ARM9 before resuming slave mode. The rest of the pins will be utilized as GPIO for reasons such as monitoring an RPG and displaying readouts on a LED bar graph.

8.4.0 Summary

In this report theories of operation as well as hardware narratives were discussed. Information regarding the individual hardware subsections of the device as well as the pin configurations on the microcontrollers was explained and discussed.

9.0 PCB Layout Design Considerations

9.1.0 Introduction

Because the Drink Mixer manipulates audio signals, the primary concern when designing printed circuit boards is noise immunity. Other major issues include fitting large analog circuits using the smallest possible board space, and placing user interface components in alignment with packaging.

9.2.0 PCB Layout Design Considerations – Overall

The main PCB required proper positioning for external ports such as the USB port and the RS-232. The LEDs and faders on the board also had to have precise positioning in order to make sure that they lined up exactly with the cutouts made for them in the top panel of the packaging. Another consideration was the VIAs required under the LED drivers for thermal dissipation [27]. The biggest consideration for the dual channel interface PCB was space. The board had to fit within an area of about 3" x 4.5". In addition to the space constraint, the RPG, LED bargraph, and pushbutton had to be the only components on the top side of the board in order to allow for correct positioning when mounted into the top panel.

The audioboard PCB also needed to be kept a reasonable size. Eventually, this PCB was split into two PCBs for economic reasons (two smaller boards cost less than one board which is over 60 square inches). In addition to that, the preamplifiers needed to remain close to the A/D pins and XLR input headers. There also needed to be separate grounds for the analog and digital ground, which were connected with a zero ohm resistor (which acts to reduce noise).

The power supply PCB has large traces and linear regulators that need to be lined up to attach to a heat sink. However, one of the power regulators does not have a ground tab, and so is attached to a separate heat sink. This PCB will be in a separate enclosure from the main packaging because of noise immunity and size considerations.

9.3.0 PCB Layout Design Considerations – Microcontroller

Using the Hammer simplifies our board layout because the ARM9, as well as its oscillator circuit, SDRAM, Flash, and power regulators are already contained within the prepackaged Hammer board. The DSP, however, was a different story. It has a JTAG interface that requires that PCB trace lengths be the same. If the lengths of these traces are over six inches, resistors must be used as well [28]. An external oscillator was chosen for this microchip instead of a crystal in order to keep continuity with the development kit (no code changes will need to be made from prototype to PCB, which simplifies debugging). There were also special requirements for bypass capacitor placement on the DSP. Certain bypass capacitors needed to be placed "as close as possible" to the DSP's reference pins [29].

The Atmega32A's were not as complicated to route as the DSP. Noise immunity is not critical for these chips, although bypass capacitors are still needed. All microcontrollers on our design have bypass capacitors on each side. As a precaution, almost all of the microchips have extra pins routed to headers in case changes need to be made and the pins need to be used later on. The DSP and ARM9 also have reset pushbuttons, while the Atmega32A's use headers instead because of space considerations.

9.4.0 PCB Layout Design Considerations – Power Supply

When laying out PCBs, the first traces to be routed were usually power and ground. They needed to be routed to all areas of the board, and must be thick enough to withstand the maximum operating current. Nowhere was this more apparent than for the H-bridges used for fader motor operation. They consume high peak currents, so some of their traces are 30 mils for safety. Ground planes also reduce noise and help with thermal dissipation. The audio PCB has ground planes for audio and digital components which are tied together at a single point with a 0 ohm resistor. The resistor provides a single point of connection between the analog and digital ground planes in order to eliminate possible ground loops. Also, power to the DSP was provided by traces that connect to main power traces at one point via 0 ohm resistors, also for noise reduction. Bypass capacitors were also important in our design, and we had a lot of them since we had 12 microprocessors! Each side of every microprocessor had at least one bypass capacitor (most had two), and (as stated above) the DSP had special bypass capacitor requirements for its reference voltage pins [29].

Since we have several printed circuit boards that require different voltages, power is provided via 18 gauge stranded wire through a system of wire harnesses from the power supply PCB to the rest of the device. From there, the various voltage rails are jumpered to each PCB as needed. On the individual channel PCBs, due to space considerations, traces were routed wherever they could fit. The Analog to Digital converter ICs on the audio PCB are actually routed so that different power rails go to different sections of the chip's footprint. This isolates digital and analog signals. The ATmega32A's have a ground pad underneath them, so ground pins were routed to a VIA under the middle of the chip, which was connected to ground. Filter caps were also routed to this VIA.

9.5.0 Summary

When designing an audio mixer, the main concern is noise immunity. Other concerns include fitting large circuits onto small board space, and placing user interface components in alignment with packaging. Noise immunity practices including ground planes, bypass capacitors, and noise isolation jumpers (known elsewhere as 0 ohm resistors) were used. Circuit board design will mesh with packaging, and was designed with power needs in mind as well.

10.0 Software Design Considerations

10.1.0 Introduction

The Drink Mixer's processing power is provided by several microprocessors. The Hammer ARM9 accepts user input from the touch screen, communicates with each channel through that channel's ATmega32A, and communicates with the DSP. The Atmega32A's load scene settings as specified by the ARM9 and monitor changes made manually to the channel. The DSP accepts, processes, and outputs audio from each channel and updates the specific effects processing according to user input given through the ARM9.

10.2.0 Software Design Considerations

10.2.1 Hammer ARM9

The Hammer runs a real time OS with a Linux 2.6.29.6 kernel [5], programmed in C++. The Hammer interfaces with each of the ATmega32A's through the I2C connection which means the kernel must be configured to enable this part of the device. The I^2C bus speed will be set slow at first, approximately 100kbps, to minimize error in the data. The SPI0 connection will be used to initialize the A/D for each channel as well as the D/A for the output. The same SPI0 connection will be used to communicate with the DSP and the two LED drivers. This interface should be able to run at a maximum speed of 25MHz, however due to the travel length in some areas this device may run at 4MHz. The SPI enable pins for each of these devices are as follows: GPB0, GPB2, GPH0, GPH7, and GPH6. After initializing the I2C connection with the ATmega32A's, the I2C connection will be tested. The QT interface will then be loaded before the Hammer begins its polling loop. The QT interface will be used to update the LCD screen via a frame buffer driver written for the Linux kernel. This interface requires connections to VM, VFRAME, VLINE, VCLK, and LCD_VD0-7 through the SFV20R (ZIF 20-pin connector). The touch screen is connected through the HFWR (ZIF 4-pin) to pins AIN0 and AIN1. This interface will communicate with the frame buffer drivers in the Linux kernel as well as interface directly with the AIN pins to get touch screen settings. The 100ms of sleep at the beginning of the code loop will keep the hammer from occupying 100% of its cycles with processing data and allow time for the QT events to occur. Within the loop, the I2C connection will be checked and the appropriate registers will be updated with changes that have been made to the individual channels. The DSP will then be updated with these changes over the SPI connection. The Hammer will use the amplitude sent by the DSP to update the LED drivers, also through SPI.

10.2.2 ATmega32A

The ATmega32A contains 32kB of flash memory, upon which the application program is stored. There are also 2143 bytes of SRAM,

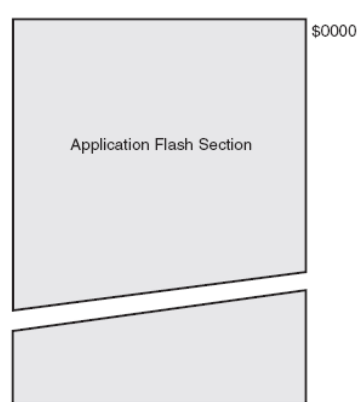


Figure 10.2.2.1 ATmega32A Flash Memory Map

and 1024 bytes of data EEPROM memory [12]. It will be programmed using the AVR-GCC compiler and AVR Studio. Because the development environment manages memory independently from the programmer, a discussion of memory addressing and data storage locations is omitted. The ATmega32A is interrupt driven in order to prioritize the various tasks it has and to control debouncing. Upon power up, initialization routines will initialize the A/D converter as well as set the GPIO settings. The processor will then sleep until an interrupt occurs. Within the timer interrupt, various peripherals will be checked and updated. The fader A/D (connected to pin ADC0) will be checked and the corresponding registers updated. Because it will be polled within the timer interrupt, the A/D will operate in a non-continuous mode. After that, button presses and any changes to the RPG will be checked. The illuminated pushbutton is connected to T0, T1, and ADC1, which are configured as GPI pins. The RPG is connected to TCK and TMS, which are also configured for GPI. A TWI interrupt occurs when the Hammer is sending information over the I2C connection. The I^2C connection is connected to SCL and SDA. If the ATmega32A is reading information from the Hammer, it will set flags for what needs to be changed and then use PWM to move the fader. OC0 and OC2 are used to communicate with an H bridge, which sets the fader position. On one microcontroller, an additional H-bridge is connected to OC1A and OC1B. A 10-segment LED bar graph, connected to TDO, TDI, TOSC1, TOSC2, and ADC2-7 (all configured to act as general-purpose output), will be updated as the Hammer commands. If the Hammer is requesting information, the register value containing current user-input information will be sent.

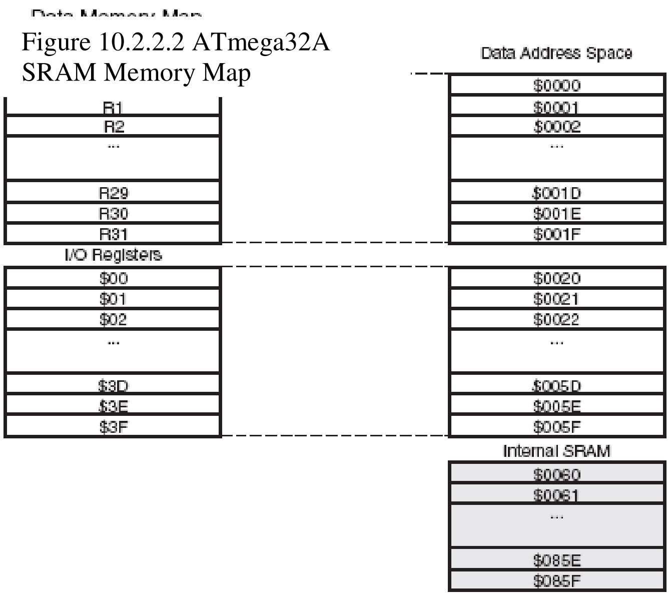


Figure 10.2.2.2 ATmega32A SRAM Memory Map

10.2.3 DSP

The DSP is interrupt driven in order to ensure that audio processing takes place as quickly as possible. Being interrupt driven is actually very common with DSP based devices and is considered one of the only ways to operate them correctly. Within the first interrupt, data will be added to the channel buffer and a filter will be applied. One of the great features of a DSP that gives it great speed improvements is the hardware implemented circular buffer. This buffer allows data to collect while processing is occurring in the main DSP. The data is received from the A/D's through an I2S connection on DAIP1-4 at 24 bits and 96kHz. It will then be sent to the output buffer and the FX interrupt will be fired. External SRAM has been interfaced using AD0-15, NWR, NRD, and ALE. The 512kB external SRAM will enable us the memory to process effects which require a fair amount of extra memory, such as delay. Within the FX interrupt, effects processing will take place before the interrupt for the output being placed in the output buffer is fired. The output is sent to the D/A's through the I^2S connection using DAIP11-14. The SPI interrupt is fired when information from the Hammer is being sent. The DSP receives the command, sends back the rolling peak amplitude for the Hammer to display through the LED drivers and then updates the filter transforms and gain settings according to the user input that has been relayed. The DSP is programmed through the JTAG header, which is attached to TMS, TCK, NTRST, TDI, and NEMU.

IOP Registers 0x0000 0000–0003 FFFF			
Long Word (64 Bits)	**Extended Precision Normal or Instruction Word (48 Bits)**	**Normal Word (32 Bits)**	**Short Word (16 Bits)**
Block 0 SRAM 0x0004 0000–0x0004 3FFF	Block 0 SRAM 0x0008 0000–0x0008 5555	Block 0 SRAM 0x0008 0000–0x0008 7FFF	Block 0 SRAM 0x0010 0000–0x0010 FFFF
Reserved 0x0004 4000–0x0005 7FFF	Reserved	Reserved 0x0008 8000–0x000A FFFF	Reserved 0x0011 0000–0x0015 FFFF
Block 0 ROM 0x0005 8000–0x0005 FFFF	Block 0 ROM 0x000A 0000–0x000A AAAA	Block 0 ROM 0x000B 0000–0x000B FFFF	Block 0 ROM 0x0016 0000–0x0017 FFFF
Block 1 SRAM 0x0006 0000–0x0006 3FFF	Block 1 SRAM 0x000C 0000–0x000C 5555	Block 1 SRAM 0x000C 0000–0x000C 7FFF	Block 1 SRAM 0x0018 0000–0x0018 FFFF
Reserved 0x0006 4000–0x0007 7FFF	Reserved	Reserved 0x000C 8000–0x000E FFFF	Reserved 0x0019 0000–0x001D FFFF
Block 1 ROM 0x0007 8000–0x0007 FFFF	Block 1 ROM 0x000E 0000–0x000E AAAA	Block 1 ROM 0x000F 0000–0x000F FFFF	Block 1 ROM 0x001E 0000–0x001F FFFF

Figure 10.2.3.1 ADSP 21262 Internal Memory Space

10.3.0 Software Design Narrative

10.3.1 Hammer ARM9

At application start the system will first initialize the SPI bus and I^2C interface by setting the appropriate registers. These registers are unimportant in mentioning here due to the fact that they are handled by the Linux kernel automatically. Once these interfaces are initialized the program will start a new process to handle the graphical user interface side of the application. The process prioritizing will be divided up based on the algorithms built into the Linux kernel.

After we have completed the initialization stage of the application a program loop will begin. In this loop a series of modules will be activated after a 100ms wait. The first of the modules will be to check to see if we have data to send out to the ATmega32A's. This data will correspond to changing values on the individual channel peripherals (i.e. moving a fader). If this information exists the I2C module of the code will be instructed to update values of the necessary channels. After this is performed the I2C module will then request updated changes from the channels, which had no new information to go out. This information is then stored in a series of variables. All user interface values are stored on the hammer. After the I^2C system has finished doing its job updated settings will be sent and requested from the QT process. This will allow the user interface to be properly updated as well as get changes that were made through the touch screen interface. This module is not yet completed.

After the communication with the QT interface is complete all the modified settings will be sent out over the SPI interface to the DSP. These settings will be sent out in a series of bytes corresponding to an 8-bit register followed by the assigned value. The DSP will interpret this information and update the effects processing filters. This module is started but not completed.

At the end of this process a request will be made for the amplitude of the left and right output channels from the DSP. These amplitudes will be sent to the Hammer over the SPI interface. The Hammer will then recycle this data and send it out to the LED drivers. The reason the Hammer is performing this task as opposed to the DSP has to do with limited GPIO pins on the DSP and the fact that the Hammer is operating as the master on the SPI bus. The program will then wait for a small amount of time and begin again. This module is not yet completed.

A separate operating process on the Hammer will be the QT user interface. While it is preferred that the QT interface process act as a thread of the main application, it will depend on issues with the QT interface and the other tasks. In the event that they can, then the user settings variables will be mutually accessible and can be used between threads via mutex locking. This interface will display detailed information regarding values on settings of each individual channel as well as effects settings. This aspect of the code has a partial set of completed parts. The initialization code for the various data interfaces as well as the frame buffer driver for the LCD screen is mostly complete. The one complication that is occurring is an outdated embedded Linux support causing failed Linux builds. We have been working with the designer of the Hammer to update this documentation and patch the latest kernel versions.

10.3.2 ATmega32A

The ATmega32A program will operate in an interrupt-driven loop. Using the timer module of the device a series of procedures will be run every 2 ms. these procedures will check if the on/off button has been pressed and set the appropriate flag. It will check the value of the fader position and update the variable corresponding to its value. It will also check to see if the RPG has been rotated and if so which direction it has been rotated. For the RPG a count will be kept with a positive or negative integer. If the number in this variable is positive then the RPG has been turned clockwise that many times and if negative, CCW respectively. This information will be stored for the next interrupt of this application.

On a periodic basis the Hammer will require updated information from the ATmega32A's. This information will be requested via the I2C interface (or in Atmel terms, TWI interface). When the interface receives a request with its corresponding device number an interrupt will be fired. The Atmel will then determine if it is a read request or a write request. If it is a read request the data stored in the variables from the previous module will be sent to the Hammer. If it is a write request, data sent from the Hammer will be stored in those variables. After this interrupt is performed a check will be made to compare the value of the fader with the desired value made by the Hammer. If these values differ, the PWM module will be initialized and instruct the fader to move until the desired value is asserted. The PWM interface is working with the H-bridge; the fader value can be read and asserted.

10.3.3 DSP

At the start of the program the circular buffer is enabled for the I2S interface on the DAI pins. The SPI interface is activated in slave mode to receive commands from the Hammer. Once the input pins are initialized to accept TDM data from the A/D's, the output pins are configured. After this is performed an interrupt table is activated.

