Hybrid Synthesizer

GROUP 32

Department of Electrical Engineering & Computer Science

University of Central Florida

Dr. Samuel Richie

Senior Design II – Spring 2015

Sponsored by Boeing and TI

Michael Smith	msmith312@knights.ucf.edu
Marlena Brammer	marlena.brammer@gmail.com
Josef Hirmann	jbhirmann@hotmail.com

Hybrid Synthesizer Table of Contents

1.0 Executive Summary

In the world of music, various electronic devices have made their way into halls of fame for their particular character or beloved versatility. This naturally extends to synthesizers. While synthesizers originated as futuristic, alien sounding keyboards that were initially written off as a fad or phase and constrained to only a handful of genres such as “techno”, that have since bled into many other forms of music, particularly modern pop. It would be hard to imagine most modern music without the synthesizer. Throw a dart at the Top 40 list, and you’d be hard pressed to find songs that make no use of some kind of synthesizer. Everyone from Lady Gaga to Katy Perry makes use of the rich, diverse sounds synthesizers can grant in their most recent works.

As such, demand for these instruments has only risen as they’ve captured the public imagination with their unique technical sound. There have been a variety of analog and digital synthesizers made available to the market in response. They can run utilizing analog components, software signal generators, or reference sampled sound libraries to create a wide array of sounds. While the instruments are fascinating and have become an important addition to most modern music, they are also prohibitively expensive to acquire. When a decent synthesizer costs $1000 on the low end, it is already a stretch for most consumers. On top of that, most synthesizers only create music using one approach (analog circuits, virtual instruments, sampling), meaning it is even more prohibitive for an individual to really get a feel for which synthesizer is best worth the investment for the sound the artist wishes to create. Additionally, most synthesizers are large and cumbersome, making them less than ideal for travel or live performance.

The synthesizer our team wishes to create would be a small hybrid synth, in that it would utilize both analog circuitry to create classic electronic sounds as well as a software environment that can simulate various well-known instruments, and even allow the user to sequence a series of notes on repeat so they could tweak and mix the presence of various aspects of the wave’s character (such as the presence of sine, square, triangle, or sawtooth waves, or the shape of the ADSR envelope). Additionally, the synth would be light, small, and portable. To save further on implementation, a custom coded keyboard application that can run on an android phone would be connected to the synth via USB, allowing the user to play real-time or sequence notes for playback either through instrument virtualization or through the analog components. This would allow the synth to retain a small profile while also enabling it to serve as a sort of audio playground, where the user could experiment real-time to get the most out of the sound.

The box would be aimed at being an affordable alternative to multi-thousand dollar synthesizers, while still providing an all-in-one quality sound synthesis source for artists with tight budgets. It would have 1/4” and potentially 1/8” (headphones) line out to be plugged into an audio interface for recording or a speaker/headphones for live playback.

2.0 Project Goals and Objectives

The Hybrid Synthesizer will conform to the following specifications:

- Needs to be light, ideally eight pounds or less
- Must be at least monophonic for at least two octaves in the analog design
- Must support at least 3 waveforms in the analog design that generate a consistent, reliable signal
- Must support at least 3 common orchestral instruments in software synthesis (piano, flute, clarinet, etc)
- Must support either plugging in an android phone with custom keyboard application, plugging in a USB keyboard, or having a physical keyboard designed (whichever is better to implement for the design)
- Must disperse heat well and ideally consume relatively low power

The above specifications were all adhered to successfully.

The design team has also designated a series of additional goals that will be supplemented or removed as necessary based on time and resource constraints and what is best for the design overall:

- Have four or more unique signal generators in the analog system that can be mixed real-time using a summing op-amp
- Have a basic envelope (2+ stages) to shape the analog sound
- Have a basic lowpass/highpass/bandpass filter that can be swept across the analog sound
- Have some form of LFO that regulates tempo or other aspects of design
- Have a sequencer, either in hardware or software, that allows the user to automate playback of notes in either the analog or virtual instrument domain
- Consider battery power for extreme portability
- Consider polyphonic sound for both software and hardware sound synthesis

However, these additional goals could not be realized beyond the first bullet due to design overhaul and time constraints.

2.1 Division of Labor

Figure 2.1.1 shows a visual representation of the division of labor and how the segments of the design and those responsible relate to each other.

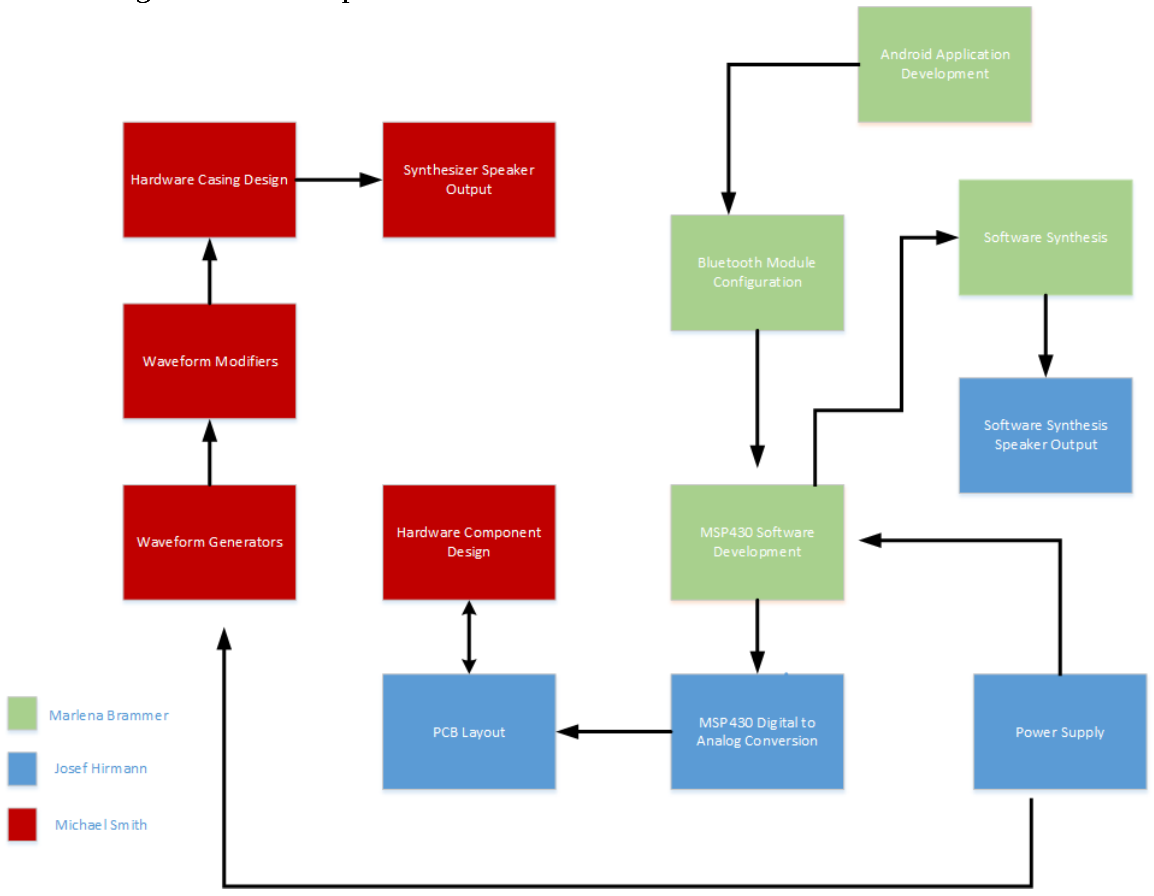

Android Application Development
Hardware Casing Design
Synthesizer Speaker Output
Bluetooth Module Configuration
Software Synthesis
Waveform Modifiers
Software Synthesis Speaker Output
Waveform Generators
Hardware Component Design
MSP430 Software Development
Marlena Brammer
Josef Hirmann
Michael Smith
PCB Layout
MSP430 Digital to Analog Conversion
Power Supply

Figure 2.1.1: Block Diagram of Labor Division. Image owned by the Hybrid Synthesizer team.

Michael was in charge of the analog circuitry and components that go into that portion of the design. The oscillators and all waveform manipulation modules related to those analog oscillators were his responsibility, as well as coming up with the design for any casing that was used for the analog components.

Marlena was responsible for coding the android application to be used on a mobile phone or tablet, as well as managing the Bluetooth communication between that device and the microcontroller. She was also responsible for the software portion of all sound synthesis, as well as being in charge of designing and coding the final website for the Senior Design project.

Josef was responsible for any control voltages and digital to analog conversion that may need to occur between the microcontroller and the analog circuitry in order to get the two halves of the Hybrid Synthesizer functioning together.

3.0 Research Related to Project Definition

Currently there are many products on the market that cater towards the market our product aims to cater to. These are discussed below.

3.1 Existing Similar Projects and Products

The basis of this project was to create a synthesizer that would use analog and digital signals in a hybrid fashion. People these days like the convenience of using a digital medium such as applications and computers, but it is difficult to produce and manipulate a sophisticated sound from a digital application alone. This is why a Hybrid Synthesizer would be the best of both worlds. So far, all the research into this project and there hasn't been a completely Hybrid Synth, only either digital or analog.

3.1.1 Arturia Microbrute

A good example of a portable Analog Synthesizer would be the Arturia Microbrute (Figure 3.1.1.1). This product is one of the more affordable synthesizers and it is also extremely portable. Common professional analog synthesizers can run upwards of $1500 whereas the Microbrute is on the low end, costing only $349.

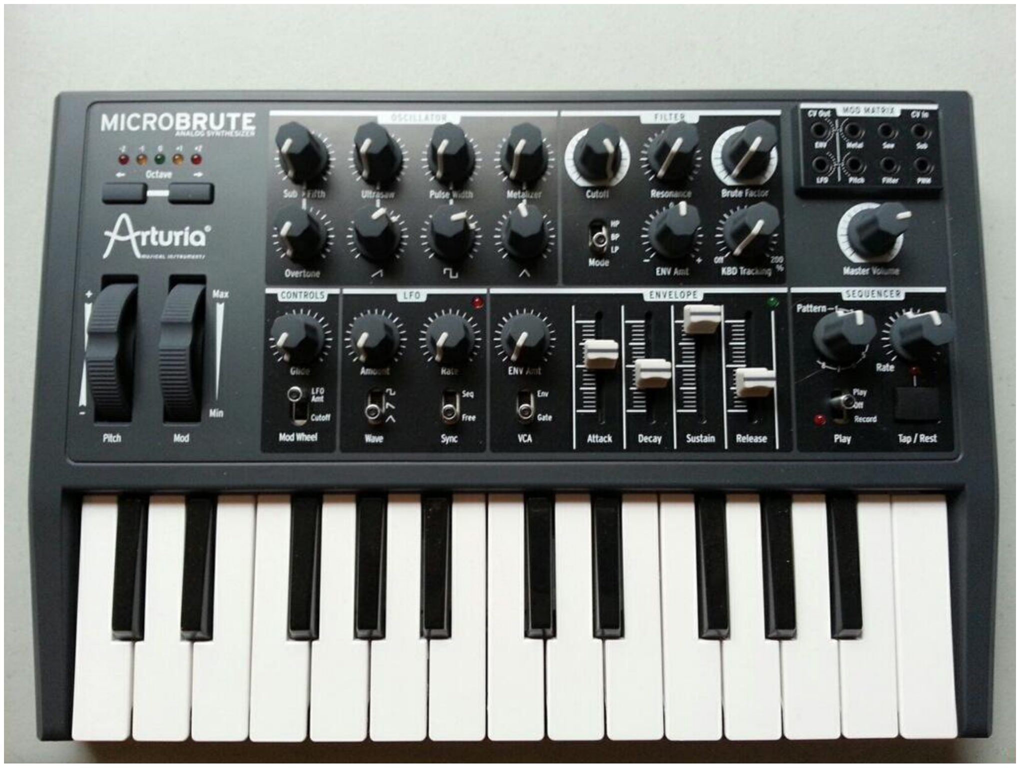


Figure 3.1.1.1: The Microbrute by Arturia. Image owned by the Hybrid Synthesizer team.

The important feature of this mini synthesizer is the fact that it is able to generate the waveforms that would be used in this project. It is able to create a saw tooth waveform, triangular waveform, square waveform and it also uses an oscillator. It has a 25 key keyboard which is attached to the device. It also has envelopes that add “jagged-ness” to the waveforms which make the sound more interesting. It only uses a 12V power supply, which is reasonable for this kind of project. Still, it doesn’t have a digital component, and if you ever wanted to save a creation that was made using this device, a USB connection to a laptop would be needed, no phones allowed.

3.1.2 Moog Sub 37

Another comparable model would be the Moog Sub 37 Tribute Edition Analog Synthesizer (Figure 3.1.2.1). It is also an analog Synthesizer with essentially the same capability as the Microbrute It has more advanced features like 37 keys, two oscillators, a pitch bend and a mod wheel. A tradeoff is that it weighs a whopping 22 pounds and uses 110V of power. It also has a hefty price tag, costing $1,579 retail. The only advantage is that it has software applications that are available for purchase. The phone apps are iOS only and the computer applications emulate the Little Phatty synthesizer by Moog. The computer applications will set you back around $70 and they have to be used with the Little Phatty Synthesizer in order to be functional. The MIDI files are transferred via USB as well.

Figure 3.1.2.1: The Sub 37 by Moog, used with permission from Moog

3.1.3 Modulus 002

There is also a small company out of the United Kingdom that was only recently founded in 2013 that has created a hybrid synthesizer which is very similar to the project that this design team wants to develop. The company is called Modulus Music, and the product is Modulus 002 (Figure 3.1.3.1). This model was released in July 2014, which is one of the newest products to hit the market. It features a touch screen user interface which has a 4.3 inch display screen, which is unique in the synthesizer world. The display changes to coordinate with the last button that was touched so there really is no menu to navigate through. The hybrid side comes from an analog/digital architecture which has very stable oscillators, and it can produce 50 different types of waveforms which can be either analog or digital. Most synthesizers on the market today are either polyphonic or analog but this one takes the best of both worlds. It also has two oscillators and two sub oscillators which can create a sound of four oscillators per voice. It has 5 octaves on the keyboard and twelve quick recall banks that are used to store presets, sequences and animations. You can also use the Ethernet jack to connect to the internet because you have access to the Modulus cloud features, which is a cloud-based server platform that allows musicians to collaborate and share sequences and sounds. It is open-source because this company wants to change the future of synthesizers by having this be the standard for all music creators. All of these luxuries come at a price, and because it is sold in British Pounds normally, when it is sold in the United States it comes out to a whopping $5,200.

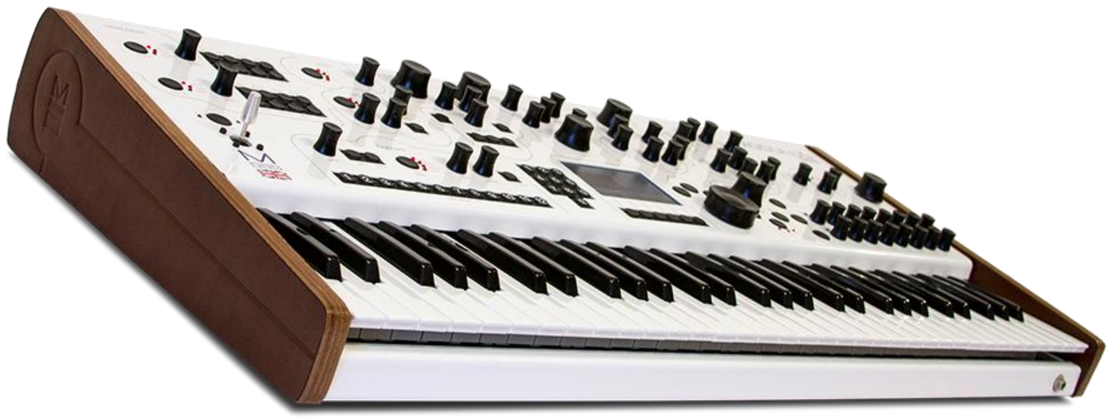


Figure 3.1.3.1: The Modulus 002 ©Modulus Limited, 2014

3.1.4 Music Synthesizer for Android

On the software side, there are many Android applications that are software synthesizers. A popular one based on the Google search results is the Music Synthesizer for Android. It is an open-source application that emulates the Yamaha DX7. The user interface shown in Figure 3.1.4.1 has a black and white keyboard and different level knobs. When a key is pressed, it turns orange as a visual aid. It also has many different sound emulators for all the different instruments that you could think of. It has three knobs, which control Cutoff, Resonance, and Overdrive. The keyboard has white and black keys, but they are in two different rows which could make it difficult to use if you are used to a standard keyboard. There is no way to record the sounds and play them back however, but since this software is being developed daily by online contributors, that could be in the near future.

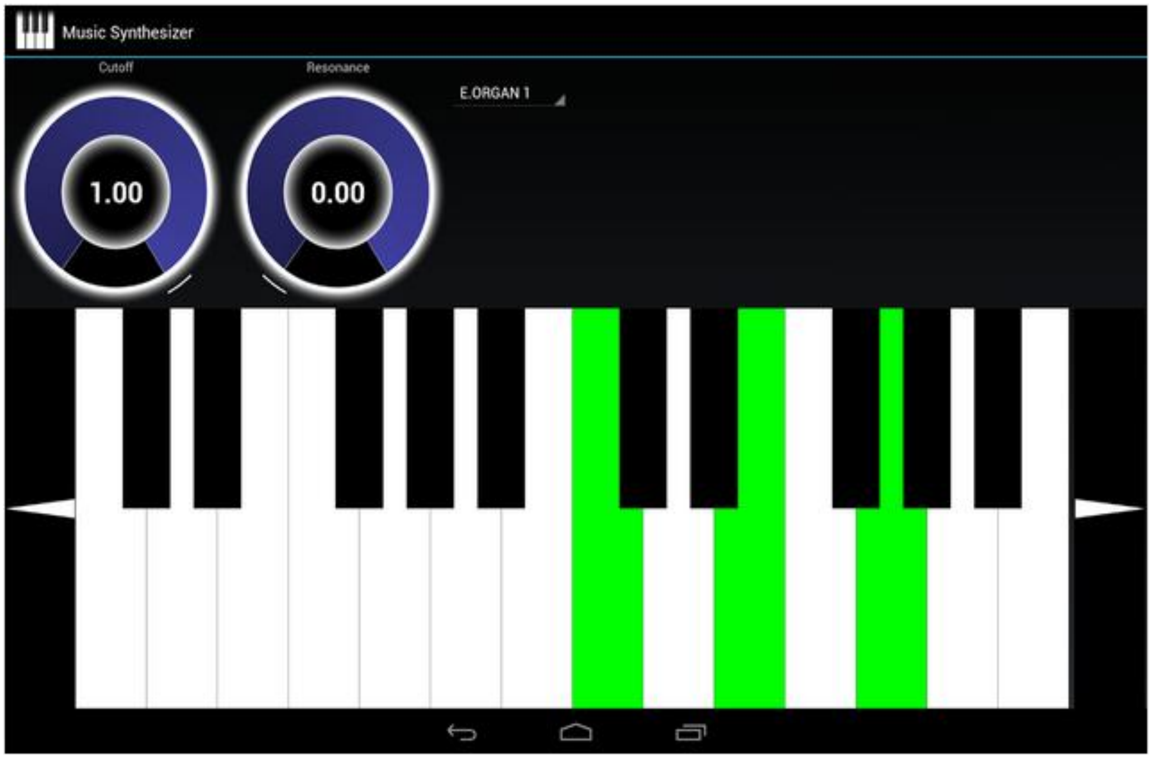


Figure 3.1.4.1: Android Music Synthesizer User Interface used with permission from Raph Leiven

3.2 Relevant Technologies

3.2.1 Components

3.2.1.1 DACs, BJTs, and Regulators

A large majority of the components used in any synthesizer are essentially similar across the spectrum when dealing with either analog or digital synthesizers. The tools and parts to send signals from the digital board to the analog circuitry will require a digital to analog converter specifically. Digital to Analog Converter (DACs) when built in circuit form usually use a resistor ladder which consists of a voltage divider network which divides the reference voltage down by ½ for each ‘rung’ of the ladder and then a summing amplifier. Voltage out depends on which bits are on or off. Devices are pre-built to provide digital to analog conversion and this component is directly relatable to the micro boards as well as various microchips which take in a digital value and send out a continuous signal to the best of the device's ability. Using digital data to create an analog signal is a potential possibility for building a synthesizer because it contains the goal that the hybrid synthesizer is attempting to reach which is combining digital and analog devices to create a music creating machine.

One such device is the Digital-to-Analog/ Analog- to-Digital Multichannel converter chip manufactured by Texas Instruments. This chip is capable of being powered by voltages as small as 2 volts to 3.8 volts which can be powered by the MSP430 Launchpad board and uses both analog and digital signals.

Microchips and voltage signals that translate over to other device and run certain analog parts or vice versa, run digital parts are an extremely delicate task due to the accurate sampling needed and the requirement for the cleanest signal available. In other words the signal must be sampled at a frequency that contains a high signal to noise ratio. To achieve one of the clearest signals to sample for any analog signal, the harmonics of the signal are required and Texas Instruments provides the "Analog to Digital Harmonic Calculator". This is a tool for excel based calculations that proves useful for determining the harmonics location in the frequency space. When searching for high order harmonics that are following the Nyquist aliasing in analog to digital converters this tool can pinpoint where the harmonics of the waveform are located. Given an analog-to-digital Sample rate, and the span of a signal, the calculator determines if the second through the ninth harmonic will fold back into the frequency band of the signal. These components will be relevant in converting signals between the digital and analog sides of the synthesizer while also discovering the best frequency to send the signals.

More components such as the Bipolar Junction Transistors are devices that rely on semiconductors for its operation and are resourceful pieces. The devices have a major amount of versatility and can be used as amplifiers, switches, or in oscillators. These devices are categorized with a Base, a collector, and an emitter and can be viewed as being controlled by the base-emitter which is a form of current or voltage. By sending a voltage to the base-emitter, the device can control the amount of current flowing to the analog circuits. Possible voltage regulators could manipulate any increase in voltage derived from a power source and expanded with a BJT. The most efficiency regulators with the lowest noise are the 78XX and 79XX families because of their fixed linear voltage regulation The 78 series creates a limiting positive voltage whereas the 79 series provides a negative voltage. The advantages of these components are the series IC's do not require additional components to provide a constant, regulated source of power which relieves space when bread boarding. Also, these regulators have built in protection against a circuit drawing too much power, or if short circuits and overheating occurs. Sometimes this device is used as a protection unit for not just itself but other parts of a circuit as well. The disadvantage is that the input voltage must always be higher than the output voltage by some minimum amount. Another disadvantage is that the input current required is always the same as the output current so the total power going in will be more than the output power provided. Taking this into calculation will resolve any design problems but a heat sink must also be provided when input voltage is significantly higher than the regulated output voltage due to heat dissipation. Conclusively, these regulators are useful for powering micro-chips like operational amplifiers which require a positive and negative supply of power to run. Combining these components will create a circuit capable of defining voltages or currents to anywhere in the circuit and can be accomplished by using a simple micro-controller to turn the BJT on or off.

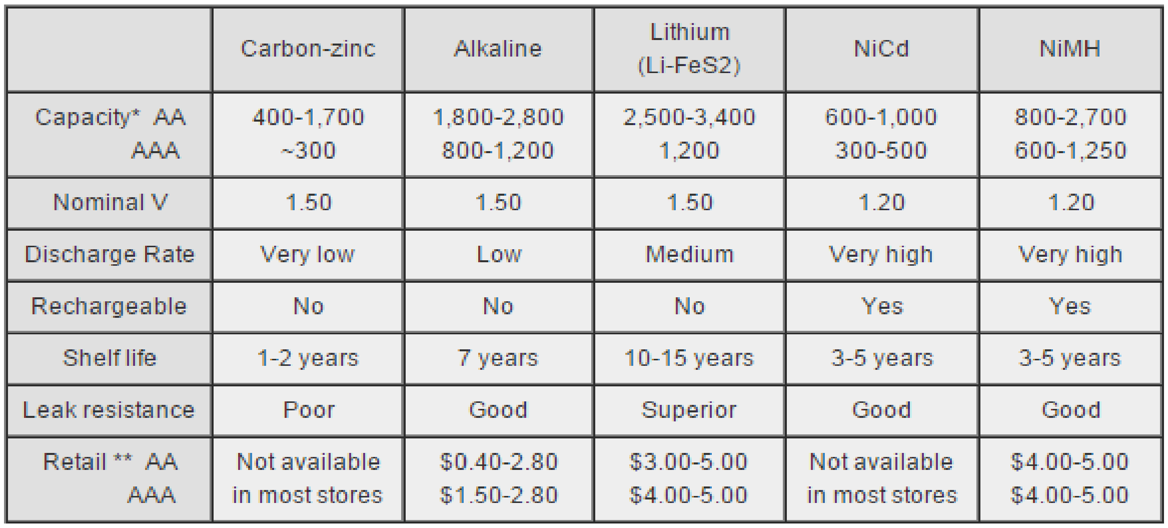
3.2.1.2 Power Supply Modules

The supply modules are Power Controls for all types of controls, interfaces (Ac and DC), slew limiters, Oscillators and LFOs. These Supply modules go beyond just those that are mentioned and truly dependent on the number of cabinets needed, the cabinet style, and the module space. Most power supplies that include the Q100 series are for large synthesizers that are used as stations within studios and other large recording facilities. A small module could be a potentially useful device considering that this entire series is designed with Synthesizers in mind. For a small portable device like the Hybrid Synthesizer, a smaller, practical, and cost efficient power supply is more in-line with the design, and that falls under the category of batteries. Battery Voltage Sources are the first and simplest option to purchase a power supply source for anywhere between 3 volt to 18 volts and the same battery could be used to power the MSP430. Batteries are one of the most common means of powering a device. They range from all different sizes and voltage levels while some are available as a rechargeable form and some are single charged. The U.S. battery industry offers specifically two large categories of battery which are the primary battery and the secondary cell. The first is a non-rechargeable battery while the latter is a rechargeable one. The major difference between the two batteries is the rechargeable, reversible, electric properties. This reaction that occurs in the secondary cell is capable or discharging and recharging the battery through the alkaline chemistry in the electrolytic cell. The primary battery only has high energy density which contains the large charges to distribute to the supply and cannot be recharged so the use of rechargeable batteries seems to be a more useful choice. However, primary batteries still place a useful role since they still contain low self-discharging rates, a specific voltage, and a small price to pay to power a circuit. The way the battery sends out the charge is through a process when the battery is connected to the two terminals but the self-discharging process is a useful feature that keeps the battery from using all the charge while the device is not on. The secondary cell will deplete itself while within the circuit so although it can recharge, the recharging must be reiterated upon its quick depletion. Overall, the primary cell batteries are conveniently used when the device is intended for single session usage and not for continuously using throughout the day whereas the secondary cell rechargeable style is used for frequently using the device such as a laptop or a cellular phone. These types of batteries contain the chemistry needed for optimum longevity in lifetime, consistency, efficiency in price, and least amount of flaws. Standard primary alkaline batteries provide 50 percent more energy that lithium-ion, rechargeable batteries however the secondary batteries have seen a great amount of growth.