The first interrupt is the data input ready interrupt. This interrupt is fired every time a series of 24-bits of data is ready for processing. This data is added to a buffer for its corresponding channel and a star-transform is applied to act as a 128^{th}-order FIR filter. Once this filter is processed it is tacked into a channel output value ready to be added to the main output after all 8 channels are processed.

The second interrupt is the effects interrupt. This interrupt is triggered after all 8 inputs have been asserted. This will combine all the final channel filter values to create the output value and also tack it into an output effects buffer. This buffer will then be assessed and its output sent to the output buffer. After this second interrupt has fired an output ready interrupt will be triggered. This third interrupt will send the final set of output data to the output circular buffer to go to the D/A's.

The fourth and final interrupt involves the SPI interface. This interface will be triggered when a full byte of data is received from the SPI interface. Once this interrupt has accumulated the full number of bits needed for the first byte instruction, some actions will be performed. If settings have been changed on channel equalizer settings then the FIR filter coefficients will need to be updated. This can be done with the DSP's built in FFT optimized functions. If gain settings need to be adjusted then the corresponding variables will be adjusted for processing. The SPI interrupt will be responsible for sending back the LED driver bits containing the amplitude of the left and right channel. So far the DSP code has successfully been able to handle basic gain control, panning, and delay via a talk through style interface. Once the PCB board is complete it will be significantly easier to develop.

10.4.0 Summary

The Drink Mixer's several processors interact to produce the desired audio effects. The Hammer interacts with the touch screen and communicates with the other components of the user interface via ATmega32A's. The DSP receives instructions from the Hammer. The ATmega32A's interpret user interface data and load scene settings. Overall, there is seamless software integration through several microcontrollers to achieve the Drink Mixer's functionality.

11.0 Version 2 Changes

With the completion of the first prototype, it became apparent that the design could be improved upon. The second version of the “Drink Mixer” would include upgrades such as more protection against hardware failure, a touch screen to replace the USB keyboard, and an Ethernet controller to allow remote control functionality.

One of the most inconvenient aspects of the “Drink Mixer’s” operation is the need for a USB keyboard. The keyboard shortcuts are not intuitive, and it is not convenient to carry around a keyboard with the mixer, especially since without the keyboard the mixer loses most of its control functionality. The second version would incorporate software support for a touch screen mounted over the LCD screen. The touch screen is already installed in the first prototype, but its development was not completed due to time constraints.

Another useful upgrade would be the inclusion of an Ethernet controller to the “Drink Mixer.” This would enable features such as remote operation via a computer network. Because the Hammer module runs embedded Linux, it would not be a difficult upgrade to make, as drivers are preexisting.

Another hardware improvement would be the inclusion of protection circuitry to prevent current leakage across the H-bridges in the fader control circuitry. The ATmega32A’s in the current prototype must be programmed without the 9V power rail (which controls the fader motors) connected, because when programming the ATmega32A’s pins are pulled high, which causes the H-bridge MOSFETs to short power to ground. A protection circuit would not only prevent this, but would serve as a hardware backup to the ATmega32A firmware which does not allow the H-bridge to short during normal operation.

12.0 Summary and Conclusions

With the development of the "Drink Mixer," the design team created a digital audio mixer which is the prototype for a great sounding board with low noise and effects processing ability. Though at the time of this writing, the prototype did not have full functionality, it allowed the team to learn a great deal about the complexities of audio system design. A system of microcontrollers was developed that communicated with each other. An intuitive user interface was prototyped and tested. Audio routines were developed, and noise was controlled in every possible way.

The skills we acquired during the development process are also worth mention. Because of the project's complexity, the top-down implementation strategy became so important to the "Drink Mixer's" design. Flowcharts and block diagrams helped tremendously in breaking the project into manageable parts, which could be prototyped and tested independently. Component selection is also a new skill for many of the design team members; navigating component selection tables on Digikey and Mouser will no longer be a challenge in our professional lives.

Knowledge of PCB design was deepened this semester for all of the team members. Learning how to overcome pervasive software glitches in PADS layout and schematic software was key to surviving the semester. Hardware debugging, soldering, and "fly wiring" were new skills for some team members. Finally, learning to program a digital signal processor, deepened knowledge of audio mixer operation and technical requirements, and learning about embedded Linux and inter-microcontroller communication protocols deepened our technical background.

13.0 References

[1] "Hammer LCD 8bit Color STN," Mar. 3, 2009. [Online], Available: http://www.elinux.org/Hammer_LCD_8bit_Color_STN. [Accessed: Sept. 12, 2009].

[2] "Korg Zero8 8-channel Digital Firewire Mixer," [Online], Available: http://www.bananasmusic.com/productdetail.asp/pid_10963/productname_Korg-Zero8-8-channel-Digital-Firewire-Mixer. [Accessed: Sept. 14, 2009].

[3] "ATmega32A," 2009. [Online], Available: http://www.atmel.com/dyn/products/product_card.asp?part_id=4404. [Accessed: Sept. 15, 2009].

[4] "PIC16F1934," Sept. 9, 2009. [Online], Available: http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en538156. [Accessed: Sept. 15, 2009].

[5] "Hammer," [Online], Available: http://tincantools.com/product.php?productid=16133&cat=0&page=1. [Accessed: Sept 15, 2009].

[6] "Atmel AVR32 32-bit Microcontroller AT32AP7000 Preliminary," Sept. 9, 2009. [Online], Available: http://www.atmel.com/dyn/resources/prod_documents/32003S.pdf. [Accessed: Sept. 15, 2009].

[7] "TMS320DM355 Pricing / Packaging / CAD Design Tools / Samples," [Online], Available: http://focus.ti.com/docs/prod/folders/print/tms320dm355.html#symbols. [Accessed: Sept. 14, 2009].

[8] "ADSP-21262: 3rd Generation Low Cost 32-Bit Floating-Point SHARC DSP," Aug 2009. [Online], Available: http://www.analog.com/en/embedded-processing-dsp/sharc/adsp-21262/processors/product.html. [Accessed: Sept. 14, 2009].

[9] Carlos Azeredo Leme et al., "Integrated Audio Mixer," U. S. Patent 6,154,161, November 28, 2000.

[10] Norman F. Williams, Jr, "Audio Mixer," U. S. Patent 5,896,459, April 20, 1999.

[11] Haruki Takita et al., "Digital Audio Mixer," U. S. Patent US D453.327 S, February 5, 2002.

[12] "ATMega32A Datasheet." ATMEL. http://www.atmel.com/dyn/resources/prod_documents/doc8155.pdf. Accessed: 29 October 2009.

[13] "LM78XX Series Voltage Regulators." National Semiconductor. http://www.datasheetcatalog.org/datasheet/nationalsemiconductor/DS007746.PDF. Accessed: 8 November 2009.

[14] "SHARC Embedded Processor." ANALOG DEVICES. http://www.analog.com/static/imported-files/data_sheets/ADSP-21261_21262_21266.pdf. Accessed: 8 November 2009.

[15] "Hammer Board." http://www.elinux.org/Hammer_Board. Accessed: 8 November 2009.

[16] MIL-HDBK-217F. United States Department of Defense. 2 January 1991.

[17] Dudash, Pete. Homework 11: Reliability and Safety Analysis, Two Wheel Deal. 4 April 2008.

[18] F. Splitt, "Environmentally Smart Engineering Education: A Brief on a Paradigm in Progress," in Engineering Education Reform: A Trilogy, 2003, pp. 2.

[19] "Lead Poisoning," [Online] Available: http://en.wikipedia.org/wiki/Lead_poisoning [Accessed: Nov. 18, 2009].

[20] "FAQ – Lead Free PCBs," *Advanced Circuits*, 2007. [Online] Available: http://www.4pcb.com/index.php?load=content&page_id=238. [Accessed: Nov. 17, 2009].

[21] T. Adams and K. Gurnett, "Achieving reliability with lead-free solders" Military & Aerospace Electronics, January 2008. [Online] Available: http://mae.pennnet.com/display_article/317439/32/ARTCL/none/none/1/Achieving-reliability-with-lead-free-solders. [Accessed: Nov. 19, 2009].

[22] "Recycling Aluminum Cans – Fun Facts" *Professor's House*, 2007. [Online] Available: http://www.professorshouse.com/your-home/environmentally-friendly/recycling-aluminum-cans.aspx. [Accessed: Nov. 18, 2009].

[23] "CCFT Tutorial," Sharp Electronics Corporation, 2009. [Online] Available: http://www.sharpsma.com/Page.aspx/americas/en/3e22ebcf-8255-43a5-9091-4969cfd3f85d. [Accessed: Nov. 19, 2009].

[24] "Table of Products that May Contain Mercury and Recommended Management Options," *U.S. Environmental Protection Agency*, October 2009. [Online] Available: http://www.epa.gov/osw/hazard/tsd/mercury/con-prod.htm#t1c8. [Accessed: Nov 19, 2009].

[25] "Soundcraft Si2 Digital Console," [Online], Available: http://www.prosoundweb.com/photos/category/C28. [Accessed: Sept. 22, 2009].

[26] P. Allison and R. Elliot, "Low Noise Balanced Microphone Preamp," May 17, 2008. [Online], Available: http://sound.westhost.com/project66.htm. [Accessed: Sept 29, 2009].

[27] "24-Channel, 12-Bit PWM LED Driver with Internal Oscillator," *Texas Instruments*, Sept. 2008. [Online]. http://focus.ti.com/lit/ds/symlink/tlc5947.pdf [Accessed: Oct. 7, 2009].

[28] D. Doyle, "Analog Devices JTAG Emulation Technical Reference," *Analog Devices*, EE-68, Apr. 15, 2008. [Online]. http://www.analog.com/static/imported-files/application_notes/ee-68_rev10.pdf [Accessed: Oct. 7, 2009].

[29] "ADSP-21261/ADSP-21262/ADSP-21266," *Analog Devices*, July 2009. [Online]. http://www.analog.com/static/imported-files/data_sheets/ADSP-21261_21262_21266.pdf [Accessed: Oct. 3, 2009].

Appendix A: Individual Contributions

A.1 Contributions of David Estes:

In this project I served primarily as the team leader and software lead on the project. I also contributed a significant amount on hardware design. In the beginning of our project I determined what parts would be needed to create the project as well as what software considerations were going to have to be made. The first goal of the project was to get a successful build of the Hammer ARM9 processor made by Tin Can Tools. This actually became quite a challenge as the documentation for it was significantly outdated and the source code for Linux would no longer compile for this board. As a result I had to make significant modifications to the kernel / rootfs (buildroot) as well as rewrite part of the init sequence of the boot loader source code as it was not compatible with gcc-4.4.x series.

The second task I had to accomplish on the Hammer was getting the LCD display to function. This turned out to be much more difficult than anticipated as the s3c2410fb frame buffer driver was also incomplete and incompatible with the screen mentioned on the Tin Can Tools wiki. With the guidance of Dave Anders I was able to get the LCD to function. This screen is a 640x480 7.8” 8-bit STN color LCD display.

The next major contribution I made was preamplifier design. I spent hours researching possible designs and even coming up with my own. After much work on this a modified version of a balanced BJT amplifier was used. This preamplifier was prototyped and proved to have a 40dB gain at 92dB SNR. This preamplifier was both balanced and high impedance which was optimal for our system.

In the project I also contributed significantly on the overall system design. I was responsible for coming up with the communication protocols and design structure for how all the different modules of the project would talk to each other. This included the implementation of the I^2C bus for communicating with Atmels as well as an implementation of SPI bus and I^2S bus. As a result of this I made a significant amount of contributions to the schematic portion of the project.

In the software area of the project I had to work significantly on the fader control system in order to get fader movement to occur correctly. I also had to write a user interface for the main LCD display to display readouts of all the fader positions and equalizer settings for each individual channel. This program not only initialized all the other project peripherals but displayed real time information pertaining to them. I also wrote code to allow the user to save “scene” settings to FLASH memory and restore them on call. When these settings were restored the faders would move back to their previously stored position and the equalizer settings would also restore themselves. This program was also responsible for conveying these changed values to the DSP such that it would adjust audio levels accordingly.

Another aspect of my contribution included writing code for the SHARC DSP. While, as of the writing of this report, audio is currently not functioning significant headway has been made into resolving the issue. Most of the issues have been locked down to too much crosstalk on a wiring harness. However, it was soon discovered the SHARC DSP has a significant learning curve as its programming structure is not anything like a typical processor. We were successfully able to program it and a boot loader was written so that the Hammer would be able to send the full DSP kernel over to it at startup.

Finally, in this project I also served as an audio expert as I have had many years of experience in digital mixing and the professional audio industry. This allowed me to create insight as to what was good for the project and what wasn't.

A.2 Contributions of Levi J. Cowsert:

I served primarily as the hardware, packaging, and power supply lead, as well as one of our audio experts. I determined our packaging needs and designed a case to accommodate them. I determined the distance necessary between channels for all components to fit, and then designed a mounting panel for everything to fit together. I originally created the concept in a 3D Google Sketchup model. Then, after the PCB layout was designed and fabricated, I had an updated spacing of components to go off of for the final panel design. I originally made the layout by hand using paper and pencil, with the assistance of a set of calipers to measure offsets. I then took the drawing to the EE Machine Shop, where they put it into AutoCAD. I then used AutoCAD to design the box part of the case, which I then had the machine shop fabricate. The design sent to the shop was for a squared box that wasn't lopsided, but somewhere along the line, the actually box is not so square. We made it work though.

The second major contribution I made to the project was the power supply. We had very significant power needs, and with that came a very large power supply. Not only do we require 150 Watts, but most of this power also needed to have incredibly low noise. As a result of this, our power needed to be linearly regulated, not switched, thus requiring large heat sinks and careful design considerations. In the end, I determined the best solution was to use a commercial 9V at 30A power supply, and regulate it down to the other required voltages. The 9V supply was used to directly power the motorized faders, as they are the only thing not requiring low-noise power. I then used linear regulators to step down to 5V, and from there to 3.3V and 1.2V. In order to power the preamps, I used a separate center tap transformer and rectifier, then regulated it to +/- 15V.

The final part of the power supply is the power distribution. We have so many different PCBs in different places requiring different power rails, that the power distribution was quite an undertaking. I have 2 different terminal block boards, 1 board is entirely for ground connections, and has 18 ground connections made to it. The other strip contains all of the DC voltages, +/- 15V, 9V, 5V, 3.3V, and 1.2V, and has 29 connections made to it. There are a total of 10 unique power harnesses.

I also made the majority of the data and audio harnesses as well. Each channel has a harness going from the gain potentiometer, the ¼" unbalanced input jack, and the 3-pin XLR connector to the audio board. There is also a harness for the ¼" output jacks. Every motorized fader also had to have a harness made for it to connect to the channel interface board. The channel interface boards also have an I^2C harness and a master reset harness going to each channel board and to the main board. Finally, there are two data harnesses, one for the A/D portion, and one for the

D/A portion that had to be made. These harnesses were actually made and re-made several times as we learned different signals creating crosstalk and inducing noise, and created different ways to cancel out and eliminate these noises. I also provided knowledge on the general idea and foundations of audio mixing, soundboard layout, and equalization of audio.

A.3 Contributions of Adam Johnson:

In our team, I functioned as a hardware and software supporter. I supported David's development of the "Drink Mixer's" firmware, and helped him with miscellaneous debugging tasks. I also assisted in PCB design in a variety of ways, from component selection to trace routing and passive component placement. During the final prototyping phase, I assisted in hardware and software debugging and learned to "fly wire" when necessary.

While David spearheaded the development of the "Drink Mixer's" software, I did smaller supporting tasks using embedded C. I was responsible mostly for the development of the individual channel interfaces. For this task, I considered several processors and selected the ATmega32A because of favorable reviews from other students and ECE staff, as well as because of its peripherals, which were well-suited for our project. I also located a driver for the ATmega32A's TWI/I^2C interface, and helped port it for the WinAVR compiler that I used for coding the "Drink Mixer's" channel interface software. With David's help, I developed an Interrupt Service Routine using the driver to communicate with the Hammer module, and assisted in testing it. Also, I programmed the drivers for the "Drink Mixer's" RPGs, faders, and LED bar graphs.

One of the greatest challenges of this project to me was the development of the fader control hardware. I helped develop the H-bridge control circuitry which the ATmega32A's use to move the motorized faders in our project. A significant setback occurred when it was discovered that the prototype did not work at all after being fabricated in PCB. David and I worked to find a solution, and I modified the hardware accordingly, performed necessary testing, and installed the modified hardware.

When the team was ready to create PCBs, I helped design the main board, and assisted with other PCBs as well. I created the schematic for the SHARC DSP, using the development kit as a model. I also sourced LED drivers for the "Drink Mixer's" amplitude display and created the schematic, and it was my idea to mount the main board under the LCD screen.

Once PCBs were fabricated, I helped teach other members of my team how to solder, and did some soldering myself. As the soldering and testing progressed, errors were detected, and I helped find and implement solutions. By the end of the semester, if there were errors on PCBs, it was assumed that either David or I would "fly wire" a solution.

Finally, my team benefitted from my technical writing. Besides writing the required two homework assignments, I wrote much of the final report, as well as the user manual. This freed up time for other teammates to concentrate on DSP-related tasks that were not finished yet.

A.4 Contributions of Susanne Schmidt:

When our team first started working at the beginning of this semester I helped out with researching parts and editing the reports written by my teammates. As the semester began to pick up, I became quite adept in creating parts footprints in PADS and was responsible for checking all of our footprints against the actual part to ensure that they would be easily soldered onto the PCBs when they came. In PADS I worked on several of the schematics, including the preamps for the audio boards as well as several of the main board schematics. I researched a variety of topics when working on the schematics in order to figure out how to interface everything correctly. This included finding engineering notes specific to the DSP and other various chips such as our SRAM chip.

During our PCB designing phase, I created the first layout for the channel interface PCB and worked on getting that routed. Once all of the PCBs were basically done, I replaced the automatic silkscreen labels on all four of the PCBs to silkscreen labels that contained the value for each resistor and capacitor, component names for transistors and other small components, as well as pin names for all of the headers. I created all of the CAM files on PADs to create the zip files needed to verify the PCBs with the manufacturer. Upon getting the preliminary reports back about PCB errors, I went through each of the PCBs and fixed clearance errors, spacing violations, missing solder mask violations, and insufficient solder mask errors. Some of these errors required changing the locations of traces and VIAs on the boards while others required changing specifications on the CAM files.

Once the PCBs came in, I did a great amount of soldering. After first learning how to solder, I soldered on almost every component on both audio boards, except for the smaller chips. I also soldered the resistors and capacitors onto the channel interface boards. Later when I was more confident in my soldering abilities I was able to solder on smaller chips, such as H bridges, when they had to be replaced. When issues were discovered with the H bridge circuit, I fly wired three of the channel interface boards and replaced the H bridges. I then determined where to cut the traces on the main board and fly wired the H bridge connections there as well. Because the changes to the H bridge circuit required level shifters, I created harnesses to connect each channel to the level shifter and back. Harnesses were also made to connect the fly wires on the main board to the level shifters and back.

In addition to working on hardware, I worked on a fair amount of the documentation. As mentioned previously, I proofread several of my teammate's reports. I also created our design review presentation and worked on much of our final documentation. Adam and I collaborated on writing the final report, the senior design report, and the user manual. I also created the poster. Although I had expected to work more on software rather than hardware, I'm glad that it turned out the other way around. I learned how to solder, how to make harnesses, and how to do a number of other useful hands-on, hardware related abilities. In the beginning I was also able to work on programming with both Adam and David a little bit. I worked with David on experimenting with different effects that could be added using the DSP and with Adam on some of the initial experimentation with getting the ATmega32A's to move the faders. I was later able to help Adam in programming the final code onto the ATmega32A's after the changes had been made to account for the H bridge change.

Appendix B: Packaging

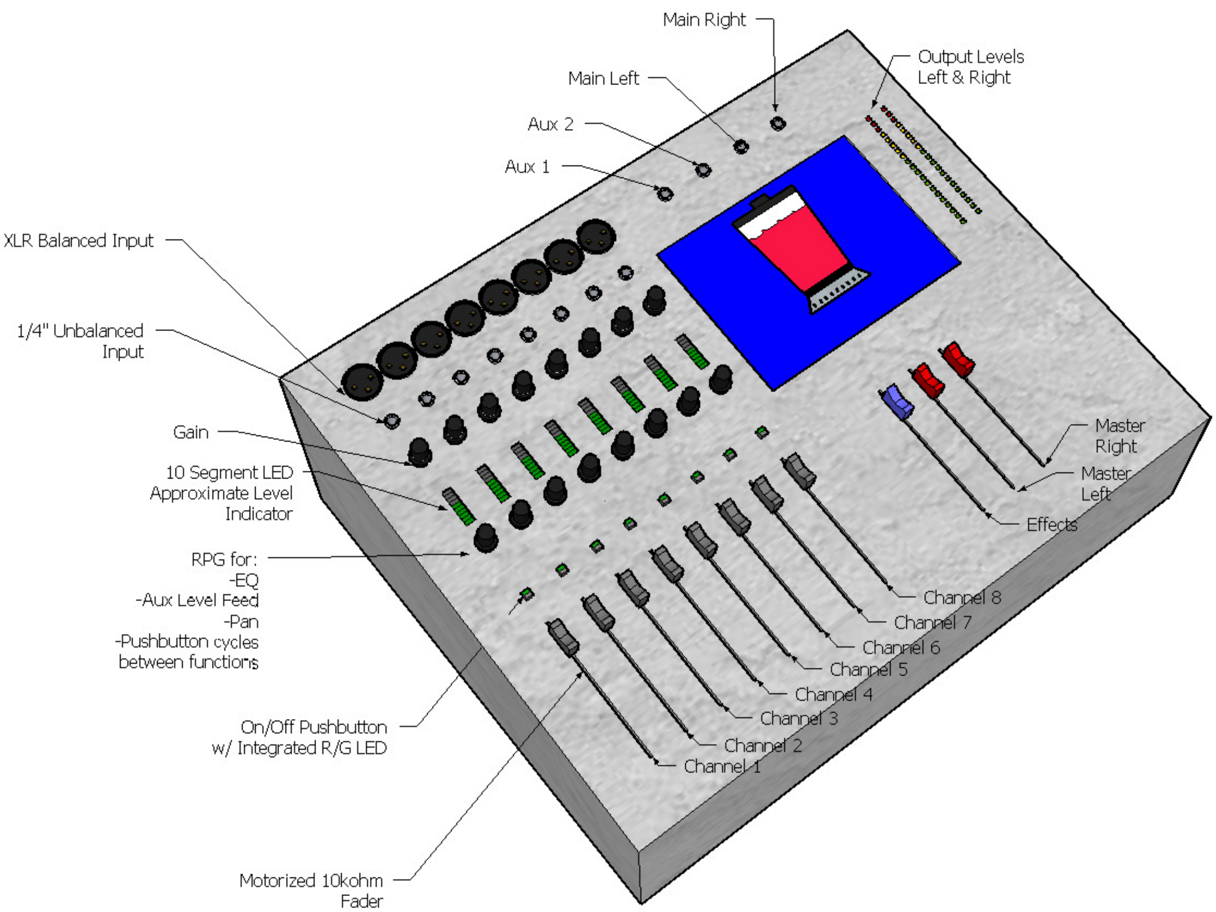


Figure B-1. Top view of product packaging, showing the functions of all controls

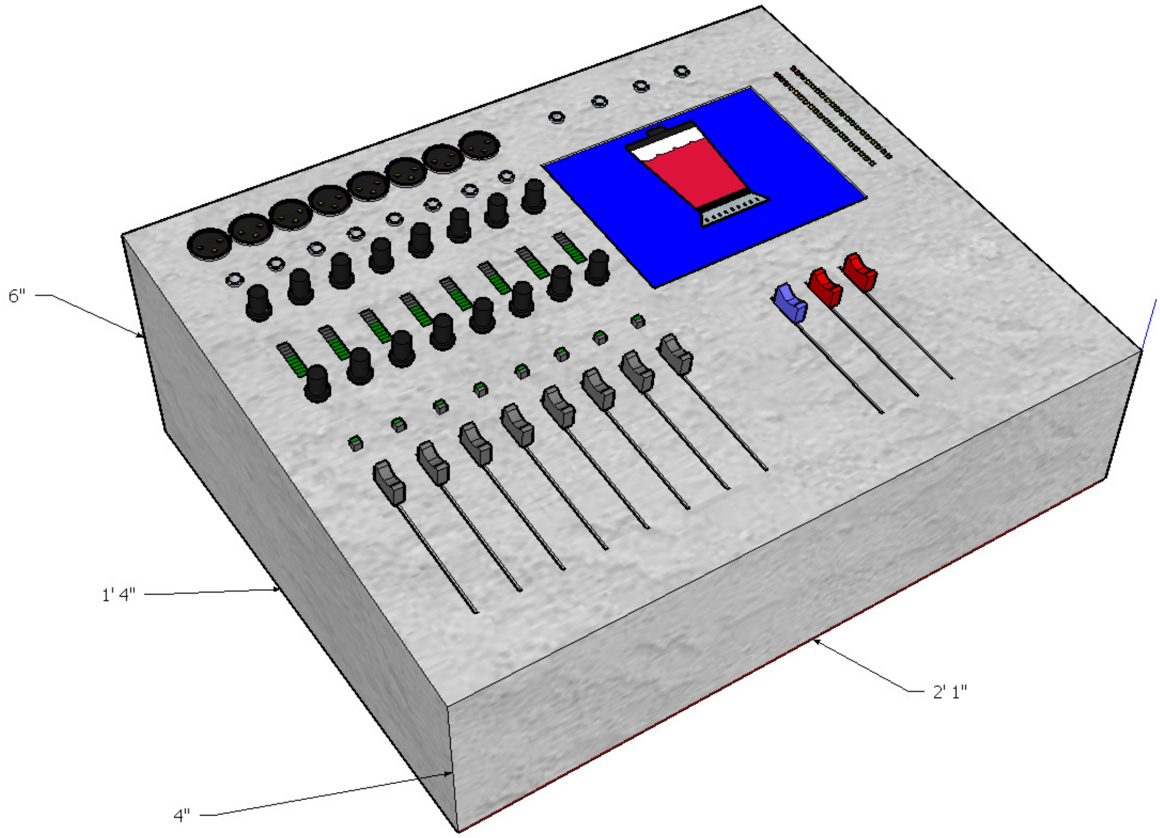


Figure B-2. Top view of product packaging, showing dimensions Drawing is not to scale.