Primary Batteries contain high specific energy, long storage times and are ready to be used instantly. A sharp performance and the increased cost of power by 30 fold comes at the cost of the inability to recharge. With a higher internal resistance, the primary batteries' discharge is limited to lighter loads. These batteries are suited to be carried to remote locations and can be used even after long storage. The prices are inexpensive, environmentally friendly and readily available as well. Carbon Zinc is the least costly of the primary batteries and usually comes along with consumer devices. Their applications include power for devices with low power drain like remote controls and flashlights. Alkaline - Manganese Batteries deliver more energy at higher load currents than carbon-zinc but is more expensive in comparison. This battery does not leak when it gets depleted which is vital but if damaged it could leak hydroxide gases due. The result is a pressure buildup which ruptures the seal and causes corrosion that can spread to neighboring parts. All primary batteries produce a gas discharge so portable devices like the Hybrid Synthesizer must have provision for venting. Lithium Iron Disulfide Battery is the most energy-dense battery and is often found in military combat. This battery is a newcomer but still has a higher capacity and lower internal resistance than the ancestral alkaline battery. Further reasons to use this battery are that is has improved low temperature performance, superior leakage resistance, and low self-discharge. The reasons not to use this battery are because it is higher priced and has air travel issues due to the lithium metal content. Recharging this battery or putting in a cell backwards or mixing it with other batteries could cause leakage or explosion.

The capacity of these batteries are measured by discharging them at a low current and the actual value of the capacitance is usually lower than the rated version by statistical means through research [batteryuniversity].The AA and AAA are the most common cell formats. AA is double the capacity of AAA at a similar price making the energy storage cost of the AAA twice that of the AA. A comparison of carbon-zinc, alkaline, lithium, Nickle Cadmium, NiMH, and nickel-zinc are shown in figure 3.2.1.2.1.

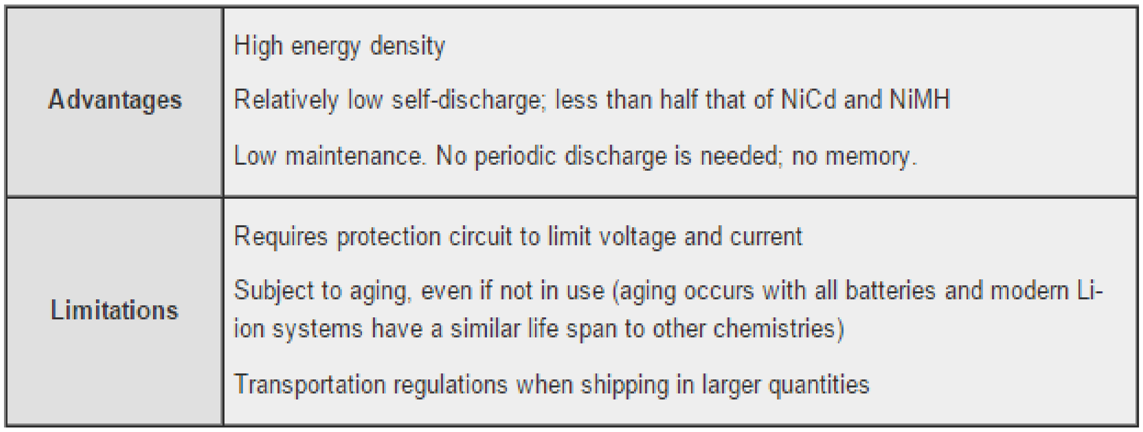
The conclusive result of choosing a primary battery source to power the Hybrid Synthesizers analog components would undeniably be the Lithium (Li-FeS2) battery due to its superior shelf-life, leakage resistance, and nominal voltage.



	Carbon-zinc	Alkaline	Lithium (Li-FeS2)	NiCd	NiMH
Capacity* AA AAA	400-1,700 ~300	1,800-2,800 800-1,200	2,500-3,400 1,200	600-1,000 300-500	800-2,700 600-1,250
Nominal V	1.50	1.50	1.50	1.20	1.20
Discharge Rate	Very low	Low	Medium	Very high	Very high
Rechargeable	No	No	No	Yes	Yes
Shelf life	1-2 years	7 years	10-15 years	3-5 years	3-5 years
Leak resistance	Poor	Good	Superior	Good	Good
Retail ** AA AAA	Not available in most stores	$0.40-2.80 $1.50-2.80	$3.00-5.00 $4.00-5.00	Not available in most stores	$4.00-5.00 $4.00-5.00

Figure 3.2.1.2.1. Comparison of Batteries Courtesy of Battery University and Panasonic, used with permission

Secondary Batteries are the rechargeable ones as previously discussed. The Lead Acid battery is the oldest rechargeable battery system that is low in cost and can handle vibrations, shock, and abuse well but is completely impractical for a device such as the Hybrid Synthesizer. Most rechargeable batteries are impractical for use in a small synthesizer device except for the lithium-ion series. The pure Lithium Ion is used for portable consumer products. The Advantages and limitations of the Lithium-ion batteries can be seen in Figure 3.2.1.2.2.



Advantages	High energy density Relatively low self-discharge; less than half that of NiCd and NiMH Low maintenance. No periodic discharge is needed; no memory.
Limitations	Requires protection circuit to limit voltage and current Subject to aging, even if not in use (aging occurs with all batteries and modern Li-ion systems have a similar life span to other chemistries) Transportation regulations when shipping in larger quantities

Figure 3.2.1.2.2 Advantages and Disadvantages of Lithium-Ion Batteries, used with permission from Battery University

The derivatives include Lithium-ion cobalt which has a high specific energy and moderate load capabilities and a reasonable service life considering its price and internal makeup. Others include Manganese and phosphate, and the first is capable of high charge and currents but has a low specific energy (usually used for power tools) while the latter has a 3.3 Volt nominal voltage per cell which includes a long life cycle. Figure 3.2.1.2.3 gives a side-by-side comparison of these and other batteries. The lithium series has one great developmental problem, the dendrite growth. This problem can compromise safety and destroy the device.

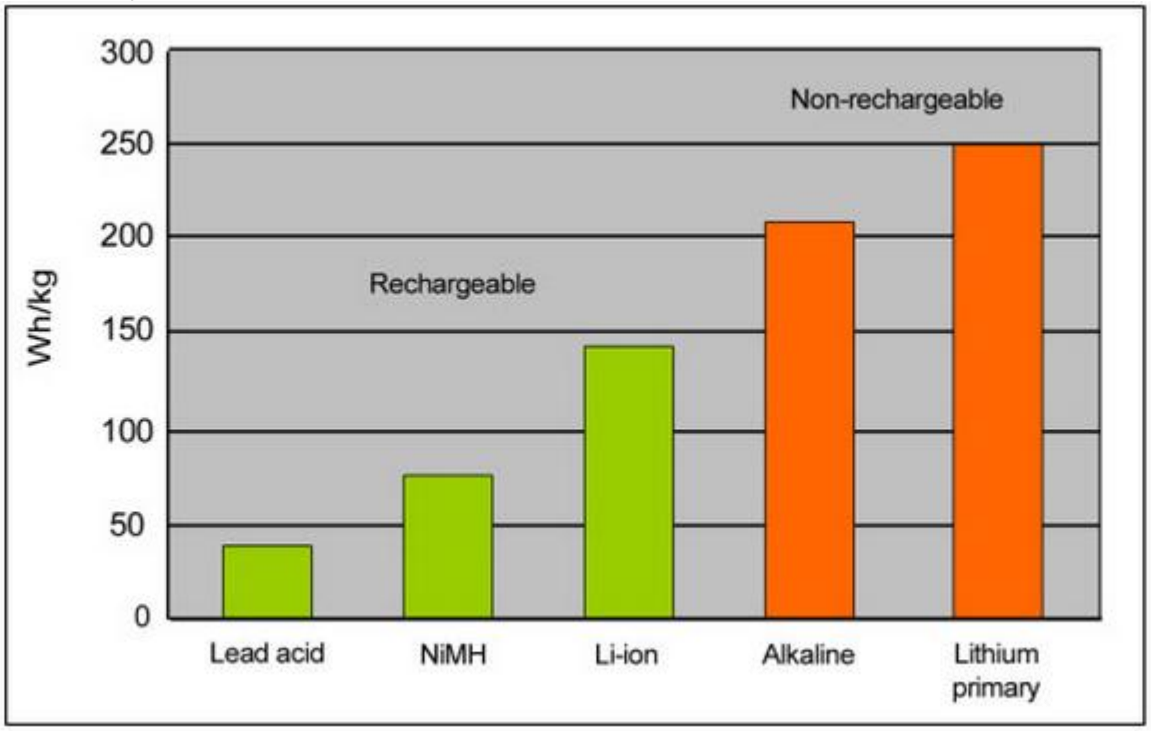


Figure 3.2.1.2.3 Specific Comparison of Primary and Secondary Batteries, used with permission from Battery University

3.2.2 Microcontroller Chips and Boards

This simplest microchip and testing board to program is arguably the Arduino series. Specifically, the VS1053b Arduino Musical Instrument Shield MIDI is a technology worth investigating. This is mostly because this particular device is a programmable board that is capable of adding MIDI sound to the Arduino interface. The design is a worthwhile board to consider as springboard into further embedded system designs. With a MIDI codec IC and a speaker connection stereo jack, proper serial commands enable this device to play music. The Arduino is not specifically an item that is wanted in the “Hybrid Synth” project nor allowed in the design however the MIDI technology and instrumental shield contains the useful concept, generating music through a wavetable stored in its memory, and as such, is relevant technology.

The most important part of this device is the coveted play-back/ play-along tracks. Whether the Hybrid Synthesizer will have play along tracks or not remains to be seen. The standard MIDI tone banks contain instruments such as keyboard, winds, strings, brass, and SFX sounds, and a percussion set. The audio playback is indeed an important aspect of creating a musical synthesizer which goes without saying, but another goal of this project it to send in keyboard commands over radio waves in RF communication, Bluetooth, or Wifi. A booster pack capable of this feat is the CC110L AIR Module booster pack**.** The Anaren Integrated Radio (AIR) BoosterPack is designed to provide wireless connectivity to the TI LaunchPad Development Tool. A low power, wireless transceiver extension for the MSP430-EXP430G2 launchpad development kit is what this device is; the Anaren Boosterpack chip is a useful device for existing applications where a simple upgrade to wireless is desired. Usually used as a Radio module with integrated antennas, this device is great for network status reporting and can provide those necessary RF communications needed. This type of board attaches to the MSP430 launchpad and just about any other launchpad in the MSP430 series. Also, based on the micro-chip installed in the launchpad, the device would be able to store the appropriate amount of code, set up reference voltages, and be run via the boosterpack module. The microchips that are under scrutiny are the MSPF Series. The F149 chip is a suitable candidate due to its common advertisement as an in-system programmable flash microchip that has implemented a solid state voice recorder.

The MSP430 can concert the analog pattern to digital with an integrated analog to digital converter and also store voice data in real time MCU Flash memory and play it back. Real-time audio could be recorded by an actual instrument, and output through speakers connected with the synthesizer. This would be an attractive feature for the user willing to hear different instruments played in the background while they themselves play a tune however, recording a few notes would not necessarily be alterable and the recording lasts for a mere 10 seconds with this chip. The MSP430F2002 is another credible and useful ultralow-power speech playback solution. Speech output can be added to any application with any other MSP device because of the small code size and minimal requirement for peripheral hardware. The architecture has 5 low power modes to optimize the extended battery life in portable applications. Also, 16 bit RISC CPU and 16-bit registers and constant generators make for very efficiency code processing. Notably, the family member MSP430F2003 has a more advanced ADC (SigmaDelta16 - 4ch) which could be a useful counter-part. Building a synthesizer using this device would mean that the audio data will be hard coded and stored in nonvolatile memory. The audio quality and length of the notes are to be tested and appropriately coded based on which board this chip ends up on. An audio sampling rate of 8 ksps allows an analog audio frequency of up to 4 kHz, which is enough for comprehensive speech. The software can be stored, and the audio data can process and output to audio signal as is the device’s function. This micro-chip seems to be much more practical for use due to its functionality and low cost.

3.2.3 MSP430 Launchpad Boards and Controllers

The MSP430 microchips are potential options for the hybrid synthesis due to their simple connection use of voltage reference through output pins, provided functional diagrams, allocated Memory, build in Central Processing Units, Memory Mapped input and output, clock generators, Interrupts and Resets, and available software for code development and continual progression towards unique, authentic designs. With so many boards available, a specific criterion is vital towards narrowing down the vast amount of launchpads and evaluation boards on the market.

The MSP430F5529 Board is one that meets part of the criteria as this ultralow power device has the architecture for an optimized extended battery life in portable applications. MSP430F5529 has the usual arrangement of 16-bit RISC CPU, 16 registers, constant generators that contribute maximum code efficiency, digitally controlled oscillators (DCO) to wake up from low power states. All technologies incorporated into this cost efficient board make for a versatile device. This board is a better equipped than the standard MSP430 Launchpad and is also able to combine with the Launchpad and other Bluetooth BoosterPacks as well.

The entire MSP430F5XXX series contains four 16 bit timers, which can clock multiple functions going on in the synthesizer. A high performance 10-bit Analog-to-Digital Converter is a required feature that all of these boards contain but the specific purpose these boards are used for do not usually include any type of audio synthesis or connection through playing music. Implementing a speaker, installing audio based microchips, and creating circuitry to make for a synthesizer with this board would not be impossible but the design would be made much easier and more efficient with a board that is created specifically for that tasks which brings us to the MSP430CC256x Audio Sink. This device is TI’s Bluetooth + MSP430 design used to creating a plethora of useful applications. One such application that is of great interest to this project is the audio streaming accessories, and because it is a cost effective audio implementation design with full files provided through Texas Instruments. This device enables RF communication and is compatible with Bluetooth audio with the low power MSP430F5529 microchip.
The device is equipped with a speaker connecter and a TAS2505 speaker amplifier chipset so we can send in an audio signal out with a Musical Note all with this one device. Combining this device with the Launchpad will make for an increase in memory and storage capacity and thus be able to manage a more efficient and complex source code. In addition the board design is portable, has a long lasting battery life, and store a variety of audio files. This device enables Bluetooth audio (SBC encode/decode) with low cost and low power via the MSP430F5529 experiments board. Connection to computers is possible through the USB-enabled MSP430F5529 and contains Up to 25-MHz System in the Clocking system. The output voltage is 1.8-V to 3.6-V operational and 128KB of flash memory that can remember what the device was doing even without a power supply attached, although only for a short amount of time. The 8KB of RAM allows for the storage of wavetables and truth tables to which the notes will be stored. Yet another feature is that this device design offloads audio processing from MCU to the Bluetooth device which enables low power audio. It is extremely cost effective as a low end wireless audio solution. The core of the solution is TI's CC2564 is best in known class Bluetooth performance (+12dBm output power). The designed compatibility with the CC256X Audio Sink and a Hybrid Synthesizer makes this board an optimal option.

3.3 Possible Architectures and Related Diagrams

3.3.1 Audio Synthesis Methods

Over the years, multiple styles and methods of creating audio have been developed. The usual method is to use oscillators controlled by voltages and amplifiers to recreate acoustic sounds. Nowadays, the fast processors, increased storage space, and data formatting allows for a greater, affordable, synthesis method.

Additive Synthesis is one such method that is a sound synthesis technique that creates timbre by adding sine waves together. This method is straight forward and east to implement. The generation of a sine wave can be creative both digitally and in analog. The underlying presupposition is that the simple summing of these waveforms allows a complex waveform to take place. The difficulty would be to recreate a complex waveform a lot of oscillators would be needed. Not available in analog synthesis.

Subtractive Synthesis is similar to additive synthesis but in an opposite method, synthesizers a partial of an audio signal that usually has many harmonics and attenuated it by a filter to alter the timbre of the sound. In other words it takes a complex waveform and sends it through a series band filters that removes harmonics to create a more practical waveform. This style would only require a single oscillator for each note and can be used for Analog or Digital Filtering but can also be used with multiple oscillators.

Frequency Modulation Synthesis is a form of audio synthesis where the timbre of a simple waveform is changed by modulating the frequency in the audio range. This synthesis is known to produce a tone that if dark and ‘gritty.’ FM synthesis can create harmonic and inharmonic sounds based on the modulating signals relationship to the carrier. As the amount of frequency modulation increases, the sound grows more complex. This use of modulators with frequencies that carry non-integer multiples can create atonal and tonal sounds like drums and ringing bell sounds. This can also be used digitally or through analog oscillators although digitally is more reliable.

Wavetable Synthesis is a digital form of sound synthesis technique that creates periodic waveforms stored in memory. A table of these waveforms are algorithmically computed and created in a simple or complex output wave. This style of synthesis relies on the sampling rate and bit-depths as they require little processing power and memory allocation which makes it a perfect candidate for the MSP430 chips.

Sample Based Synthesis is a form of audio synthesis that can be contrasted to either subtractive or additive synthesis. The same principles from the wavetable synthesis are used in this method however a sound file is used to generate the audio output. This method can aid the Hybrid Synthesizer in replicating acoustic instruments, pianos, organs, etc. By recording each note from the instrument and passing it through an analog to digital converter, the recording would get loaded as a sound file into the synthesizer. Obviously, this method requires a specified amount of digital storage requirements which ideally will be increased in the Hybrid’s design through MSP boosterpacks. Some professional libraries run into a Gigabyte to Terabyte range so Memory Storage will definitely be a vital role when using this type of audio synthesis.

Granular Synthesis is yet another synthesis method that operates on the micro-sound scale. Based on pre-processed sampling sounds, each pieces of sample is split into “grains” that have the potential to be layered on top of each other and played at different speeds, phases, volumes, and frequencies. It is not sample based synthesis nor wavetable but rather a combination of the two.

Direct Digital Synthesis Devices is used for creating arbitrary waveforms from a single, fixed-frequency reference clock. This technology includes applications for signal generation, local oscillators, function generators, mixers, modulators, and sound synthesizers. The oscillator provides a time base and determines the frequency accuracy of the device and outputs the quantized from the version of the sinusoid output waveform. The sampled digital waveform is overall converted to an analog waveform by the DAC. The output reconstruction filter produces a zero-order hold inherent in the analog conversion process.

Conclusively the Wavetable synthesis and the Subtractive synthesis are ideal methods to be used within the Hybrid Synthesizers Architecture due to the components used and the methods available. The MSP micro board is a tool that is capable of storing hard coded data what the wavetable requires and the sampling methods and bit-depth is all available in this micro-board. The Subtractive synthesis is great for Analog and Digital methods which is exactly what is called for in a Hybrid board due to the multiple oscillators and waveforms that are to be generated.

3.3.2 Sampling, Signal Conversion, and Audio Resolution

The method for receiving multiple notes from the keyboard input and sending them out to the audio output speaker is an intricate process and requires methods of receiving multiple signals within a specified time domain, altering them appropriately and outputting them in a fast period. When a single note is played the goal is to output the note quickly and when multiple notes are played the goal is to play each of the notes simultaneously. Time Division Multiplexing is a method of transmitting and receiving independent signals over a common signal path by means of synchronized switches at the end of each transmission. This method of sending and receiving signals could be a useful advantage for the hybrid synthesizer because playing multiple notes at a single times and still receiving all the notes for audio output is important. Also, it can be used for both digital and analog signals by dividing the time domain into several time slots of fixed length.

With the Hybrid Synthesizer, a potential aim is to utilize features of the MSP430 device to convert between the analog and digital components which will be used as a peripheral Digital to Analog Conversion (DAC) and use the internal clocks and source code to sample the signals being sent. To send signals from the digital MSP430 to the Analog components as a waveform or in reverse the method of Digital to Analog Conversion is necessary. Doing this enables us to measure values on a digitized level and send commands based on the value of the number. The digital value is to be sent high or to zero which would be converted into subcategories by the digital to analog converter. The main requirement is that a wide amount of sampling, quantizing, and encoding of this analog signal must be used to convert the signal into digital values when receiving an altered waveform and analog when outputting a digital waveform. The digitized sample could do with conversion into an analog audio signal for clearer listening and to do that is must be sampled.

The sampling theorem states that a signal may be reconstructed without error from regularly spaced samples taken at a rate in samples per second, which is at least twice the maximum frequency present in the signal. To read in signals between the digital components and the analog components require complex modulation and sampling. The MSP430's built in ADC uses a method known as Pulse Code Modulation to digitally represent sampled analog signals. It is the standard form of digital audio in most devices around the house. The workings are based on the amplitude of the analog signal, which is sampled regularly at uniform intervals. Each sample is quantized to the nearest value within a range of digital steps. This is the analog to digital conversion and to recover the original signal, a "demodulator" can apply the produce in reverse.

The average MSP430 contains a peripheral for DAC and ADC so transmitting signals between the two sides would be an easier feat than creating an individually integrated circuit. The amount of resolution for a Digital to Analog or Analog to Digital converter depends on the number of bits it has: Resolution = (2) ^N where N = Number of bits.

Therefore a 10-bit analog to digital converter like the one in so many MSP430 boards contains a resolution of 1024. Depending on the voltage reference, the voltage resolution would bout the Voltage reference divided by the number of bits. The bits needed if this architecture is integrated into the design would most likely be around 4 bits because of the 16 combinations that can account for the 12 notes within an octave. If the voltage reference was 8 volts and 4 bits of data, for example then the voltage resolution would be 8 volts / 16 bits = 0.5 volts. An analog signal, albeit not a perfect sine wave, does have various curved values that rise and fall like figure 3.3.2.1. The quantizing takes in the values and converts them to levels than computers can understand and is seen in figure 3.3.2.2. The pulse amplitude is encoded digitally for Pulse-Code Modulation transmission shown in figure 3.3.2.3 and there lies the digital values. The quantization process prevents exact reconstruction of the digitized waveform. The error introduced earlier, the quantization error, is problematic most when a person is listening to a reconstructed speech signal and perceives the quantization error and noise which is known as quantization noise.

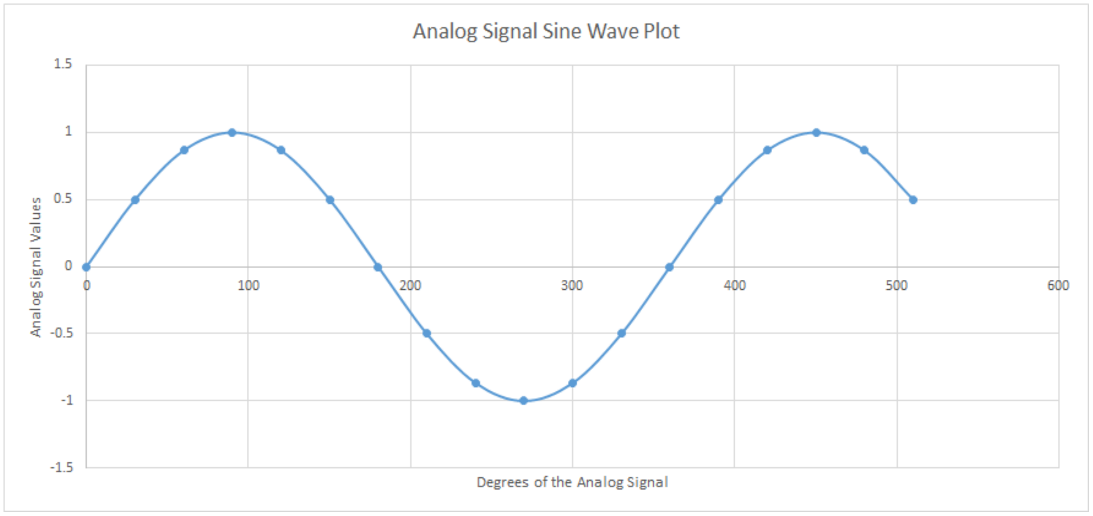


Figure 3.3.2.1: Input waveform and the quantization levels. Image owned by the Hybrid Synthesizer team.