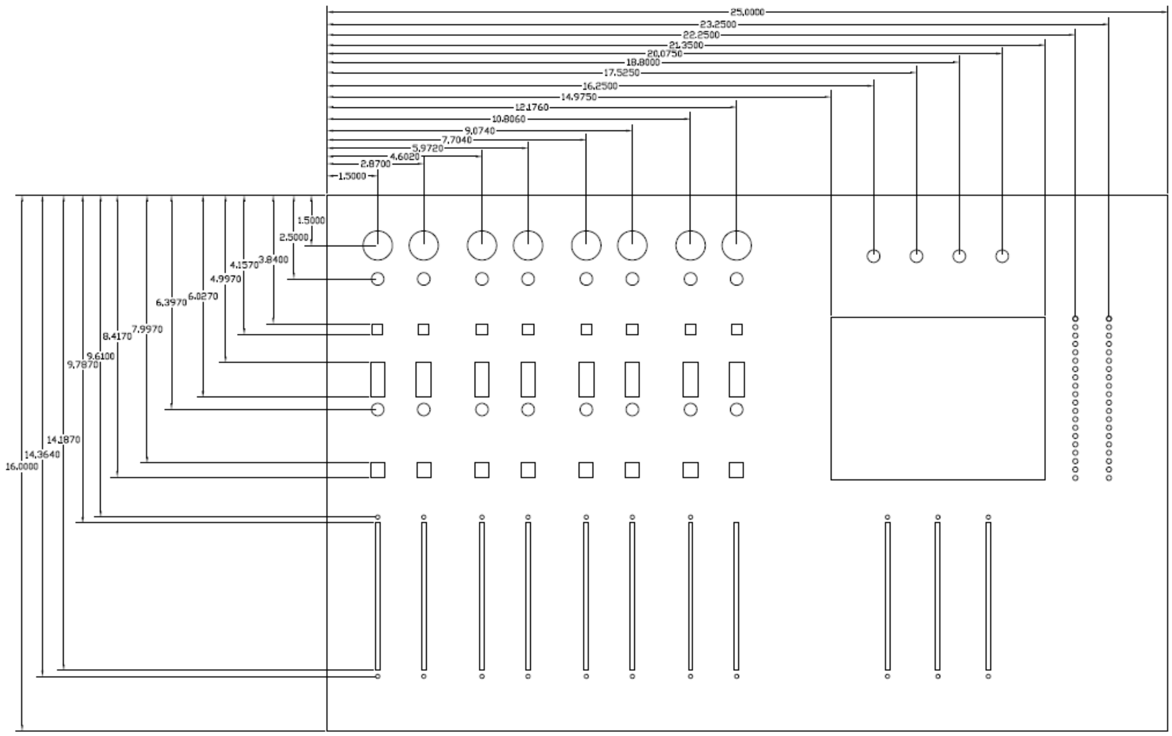


Figure B-3. AutoCAD drawing of top case layout, showing dimensions

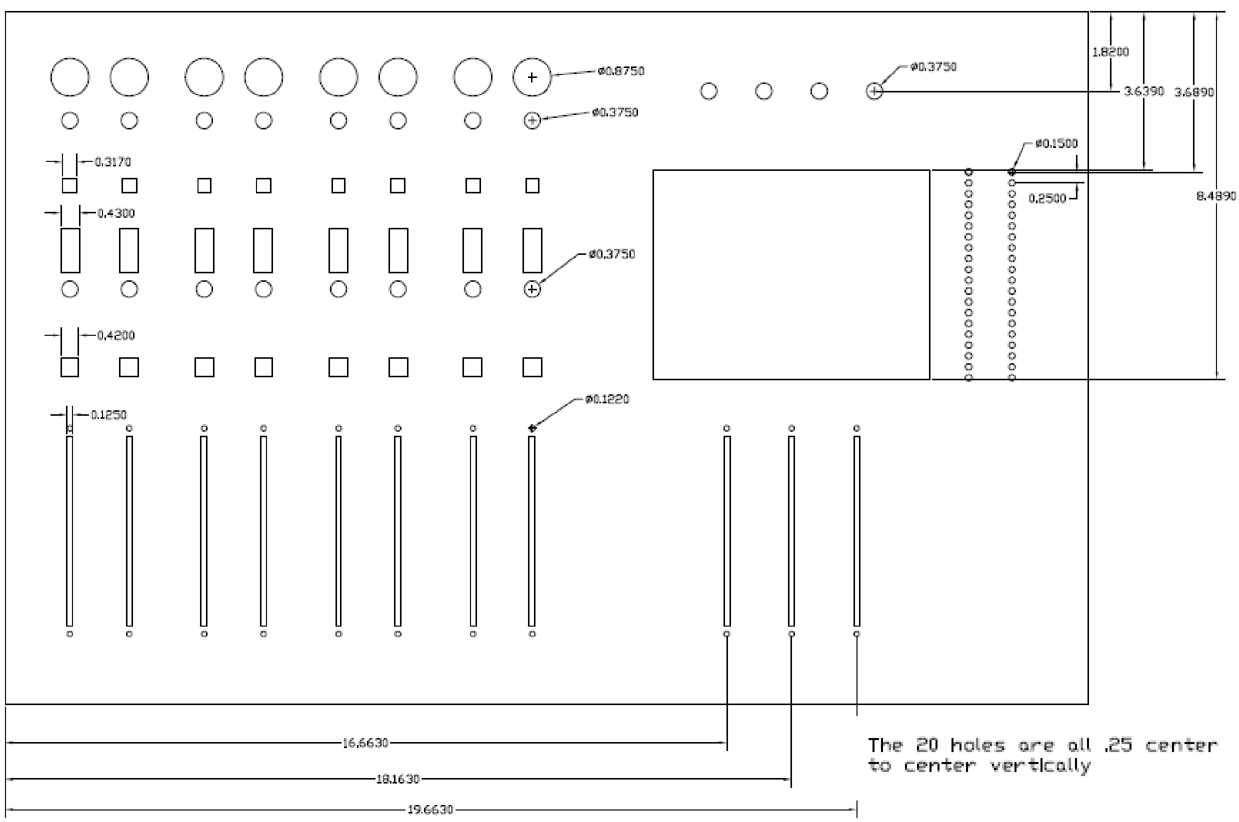


Figure B-4. AutoCAD drawing of top case layout, showing dimensions

Appendix C: Schematic

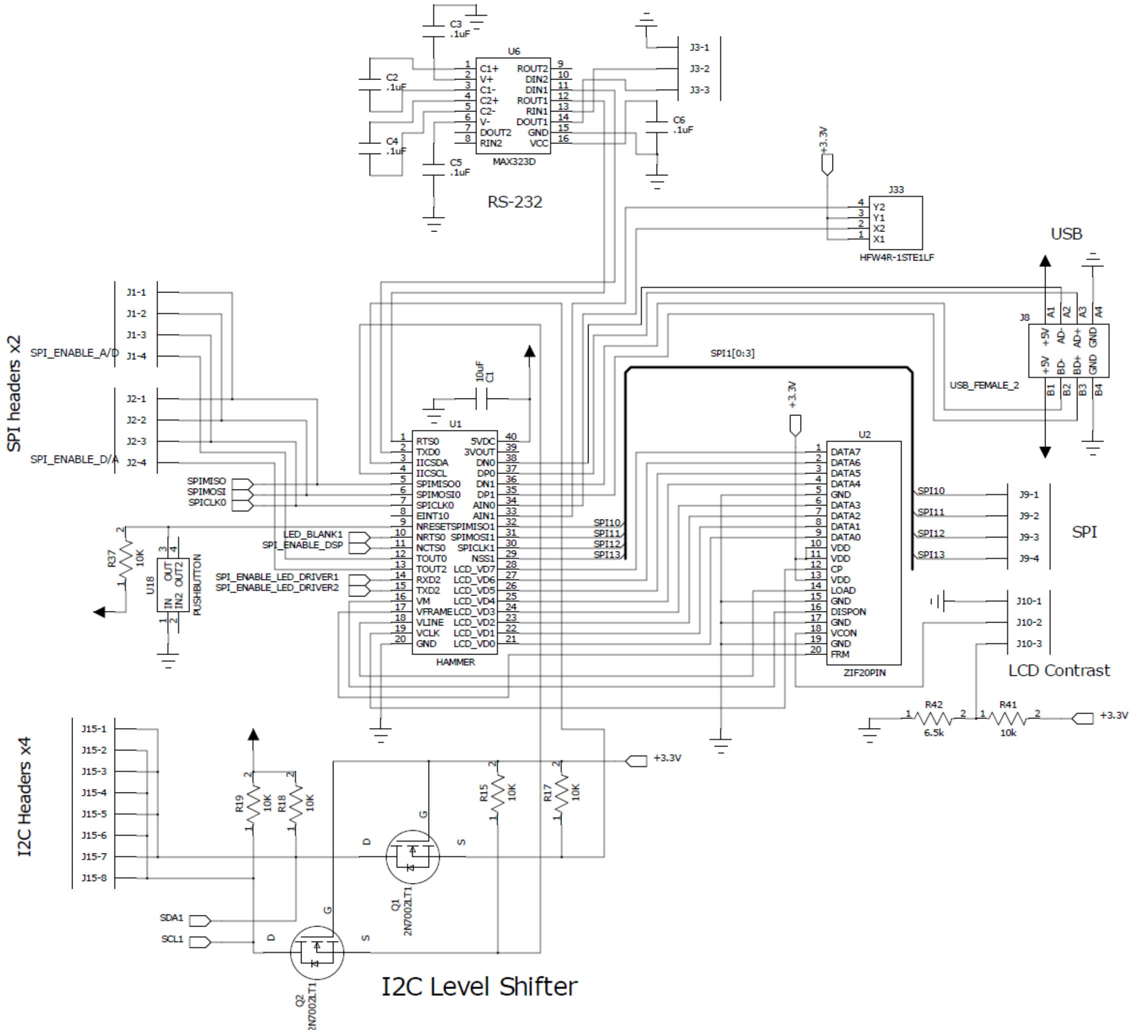


Figure C-1. of Hammer board connections on Main board PCB.

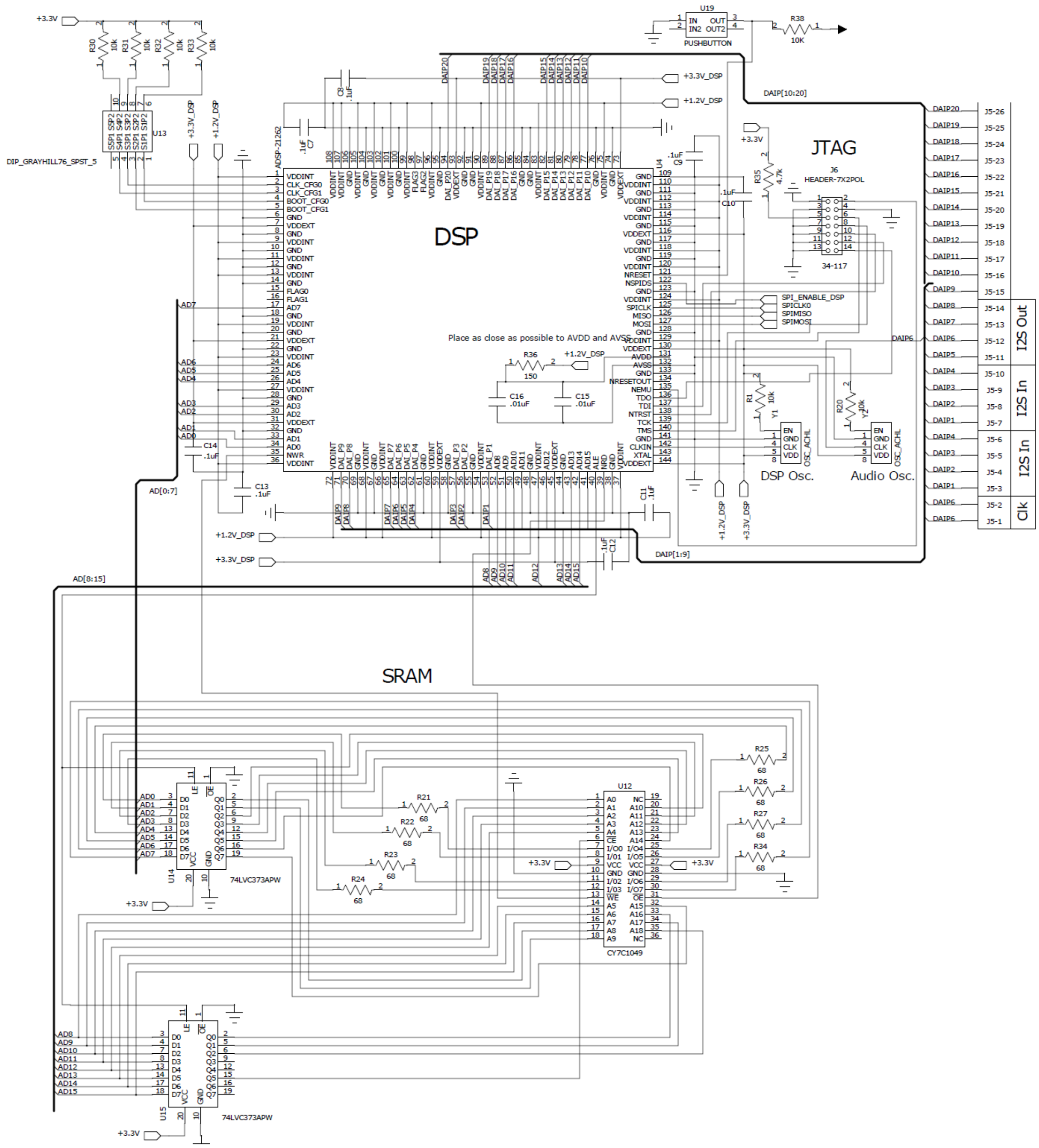


Figure C-2. Schematic of DSP connections on Main board PCB.

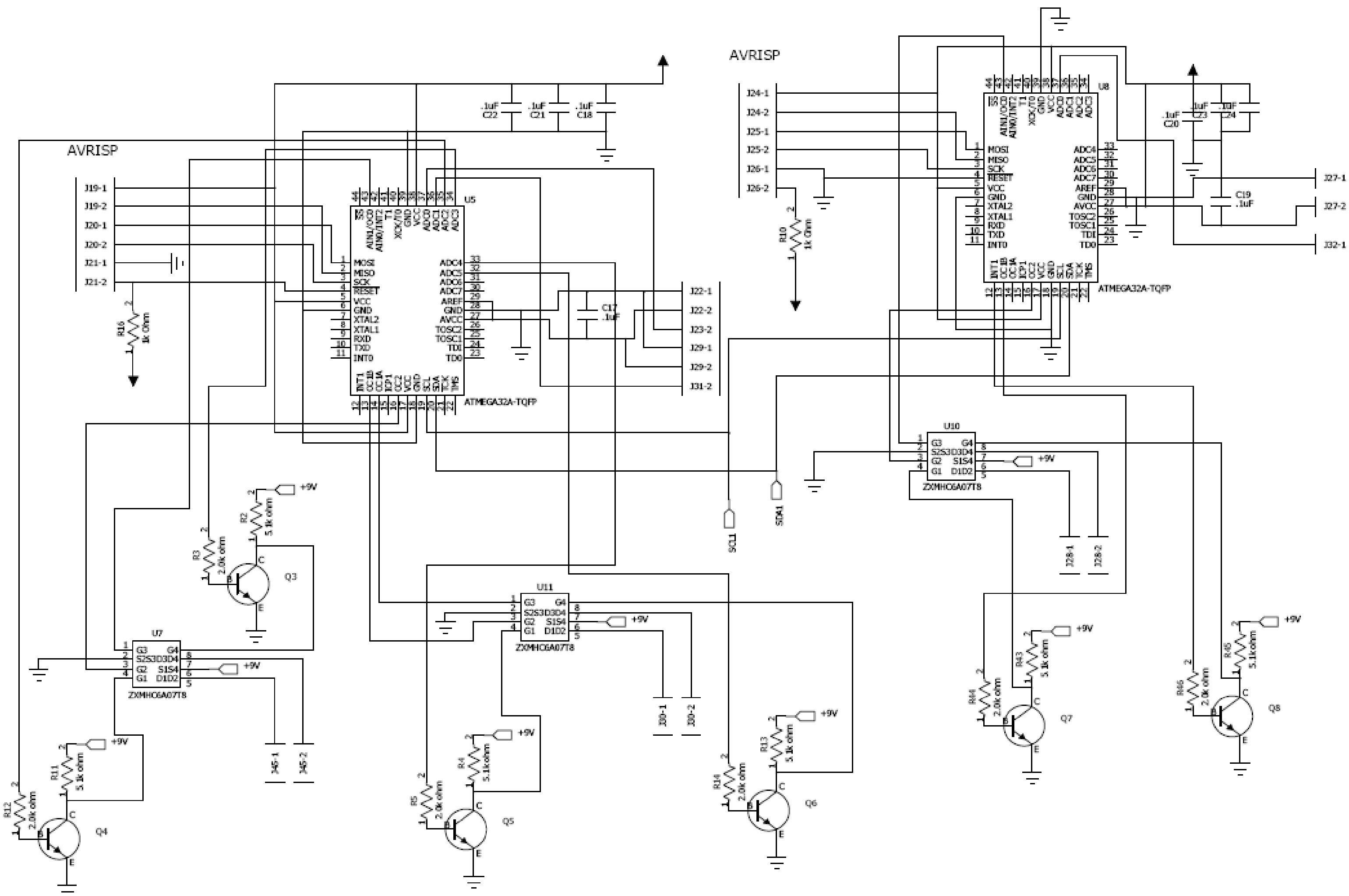


Figure C-3. Schematic of ATMega32A connections on Main board PCB.

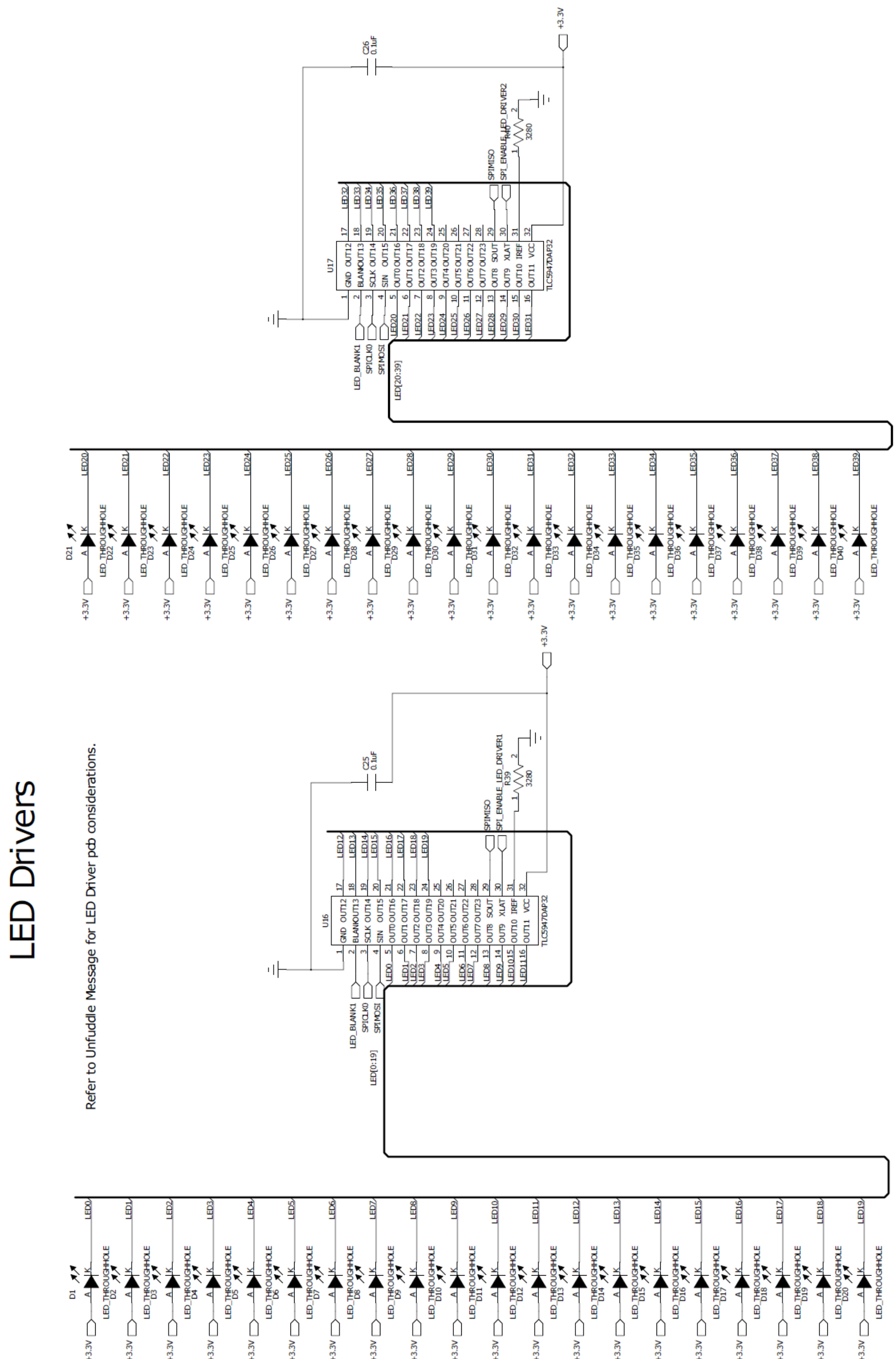


Figure C-4. Schematic of LED driver connections on Main board PCB.

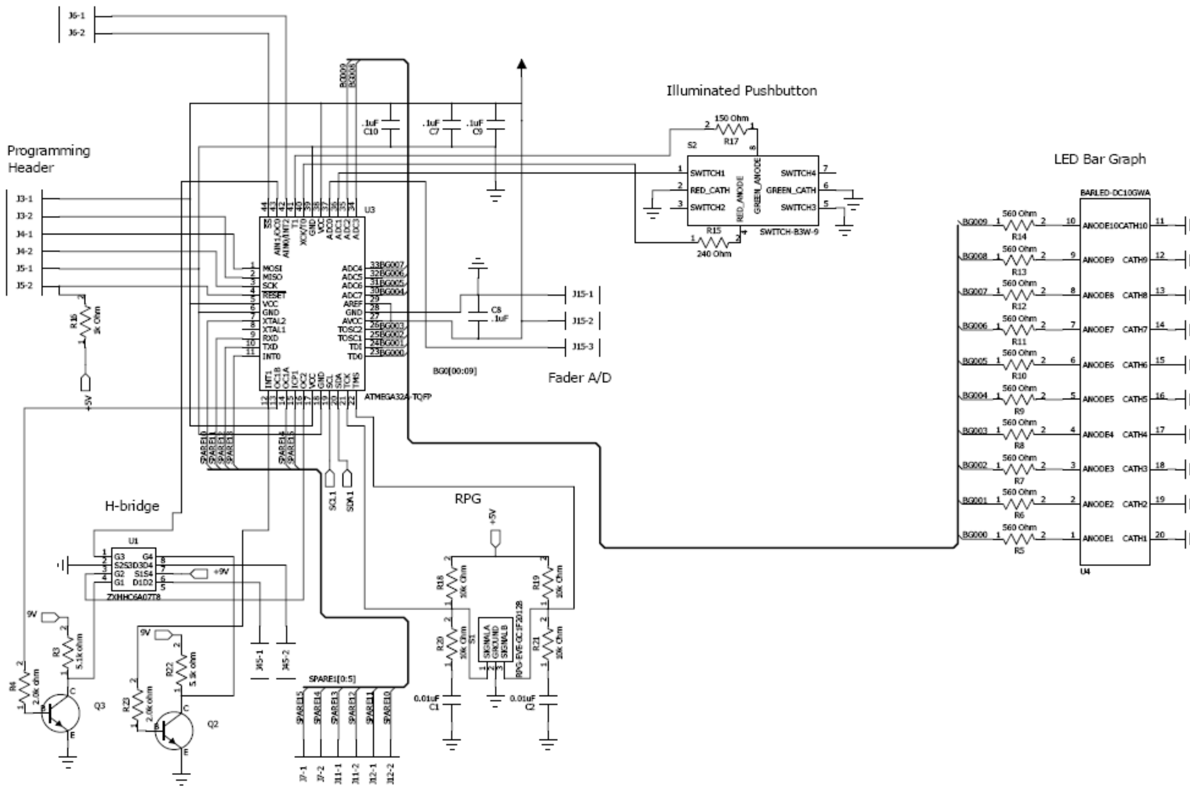


Figure C-5. Schematic of a single channel's user interface: Two of these exist per Channel PCB. Four channel PCBs are manufactured, for a total of eight channels.