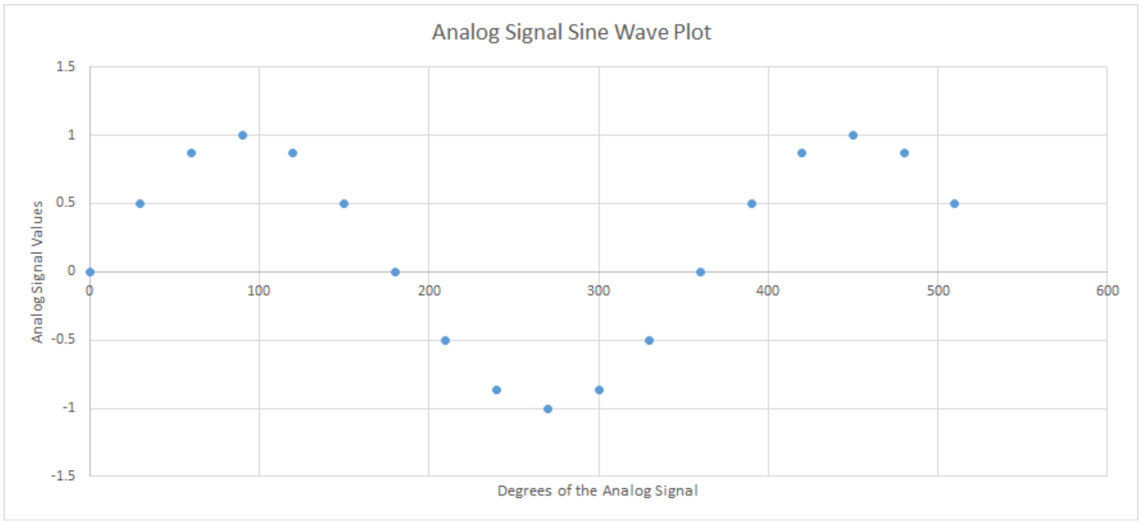


Figure 3.3.2.2: Quantization Samples. Image owned by the Hybrid Synthesizer team.

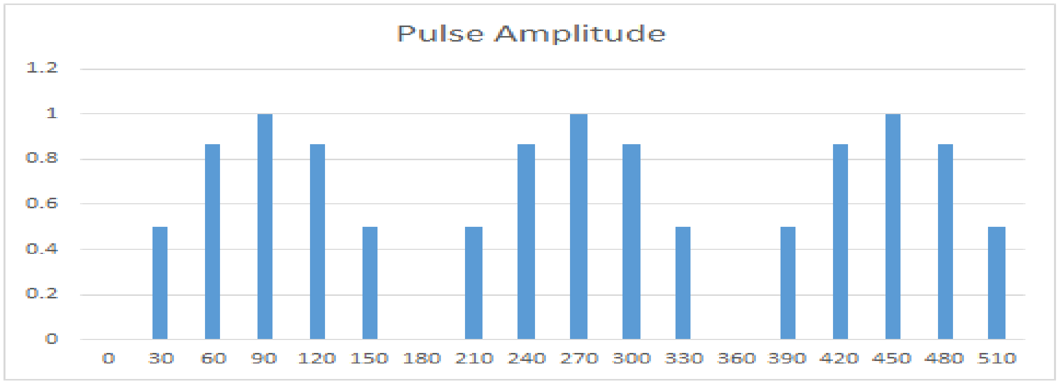


Figure 3.3.2.3: Digitally Encoded Signal for PCM Transmission. Image owned by the Hybrid Synthesizer team.

Quantization error an audio signal is that the audio signal no longer sounds like a full instrument or rich tune but more like a robotic version of what is once was. This error is defined as the difference between the actual analog value in terms of voltage and the discrete voltage step value corresponding to the converted digital value. The more errors involved, the more robotic the output sound becomes. The Nyquist theorem demonstrates earlier and called the sampling theorem states that a minimum sampling rate is to be set so that the frequency of the sampled signal can be faithfully reconstructed from the digitized version. Nyquist Sampling Rate is equal to 2 times maximum frequency of the signal however the goal is to overshoot the frequency by a factor of 5 or 10 instead of two in completely optimize sound value quality and minimize errors. A few other alternative methods to digital to analog conversion are the Comparator Method and Successive Approximation.

The comparator method is the backup plan to create a conversion between the two signals, and can be used as a contingency by taking an analog input and sending it through a multiplexer which then gets sent into a physical device converter. Schmitt triggers on the Input Ports of the MSP430 are also substitutes for comparators to perform these analog level measurements but using the peripheral ADC and DAC is much simpler and less noisy. By using two compare registers in the timers to keep track of the both frequencies of the waveforms, another form of architecture can be created. Successive Approximation is another means for analog to digital conversion and the following pieces: a Digital to Analog converter, a comparator, and a register to convert an unknown voltage into a digital value but the method is far too slow for the hybrid synthesizer project.

3.3.3 Comparable Synthesizer Architectures

To create a successful and working hybrid synthesizer, searching and finding another already successful hybrid synthesizer was a helpful pursuit. The features Hybrid 3.0 is an AIR created synthesizer with a high definition virtual interface that combines with analog synthesizers to produce a full range of digital alterations. These alterations are created by adding a plethora of sounds and effects stored internally. The result of the Hybrid 3.0 is a virtual instrument with a comprehensive set of adjustable components that can sound like many types of instruments. Ideally, this machine shares a design that is a goal for the Hybrid Synthesizer which is to create analog components with a large bank of sound effects peppered in and to play the audio output by means of digital calculations and stored instruments. Being another type of hybrid synthesizer, this architecture has similar specifications to the Senior Design Team's Hybrid Synth. A total sum of the device has two simultaneous parts, each loaded with 3 oscillators and a sub-oscillator, filters, LFOs, envelopes, and effects for creating complex patches. These technologies and devices show what is needed to create an efficient design for a synthesizer capable of both analog and digital makeup.

Dual, series and parallel filters are used in this design with Voltage Controlled Filter modes. All analog synthesizers have an active filter built in to alter the harmonics, the timbre, and the pitch. A low pass filter is used on a square wave oscillator signal creates a hollow / woody sound like a clarinet. A classic saw tooth wave signal sounds like a cello. The cutoff frequency and resonance can be controlled by a control voltage signal which can be sent from a keyboard, the envelope generator, or the LFO. The Dual Filter and Saturation Modes have filters that run in parallel or series. These filters can be assigned one per channel to the stereo outputs so Oscillator 1 feeds Filter 1 while the other Oscillator is routed to filter 2. Another interesting architecture here is the concept of stacking oscillators together which creates huge pads. Pads are sustained tone generator. These sustained tones play for extended periods of time when the note is pressed. The internal architecture of this device also reveals that two multi-functional oscillators can recreate the classical subtractive synthesis waveforms for an old analog sound. To explain this further, the Oscillators have a choice of different modes such as: Multi-Square, Saw Sync, Saw Cross Modulations, Saw Multi, Saw Sync, Square Cross, Square Pulse Width Modulation, and wavetables. The Oscillator circuit constantly alternates between two voltages and when the input and output are tied together, the circuit oscillates rapidly between the two values. As 0 goes in, a 1 comes out and vice versa. Adding an RC delay circuit will slow help to adjust the time constant and slows the oscillations to audible frequencies. This technology can be improved to allow the user to have frequency manipulation by means of replacing the constant resistor R1 with a potentiometer.

Modulation options are abound in the Hybrid 3.0 design due to 8 LFO’s capable of creating all sorts of variations in signal. Modulation matrices and multiple step sequencers are also integrated to modulate multiple signals and sequence them in a way that many sounds can be played. The Modulation Matrix technology has 16 assignable modulation sources and more than 20 destinations each. Also, it can assign modulations to an oscillator or filter and allows multiple assignments of sources and destinations with adjustable intensities. In terms of memory, the minimum RAM required for such a feat as the Hybrid 3.0 is 4GB although 8GB is recommended. The replacement of built in sounds and nature of the Hybrid Synth design will allow some relation between some of the Hybrid 3.0 Architecture. These pieces together will account for every corner of the synthesizer realm as this device is at the peak of synthesizer architecture. Following the successful leaders who are already established in this sort of technology, while still in the scope of capability will create a well-established Synthesizer Project. This device gives great insight however matching such architecture is considered ambitious for the Design Project Hybrid Synthesizer.

3.4 Power Regulation Architecture

Simple Regulator Circuits can be created from a resistor in series with a diode. This is one of the simplest, but also most important, designs to be consider due to the fact that an incorrect voltage level could destroy microchips, fry analog components, and completely disrupt and Launchpad boards present. This is the reason that regulator circuits will be attached to the voltage source and throughout the circuit where necessary. The voltage across the diode changes only slightly due to current drawn in from the input. Feedback voltage regulators operate even better than the simple average regulator circuits because they compare the actual output voltage to some fixed reference voltage and any difference is amplified and used to control the regulation element.

The Shunt Regulator contains the regulating element in parallel with the load. The schematic shown in 3.4.1 is the shunt regulator and the load is any circuit powered from regulated voltage. An advantage and possible disadvantage of this type of regulator is that a shunt regulator draws a constant current from the source. The current drawn from the source is the current that flows through the series resistors and that is only a function of the source voltage. Putting a zener diode at the load would have the zener voltage draw the excess current away from the load. This process and architecture will regulate voltage until the load draws more than the current that the zener draws so to increase the regulation, the resistance attached to the input voltage must draw less current that the load impedance of the zener diode. A more advanced version of the Shunt regulator is seen in Figure 3.4.2 and the output voltage is lowered at a smaller value than the input voltage. Depending on the amount of voltage battery that is chosen and if an analog circuit device requires current drawn away from the input, the shunt regulator may be used in the Hybrid Synthesizer architecture.

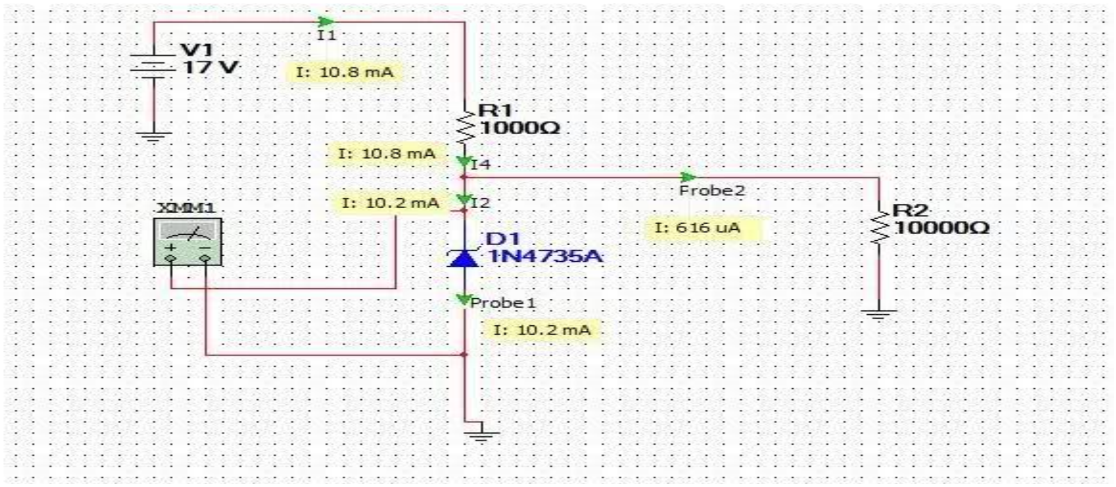


Figure 3.4.1: Shunt Regulator. Image owned by the Hybrid Synthesizer team.

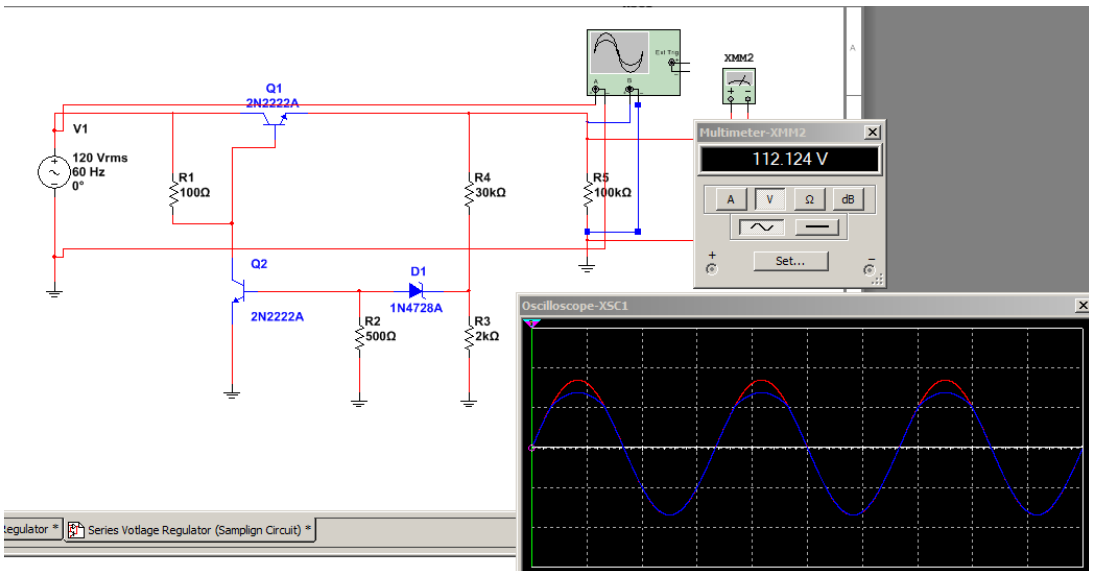


Figure 3.4.2: Advanced Shunt Regulator. Image owned by the Hybrid Synthesizer team.

To power certain components throughout the synthesizers analog components, the use of batteries is the chosen architecture and although having multiple levels of different batteries is impractical, using voltage regulator circuits at attune the correct frequencies to the correct components is a reasonable feat. The circuit in Figure 3.4.3 is a means for providing a final voltage regulation in a linear voltage regulator circuit. Figure 3.4.3 is a Series Voltage Regulator, sometimes this circuit is known as a pass regulator, is the most commonly used circuit for voltage regulation. The circuit is built with a variable element placed in series with the resistors. The advantage of the series voltage regulator is that the amount of current drawn is used by the load and, unlike the shunt regulator, the series regulator does not draw the full current. The series regulator is considerably more efficient. To provide more improved levels of performance it is possible to add a feedback network into the regulator circuit to enable the output to be sampled and compared with a stable reference voltage. The error is then used to correct the output voltage and can minimize signal problems like spikes and ripples. The figure 3.4.3 contains a spike because no feedback network has been added to the circuit yet. These spikes can cause damage to components if not effectively handled.

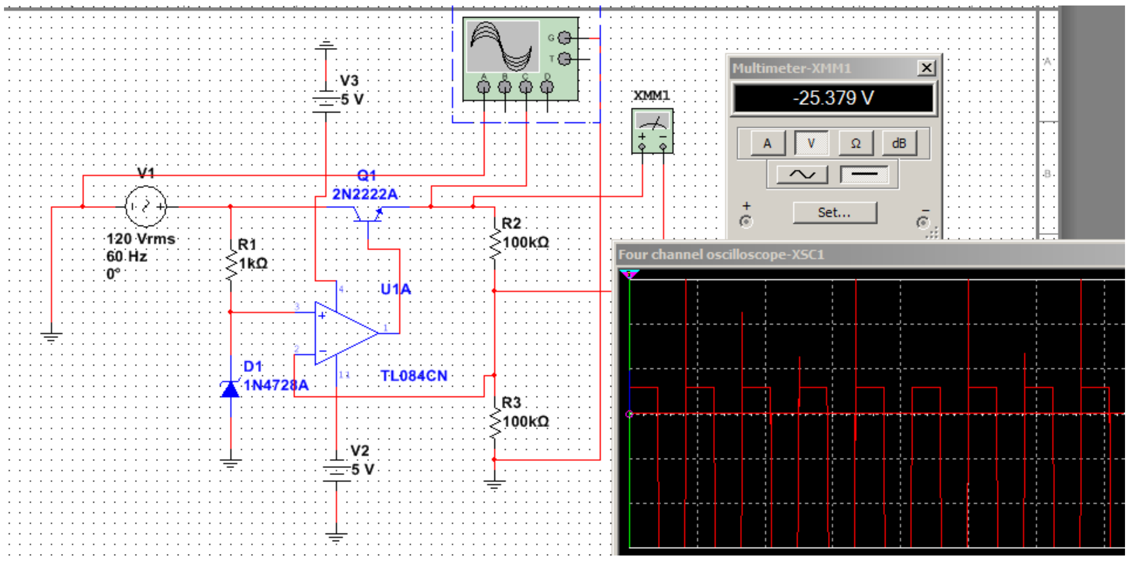


Figure 3.4.3 Series Voltage Regulator: Image owned by the Hybrid Synthesizer team.

4.0 Project Hardware and Software Design Details

This Synthesizer is predicted to include layers of Embedded System Development along with Electronic Circuitry Design, and Computer Programming with Signal Transmission skills. Beginning with an android application that sends commands to the MSP430 via Bluetooth connectivity, the synthesizer device contains a programmed source code capable or reading and playing notes from specific instruments.

4.1 Initial Design Architectures and Related Diagrams

The power source runs the MSP430 and gets and the Analog Circuitry powered up through regulating the voltages and protecting from any unnecessary discharge. The MSP430 uses internal peripherals and the specified microchip to measure temperature control, light LEDs, and check for data overflow detection. When appropriate, the MSP430 will send a reference voltage signal to the analog components telling the device it is time to turn on and create an analog audio wave. The result will be an Audio output from a digital device or an analog circuit based on whichever command is sent from the web application. The principles of how to create this design stem from the previous research into project hardware and software information.

Below is a simple block diagram in Figure 4.1.1 of what the Hybrid Synthesizer's Architecture will look like. The signal gets transmitted from the Android application through the Bluetooth connection, and then sent to the MSP430. The data signal is then read and processed through the MSP430F5229 processing chip and the appropriate commands are brought to the Central Processing Unit for direction implementation.

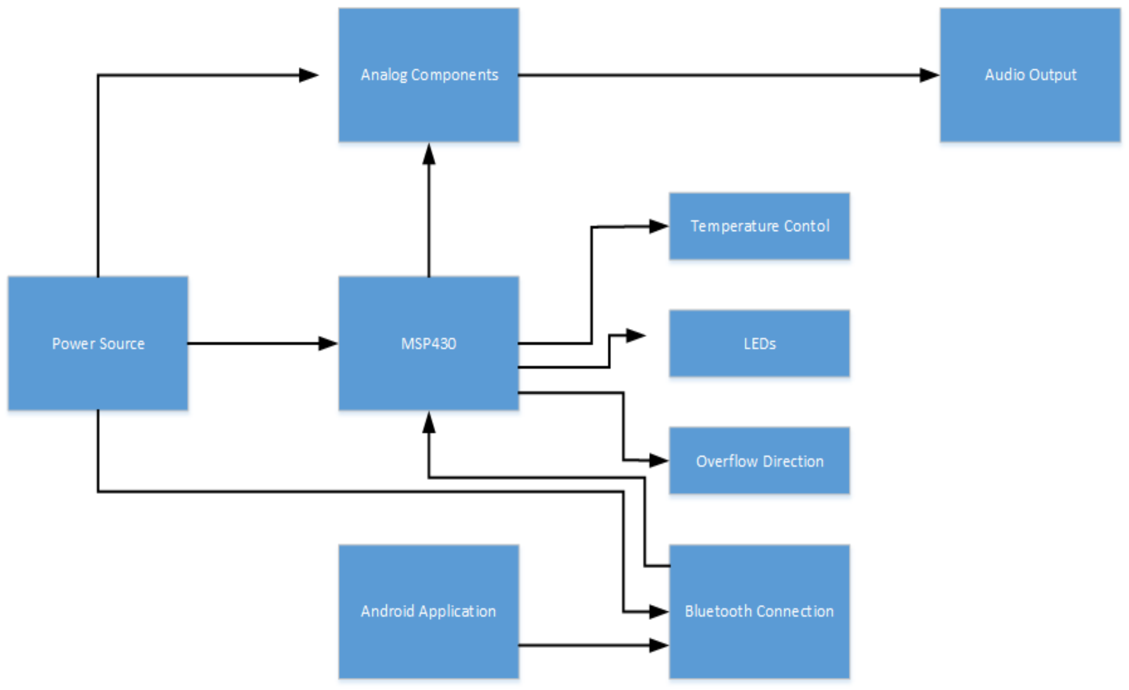


Figure 4.1.1: Hybrid Synthesizer Block Diagram. Image owned by the Hybrid Synthesizer team.

4.2 Project Hardware Design Details

4.2.1 Potentiometers

For the average person, potentiometers are the most visible and accessible of the components that go in to an analog synthesizer (or analog equipment in general), though you would be hard pressed to find an average person who knows them by this name. Potentiometers are variable resistors, and they come in a variety of shapes and sizes and with a variety of triggers that their resistance can vary by. For most analog equipment, especially synthesizers, there are two physical iterations of potentiometer that an individual is likely to see: knob-based and slide-based potentiometers.

Both knob-based and slide-based potentiometers function in a similar fashion, in that they both vary in resistance across their range of motion. This rate of variance is referred to as “taper” and comes in one of two forms: linear or logarithmic taper. Linear taper varies linearly across the entire range of motion of the potentiometer while, as expected, a logarithmic taper increases resistance logarithmically as the position of the dial varies across the potentiometer's range of motion. Logarithmic taper is also referred to as audio taper, which is due to the fact that the human ear recognizes a sound's volume on a logarithmic curve. As such, potentiometers with audio taper are often ideal for volume control on analog synthesizers, as their effect on the audio signal's gain will appear linear due to the way the human ear perceives it (whereas a linear taper potentiometer would appear to grant diminishing returns as it was "turned up" by the user).

Both knob-based and slide-based potentiometers can come with either linear or audio taper, so the selection of which to use is governed more so by their size relative to the physical layout of the circuits and the workflow and usefulness to the end user than by their technical specifications. While knob-based potentiometers are clearly more efficient from a space perspective, as they take up less surface area on the console, slide-based potentiometers are more useful for visualizing scale of whatever is tied to them.

Our selection of potentiometers will vary depending on the functionality of each module. Our mixing console is intended to be a simple waveform blending stage used to sum waveforms together to taste, which is rooted in artistic expression rather than explicit measurement. Considering that most recorded audio is "mixed" to a much more precise level by an audio engineer for any recorded work, the level of precision that large slide potentiometers on a mixing console provide in that instance is unnecessary for our travel-sized instrument, as that stage of balancing will be handled by a live or studio soundboard our synthesizer would be routed to.

Similarly, any simple high pass/low pass filters implemented in our synthesizer would benefit more from the size efficiency of knobs over sliders, as such filters are often used for frequency sweeps that, again, are rooted more in granting a specific "feel" to the sound adjusted by the feedback loop formed by the musician's ear, mind, and arm rather than having an easily traceable quantified visual path. However, for modules such as the ADSR envelope module, a series of moderately sized slide potentiometers placed in parallel would grant the musician a much more useful visual metric of comparison between the stages of the envelope, so in that case some faceplate space should be consumed in order to give additional useful functionality to the user.

4.2.2 Oscillators

Oscillator circuits form the core of analog synthesizers. As such, the team will make sure to include several possible voices so that the user has options for a variety of waveform characters when performing. The most classic and well known waveforms used by analog synths are sine, square, triangle and sawtooth waveforms.

4.2.2.1 Sine Wave Generator

A sine wave is a very important signal for a musician's arsenal, as it is the purest waveform, consisting of energy at a single frequency. While almost grating on its own, giving the musician the ability to blend in a single pure sine wave tone allows them to bring the fundamental frequency of their note forward as much as they need, allowing a musician to give their waveform a little more nuance by lowering the relative level of the harmonics present in the final blended waveform. For a sine wave generator, we have a few implementation options. The first worth considering is a standard Wien-Bridge Oscillator.

The figure below (Figure 4.2.2.1.1) depicts a Wien-Bridge Oscillator, which is governed by the following mathematical characteristics:

$$C1 = C2 = C, \qquad R1 = R2 = R, \qquad R4 = \frac{R3}{2}$$

$$f_0 = \frac{1}{2\pi \mathrm{RC}}$$

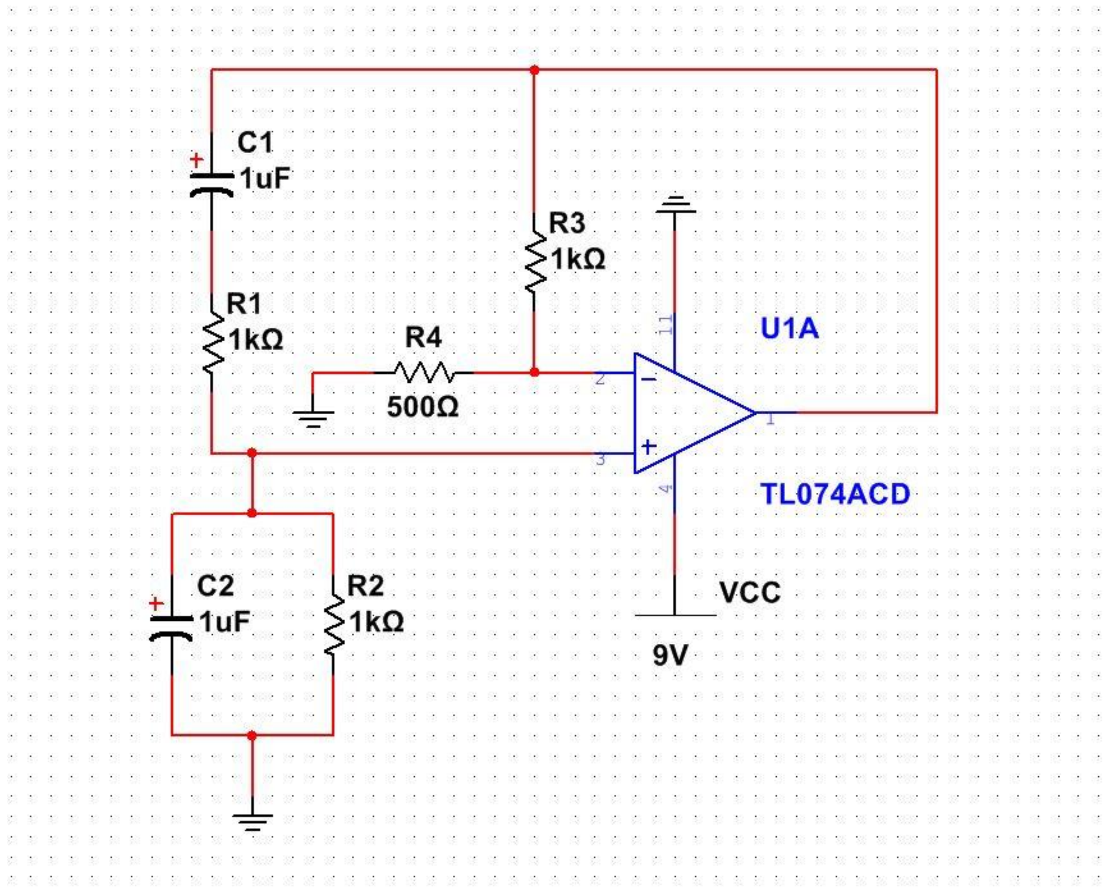


Figure 4.2.2.1.1: Wien-Bridge Oscillator. Image owned by the Hybrid Synthesizer team.

The above design is efficient in that it only needs one op-amp to output a very stable sine wave. However, as the diagram indicates, the oscillating frequency is tied to two separate resistors, which means the designers would have to utilize a dual element potentiometer, which would increase the cost for individual parts.

The figure below (Figure 4.2.2.1.2) depicts a quadrature oscillator, which is governed by the following mathematical characteristics:

$$C1 = C2 = C3 = C, \qquad R2 = R2 = R$$

$$f_0 = \frac{1}{2\pi RC}$$

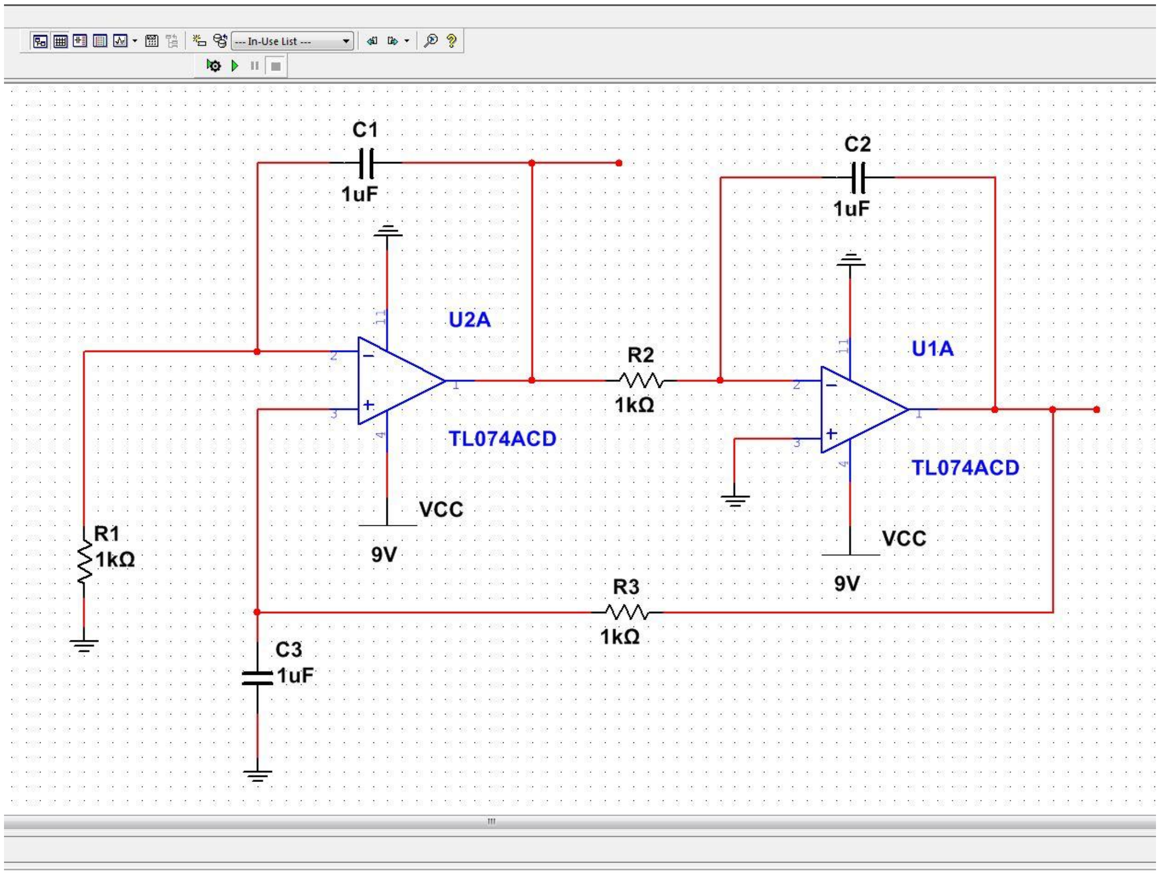


Figure 4.2.2.1.2: Quadrature Oscillator. Image owned by the Hybrid Synthesizer team.

The above design gives both sine and cosine wave outputs. However, the problem of having two separate resistors govern the cutoff frequency persists, so this configuration could only be justified if a use could be found for the cosine wave, as otherwise there is no reason to include a second op-amp.

4.2.2.2 Square Wave Generator

Square waves lend a hollow, retro video-game quality to an analog synthesizer's arsenal, and can be used for modern sounding bass lines as well as higher electronic melodies. They can also be used to emulate various woodwind instruments, such as clarinets or flutes. They're a staple for most modern dance music, which means its inclusion would see a significant amount of use by most artists. Below we can see a simple square wave generator.

In the figure below (Figure 4.2.2.2.1), a Transistor-based square wave generator is depicted, which functions based on the following mathematical characteristics:

$$R1 = R2 = R, \qquad C1 = C2 = C$$

$$f_0 = \frac{0.7}{RC}$$

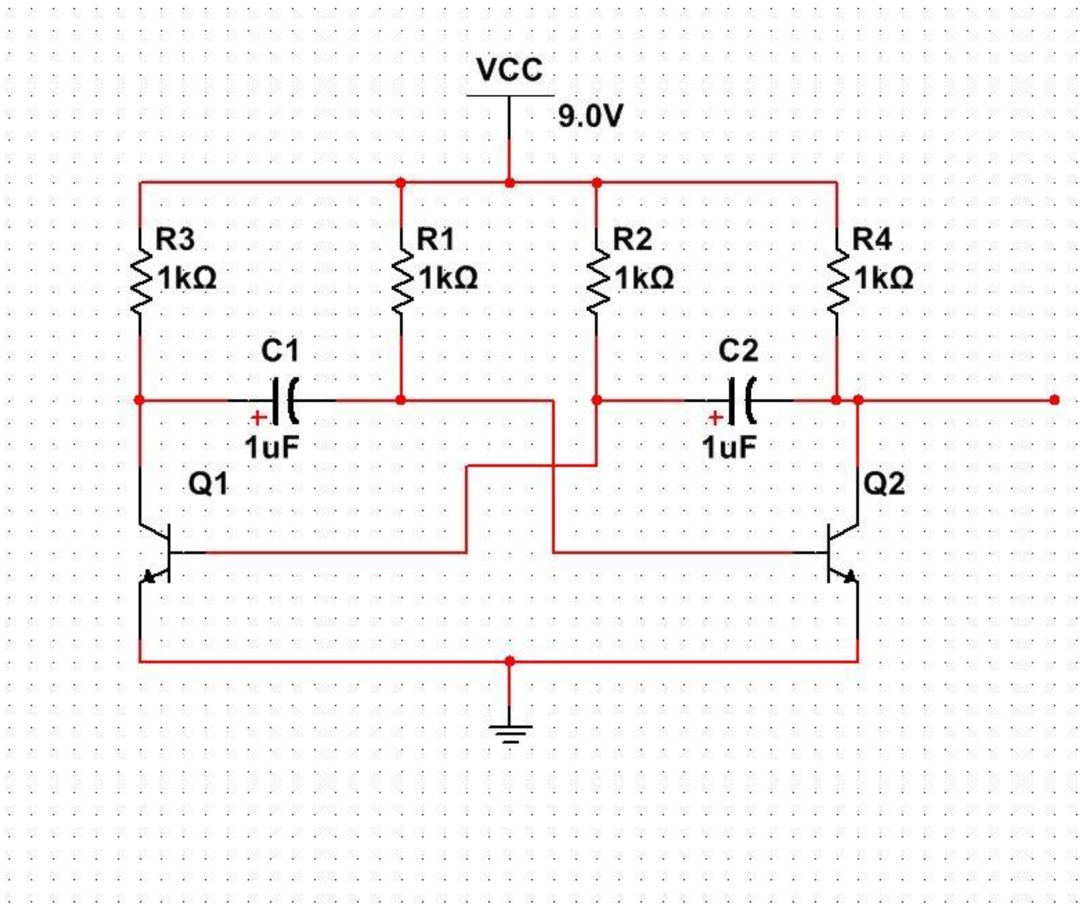


Figure 4.2.2.2.1: Transistor Square Wave Generator. Image owned by the Hybrid Synthesizer team.

The above design is incredibly simple. However, a dual element potentiometer would be needed to be able to modulate frequency. Additionally, op-amps will provide a significantly higher sound quality for just as much space consumption. This signal quality is fine for circuit testing, but its reliability for audio is questionable. Since the sine wave generator will need an op-amp, and op-amps can be packaged together easily to save space, using an op-amp to generate a square wave may be more ideal.

In the figure below (Figure 4.2.2.2.2), a Schmitt Trigger square wave generator is depicted, which is governed by the following mathematical characteristics:

$$R1 \approx \frac{R_f}{3}, \qquad 2R1 < R2 < 10R1$$

$$f_0 = \frac{1}{2R_f C \ln\left(\frac{2R1}{R2} + 1\right)}$$

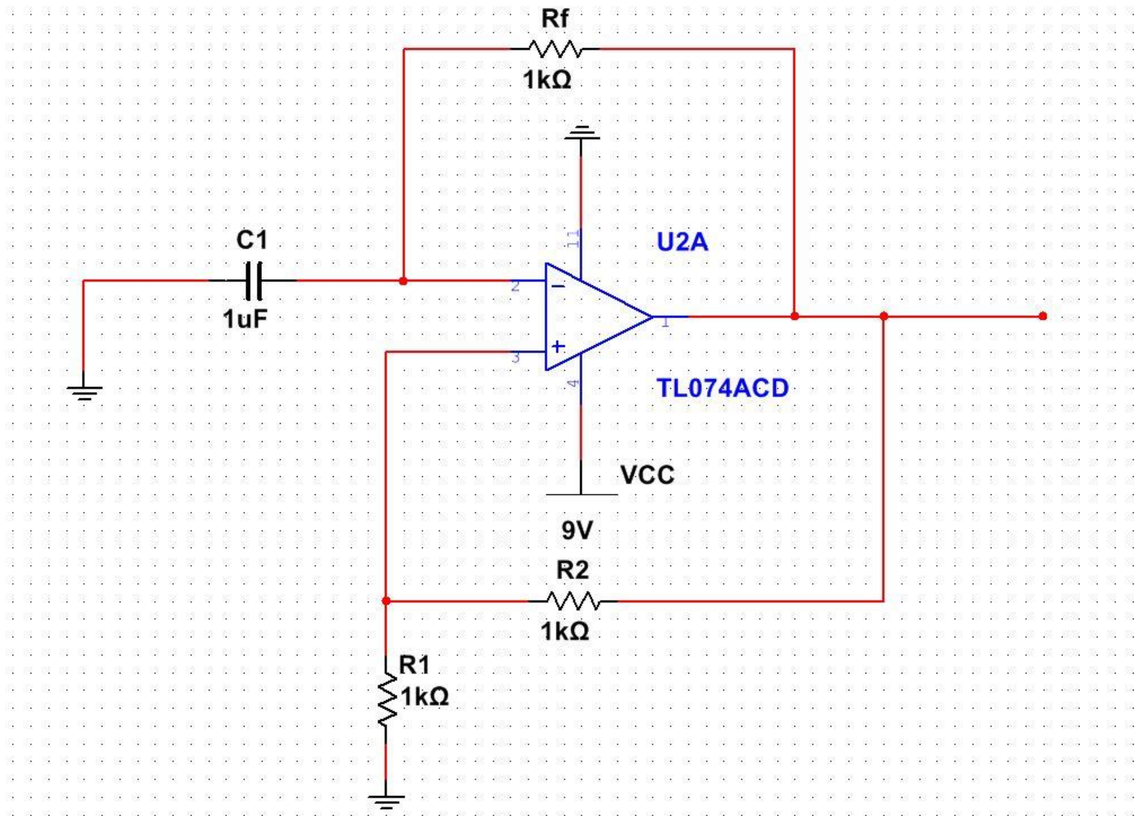


Figure 4.2.2.2.2: Schmitt Trigger Square Wave Generator. Image owned by the Hybrid Synthesizer team.

The above design is simple, which makes it a strong contender from a parts, heat, and breadboard size perspective. Additionally, with the likely use of several op-amps, this generator would be a part of a larger count package (2 or 4), further saving on space on the breadboard. As a result, the size comparison between this circuit and the prior circuit is negligible.

In the figure below (Figure 4.2.2.2.3), a different take on the square wave generator is displayed, using a 555 timer as an astable multivibrator, which is governed mathematically like so:

$$f_0 = \frac{1.44}{C(R1 + 2R2)}$$

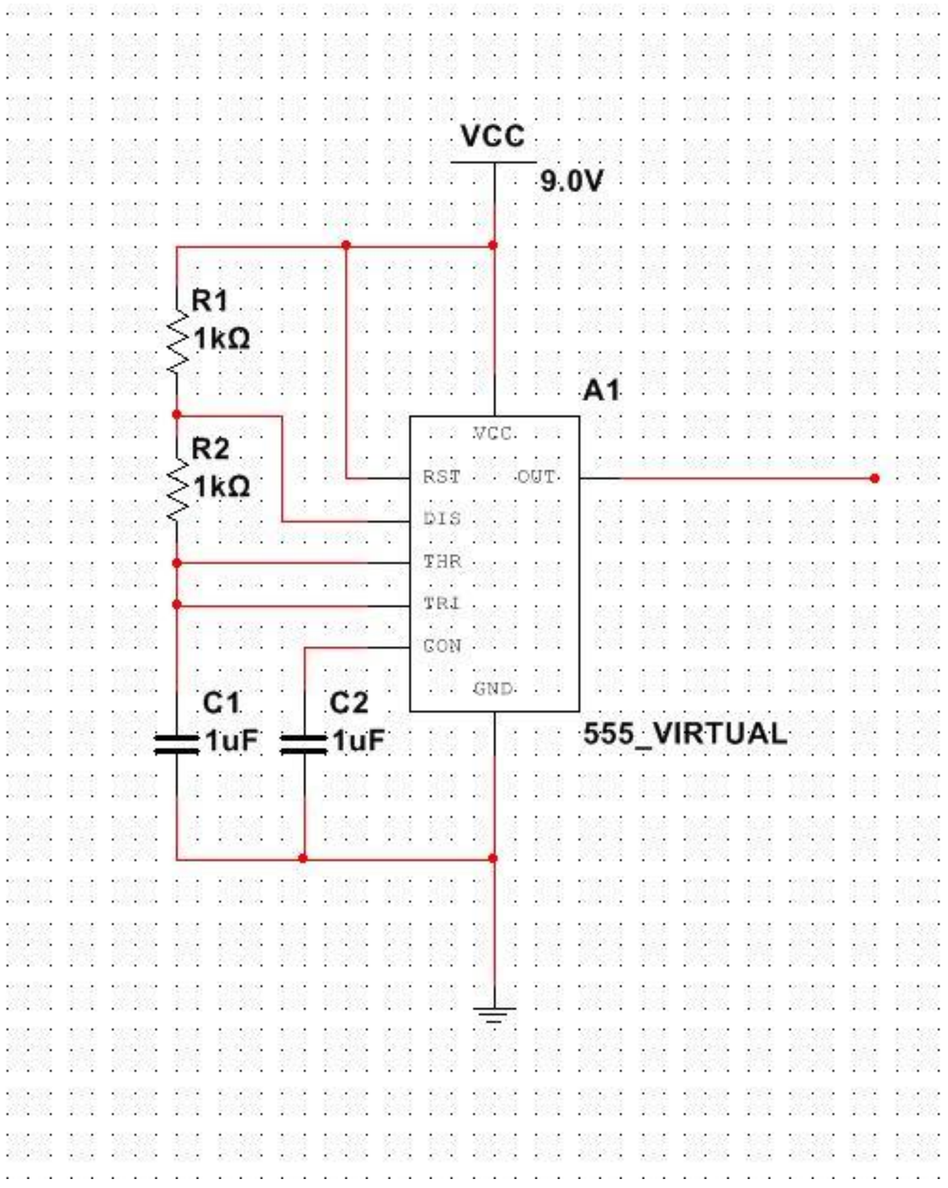


Figure 4.2.2.2.3: 555 Timer Square Wave Generator. Image owned by the Hybrid Synthesizer team.

By installing a potentiometer between the trigger and discharge pins, the duty cycle of the square wave output can be adjusted to create a rectangular pulse wave of varying widths. This brings a useful degree of flexibility if it were used to fuel a sawtooth/triangle wave generator, as the sawtooth could then have a positive or negative slope, or become a triangle wave depending on the user's adjustments. However, this circuit also makes use of a 555 timer instead of op-amps like the other designs would be using. This may not make a significant difference when comparing the outputs, however consistent signal fidelity is an important consideration for the design, as the musician should expect the same signal quality from all waveform generators.

4.2.2.3 Triangle Wave Generator

The triangle wave contains the same odd harmonics as a square wave, so as a result it has a similar aural character. Since these harmonics fall off exponentially rather than linearly, the triangle wave has a more muted quality to it when compared with its square wave cousin. This makes it a great waveform for filling out sound, as it has enough harmonic character to add meat without getting in the way of the timbre the other more complex waveforms are adding to the sound.

The design below (Figure 4.2.2.3.1) depicts a triangle wave generator governed by the following mathematical characteristics:

$$f_0 = \frac{1}{2\pi R1C1}$$

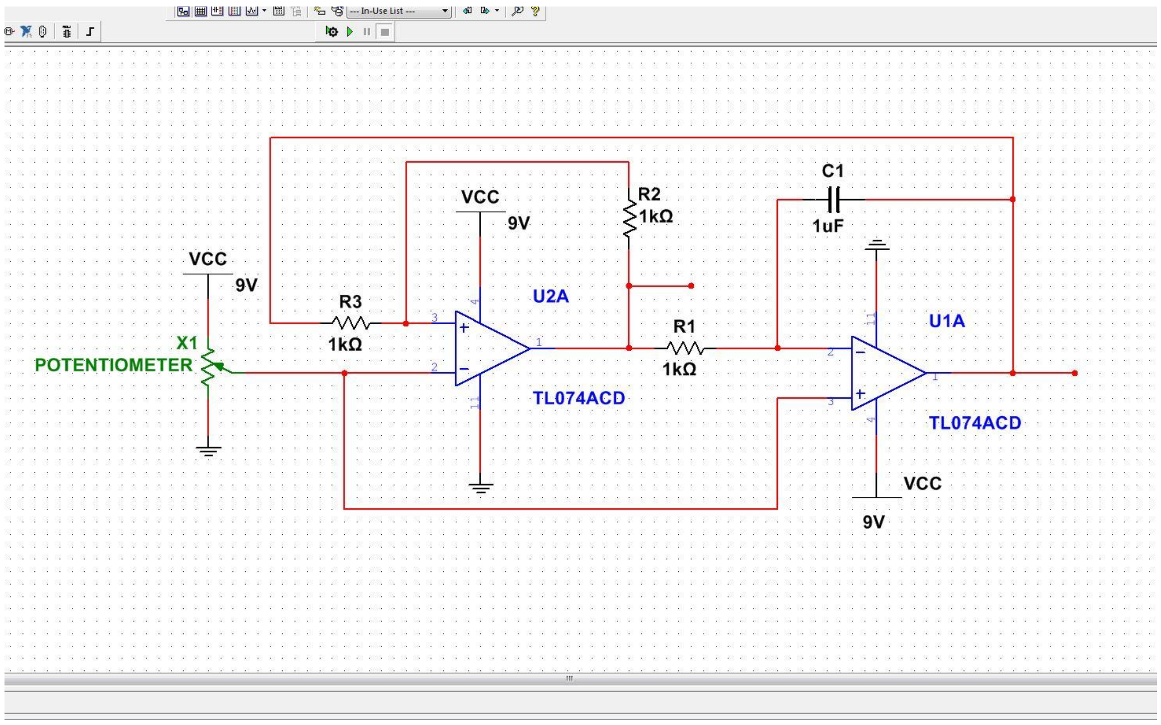


Figure 4.2.2.3.1: Triangle Wave Generator. Image owned by the Hybrid Synthesizer team.

Not only will it produce a stable triangle waveform, it also supplies a square wave from the same circuit, as a square wave is used to create the triangle waveform. This is an efficient use of op-amps from a packaging standpoint, and it solves the dual element potentiometer problem seen earlier. If the two waveforms can be pulled into the mixer without any adverse effects making their way back into the oscillators, this would be an ideal solution.

4.2.2.4 Sawtooth Wave Generator

Sawtooth waves have the most character out of the waveforms that have been considered so far. Sawtooth waves are brassy and buzzy, and when combined with a filter offer extreme flexibility for both bass lines and supporting harmonies/countermelodies. As their bright sound would imply, sawtooth waveforms can also be used to emulate the presence of brass instruments. Below is an example of a sawtooth wave generator (Figure 4.2.2.4.1).

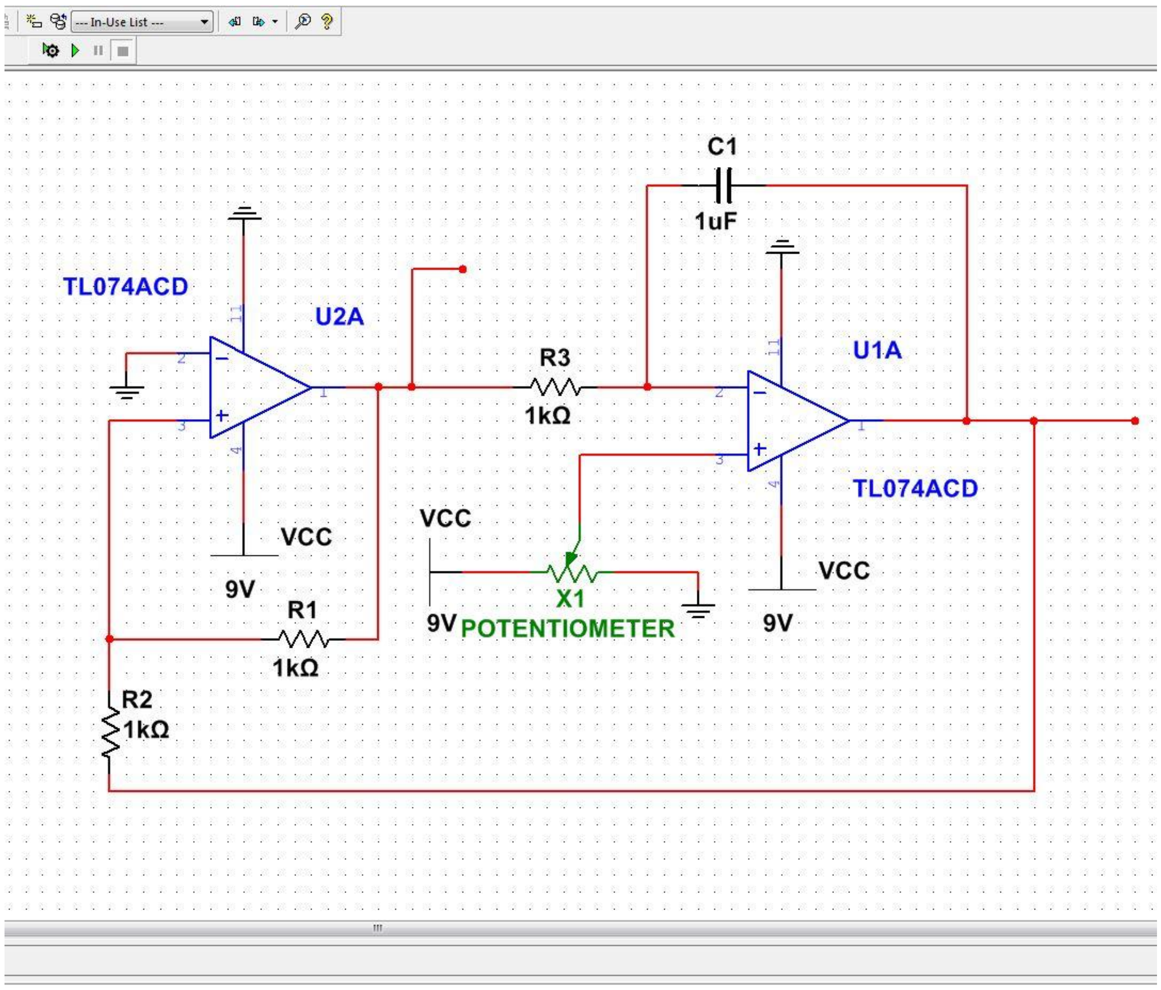


Figure 4.2.2.4.1: Sawtooth Wave Generator. Image owned by the Hybrid Synthesizer team.