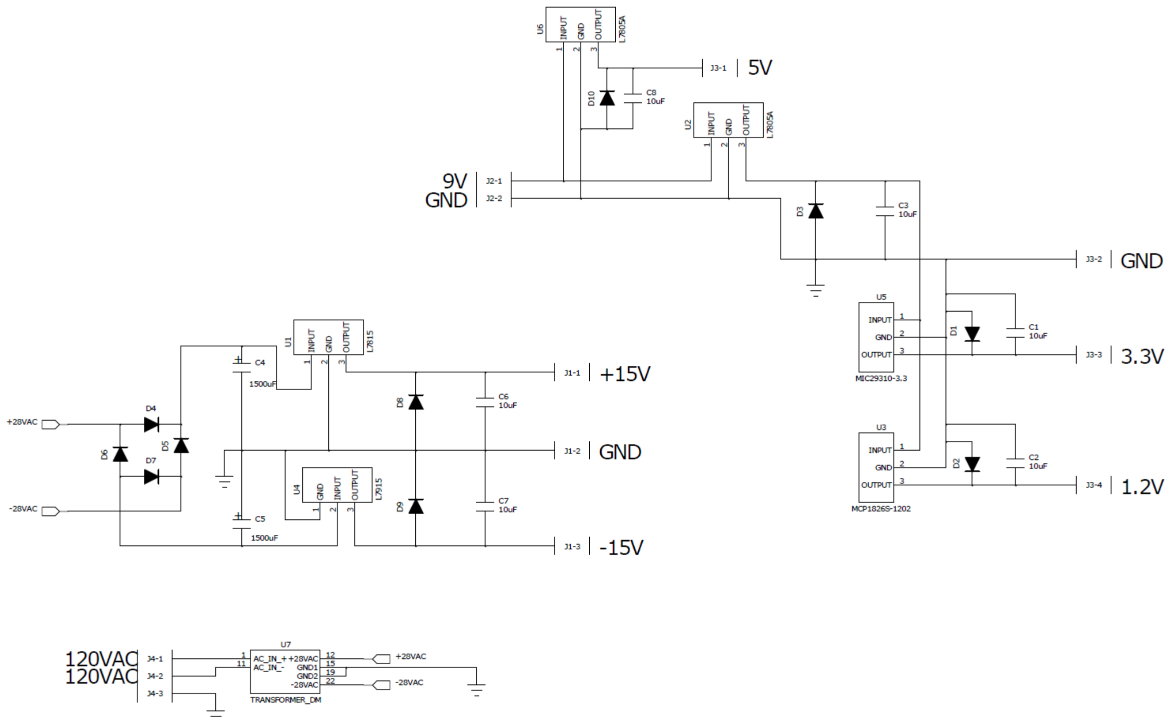


Figure C-6. Schematic of Power Supply PCB: A power supply (not shown) outputs 9V from 120VAC. 15V is produced from 120VAC via a transformer (left schematic). 5V, 3.3V, and 1.2V are produced from the 9V supply (right schematic).

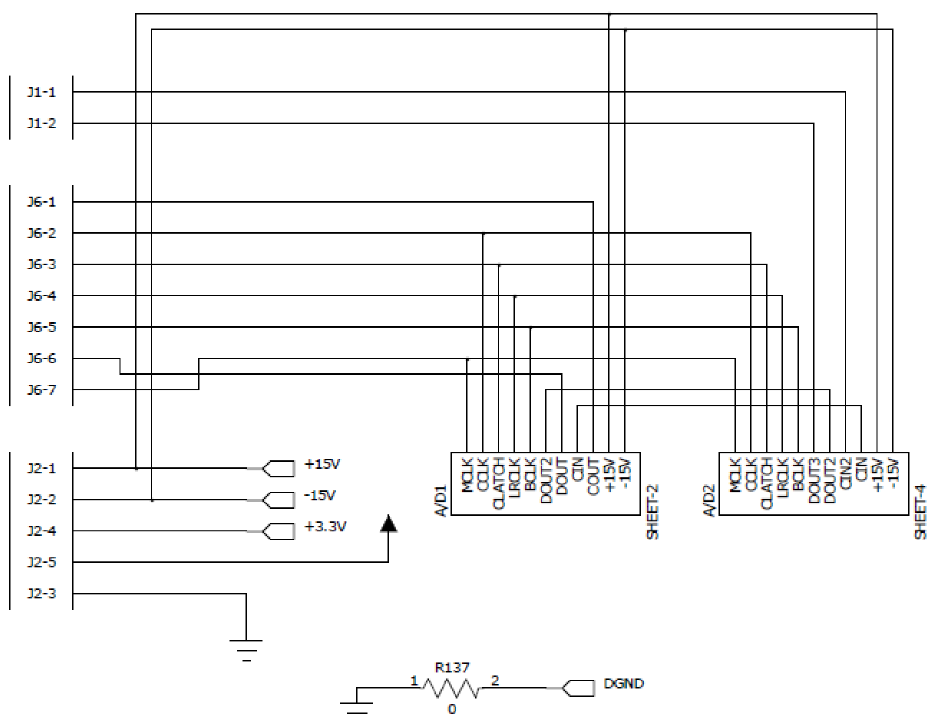


Figure C-7. Schematic of Left Audio Board channel interconnections. These are the connections between the board, power, and two A/D converter circuits.

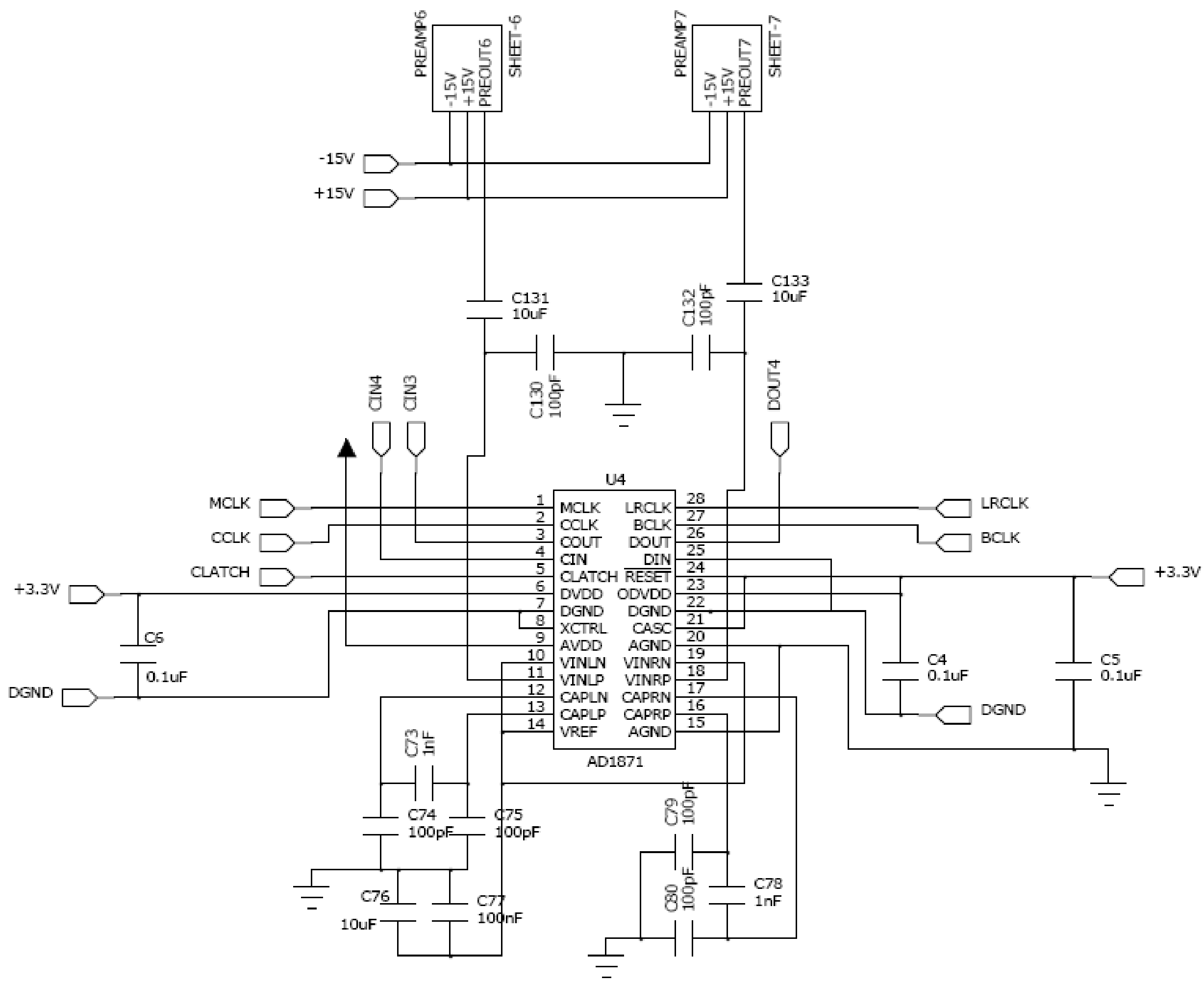


Figure C-8. Schematic of an Audio Board A/D converter: There are two of these on the Left Audio Board, and another two on the Right Audio Board.

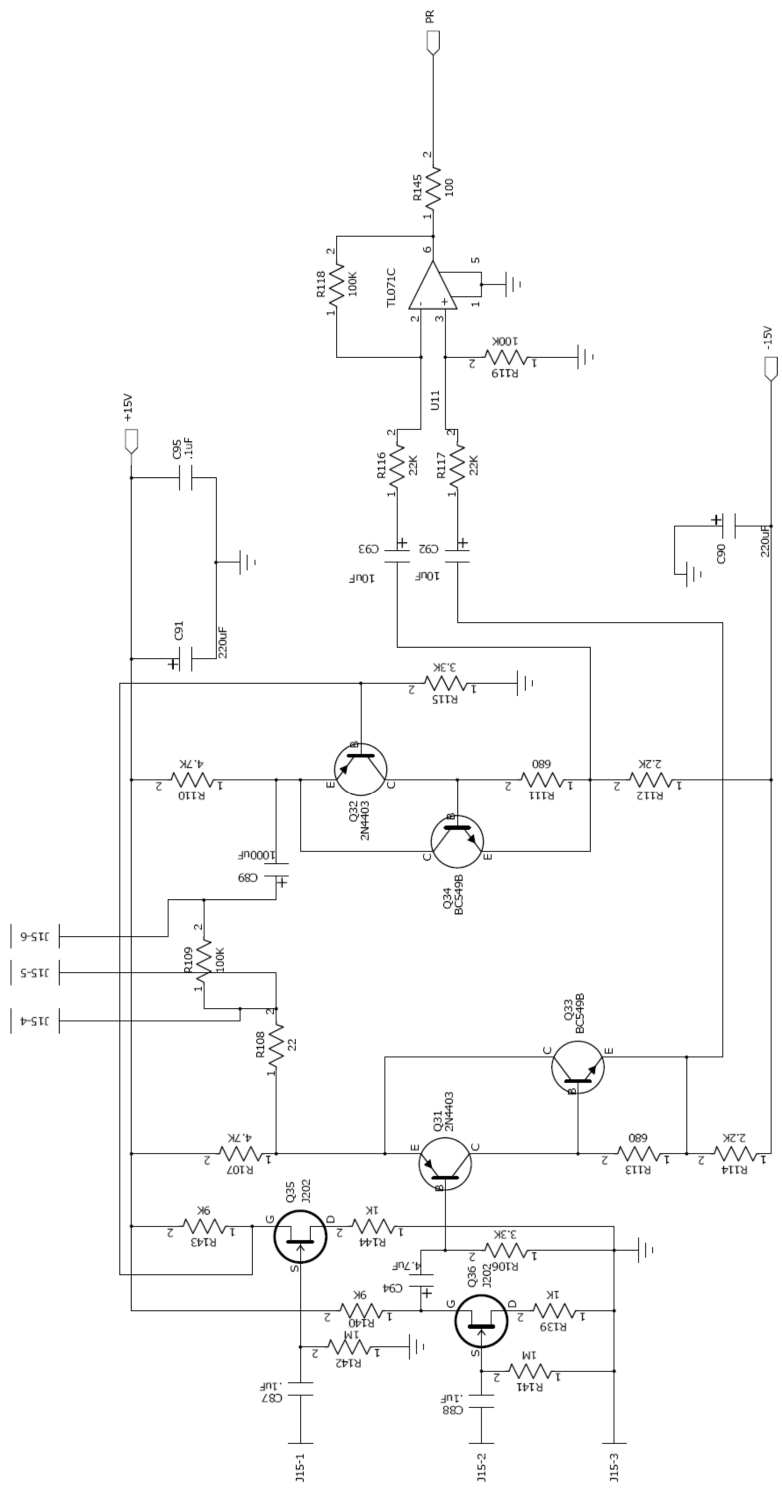


Figure C-9. Schematic of a Left Audio Board Preamplifier: There are four of these on the Left Audio Board, and four on the Right Audio Board.

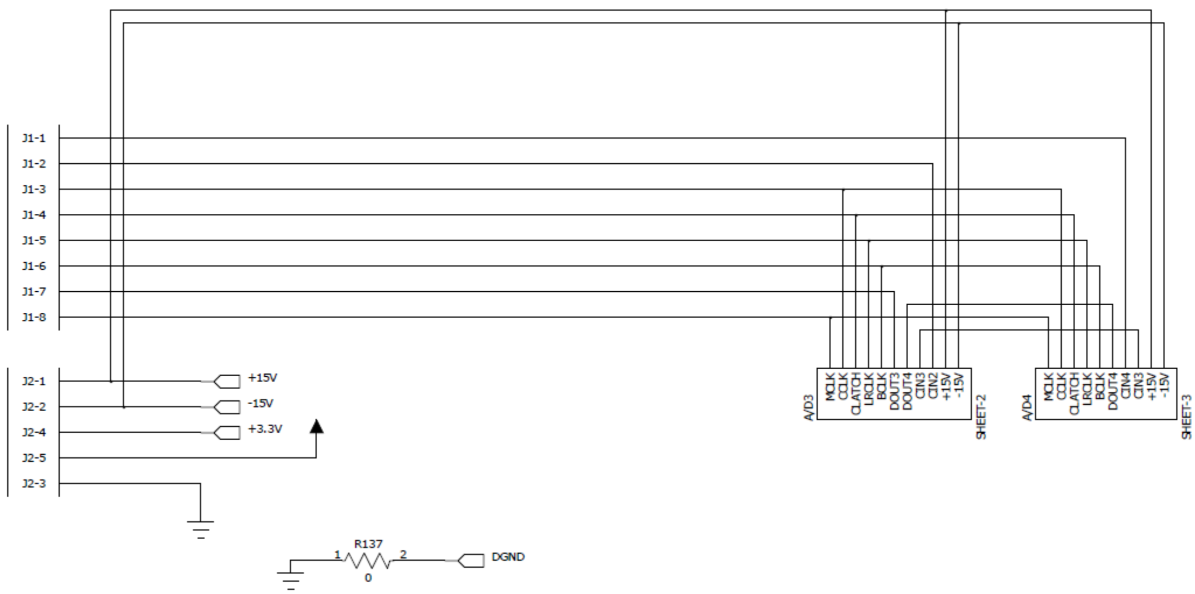


Figure C-10. Schematic of a Right Audio Board channel interconnections. These are the connections between the board, power, and two A/D converter circuits.