The above circuit is a modified variant of the triangle wave generator considered earlier, with the following mathematical characteristics:

$$f_0 = \frac{R1}{4R2R3C1}$$

This particular design features a pulse width control for the system. This is a very useful feature, as it greatly increases the flexibility of the sawtooth wave as well as adds the potential to sample a rectangular pulse wave into the mixer, which would add much more variance and depth to the musician's arsenal. Now the sawtooth can have a positive or negative slope and the musician gains the uniqueness of a pulse waveform generator.

4.2.3 Voltage Controlled Oscillators

Until this point, only traditional RC oscillator circuits have been discussed. These circuits rely on a transfer function governed by their RC components, which are only made dynamic by the substitution of a potentiometer for key resistors in the configuration of the circuit. Because this circuit is intended to receive key-press information from an android application routed through some form of microcontroller, it is important to consider that a design where the frequency varies based on voltage control rather than resistance variance may be a better source for analog waveform generation. As such, two forms of voltage controlled oscillators will be discussed.

The figure below (Figure 4.2.3.1) depicts a Hartley Oscillator, which is governed by the following mathematical characteristics:

$$f_0 = \frac{1}{2\pi\sqrt{LC}}$$

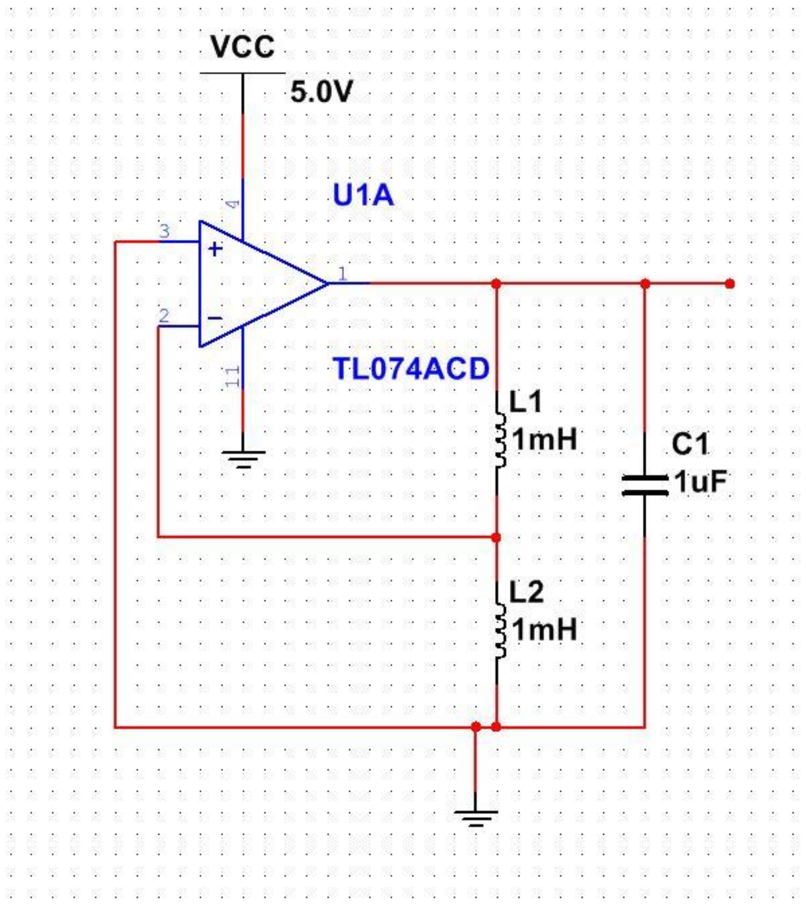


Figure 4.2.3.1: Hartley Oscillator. Image owned by the Hybrid Synthesizer team.

Hartley Oscillators can also be implemented utilizing a transistor and a resistor network for simplicity. The Hartley Oscillator is controlled by a tank circuit consisting of two inductor coils placed in parallel with a capacitor. This tank provides a control voltage to the amplifier, and the frequency of oscillation is directly related to the resonant frequency of the tank circuit.

While this circuit is a reliable voltage controlled oscillator, it relies on a large tank for its control voltage supply. The need for two inductors makes this a comparatively enormous circuit to implement, and we are left with a similar problem to the prior topologies in that the resonance of the tank controls the resonating frequency of the circuit, thus leaving the frequency to be controlled by a variable capacitor, which is a significantly more monetarily expensive and design limiting part to implement.

The figure below (Figure 4.2.3.2) depicts a Colpitts Oscillator, which is governed by the following mathematical characteristics:

$$f_0 = \frac{1}{2\pi\sqrt{L\left(\frac{C1C2}{C1+C2}\right)}}$$

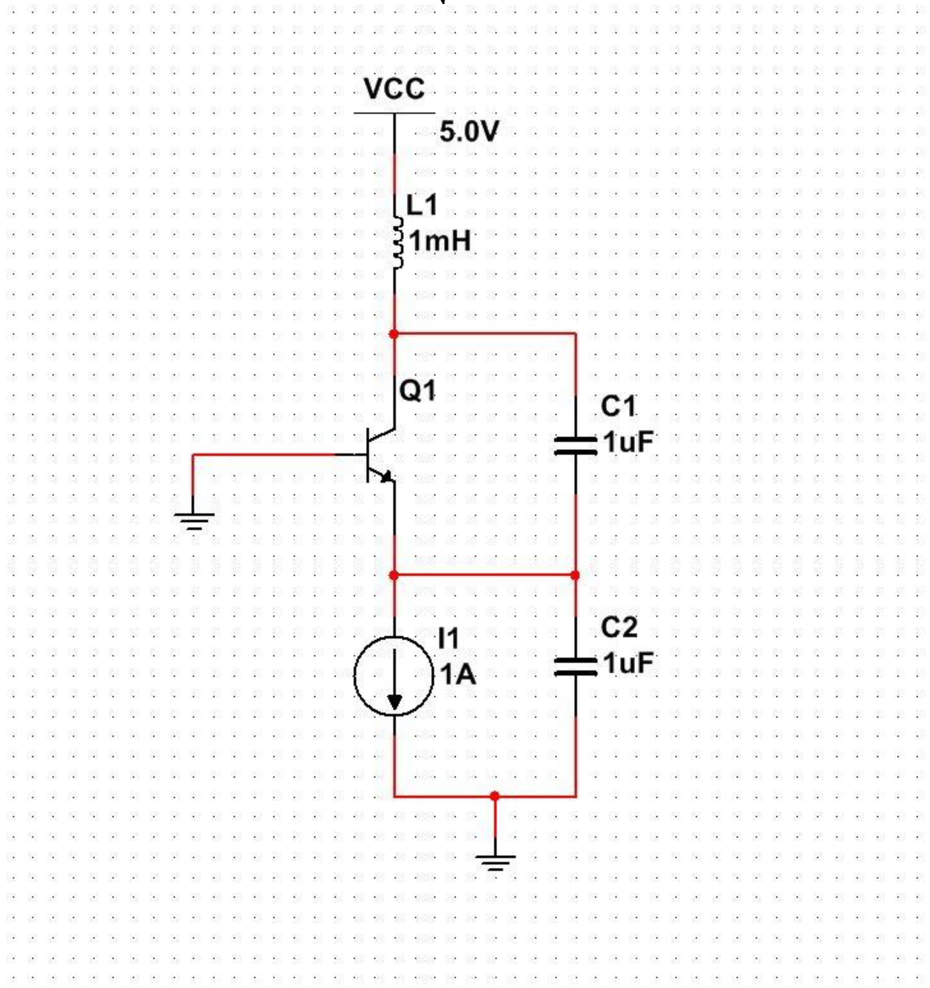


Figure 4.2.3.2: Colpitts Oscillator. Image owned by the Hybrid Synthesizer team.

The Colpitts Oscillator is effectively the electrical duel of the Hartley Oscillator, in that is similarly relies on a control feedback signal from a tank consisting of two capacitors in parallel with an inductor. While it is a smaller circuit than the Hartley Oscillator, it still relies on an inductor, which makes its size implementation cost huge in comparison to other circuits that have been considered. Additionally, the Colpitts oscillator functions most ideally in the RF range, and it too is plagued by the same issue of needing a variable inductor to alter the resonating frequency of the circuit.

However, considering these voltage sources being independent of the power supply brings the design team to an important consideration, as discussed with the following image (Figure 4.2.3.3).

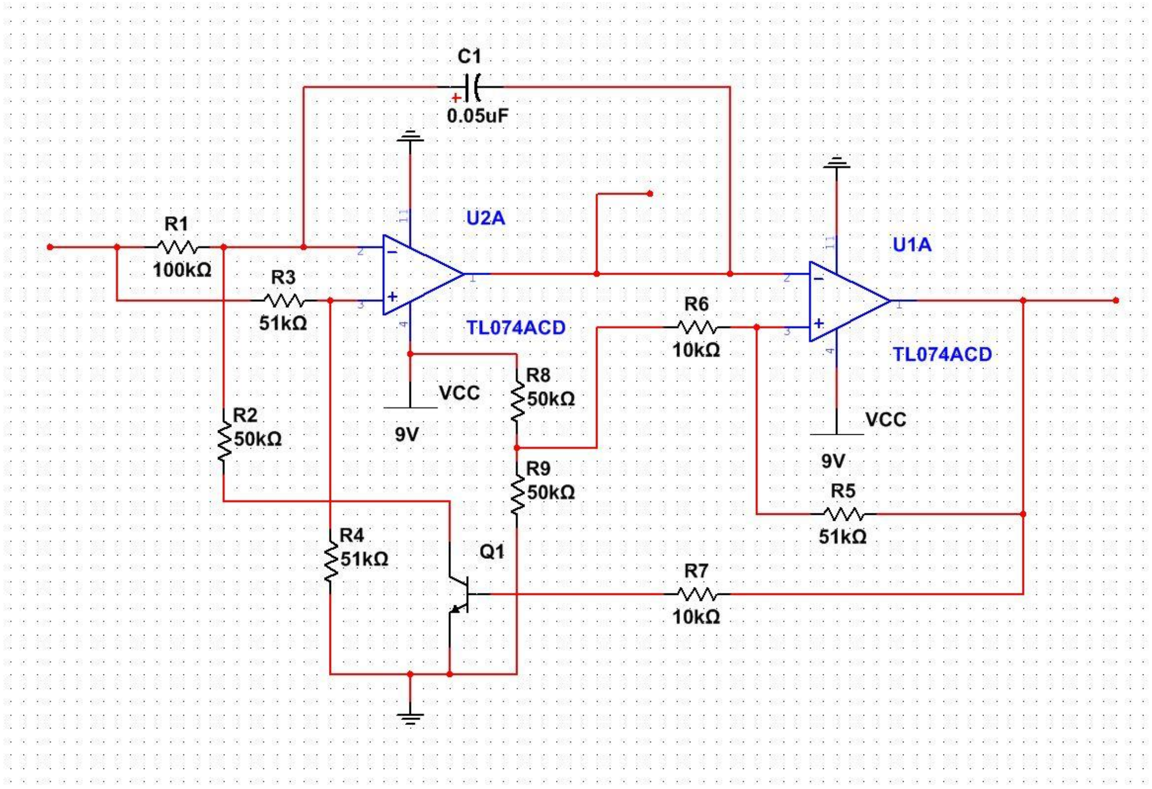


Figure 4.2.3.3: Voltage Controlled Oscillator. Image owned by the Hybrid Synthesizer team.

The above figure depicts a voltage controlled variant of a square and triangle wave generator. Looking back at the prior designs, it is clear that they are all effectively voltage controlled oscillators. The only difference is that prior designs rely on the use of a potentiometer to vary a control voltage input between a supply voltage and zero, which is then routed to one of the terminals of the op-amp. If that signal were to instead be regulated and sent via a microcontroller, then the topologies have to change very little to accommodate this form of voltage control. Instead, the design team now only has to tweak the values of parts around the control voltages they can send from their microcontroller. Provided a stable range of output voltages can be implemented from the microcontroller, this may be the ideal solution to controlling the circuits externally via an android application.

4.2.4 Gain Stages

The design rationale surrounding any and all gain stages will revolve primarily around power consumption and usage. The designers wish for this system to be as portable as possible, and having to drag a large power supply around with the unit (or worse, mount a large, heavy power supply inside the unit). As such, it will likely be assumed the synthesizer is intended to be routed either to headphones to monitor sound, a preamp for recording or live performance, or a powered speaker for monitoring or live performance. Assuming the above is the case, the most important gain stage in the signal flow would be at the mixing stage, as it is at this stage where the artist will be able to adjust the levels of the various oscillators to craft the exact sound desired.

The figure below (Figure 4.2.4.1) depicts a typical audio mixer with the following mathematical characteristics:

$$-Vout = \frac{R_f}{R_{IN}}(V1 + V2 + V3 + \cdots + VN)$$

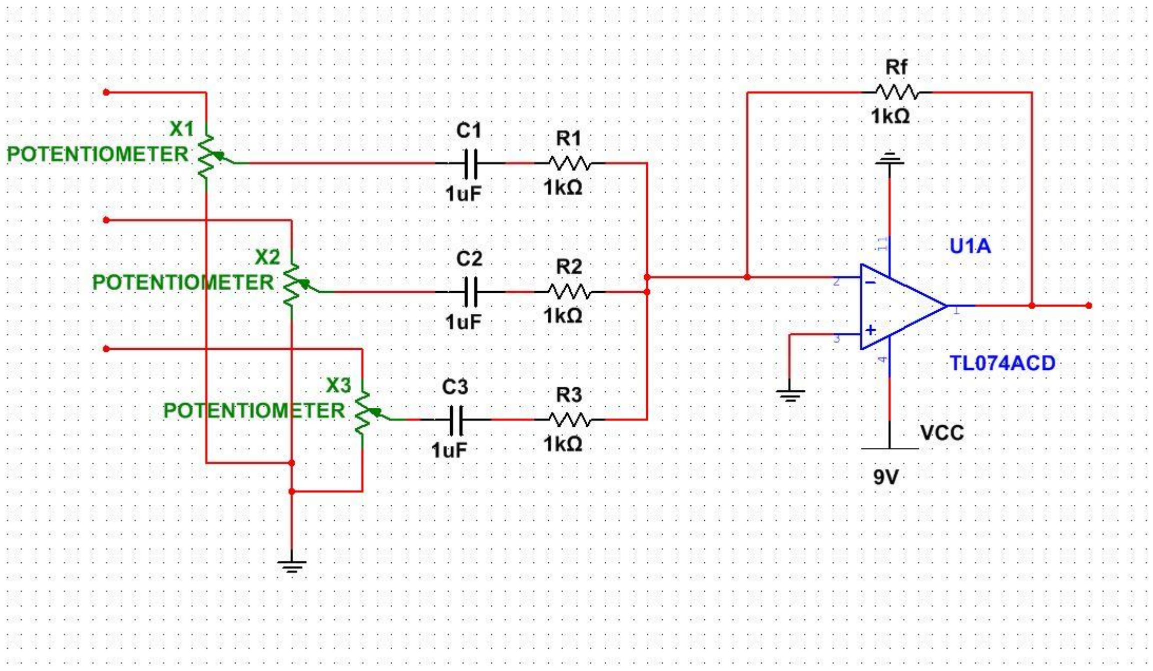


Figure 4.2.4.1: Audio Mixer Module. Image owned by the Hybrid Synthesizer team.

Potentiometers are tied to the inputs of a simple summing amplifier. Capacitors are used to block any DC offset and reduce noise. The number of these summing amplifiers will have to be considered. With every additional channel connected to the adder, there will be some loss to the overall circuit. It is best to keep four inputs to an adder just to ensure signal fidelity. If the design implements more than four waveforms, then two audio mixer stages like the one depicted above may be needed. If so, they would be routed to an additional summing gain stage that operated independent of user input.

Additionally, there should be a master volume control gain stage at some point after the mixing stage, so that if the artist wishes to scale the mixed waveform, he or she does not have to try and remix the signal at a new output volume through the mixing console. Fortunately, this form of amplification is the simplest design concern to be considered thusfar.

Below, a simple inverting op-amp is depicted (Figure 4.2.4.2), which adheres to the following mathematical characteristics:

$$Av = -\frac{R2}{R1}$$

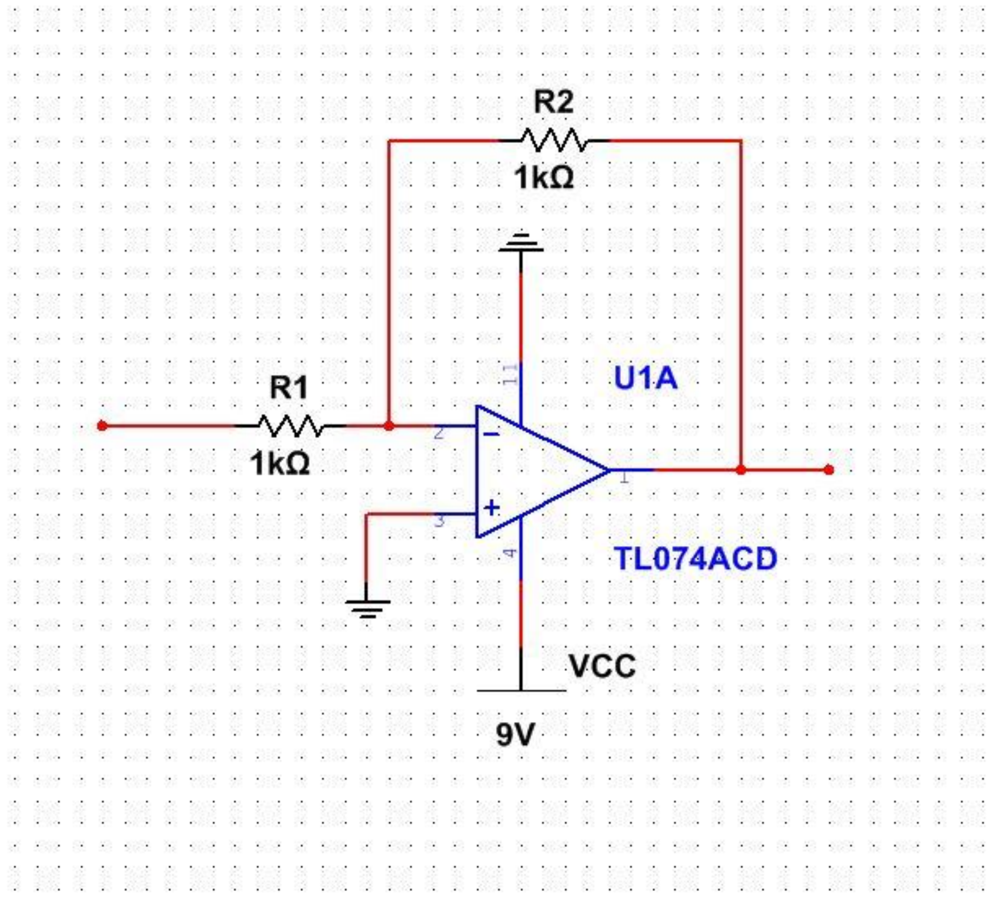


Figure 4.2.4.2: Inverting Op-Amp Gain Stage. Image owned by the Hybrid Synthesizer team.

This is a simple circuit to implement, and a potentiometer can easily be used in the place of the resistor to create variable gain. Since the oscillator will be producing a mono (single channel) sound, the inversion of phase is irrelevant to the final output waveform, and may not even need to be corrected by another inverting amplifier with unity gain. Depending on the amount of gain needed, however, this design may be unreliable or introduce some distortion. As such, the following design can also be considered (Figure 4.2.4.3).

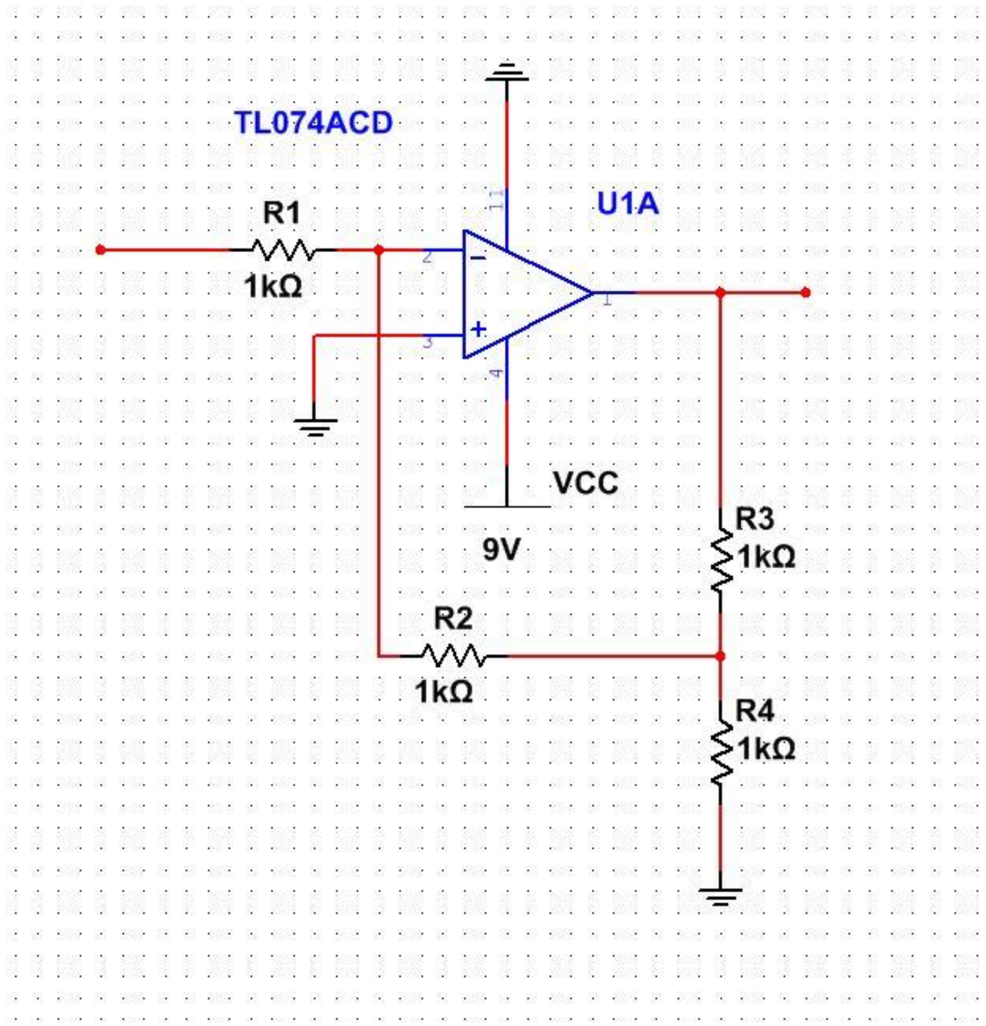


Figure 4.2.4.3: High Input Impedance Op-Amp. Image owned by the Hybrid Synthesizer team.

The above figure demonstrates a high-input impedance op-amp with the following mathematical characteristics:

$$Av = -\frac{R2(R3 + R4)}{R1R4}$$

If a very high maximum gain value is needed, the above circuit could be implemented as a gain stage, provided the bandwidth of the op-amp in question can handle the signal being passed to it.

4.2.5 Envelope Generator

In the case of analog synthesizers, an envelope generator is a circuit module that creates an envelope that will modulate the amplitude of the signal it's applied to. It's often referred to by an acronym composed of the stages the envelope goes through. A basic envelope consists of at least two states, Attack and Release (referred to as an AR envelope). Envelopes can have additional states to create more complex sounds. The number of possible additional states is effectively infinite; however a four stage envelope has shown in practice to be the most comprehensively useful type of envelope. This envelops is referred to as the ADSR envelope, with its acronym referring to the four states: Attack, Decay, Sustain, and Release(Figure 4.2.5.1).

Attack refers to the time it takes for the note played to reach its highest amplitude value. Decay refers to the time it takes for the note to reduce in volume to its assigned Sustain value. Sustain refers to the amplitude level the note will converge to while being held. Once the key is released or the source of the sound is removed, the Release portion of the envelope engages, and this refers to the time it takes for the note to decay from its sustained value to zero.

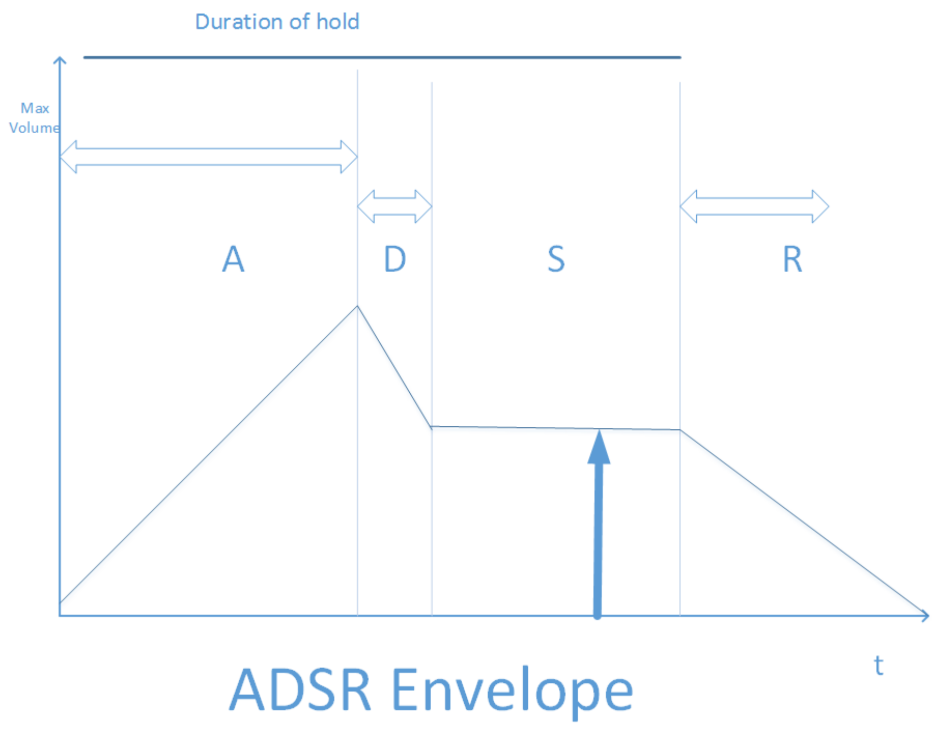


Figure 4.2.5.1: ADSR Envelope. Image owned by the Hybrid Synthesizer team.

There are other additions that can be made to envelopes like the above. For example, some synthesizers make use of an ADSHR envelope. The H in the acronym refers to a Hold parameter, which would allow the user to set each note to sustain for a fixed length of time instead of being tied to the key being held to continue. Another example would be a DADSR envelope, with the first D referring to a Delay parameter. This Delay parameter could be set to cause each note to play a fixed length of time after a key is pressed or input is given.

Another interesting technique to consider is the inversion of the ADSR envelope, which, as expected, causes the reverse effects from what has been described previously for each stage. Larger envelopes (6+ stages) typically incorporate a few inverted ADSR stages, which can be used to give additional pulse or reverb effects to the sound generated.
The envelope generator will most likely make use of slide potentiometers, as being able to see the relative scale of the different time delay values of the envelope would be very useful to the musician trying to craft the envelope. Based on this design choice, a 6+ stage envelope would chew up a large amount of surface area, in addition to requiring a significantly larger amount of supporting circuitry.

Additionally, including additional parameters such as Hold or Delay are usually features that are exclusive to a particular model of synthesizer. Our synthesizer is not part of a line of varied products, and thus having one of these implementations moves our synthesizer away from its intended general-purpose workhorse design.

Below is a simple two-stage envelope (Figure 4.2.5.2), containing Attack and Release stages. The advantage to this design is that it is easy to implement and requires few parts. The disadvantage is that an AR envelope is not a comprehensive envelope generator. Instruments that use air to generate notes, such as trumpets or flutes, can be represented by an AR envelope, as their sounds tend to sustain at the initial attack value (barring any special musical accents played by the musician). Instruments with oscillating components, such as guitars or pianos, cannot be accurately represented by an envelope with this few stages, as their amplitudes decay significantly after the initial attack before converging to a sustain amplitude due to the nature of how the oscillating strings disperse their energy once plucked.

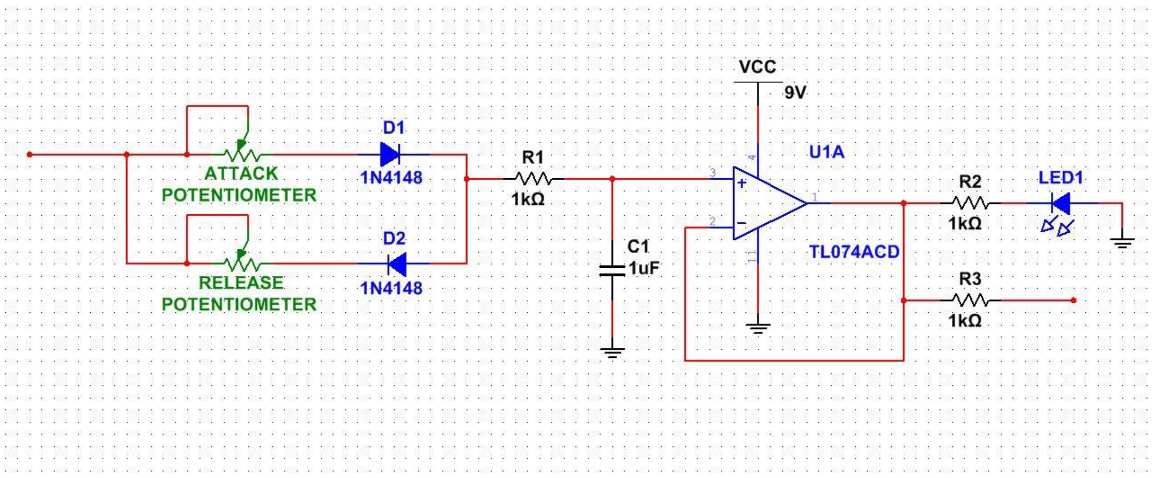


Figure 4.2.5.2: AR Envelope. Image owned by the Hybrid Synthesizer team.

While the analog side of our synthesizer has no intention of trying to match the sound of another specific instrument, the envelope is used to give the waveform additional character, and a more "plucked" sound may be desired by the musician. As such, an envelope like the one pictured below (Figure 4.2.5.3) may be more desirable, as having an attack and decay value allows for a greater degree of flexibility in constructing waveforms with different aural properties. However, the drawback here is that there is a significant increase in the number of parts. A 555 timer is implemented to create the envelope, and a BJT network buffers the gate input into an impulse needed to trigger the timer.

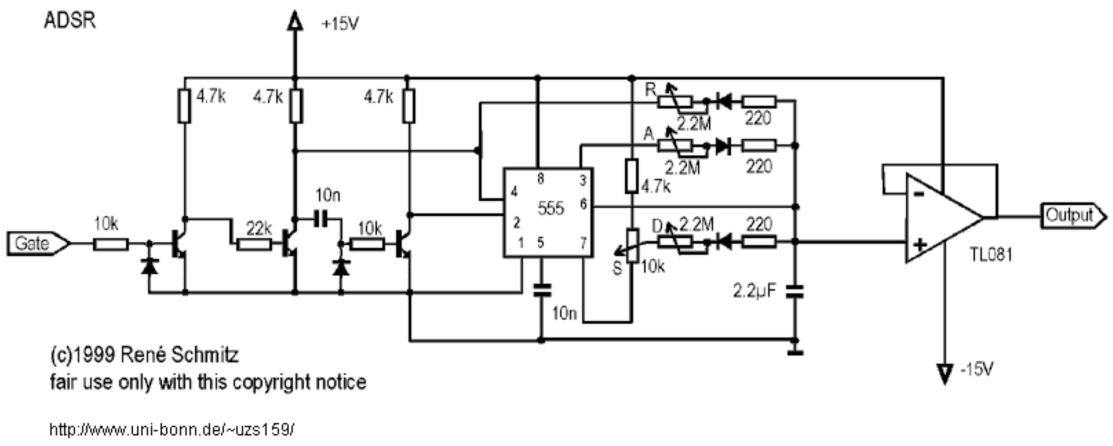


Figure 4.2.5.3: ADSR Envelope. Image owned by Rene Schmitz, reprinted under displayed permission.

The above circuitry is more complex than the AR envelope described previously; however it is not more complex by a significant margin, and the added flexibility it grants the musician is likely worth the inclusion of a few more parts.

4.2.6 Filters

Filters are some of the most commonly used effects in audio applications. From emulating audio effects such as phone calls or cleaning and refining a signal from noise, such as Equalization (EQ) filters, they see a variety of uses in a variety of applications. The filters implemented in live performance analog synthesizers tend to be simple ones, and for our design, we don't expect the musician to be doing significant EQ, as this is usually taken care of by a mixing engineer or sound board, depending on if the performance is recorded or live. As such, we'll want to keep the design of the filter simple, to be used for artistic effect rather than for intensive waveform sculpting.

The figures below (Figure 4.2.6.1, 4.2.6.2) show elementary passive low-pass and high-pass filters respectively, with the following mathematical characteristics:

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

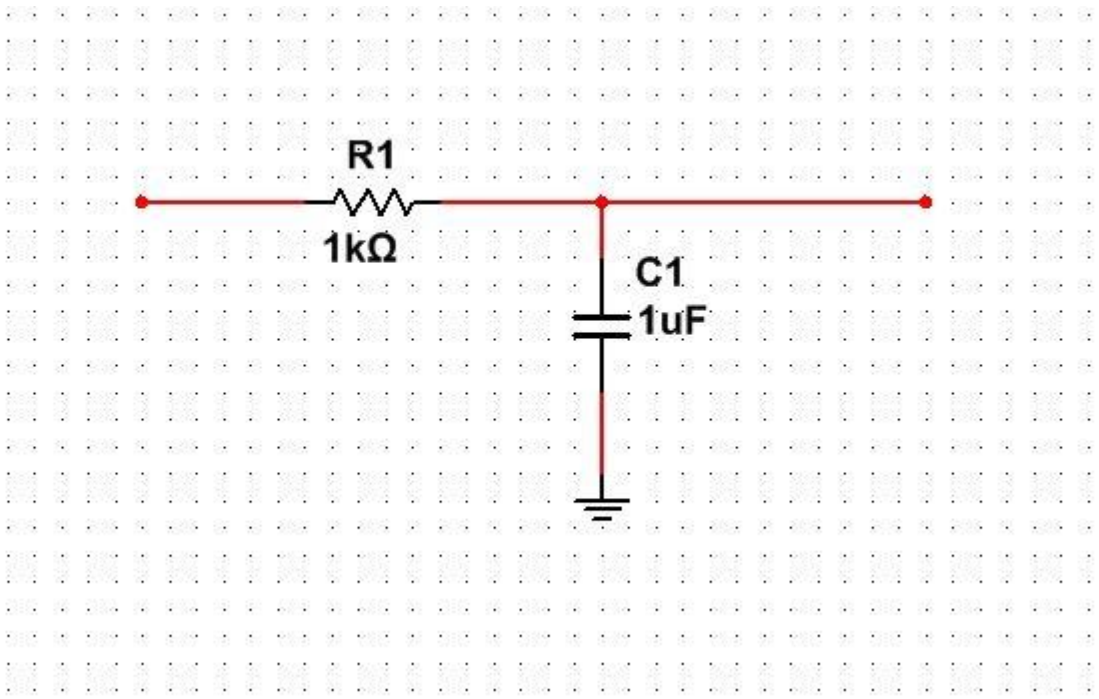


Figure 4.2.6.1: Passive Low-pass Filter. Image owned by the Hybrid Synthesizer team.

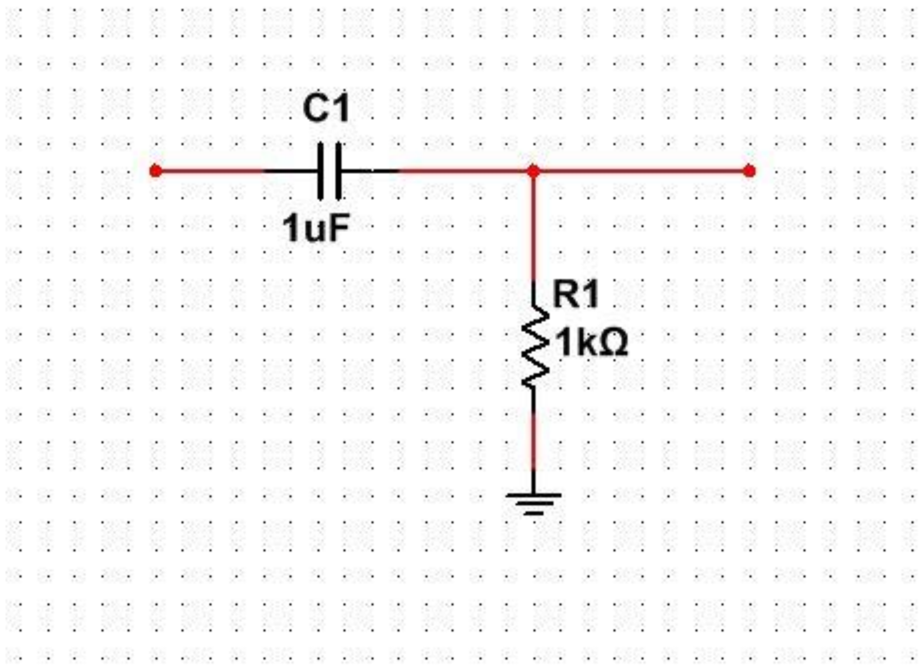


Figure 4.2.6.2: Passive High-pass Filter. Image owned by the Hybrid Synthesizer team.

These filters are exceedingly simple to implement and would take up a negligible amount of space on the PCB layout, aside from only requiring two parts apiece. Use of a potentiometer for the resistor would allow for a flexible cutoff frequency that could be swept at a variable rate by the musician over time to give his or her music a certain build or feel when performing. However, these filters are rather brutish for audio applications, and may be a little too simple, or may inject too much noise from the resistor. As such, it is worth considering active filters as well (Figures 4.2.6.3, 4.2.6.4).

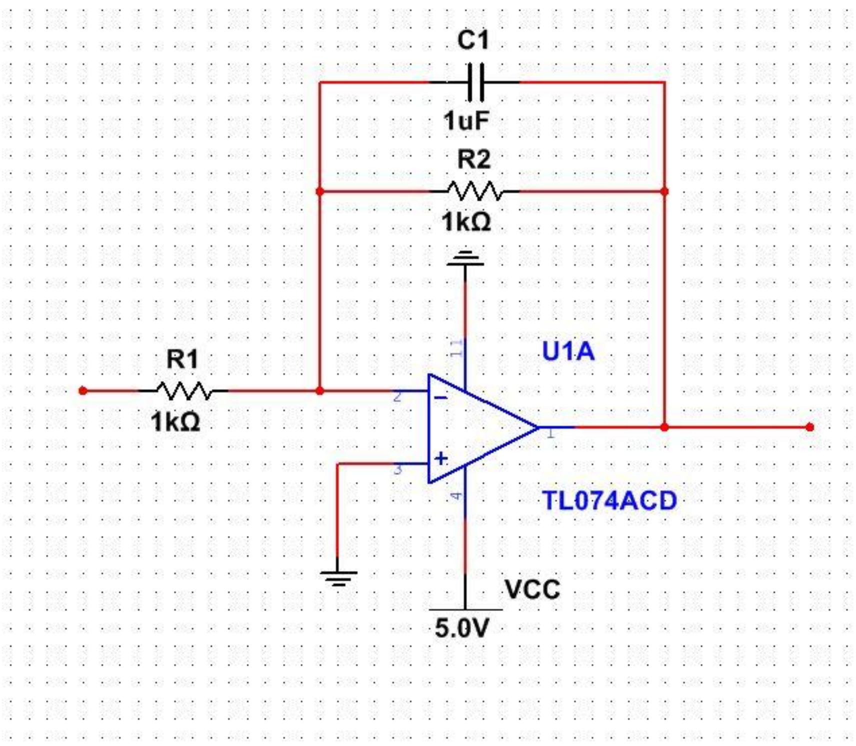


Figure 4.2.6.3: Active Low-pass Filter. Image owned by the Hybrid Synthesizer team.

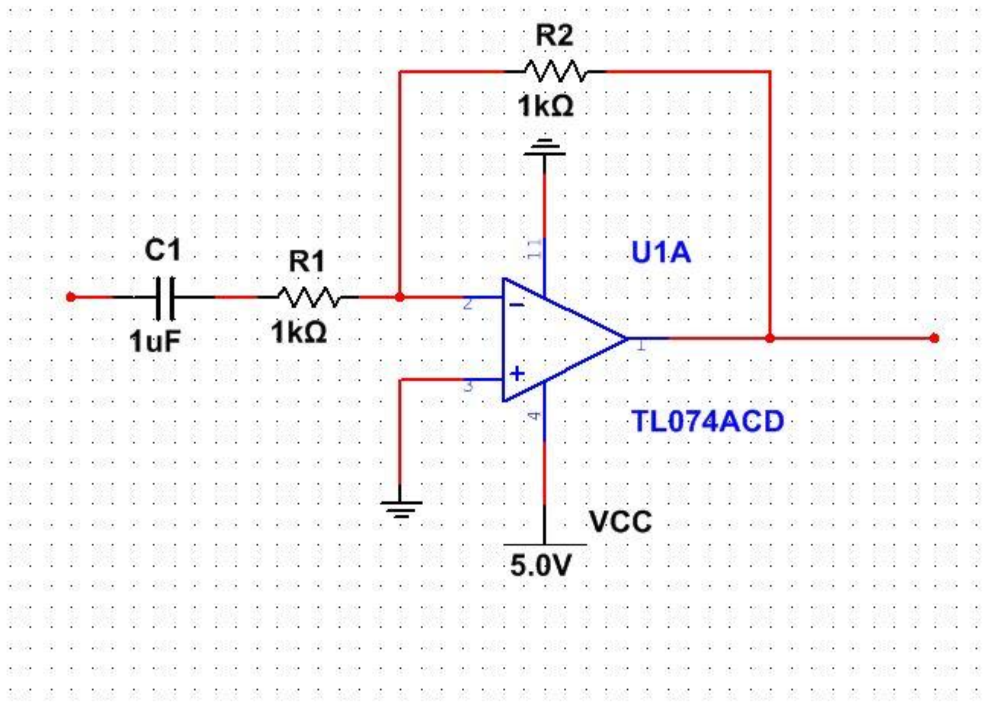


Figure 4.2.6.4: Active High-pass Filter. Image owned by the Hybrid Synthesizer team.

The above figures show an active low-pass and high-pass filter respectively, with the following mathematical characteristics:

$$f_c = \frac{1}{2\pi R_2 C}$$

for the low-pass, and

$$f_c = \frac{1}{2\pi R_1 C}$$

for the high-pass. The gain for both circuits is

$$Av = -\frac{R2}{R1}$$

The downside compared to the prior design is the significant increase in part size, amount, and cost, with the inclusion of the op-amp. However, the inclusion of the op-amp provides greater signal fidelity and will drop off at a clean -20 dB per decade, while still allowing the cutoff frequency to be adjusted via a potentiometer. Additionally, although the filter above is first-order, implementations using an op-amp make it easier to create multi-order filters should the design change, and the inclusion of the op-amp introduces another gain stage that may be of use depending where the module ends up in the pipeline.

The figure below (Figure 4.2.6.5) depicts a possible implementation of a band-pass filter. The problem with the above design is that it relies on the use of three inductors, making it an enormous circuit to physically implement. It would also require the use of variable capacitors and inductors, which are much more expensive parts to use.

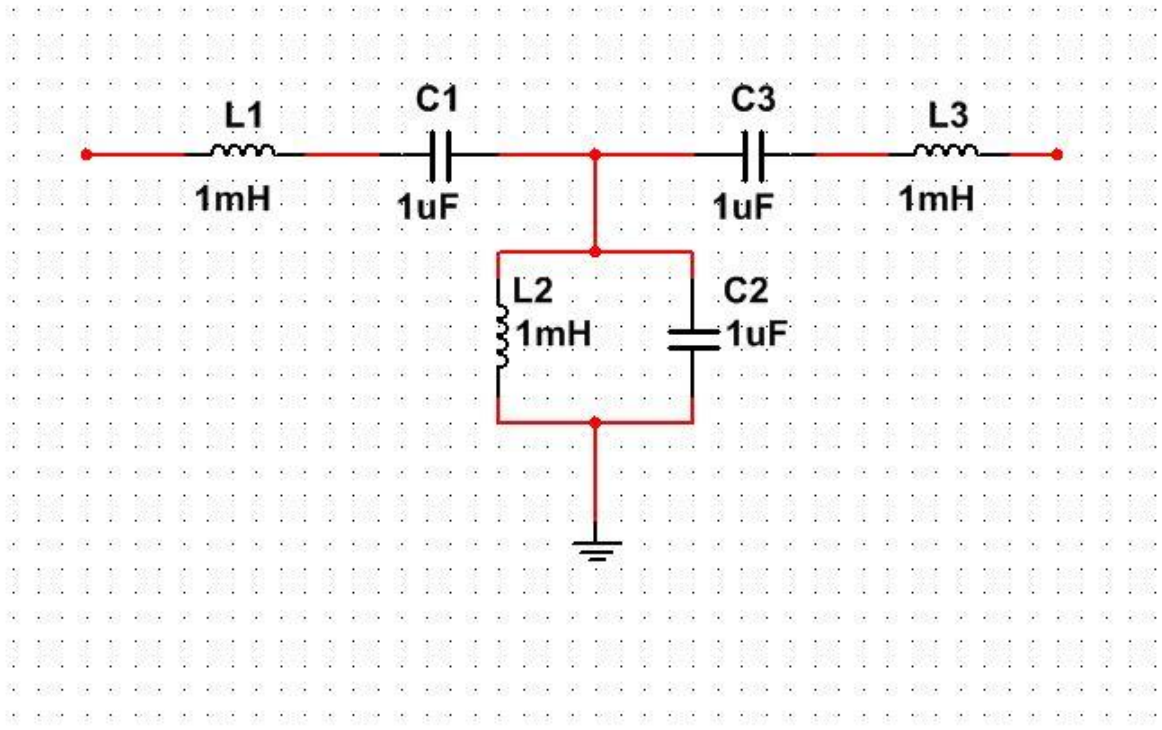


Figure 4.2.6.5: Band-pass Filter. Image owned by the Hybrid Synthesizer team.

The functionality of a band-pass filter may be useful to the artist. However, it is also possible to chain a high-pass and low-pass filter together in series to form a band-pass filter. If the prior passive or active filters are chosen by the design team, they could be arranged in series, and the two cutoff frequencies can still act independent of each other and be utilized as the artist sees fit. If the design team can implement a band-pass filter this way, then a significant amount of space could be saved within the unit, in addition to cutting down on additional parts.

4.2.7 Signal Out

Since an analog blended waveform will be output by the analog components of the synthesizer, and the microcontroller will output software instrument waveforms, either of these outputs should be able to be sent to the same output jack mounted somewhere on the synthesizer.

At the very least, a 1/4" audio jack should be used as a complete audio out port. A 1/8" audio jack, otherwise known as a headphone jack, would also be a useful port to include, so that the musician is free to monitor any recorded performances real time, or can choose to use the workstation for non-invasive personal use without the need for additional adapters.

Inclusion of a headphone jack may require an additional gain stage to ensure the output waveform does not overpower and blow out the headphones. If such a gain stage is necessary, it will likely take the form of one of the amplifiers discussed in Section 4.2.4, and may include a gain adjustment feature specific to that port to allow the musician to adjust monitoring volume to taste.

4.2.8 Mobile Power Source Options

Since one of the intents is to make this synthesizer portable, it is worth considering powering the analog components with batteries. If batteries were to be used, it is likely a 9V would be selected. Regardless of what battery is used, there will need to be some circuitry to house and convert the battery's supply to a proper voltage source.

The circuit below (Figure 4.2.8.1) makes use of the LM7805 5V voltage regulator. This would ensure the voltage source would stay at a consistent 5V, which allows for clean low-voltage operation that the team will be designing around. The regulation capacitors can be acquired from the datasheet for the part. An LED is included to indicate the system is indeed receiving power from the battery.