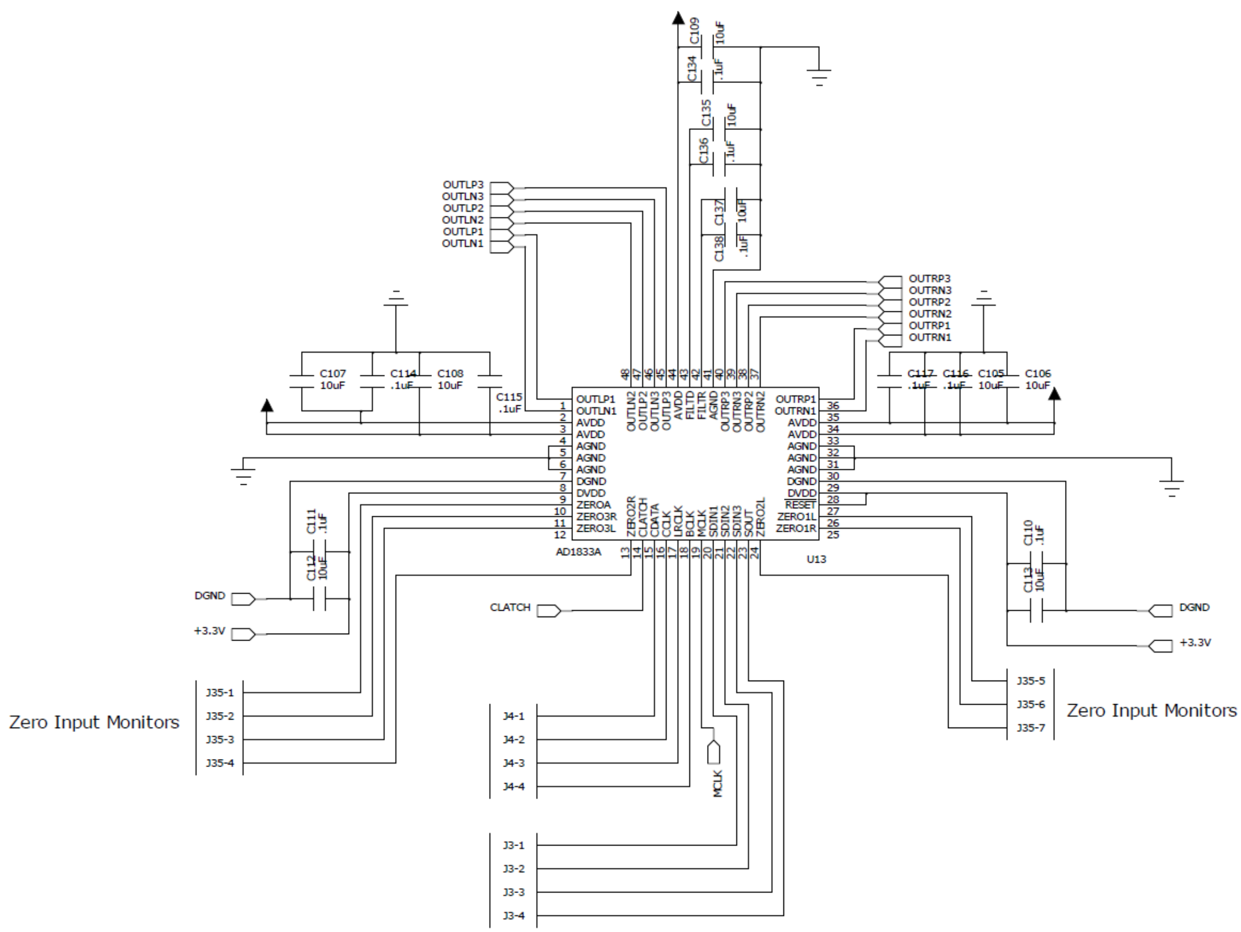


Figure C-11. Schematic of the D/A Converter on the Right Audio Board:

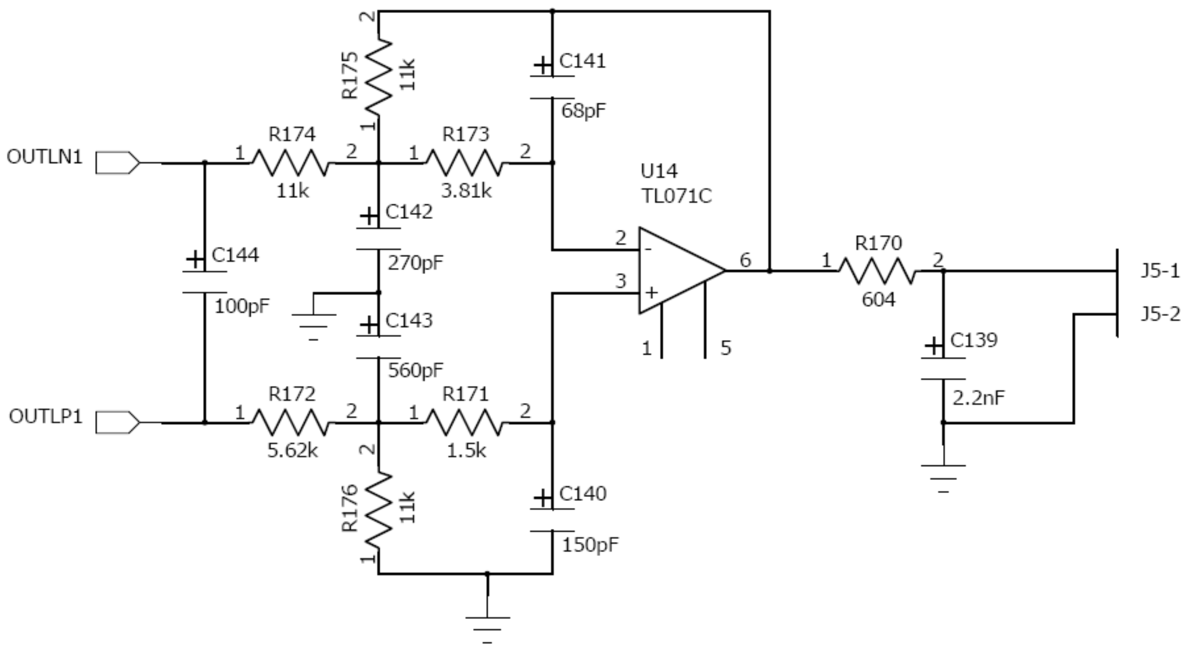


Figure C-12. Schematic of an Output Preamplifier: There are four of these on the Right Audio Board.

Appendix D: PCB Layout Top and Bottom Copper

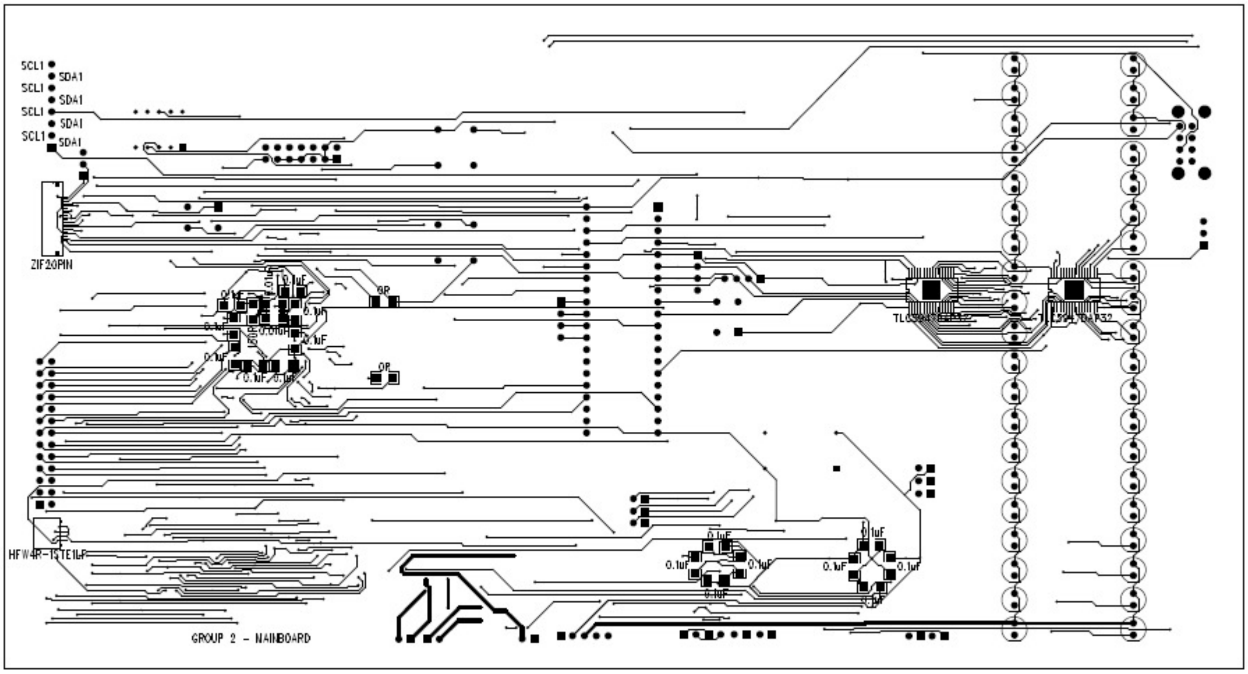


Figure D-1. Main board PCB: Top Copper and Silkscreen

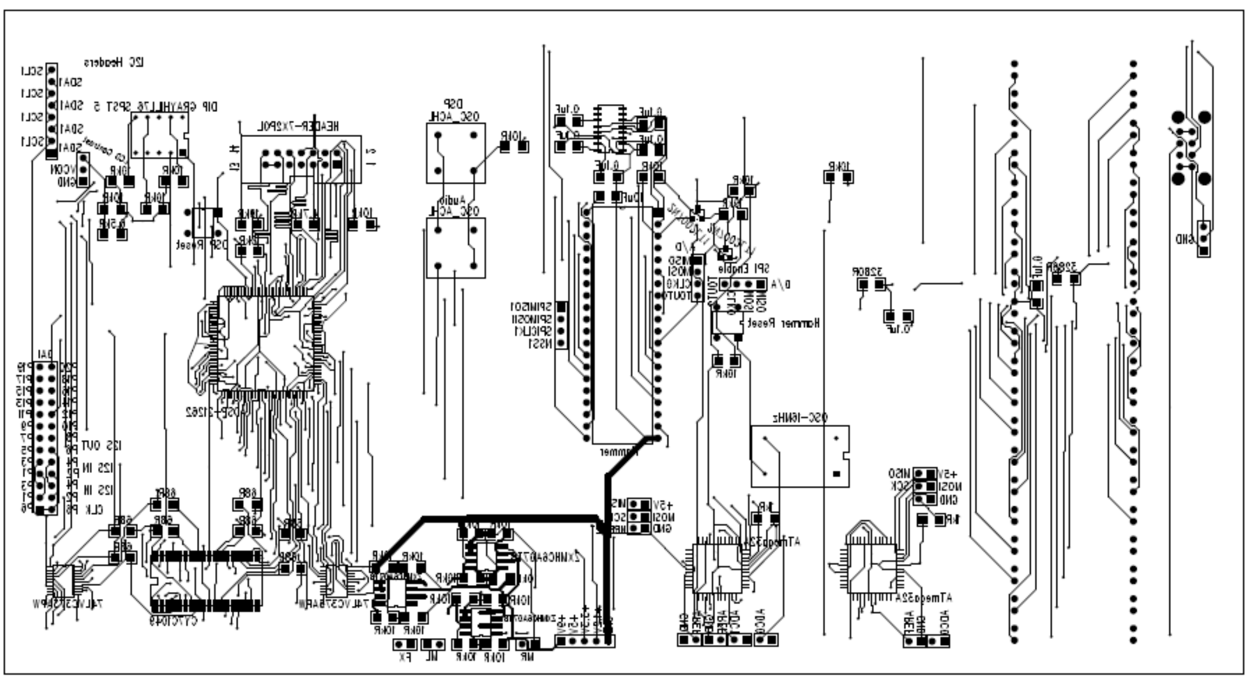


Figure D-2. Main board PCB: Bottom Copper and Silkscreen

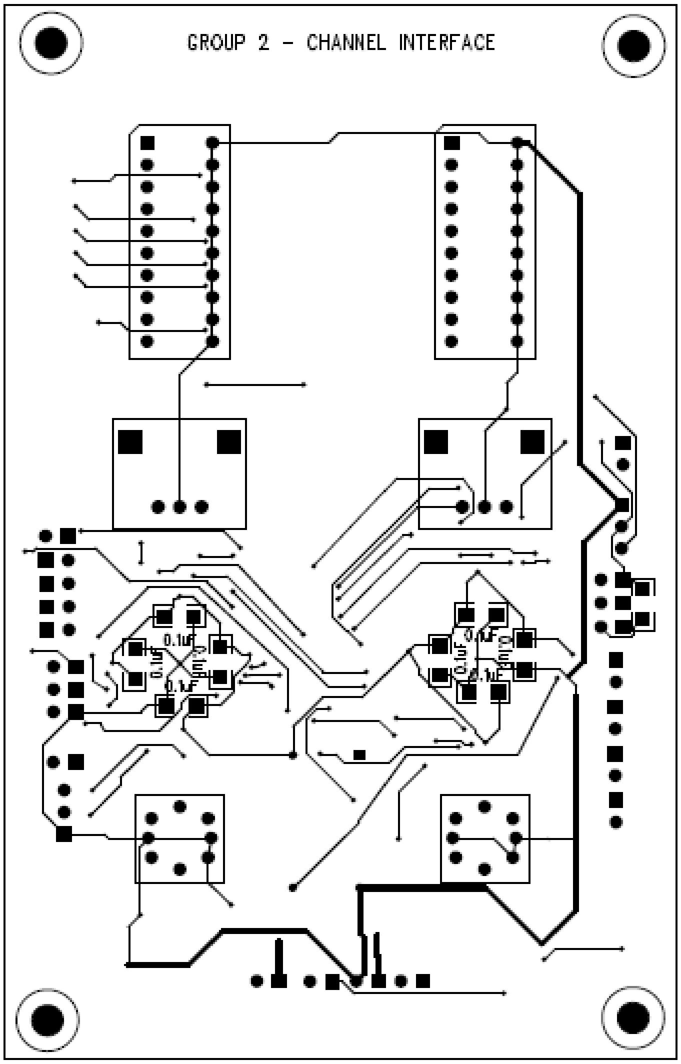


Figure D-3. Channel Interface PCB: Top Copper and Silkscreen

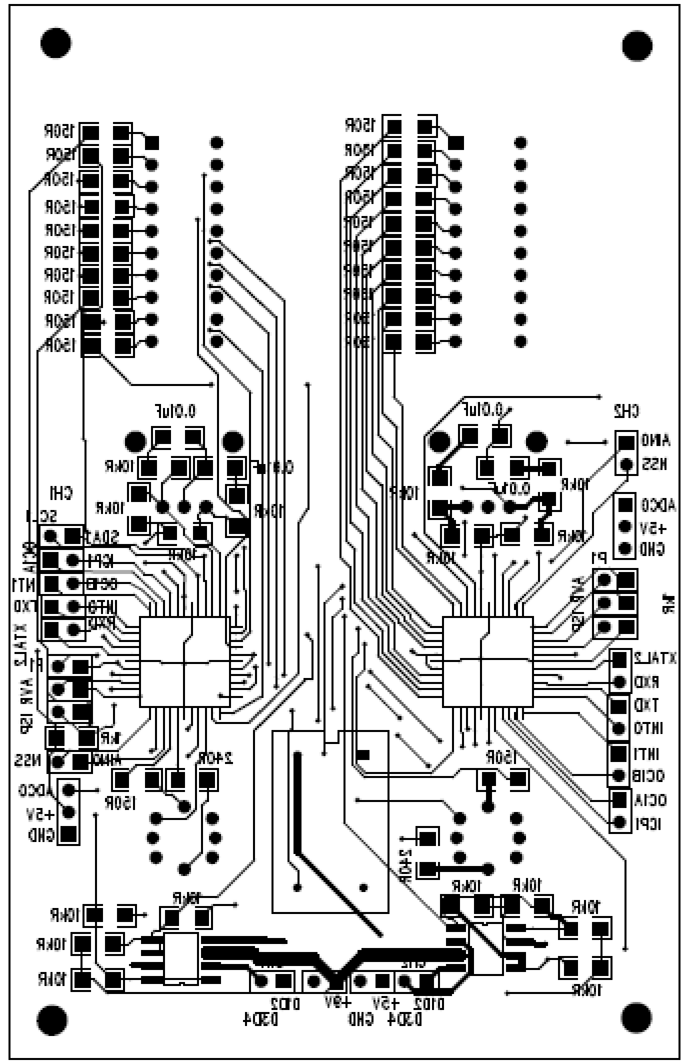


Figure D-4. Channel Interface PCB: Bottom Copper and Silkscreen

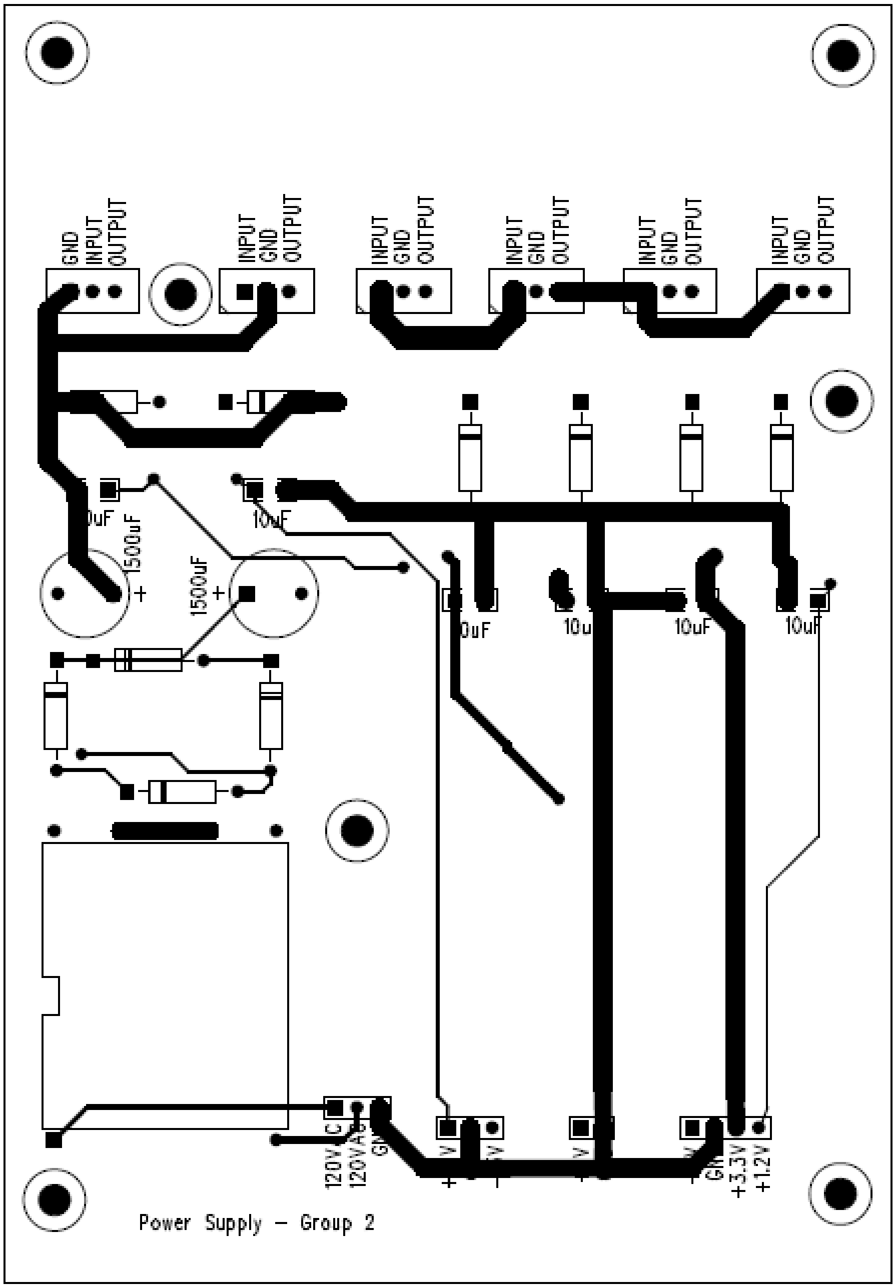


Figure D-5. Power Supply PCB: Top Copper and Silkscreen

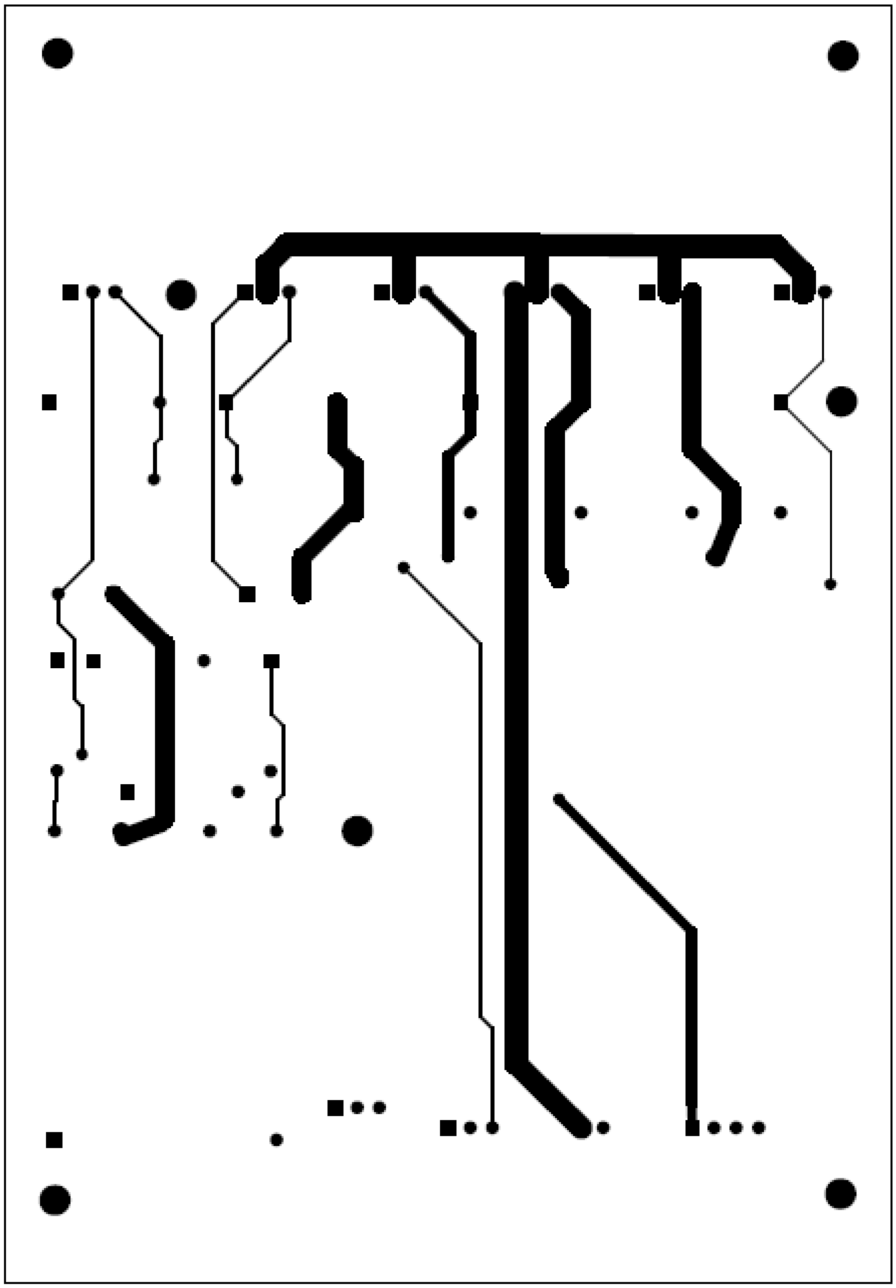


Figure D-6. Power Supply PCB: Bottom Copper and Silkscreen

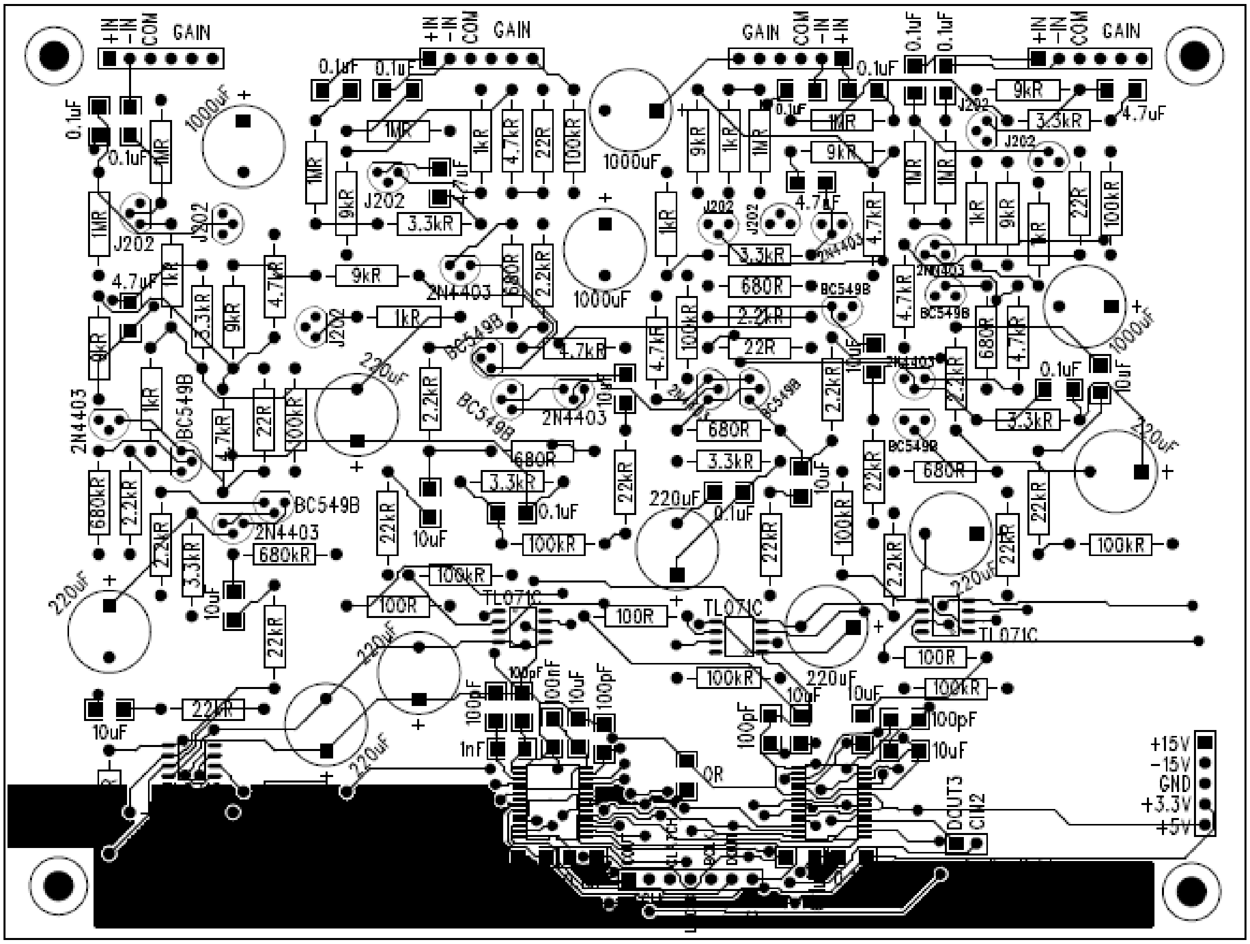


Figure D-7. Left Audio Board PCB: Top Copper and Silkscreen

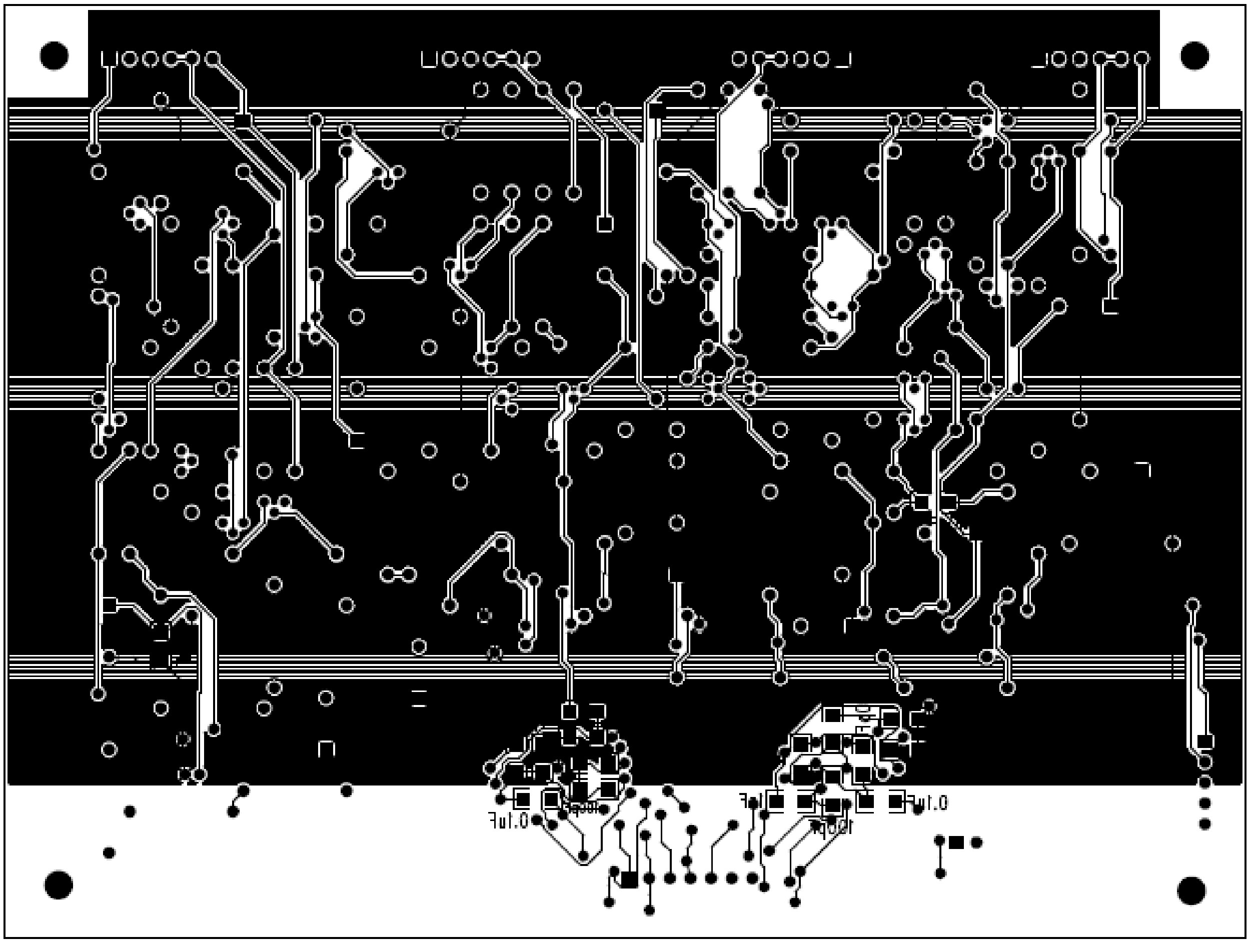


Figure D-8. Left Audio Board PCB: Bottom Copper and Silkscreen

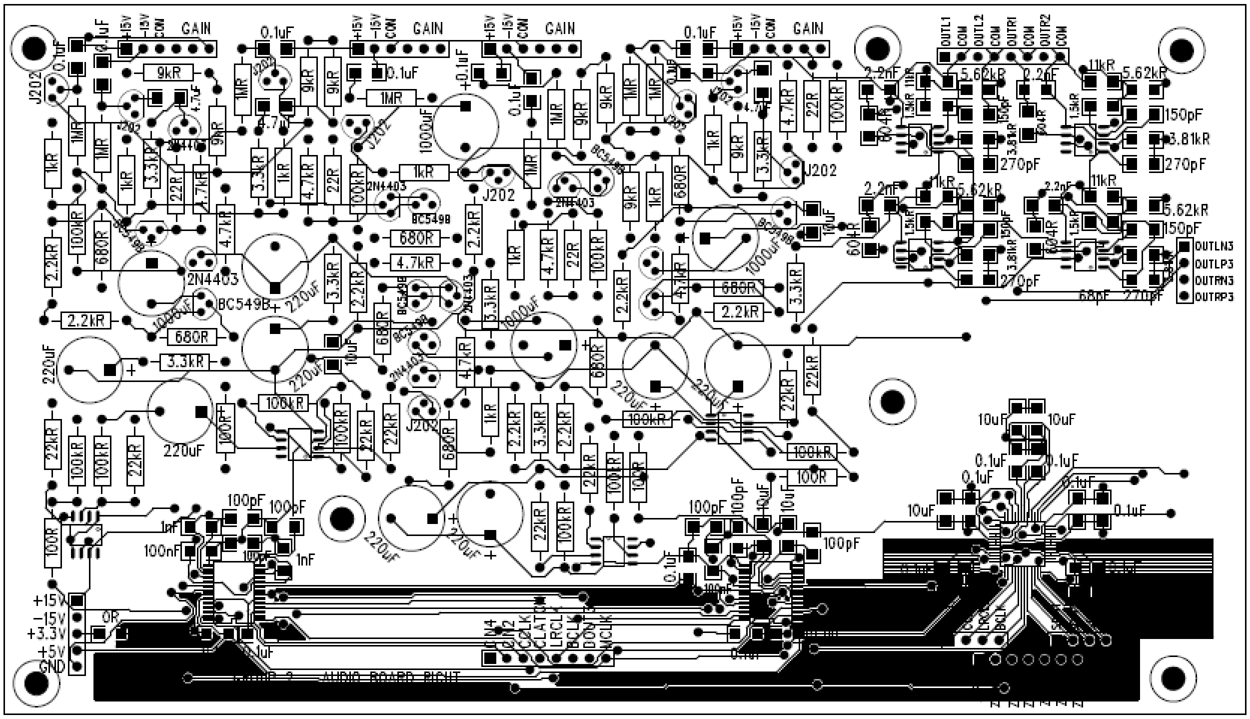


Figure D-9. Right Audio Board PCB: Top Copper and Silkscreen

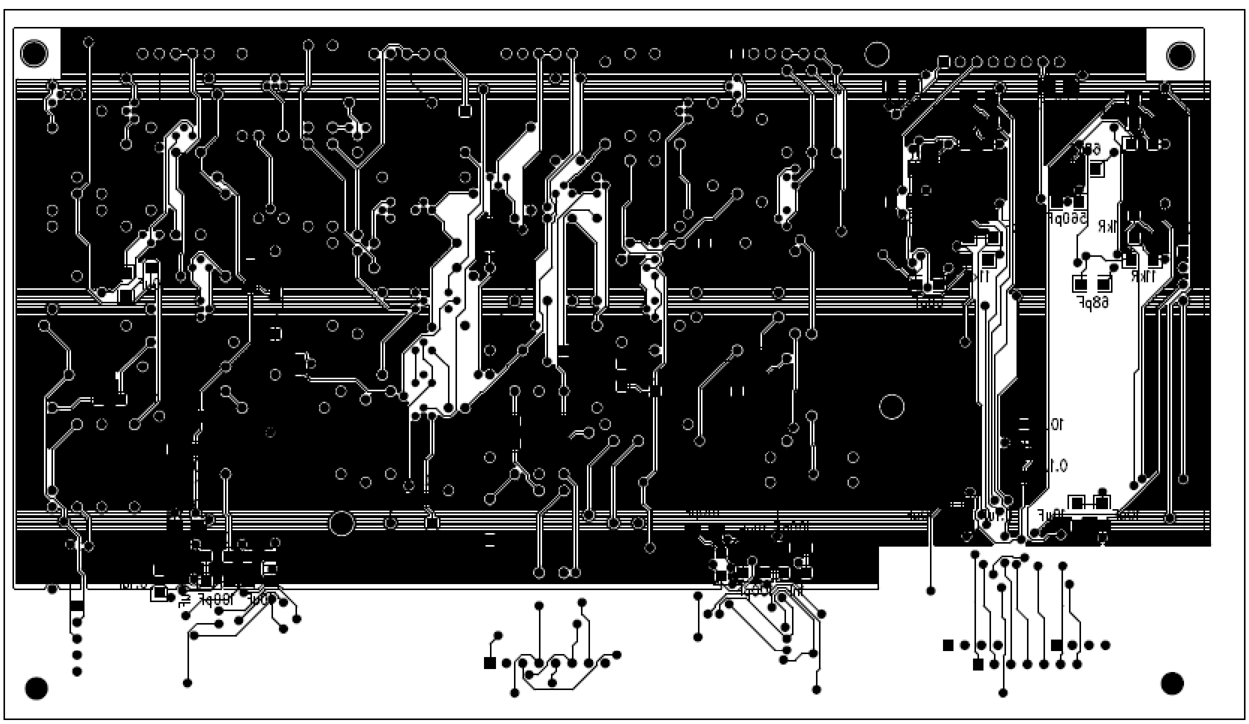


Figure D-10. Right Audio Board PCB: Bottom Copper and Silkscreen

Appendix E: Parts List Spreadsheet

Vendor	*Manufacturer*	*Part No.*	*Description*	*Unit Cost*	*Qty*	*Total Cost*
-	-	-	**MAIN BOARD**			
earthLCD	Panasonic	EDMGRB8KJF	7.8" Color STN Touch screen LCD	99.00	1	99.00
Tin Can Tools	Tin Can Tools	Hammer Carrier S3C2410A ARM920T	Development Kit and Hammer ARM920t based on Samsungs S3C2410A	239.00	1	239.00
Digikey	Analog Devices	ADSP-21262SKSTZ200	SHARC ADSP-21262 DSP Processor	30.69	1	30.69
Mouser	STMicroelectronics	L9997ND013TR	Dual H-Bridge Driver 7-16V 1.2A	3.00	6	18.00
Mouser	Kobiconn	161-0023-E	1/4" Audio/Video Connector	1.19	13	15.47
Mouser	Amphenol Audio	AC3FAV-AU-B	XLR Connector	1.44	8	11.52
Mouser	JKL Components	BXA-501	CCFL Inverter	15.12	1	15.12
Chuck	Cypress	CY7C1049	External SRAM	0.00	1	0.00
Digikey	NXP Semiconductors	74LVC373APW	IC Octal Transparent Latch; 20 TSSOP	0.50	4	2.00
Digikey	Texas Instruments	MAX232D	RS-232 Level Translator (MAX323E on schematics)	1.16	2	2.32
Mouser	ABRACON	ACHL-24.576MHZ-EK	Oscillator 24.576 MHZ	1.30	1	1.30
Mouser	ABRACON	ACHL-25.000MHZ-EK	Oscillator 25.000 MHZ	1.30	1	1.30
Parts Room	ECS Inc.	ECS-100AC	Oscillator 16MHz	1.88	5	9.40
Mouser	Texas Instruments	TLC5947DAP	LED-Drivers; 24Ch 12B PWM	4.95	2	9.90
Mouser	FCI USB Connectors	72309-8034BLF	USB Connector (PCB Mount)	1.00	1	1.00
Mouser	ON Semiconductor	2N7002LT1G	N-Channel TMOSFET	0.16	4	0.64
Parts Cabinet	Grayhill	7605SPST Rocker	Thru-Hole Dip Switch	0.00	1	0.00
Digikey	FCI	HFW4R-1STE1LF	ZIF 4 PIN	0.68	3	2.04
Digikey	FCI	SFV20R-1STE1LF	ZIF 20 PIN	1.81	3	5.43
-	-	-	**POWER SUPPLY**			
Mouser		7915 Regulator	(-)15V Regulator	0.48	2	0.96
Mouser		7815 Regulator	(+)15V Regulator	0.48	2	0.96
Digikey	Nichicon	UPW1V152MHD6	1500 uF Cap (for P.S.)	1.09	2	2.18
Digikey	Micrel Inc.	MIC29310-3.3WT	3.3V Regulator	3.92	3	11.76
Digikey	Microchip Technology	MCP1826S-1202E/AB	1.2V Regulator	1.66	2	3.32

Mouser	Fairchild	KA78T05	5V Positive Voltage Regulator	1.37	3	4.11
Mouser	Hammond	164J16	Transformer (for P.S.)	16.69	1	16.69
Parts Room			Diodes	0.00	10	0.00
Parts Room			Power Entry Module for AC plug	0.00	0.00	0.00
-	-	-	**CHANNEL INTERFACE**	-	-	-
Mouser	Atmel	ATmega32A	AVR 8-bit RISC microprocessor	4.16	11	45.76
Mouser	Bourns	PTV111-3415A-B103	Potentiometer; 10K 20% 12mm (gain control for each channel)	0.96	8	7.68
Mouser	Kingbright	DC10GWA	LED Bar Graph	1.24	11	13.64
Digikey	Panasonic - ECG	EVE-GC1F2012B	RPG - 12 mm vertical 12 ppr	0.76	8	6.08
Mouser	Diodes Inc. / Zetex	ZXMHC6A07T8TA	H-Bridge	1.65	11	18.15
Mouser	Omron Electronics	B3W-9000-RG2N	Red/Green Pushbutton	2.68	9	24.12
-	-	-	**AUDIO BOARDS**	-	-	-
Newark	Fairchild Semiconductor	J201	J201 N-Channel JFETS	0.11	60	6.72
Newark	Analog Devices	AD1871YRSZ	24-bit Stereo A/D Convertor	9.52	4	38.08
Mouser	Fairchild Semiconductor	2N4403TF	Bipolar Transistor, PNP	0.05	16	0.80
Mouser	Fairchild Semiconductor	BC549B	NPN Transistor	0.09	16	1.44
Digikey	Analog Devices	AD1833AASTZ	DAC Audio 24 bit - 48 LQFP	9.90	2	19.80
Newark	Analog Devices	AD1871YRSZ	24-bit Stereo A/D Convertor	9.52	4	38.08
Mouser	STMicroelectronics	TL071ID	OP-AMP	0.62	5	3.10
Mouser	Vishay/Dale	CCF07680RJKR36	1/4 Watt Metal Film Resistors (680)	0.06	18	1.08
Mouser	KOA Speer	MF1/4LCT52R102J	1/4 Watt Metal Film Resistors (1K)	0.05	18	0.90
Mouser	KOA Speer	MFS1/4LCT52R222J	1/4 Watt Metal Film Resistors (2.2K)	0.09	28	2.52
Mouser	KOA Speer	MF1/4LCT52R332J	1/4 Watt Metal Film Resistors (3.3K)	0.05	18	0.90
Mouser	KOA Speer	MF1/4LCT52R472J	1/4 Watt Metal Film Resistors (4.7K)	0.05	18	0.90
Mouser	KOA Speer	MF1/4DCT52R9101F	1/4 Watt Metal Film Resistors (9.1K)	0.05	18	0.90
Mouser	KOA Speer	MF1/4DC1102F	1/4 Watt Metal Film Resistors (11K)	0.06	12	0.72
Mouser	KOA Speer	MF1/4LCT52R223J	1/4 Watt Metal Film Resistors (22K)	0.05	18	0.90

Mouser	KOA Speer	MF1/4LCT52R104J	1/4 Watt Metal Film Resistors (100K)	0.05	26	1.30
Mouser	KOA Speer	MF1/4LCT52R105J	1/4 Watt Metal Film Resistors (1M)	0.05	18	0.90
Parts Room			50V 220uF AE6 capacitors	0.50	16	8.00
Parts Room			50V 1000uF AE6 capacitors	0.50	8	4.00
-	**-**	**-**	**OTHER**	-	-	-
Ebay seismoman	ALPS	RSA0N11M	10k Motorized Faders	3.98	20	79.60
Mouser	Kobiconn	161-0023-E	1/4" Mono Black Audio/Video Connectors	1.19	8	9.52
Mouser	Kobiconn	161-0023-E	1/4" Mono Black Phone Jacks	1.19	5	5.95
Mouser	Bourns	PTV111-3415A-B103	10k 20% 12mm Panel Mount Potentiometers	0.96	8	7.68
Mouser	Amphenol	AC3FAV-AU-B	Plastic A Panel Audio/Video Connectors	1.44	8	11.52
				TOTAL		**864.85**

Appendix F: FMECA Worksheet

Table 1 – Hammer ARM9

Failure No.	Failure Mode	Possible Causes	Failure Effects	Method of Detection	Criticality	Remarks
1A	Micro remains in reset mode	Reset switch is broken and stays in “pressed” state	Microcontroller fails to run program, also cannot reprogram memory	Observation with DMM	Medium	Medium criticality because it disables the functionality of the system
1B	ATMELS and Hammer cannot communicate because Hammer cannot understand 5V logic	I^2C level shifter damaged	User interface seems to be working, but audio is not	Observation with DMM and Logic Analyzer	Medium	Medium criticality because it disables the functionality of the system
1C	Contrast is set either all the way up or all the way down	Contrast voltage divider resistor is shorted	Cannot adjust the contrast on LCD	Observation with DMM	Low	Low criticality because it is simply a nuisance to the user
1D	LCD does not receive data	ZIF connector has bent pins or Hammer has burned out pins	LCD will not change the display, but the touch screen works	Observation with Oscilloscope	Low	Low criticality because it is simply a nuisance to the user