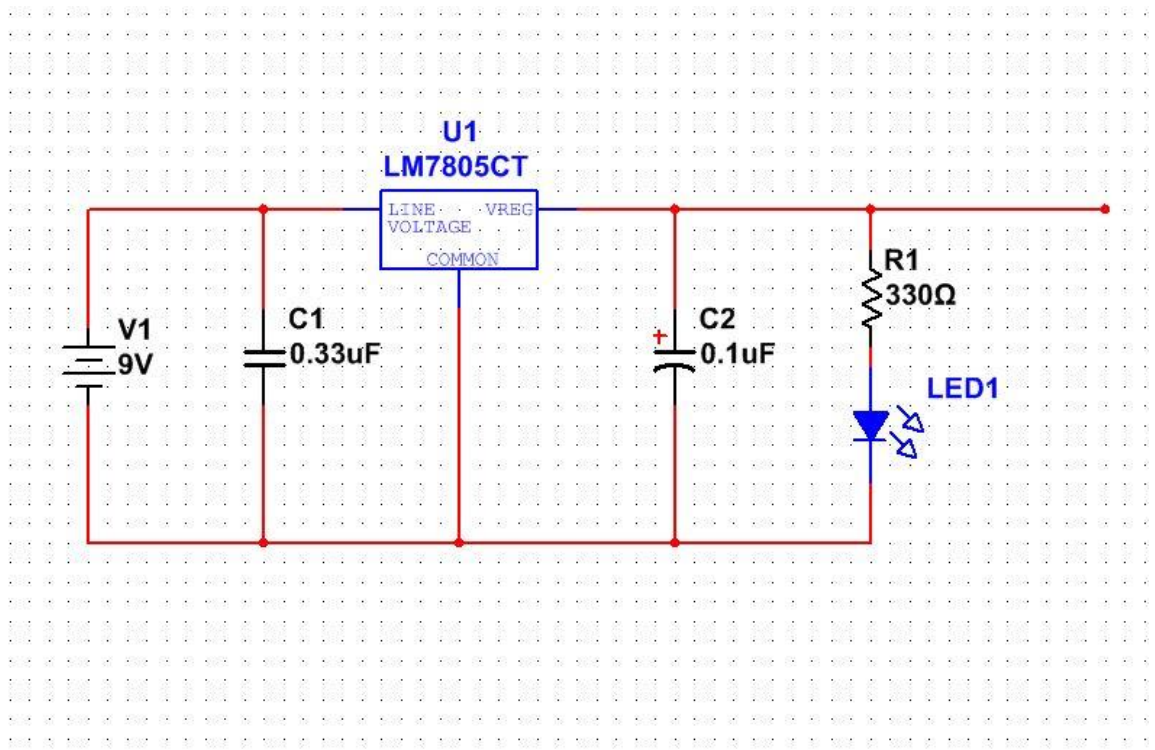


Figure 4.2.8.1: Battery Conversion Circuit. Image owned by the Hybrid Synthesizer team.

4.3 Mobile Application Research

The project is very dependent upon a software application that will be created using a mobile application. There are many good platforms on the market today that would have suited our needs, but seeing how the project has certain limitations, a major one being cost, the team had to weigh the pros and cons in determining which platform to use. In Figure 4.3.1, a distribution of the number of mobile users that are using a Mobile Device or tablet in the month of September 2014 is shown. From the graph, it can be determined that 43.5% of users are using the iOS operating system by Apple. Roughly 39.85% of users are using the Android operating system.

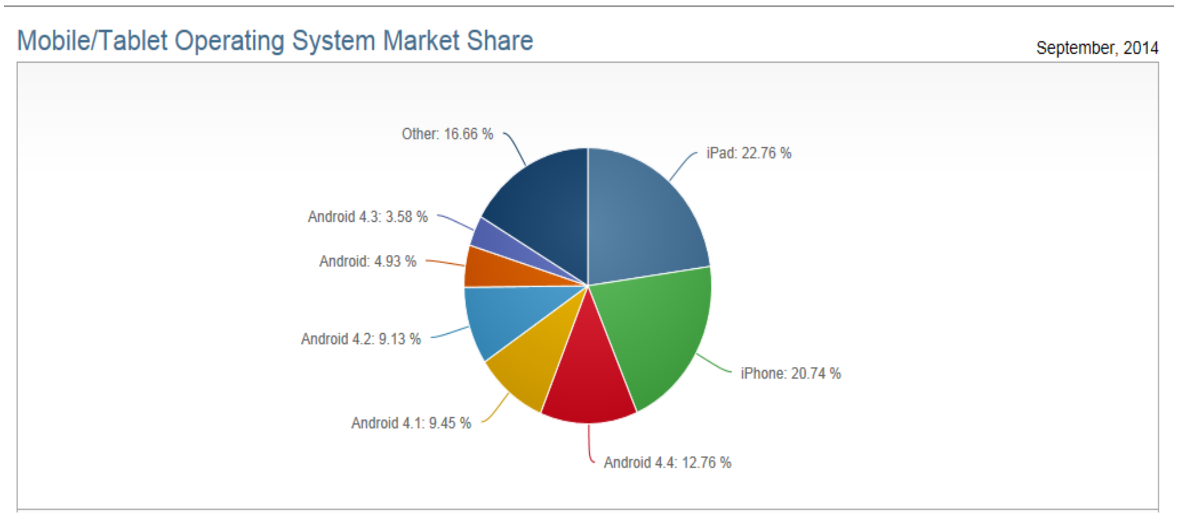


Figure 4.3.1: Market Share Distribution for Mobile, used with permission from netmarketshare.com

4.3.1 Apple iOS

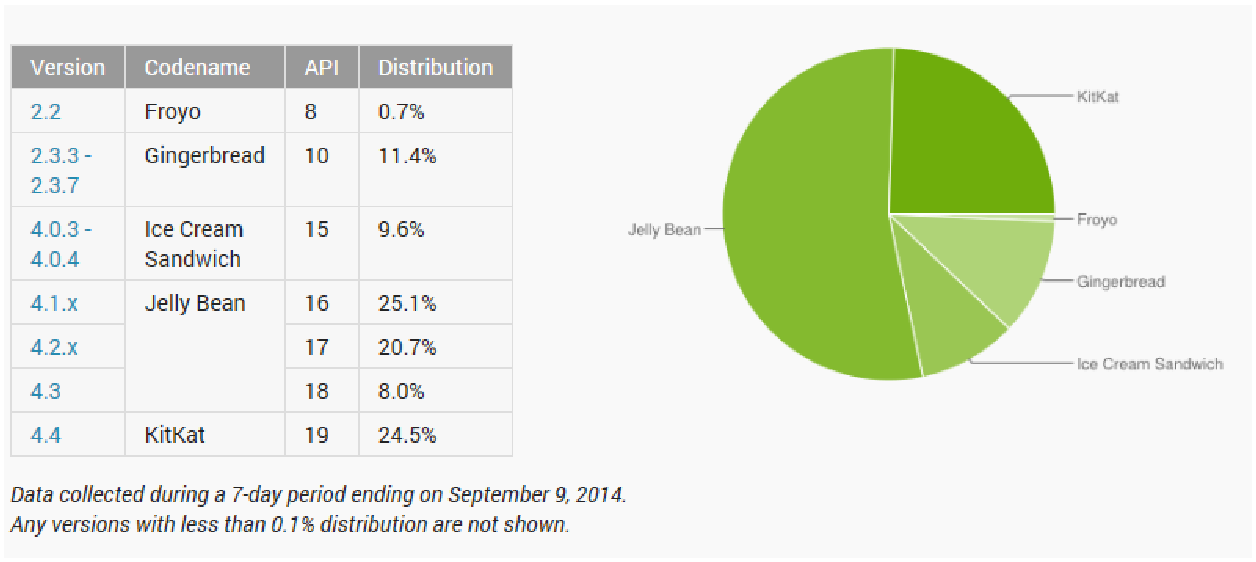
Looking into it, it would seem that developing on the iOS platform would reach the most users; however, there are a lot of costs associated with using the Apple mobile platform. In order to develop iOS you need to have a Mac computer that uses the Mountain Lion Operating System. Since none of the developers on the team own a Mac, this feat would eat into our budget tremendously. Also, these applications are written in Objective-C, and the newer versions will be created using a programming language called swift. Since no one on the team was familiar with either of these languages, it seemed that for the sake of knowledge and time that the design team decided to go with the Android operating system.

The Android OS seemed like the logical choice for a variety of reasons. First being that Android is Open Source, meaning that it is free for anyone to use and develop with.

Android, unlike Apple's operating system iOS, does not require any specified hardware to develop on. You can create an Android Application from any computer, whether it is running Windows, Mac, or Linux.

4.3.2 Android

The Android Integrated Development Environment (IDE), Android Software Development Kit (SDK), is also free and runs off of the Java Eclipse platform. The Android team at Google is also working on a new IDE which will be named Android Studio, but it is still in Beta, so the team elected to go with the standard SDK for the development environment. Choosing the version was also a challenge. The research that the team did was to look at the amount of users that were using each version to see where the demographic would be (Figure 4.3.2.1), while also keeping in mind which version our hardware for the project was running.



Version	Codename	API	Distribution
2.2	Froyo	8	0.7%
2.3.3 - 2.3.7	Gingerbread	10	11.4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	9.6%
4.1.x	Jelly Bean	16	25.1%
4.2.x		17	20.7%
4.3		18	8.0%
4.4	KitKat	19	24.5%

Data collected during a 7-day period ending on September 9, 2014.
Any versions with less than 0.1% distribution are not shown.

Figure 4.3.2.1 Percentage of users for each version of the Android Operating System, used with permission from developer.android.com

Since the hardware (tablet) was using the Android Kit Kat operating system it was important to use that as a starting point and work our way backwards. Android is backwards compatible in a sense, but if you want the UI to comfortably display both tablet and phone configurations you would need to explicitly declare this in the manifest file while writing the application. Since Android 3.2 was the first time developers were able to declare layout sizes for tablets, the team would have to make sure that the application was backwards compatible up to that point. Since there is a new API, Android 5.0 (nicknamed 'L'), coming out in the next few months, it would make sense to use the previous iteration of the Android API just to make sure that there are not unnecessary bugs.

4.3.3 Graphical User Interface

The graphical user interface (GUI) is going to be simple, clean and user friendly. It is going to have a welcome screen that will display for three seconds when the application first opens up. It will have a picture, with a slight glaze over it and the name of the application, “Hybrid Synthesizer”. After that screen is displayed, the application will check to see if the device’s Bluetooth is turned on, and if not there will be a message box telling the user to check their Bluetooth connection.

4.4 Power Generation

The power generation for the hybrid synthesizer was one used to power the analog components while a battery source powered the digital components in this device. Ideally, to run on a voltage battery would be the simplest means of power such a device however certain components of the device required specific voltages as to not overpower the integrated chips. Traditional Analog synthesizers are run off of direct AC current sent through a series of components to produce a manageable DC voltage to run the device. As technology improved and synthesizers became more portable and simplified the simple power supply of voltage batteries were enough to power some of the devices.

Most of these component devices and circuits within a power supply are built-in to the power boxes. The box connects to AC voltage from the wall and outputs a specified voltage. This voltage has to be attenuated, rectified, and filtered obtain AC voltage. The power regulator is the way to build this could either be done by using two transistors in a circuit with an input resistance, a zener diode attached to the emitter of the second transistor, and a load impedance. The first transistor acts as the pass transistor to allow voltage and the second transistor, which would be connected to the Base, acts as the comparator. An operational amplifier could also be straightforward and would be able to be used as a comparator to provide better feedback levels and also better regulation. Previously mentioned was that often times trouble can arise when an output voltage is too high because elements that regulate the circuit will normally be commanded to drastically drop the voltage. This causes a large amount of heat to be dispersed so a heat sink will be attached and fortunately, most regulator elements nowadays have over-current protection so that they will stop sourcing if the output current is too high or even might shut down is the input voltage is beyond certain ranges.

To truly compare a self-made power source with a battery, and with other power box sources, looking at the positive traits and limitations of the items under scrutiny will demonstrate the most efficient path to choose. First of all, the primary battery as an energy source is the simplest and easiest way to go. The energy storage is important because a battery must store the energy well and for a reasonably long amount of time whereas the wall plug cannot store energy unless it is plugged in and then must attach to a rechargeable battery. Specific energy of the battery must be sufficient enough to hold adequate energy for portable use and a long amount of time. The responsiveness of a primary battery has an advantage over larger power sources in terms of quickness because no warm up time is needed however if the primary battery runs out of charge, the user must purchase a new one. The environment a battery can withstand must also be vital because circuitry can tend to get hot and malfunction and if it does so, the battery must be able to keep a cool constant source of power to enable the circuitry to keep running. If the battery begins to leak, then damage can be caused to the circuitry which is why the chosen Lithium battery has the least amount of leakage, statistically speaking and the largest amount of efficiency in energy charge. The closer the battery is to 100% and the lower the discharge losses, the more ideal the battery is and the Lithium battery is the most ideal battery at this time. Easy installation is another bonus with using a simple primary battery and the ability to tolerate shock and vibration will keep the power supply from breaking. Some batteries require being in an upright position which is not practical for the Hybrid Synthesizer but the smaller Lithium primary battery is right in the ballpark.

The second option would be to use a pre-built power box supply units and distribute the voltage to the various components using voltage dividers and other relevant techniques such as operational amplifiers and attenuators. By using the gain incrementation of op-amps to boost an input voltage, the option of sending larger amounts of voltage would be possible task but the risk level is far too high when dealing with AC voltage. The DC voltage power boxes are options due to being easily attainable but the need to plugging into a wall is not an ideal design for a portable synthesizer.

Conclusively, constructing a self-made power supply is dangerous for those who are not certified and ideally should be built by the professionals. This being the case, a power module can be purchased to run the analog components. Power Box supplies ranging from 5 volts, 8 volts, 12 volts, 15 volts, and 18 volts are all viable options for the analog portion of the hybrid synthesizer. Many of the Operational Amplifiers and chips require a specific voltage to power the two terminals also. A +/- 15 Volt Power Source is the requirement for the LM series of Operational amplifiers alone. One voltage source, the positive source, gets plugged into the +Vcc terminal on the operational Amplifier chip while the negative source gets plugged into the –Vcc terminal. The power supply box is one that requires input from the AC derived from wall plugs as it goes through all types of manipulations to create the output voltage. Using a wall plug power supply will require input socket be attached to the analog circuits will be required. The digital components will be independently powered. The MSP boards can be powered and a majority of board types by a small voltage battery. The amount of boards needed to be powered will increase the number of required batteries but the application used to send signals over Bluetooth, to the MSP is powered by its own company's battery source.

4.5 Software Research

In the android application, there will be a few key development steps. An important factor in any Mobile Application is the graphical user interface or GUI, for short. The developers need to make sure that it is simple enough for the user to access every part of the application easily. First, we will have an application icon that will be displayed on the Android smart phone or tablet. This is the first piece of the application that the user sees so it would need to be designed well. I would like the icon to have a small drawing of a synthesizer or just the keys to a Piano.

Once the user clicks on the application, a welcome screen should appear for approximately 3 seconds. This will have the words “Hybrid Synthesizer” on it, and the same picture. If we need to load anything in the background of the application that takes longer than 3 seconds, the welcome screen should have a loading graphic which can be the same picture with the keys of the piano being pressed.

Once the application has loaded, the background code will run and it will determine if the Bluetooth functionality on the phone or tablet is turned on, if it is then no message will appear, and if it is not, then the user will be prompted to turn Bluetooth on. From there, the application will go to the Settings in Android and it will give you the option to turn Bluetooth on.

After Bluetooth is fully connected, we are ready to jam! The application interface will rotate 90 degrees and it will be in landscape mode from now on. This will allow for the user to have more screen real estate for the synthesizer piano. There will also be an interface for the user to be able to change the instrument and sound of the synthesizer. This will be done either with a button or a wheel that will scroll through the different options. A graphic of this process is documented below.

Once the user has pressed the note, this will enable the background code in the application to send a string that corresponds to the note’s frequency value to the Bluetooth module. This frequency value will be processed by the MSP 430 and routed to the analog circuit where it will be passed through the different waveform circuits and the sound will be altered.

The design team also hopes to have the functionality that will allow the user to change the instruments and create a different sound. This can be implemented by creating different look-up tables within the MSP430 code and these values are stored as an array. The frequencies are varied slightly to account for the timbre of each different instrument.

The code on the MSP 430 will be very different from the Android code because it will be written in C. The look-up tables with the different frequency values will be stored in the header files. These frequencies are calculated using the following formula:

$$f = 440 * a^n$$

Where 440 is the frequency for the note A located above Middle C which is referred to as A_4. This will be our reference note and all of the other notes will be calculated using this as a starting point. The variable a in this equation is the value for $a = 2^{\frac{1}{12}}$ which is approximately equal to 1.059463094359. The variable n is the number of half steps away the note is. This number can be negative or positive depending on its relation to the reference note A_4. The time duration of each note is calculated using this formula:

$$\text{Time} = 1/F(n)*1000000\mu s.$$

Using these two equations the team can calculate all that we need to create music on an MSP 430 or really any microprocessor. Since there will not be any hard-coded songs for the user to automatically play, and they will choose their own notes, these frequencies will be passed into a function on the MSP 430. They will be passed from the App through the Bluetooth module. This process is shown in the Figure 4.5.2.

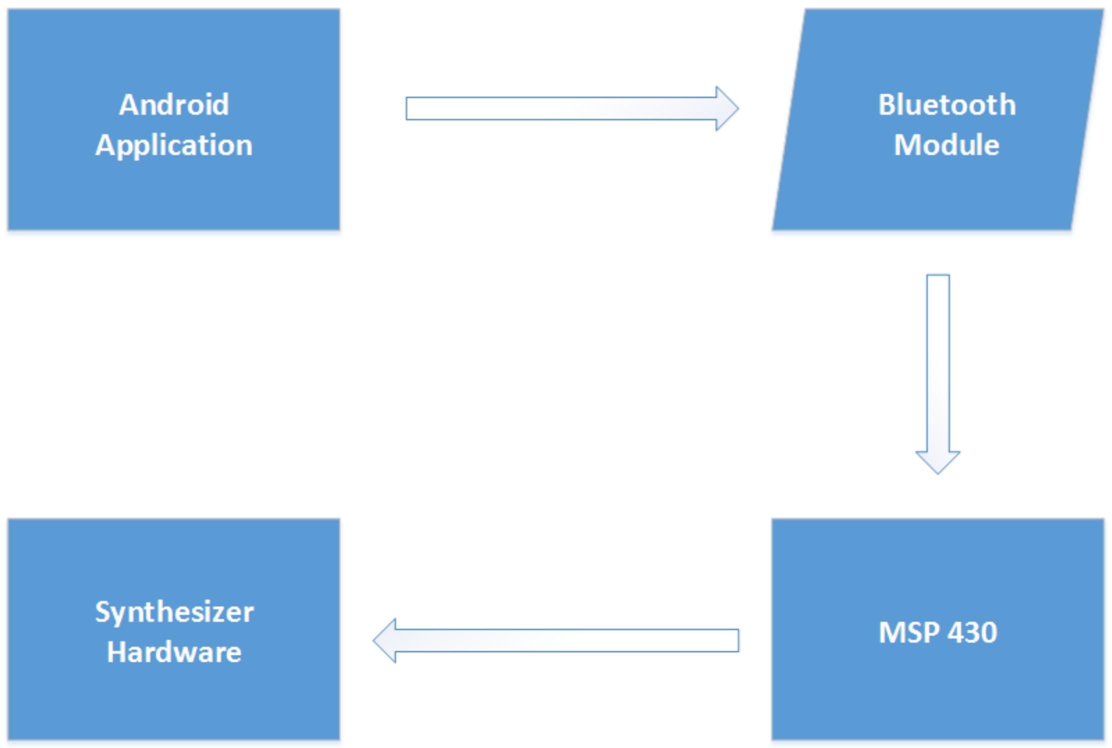


Figure 4.5.2: A Flow Chart of Data Transfer Between Devices. Image owned by the Hybrid Synthesizer team.

The code that is going to go between the MSP 430 is going to be fairly simple. The android application will send a string to the Bluetooth module, which will then be processed by the MSP 430. This string will contain the note name and the instrument that is supposed to be simulated by the note. The C code on the MSP430 will contain lookup tables that will correspond to the different frequencies that each note can produce. These "tables" will be stored in header files that we will import into the project using Code Composer Studio, an Integrated Development Environment created by Texas Instruments. We will also have to import the header files that correspond to the exact MSP430 part that we are using.

It would also be nice to have LEDs that were able to let the user know about different functionalities. For instance, one LED could be used to signal to the user whether or not the Bluetooth devices are paired. That way, if music was not being played for whatever reason the user could look at the LED and immediately know that there was a problem with the Bluetooth connectivity. Another LED could be lit up when notes were being passed to the device; it could also remain lit if the user held out the note for a very long time.

In the MSP430 code there will be functions for playing, which will pass the information from the Bluetooth as a parameter. We also will have a function for rests which will simply be an interrupt that is waiting for the next available input from the user. The synthesizer code will contain the main function that will call the other functions. It will also control the output which will act like the microprocessor is pushing the signals and the different frequencies to speakers, but in reality we are pushing them out to the Synthesizer.

4.6 Wireless Communication

For this synthesizer, the project will need to use wireless communication from the Android device to the MSP 430. This will allow us to wirelessly transmit the data of what note is being pressed on the virtual keyboard. The team considered some different options before finally settling on one certain type of communication.

One consideration was to run a USB cable from the Android tablet to the Synthesizer, but that would be cumbersome and it would probably interfere with the aesthetics of the design. So the team knew a wireless option would be best.

4.6.1 Wi-Fi

After attending a few workshops, the design team was also exposed to a Wi-Fi chip that could be used with the MSP 430. It was the SIMPLELINK WI-FI CC300 by TI. Wi-Fi has its advantages and disadvantages. One huge advantage is the range that you can connect to a Wi-Fi signal, as it can get a range of 100m. Seeing as our project only has a small Synthesizer that is being used with a tablet right next to it, range really is not a problem for our project. Also, there are cases where concert venues are outdoors or in places that are not easily connected to an internet service provider, and you would also need a router and a wireless access point. The Wi-Fi chip is probably not the best way to have our Synthesizer communicate with the application.

4.6.2 NFC

One other option that was considered for this project was Near Field Communication or NFC. This technology works with another device that has the NFC chip. It works by touching the two devices together. While Bluetooth and WiFi use radio frequencies for their wireless communication, NFC is a offshoot of radio-frequency identification or RFID but it needs the devices to be very close to one another. NFC works by using magnetic induction. The passive device usually called a reader emits a small electric current which generates a magnetic field that in turn bridges the physical space between the devices. The field is received on the client device which turns this magnetic field back into electrical impulses in order to communicate the data. It works like RFID in the fact that it uses the 13.56MHz radio frequency spectrum and it uses less than 15mA of power to send information. This technology has been implemented in various ways such as electronic ticketing and using it in the new technology Apple Pay. The downside for this implementation is the fact that NFC usually only works when devices are less than 20cm away. This creates an extremely short range in which the two devices, the mobile device and the synthesizer, will be able to function. If this device was used in a concert venue setting, it would be very difficult to be kept in this range and still put on a performance.

4.6.3 Bluetooth

Another popular option in wireless transmission is Bluetooth. It is similar to the Wi-Fi chip in the fact that it can easily be mounted to a Printed Circuit Board. There are also Bluetooth stacks that come with certain Bluetooth modules that make it easier to install the firmware on the MSP 430. There are standards that Bluetooth is held to, so developing with it is easier because it is so regulated. It only has a range of 10 m but that is not a problem seeing as the Hybrid Synthesizer team would like to keep the Android device close by the Synthesizer.

Our goal would be to create a Personal Area Network (PAN), or a piconet using our Bluetooth device already embedded in the Android tablet and our Synthesizer. When two Bluetooth devices connect together, this is called pairing. The devices have a master-slave structure where up to seven devices can connect to a single master. This relationship is packet based and the master receives packets from the slave and vice versa. The developer would configure the Bluetooth module to always be discoverable when it is powered on, this would make it so that when the Synthesizer turned on, it would automatically connect to the application without having to manually pair the device every time.

Low power was also an important factor because the team eventually wants to run our synthesizer off of battery power, and the team doesn't want to overheat the system. Bluetooth 4.0 is specifically designed to use less power than previous iterations of Bluetooth. It goes into a "sleep mode" when not being used, so there are less pings while data is not being transferred. Bluetooth also works very well in transferring audio, which is the basis of our Senior Design project.

4.6.4 HC-05 Bluetooth Module

The module that would worked best for our design was the probably the HC-05 Bluetooth Module which is compatible with most development boards. It is designed specifically are cost efficiency and easy usability. It also has numerous amounts of resources available to help us understand how to implement it into our design such as instructables and AT command sources.

Something else to consider was once the information was sent wirelessly, how was it going to talk to the Arduino? The UART Serial capability made this an easy problem to solve. UART (Universal Asynchronous Receiver/Transmitter) is a system for pushing a byte of data from one device to the other, so long as both have UART implemented. It was an easy way to transmit information and it is going to solve a huge part of our design. Knowing that the HC05 uses UART, it fit into our design perfectly.

4.7 Arduino

The Development Board device chosen to be implemented in the design of the Hybrid synthesizer is the Arduino Uno due to its specific hardware and software features necessary for creating audio output. An effective link for data transmission is bridged by the HC05 blue-tooth module and the Android Hybrid Synthesizer Keyboard Interface. Extended features support an external high order filter and connected to an audio jack to output digital audio. More features of this versatile development board includes 14 digital pins, 6 analog input pins, 6 pulse width modulation pins, a 16 MHz ceramic resonator, a USB connection, a powerjack, an ICSP header,. The UART function is installed in this board for serial communication capabilities as well as reset buttons. The device allows several on-chip peripherals that enable easy to communication wireless connection with a host system and the Bluetooth core. The transmitter – receiver architecture is seen in Figure 4.7.1.