1E	Erroneous/Sporadic data sent to the DSP	ARM9 is non-functional	Audio levels are sporadic. Possibly very high output levels.	Observation with Logic Analyzer	High	High criticality because if levels are too high, they can be harmful when amplified

Table 2 – DSP

Failure No.	Failure Mode	Possible Causes	Failure Effects	Method of Detection	Criticality	Remarks
2A	Micro remains in reset mode	Reset switch is broken and stays in “pressed” state	Microcontroller fails to run program, also cannot reprogram memory	Observation with DMM	Medium	Medium criticality because it disables the functionality of the system
2B	Memory space is too small	SRAM chip burned out	Audio Processing is greatly lagging	Observation with DMM and Logic Analyzer	Medium	Medium criticality because it disables the essential functionality of the system
2C	- No power sent to individual LEDs (likely if some LEDs still functioning) - SPI signal not present or sampled incorrectly (likely if no LEDs are functioning)	-If some LEDs still function: LED driver is burned out - If no LEDs are functioning: SPI is not working on DSP	Output amplitude LEDs are not lighting	Observation with DMM and Logic Analyzer	Low	Low criticality because it is simply a nuisance to the user (Although it is one of our current PSSCs, so it is critical)
2D	Erroneous/Sporadic output levels	SHARC is non-functional	Audio levels are sporadic. Possibly very high output levels.	Observation with Logic Analyzer	High	High criticality because if levels are too high, they can be harmful when amplified

Table 3 – ATMega32A						
Failure No.	**Failure Mode**	**Possible Causes**	**Failure Effects**	**Method of Detection**	**Criticality**	**Remarks**
3A	Micro remains in reset mode	Reset jumpers are shorted, thus created an effective "button pressed" state	Microcontroller fails to run program, also cannot reprogram memory	Observation with DMM	Medium	Medium criticality because it disables the functionality of the system
3B	Micro not communicating with ARM9	ATMEL is non-functional or I^2C not configured properly for that channel	Nothing works on one individual channel	Observation with Logic Analyzer	Medium	Medium criticality because it disables the functionality of the channel
3C	Erroneous/Sporadic information about audio levels is sent to ARM9	ATMEL is non-functional	Audio levels are sporadic. Possibly very high output levels.	Observation with Logic Analyzer	High	High criticality because if levels are too high, they can be harmful when amplified
3D	PWM is only working on one channel	PWM is disabled or fried	Fader will only move automatically in one direction	Observation with Oscilloscope	Low	Low criticality because it is simply a nuisance to the user.

Table 4 – Channel Interface

Failure No.	Failure Mode	Possible Causes	Failure Effects	Method of Detection	Criticality	Remarks
4A	H-Bridge is not providing power to the fader motor	H-Bridge burned out	Faders don't move automatically	Observation with DMM	Low	Low criticality because it is simply a nuisance to the user.
4B	Channel remains enabled or disabled	On/Off button broken and not creating contact when it is pressed.	Channel audio is not heard and button does not change color when turned on	Observation with DMM	Medium	Medium criticality because it disables the functionality of the channel
4C	LEDs do not light up	LEDs are burned out	LEDs in pushbutton do not light up when they are supposed to	Observation with DMM	Low	Low criticality because it is simply a nuisance to the user.
4D	LEDs do not light up	LEDs are burned out	Some of the LEDs in the bar graph do not light up.	Observation with DMM	Low	Low criticality because it is simply a nuisance to the user.
4E	Square wave is not generating properly	RPG is broken	Levels do not change when RPG is turned	Observation with Oscilloscope or Logic Analyzer	Medium	Medium criticality because it disables the functionality of the channel

Table 5 – Power Supply

Failure No.	Failure Mode	Possible Causes	Failure Effects	Method of Detection	Criticality	Remarks
5A	Excessive current draw, fuses continuously blown	Power rails shorted together	Short causes a blown fuse, burnt out components, or even a fire	Observation with DMM and continuity check	High	High criticality because if power traces are shorted, they can cause a fire
5B	Excessive current draw on regulator	Regulator is blown	Devices on a particular power rail will not power on	Observation with DMM	Medium	Medium criticality because it disables the functionality of the unit
5C	Rectifier circuit is degraded and goes below dropout for regulator, causing a noisy voltage supply	Rectifier diodes or Capacitors are blown	Preamp is noisy	Observation with DMM	Medium/Low	Medium/Low criticality because it is a nuisance to the user, but also degrades the quality of the audio signal.

Table 6 – D/A and A/D

Failure No.	Failure Mode	Possible Causes	Failure Effects	Method of Detection	Criticality	Remarks
6A	A signal is being passed into the A/D chip, but no signal is coming out.	A/D chip is non-functional	No audio input to DSP	Observation with DMM and Logic Analyzer	Medium	Medium criticality because it disables the functionality of the unit
6B	A signal is being passed into the A/D chip, but no signal is coming out.	D/A chip is non-functional	No audio output from DSP	Observation with DMM and Logic Analyzer	Medium	Medium criticality because it disables the functionality of the unit
6C	Reference voltages are noisy and can create misreads of digital information.	Filter capacitor is blown and open circuited	Audio signal is garbled	Observation with Capacitance Meter	Medium/Low	Medium/Low criticality because it is a nuisance to the user, but also degrades the quality of the audio signal.

Table 7 – Preamps

Failure No.	Failure Mode	Possible Causes	Failure Effects	Method of Detection	Criticality	Remarks
7A	Potentiometer is not changing the voltage divider on the preamp	Gain potentiometer is broken	Gain knob does not change gain	Observation with DMM	Low	Low criticality because it is simply a nuisance to the user.
7B	Missing components add noise to the signal that is very small to begin with	Resistor, capacitor, or transistor is blown	Preamp is noisy	Observation with DMM	Low	Low criticality because it is simply a nuisance to the user.
7C	Op-amp does not pass the signal through	Op-amp is blown	Preamp has no output	Observation with DMM	Medium	Medium criticality because it disables the functionality of the unit