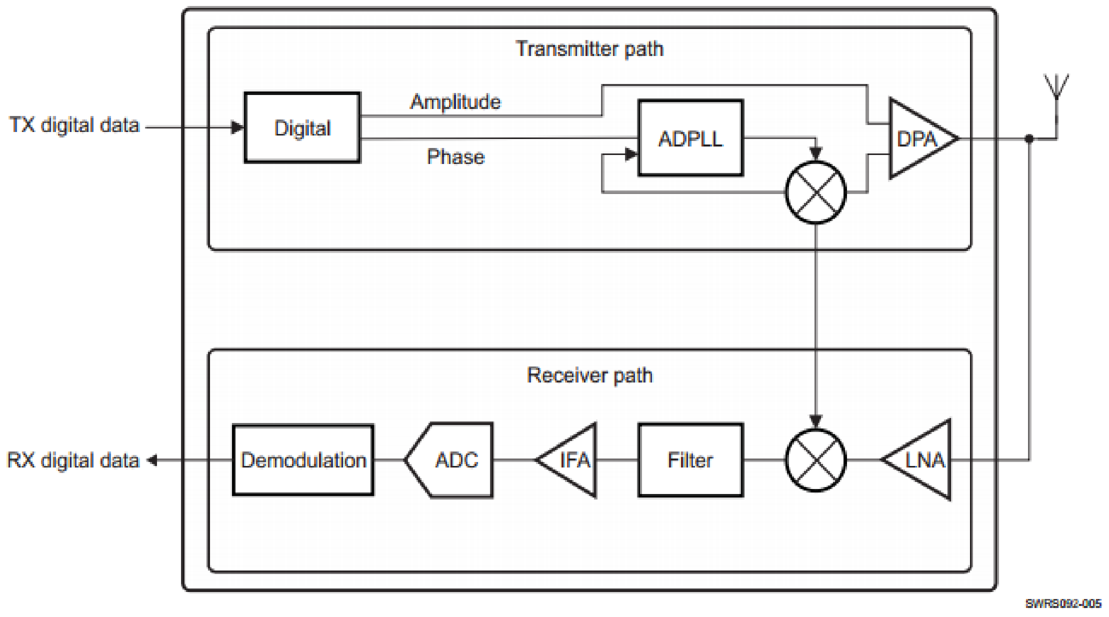


Figure 4.7.1: Transmitter/Receiver Architecture. Used with permission from Texas Instruments

The Arduino Uno microcontroller ATmega328 contains a 5 volt operating voltage which is necessary for powering the HC-05 blue-tooth controller and other components. The input voltage requires 7-12 volts which would be easy to power with a typical battery and the limits are 6 volts to 20 volts so there would not be much of a problem to overpower the board. The memory capacity in the Flash Memory is 32KB which is a massive amount of variables and code capable of being stored in the program memory. The Flash Memory capacity is another reason for choosing this chip.

5.0 Design Summary of Hardware and Software

After cross-comparing various techniques for approaching the hardware and software design, the design team narrowed down a series of relevant technologies and design methodologies to use for their final design. This section will discuss those design decisions.

5.1 Hardware

The oscillator is the main tone generating module for an analog synthesizer. The design team wished to make the oscillator voltage controlled to more easily interface it with the chosen microcontroller. A standard step for voltage controlled oscillators in synthesizers commercially produced follows CV/Gate implementation, at a scaling rate of 1V/Octave.

This method was initially popularized by Bob Moog, and seemed to be the most convenient standard to utilize during the design process. This gave the design team the most flexibility concerning their microcontroller's built-in PWM functionality and range, provided the team could get the resolution and accuracy of semitone steps down to 80mV. That value is the difference between two adjacent semitones when following 1V/Octave scaling. The oscillator, through the use of a matched pair of discrete transistors and a temperature compensator, exceeds the team's design requirements and tracks accurately over at least 6 octaves of range, and can be tuned to track accurately above and below the range of human hearing.

The circuitry for the VCO shown in Fig. 5.1.1 demonstrates the hardware implementation of the CV/Gate 1V/Octave converter as well as the tone generator for the entirety of the oscillator. The team decided to opt for a ramp core design, meaning all other voices would be filtered from a single ramp wave oscillation source. This was both to ensure all voices would scale correctly together across the entire operating range (meaning all four voices would remain in tune with each other without any additional correctional circuitry), as well as to make the filtering and shaping of all subsequent voices (square, triangle, sine) easier, which will become more apparent as this description continues.

UA-1 sums control voltages from up to four additional sources and applies the summation as a control voltage bias. Our design did not implement multiple control inputs, so UA-1 simply buffers the tuning bias provided by the R2 and R3 potentiometers, which can be adjusted to shift the range of control voltages applied up or down as appropriate to the hardware interfacing with the oscillator. This CV summing amplifier at UA-1 has a gain of -1/50th, scaling inputs to 20mV per volt of input. The trim potentiometer R16 allows further adjustment down to 18mV per volt, which drives the current reference discrete transistor pair to convert the voltage exponentially into current flow. Thus, linear changes in the CV inputs are converted to exponential current changes at the current sink, doubling the current flowing into the transistor pair for every volt of CV. The transistors are compensated to continue to track accurately through temperature changes through use of a regulatory 2K platinum film +3300 PPM temperature compensator mounted in thermal contact with the transistor pair.

The circuitry made up of D1, R9, and R11 serve as compensation circuitry due to the fact that there is a finite time required to discharge the integration capacitor C6 while the circuit is oscillating. This circuitry allows U1-B to boost the current at the current sink when the oscillator is operating at higher frequencies (above 4KHz), preventing the tone from going flat as it ascends beyond this threshold.

The exponential current flow created by the discrete matched transistor pair causes the integrator formed by U2-A and C6 to ramp upward linearly from ground toward V+. This ramping voltage is fed into the comparator U2-B, which sits at an output of approximately -10.6V. It will remain at this voltage until the comparator is tripped by U2-A's output crossing the 2V threshold.

Once the ramp voltage reaches approximately 2.2V, the comparator's reference pin has been breached and the output of the comparator pulses up to 8V for about 3μS. This forward biases D8, turns on the PN4391 JFET, and charges C17, pulling the integrator's output back down to ground. After a reset delay of about 1 μS, the integrator then begins to ramp up linearly again, generating a reliable ramp wave at the frequency determined by its corresponding control voltage. As a result, the ramp wave output from the RAW terminal varies in amplitude from 0V to about 2.2V, with the frequency determined by the current flowing into pin 1 of the discrete transistor pair.

The circuitry for the VCO shown in Fig. 5.1.2 converts the ramp wave into triangle, sine, and square waves for a total of four analog voices oscillating at the same frequency. The raw ramp wave is fed into an inverting amplifier U4-A, where the voltage is boosted to 10 V peak-to-peak, flipped to become a sawtooth wave, and adjusted through R5 to oscillate evenly around ground, giving it an amplitude of +/-5V peak-to-peak. This adjustment is important for the first filtering process.

The sawtooth wave is fed to U4-B, where it is inverted again at unity gain to produce the ramp wave originally generated. These two waveforms are then rectified by D2 and D3, resulting in the positive half of their waveforms being preserved to sum together and create a triangle shape from 0-5V, as can be seen in part 1 of Fig 5.1.3.

They are dropped across R21, and fed into U4-C with a gain of 2 to return the rectified wave to 10V peak-to-peak, again adjusted to oscillate around ground through use of the trim pot R7 (part 2 of Fig. 3). Segments 3 and 4 of Fig. 3 demonstrate the appearance of an improperly trimmed triangle wave, resulting in odd overlaps between uneven positive peaks of the two ramp-based waveforms.

At the time the initial ramp wave generator is reset by the comparator circuit, an audio glitch is created at the peaks of the triangle wave for about 6 µS. Although the integrator's fall time is 1 µS, the slew rate limiting of the operational amplifier U4-A causes a delay of 3 µS. U4-B then also takes an additional 3 µS to pull low. The glitch happens so quickly that it is inaudible to the human ear, and it's amplitude is reduced by C14. The design team felt this level of inaccuracy was acceptable for the product as it would not affect the musician or output of the music on any notable level.

At this point, both ramp and triangle waves have been created. The triangle wave is fed to the LM13700 (U5), which applies a non-linear distortion to the wave to help it approximate a pure sine tone. Three trim pots are used to shape the wave so that it achieves negligible harmonic character, as seen in Fig 5.1.4. R60 carves out the general shape of the sine wave, and R64 fluctuates the curvature between square and triangle shapes and should be adjusted until minimal sine distortion is achieved. R59 adjusts the level of the waveform, and should be used to bring the waveform back to 10 V peak-to-peak centered around ground.

The rectangular wave comparator uses the ramp wave from U4-B to forward bias D4 and charge C13 with the positive excursions of the waveform, and forward bias D5, 6, and 7 and negatively charge C12 with negative excursions of the waveform. These two voltages are buffered with U6-A and U6-B respectively and are then applied to a panel mounted potentiometer R39. This potentiometer allows the user to adjust the shape of the rectangular wave between a 10% and 90% duty cycle, as the output of the potentiometer is used as a comparator input for U7. When the voltage of the ramp waveform being referenced by the non-inverting input of U7 goes below the threshold set by the potentiometer R39, the output of U7 goes low. Likewise, when the voltage of the ramp waveform being referenced on the non-inverting pin of U7 goes above the threshold set by R39, the output of U7 goes high. The PWM input allows for voltage control of the pulse width, however our design chose not to make use of this feature, instead leaving it as a modular option for future expansion.

The design team selected a matched pair of 2N3904 transistors for the current converter because they will track accurately over the desired range of operation, producing minimal harmonic distortion due to accurate pairing and temperature compensation. These transistors were the spiritual successors to hobbyist favorite current mirrors such as the SSM2210, which are no longer in production.

The OP275 was selected for its excellent linear scaling as a summing amplifier. It too comes highly recommended in the hobbyist community for its reliability when paired with the SSM2210 or equivalent circuitry for current driven oscillators.

For all other gain stage operational amplifiers, as well as the oscillator for the initial ramp wave, the TL07X family of JFET general purpose operational amplifiers was selected due to its low-noise operation capability, with an incredibly low Total Harmonic Distortion value of 0.003%. In addition, its low power consumption and its excellent linearity were also deciding factors.

The integrator's capacitor C6 needed to be temperature stable to ensure proper tracking for the oscillator across the range of operation of the system, therefore the team decided to opt for a polystyrene capacitor, due to its excellent temperature stability.

To properly sculpt the shape of the sine wave from its triangle wave input, the LM13700 was selected for its flexibility as a signal filter and its ability to linearize or de-linearize signals passed through it.

To protect from temperature drift, 1% metal film resistors were used throughout the entirety of the VCO design, to allow for high performance across the entire operating range with minimal variance.

The final analog waveforms will then route from their respective output nodes on the VCO to an audio mixer that can be manipulated by the user to adjust the relative volumes of each signal in the mix, from having all four waveforms blasting at 100% volume to isolating one or more waveforms, and everything in between.

To allow the user this level of functionality, 100kΩ Audio (Logarithmic) Taper panel mount Potentiometers were selected. These audio taper potentiometers, in contrast to their linear counterparts used elsewhere in the VCO, account for the fact that humans hear on a logarithmic scale, and this linear changes in amplitude will at first scale as expected by the listener but then appear to "pan out" rather than increase in volume at the same rate. Both slide and knob potentiometers were considered, and while slide potentiometers are more traditionally used as volume faders in the audio world, the design team felt that saving space on the faceplate would contribute to the synthesizer's portability, so only knob potentiometers were used.

The audio mixer is a simple summing amplifier that will take the incoming synthesizer voltages and apply a gain of 1/50th to them so that they will be brought down to headphone voltage (+/- 100mV per channel). Due to the additional drop that occurs when multiple channels are summed through one amplifier, the sum of all four voices at maximum volume should fall just short of +/- 400mV, which is the expected maximum operating range for most audio amplifiers, and is also pushing the maximum range expected by headphones. The resulting waveform will then be routed to an 1/8" audio jack (headphone out), which can be listened to through headphones or a powered speaker that accepts 1/8" cables.

In addition, a JFET based muting circuit was implemented in between the summed analog waveforms and the final audio summer that combined the analog and digital waveforms. This was due to the fact that the VCO oscillated at any time when receiving power. Each keypress sent from the app toggled on a mute control signal that would unmute the signal and allow it to pass through. Removal of a finger from the keyboard returned the circuit to mute state without clicks or pops (Fig.5.1.5).

Additional analog modules such as filters, an ADSR envelope, and a sequencer were not added due to time constraints. However, the design was left modular in case future revisions were ever attempted.

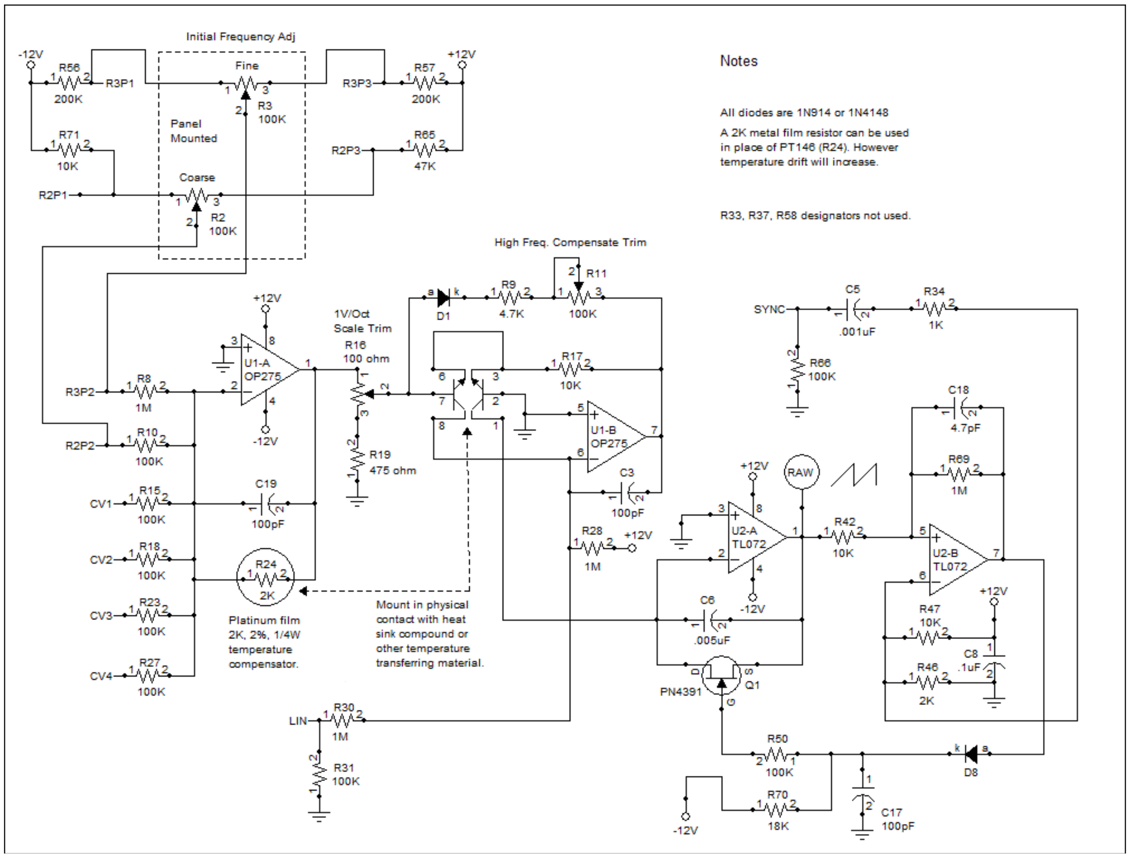


Figure 5.1.1: VCO Schematic Page 1. Image owned by Ray Wilson, reprinted with displayed permission

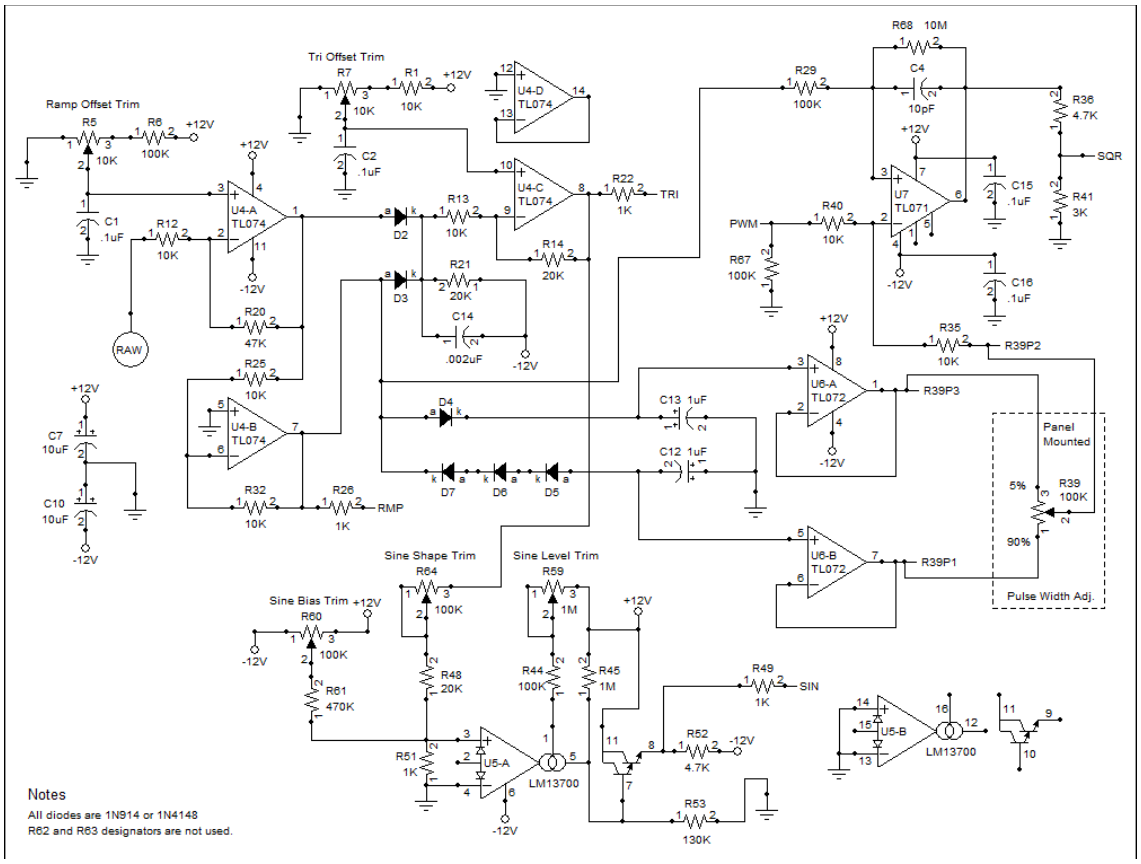


Figure 5.1.2: VCO Schematic Page 2. Image owned by Ray Wilson, reprinted with displayed permission

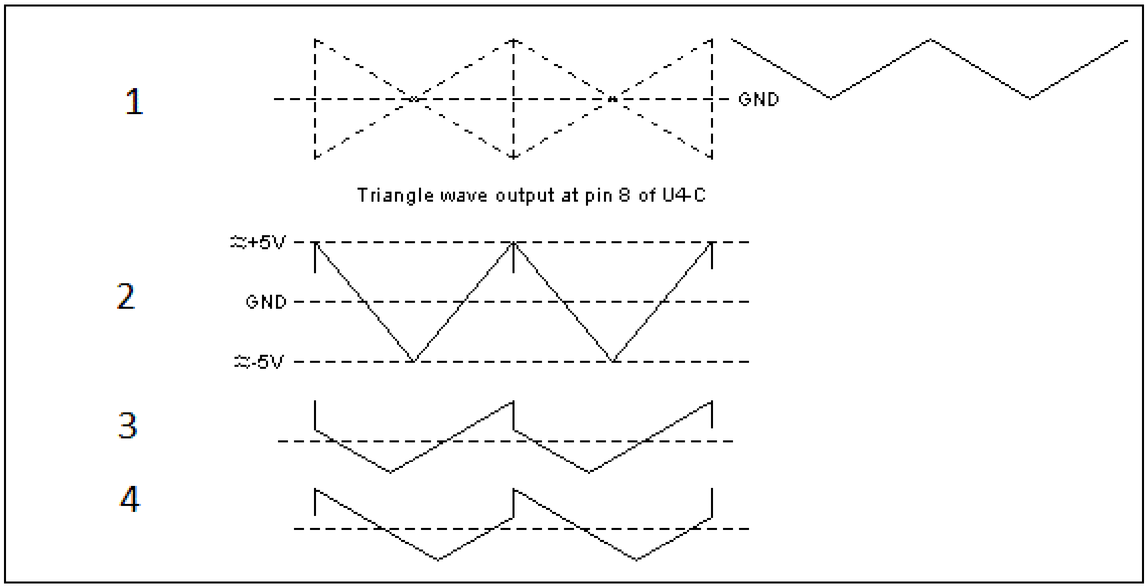


Figure 5.1.3: Triangle Waveform Synthesis and Adjustment. Image owned by Ray Wilson, reprinted with displayed permission

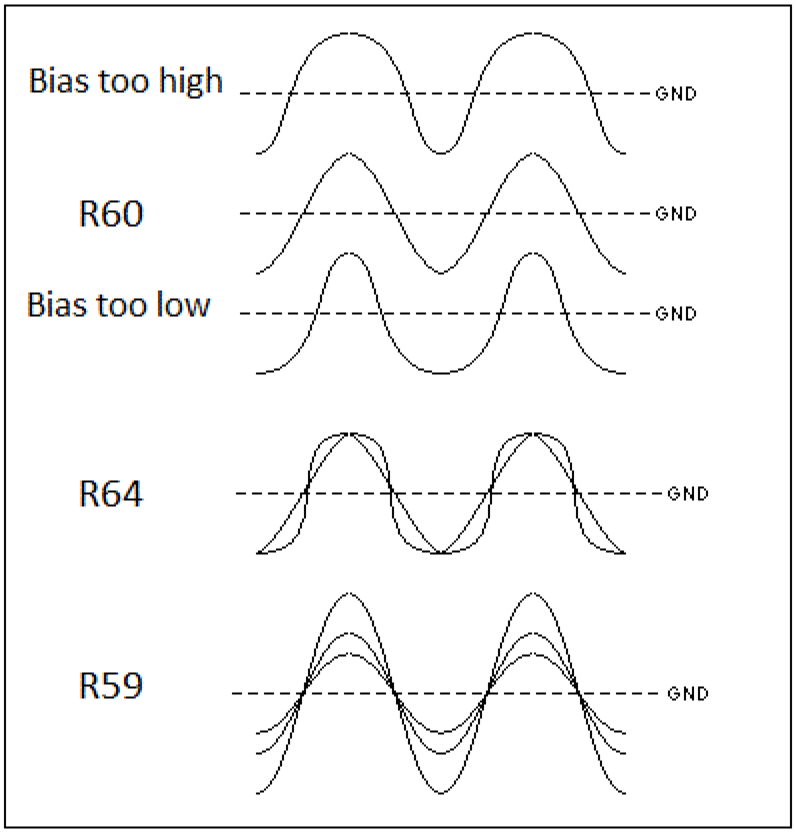


Figure 5.1.4: Sine Waveform Synthesis and Adjustment. Image owned by Ray Wilson, reprinted with displayed permission

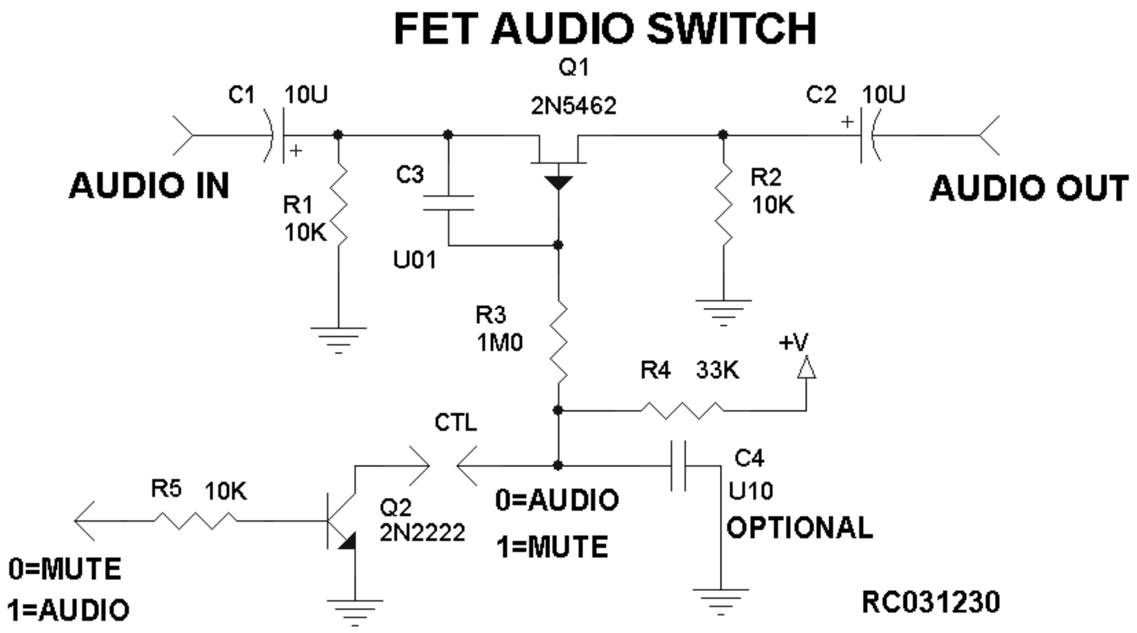


Figure 5.1.5: JFET Audio Mute. Reprinted with displayed permission

5.2 Software

The basis of our software design is to get the user input to the Synthesizer so that the sound can be manipulated. Once the user pressed a key on the virtual keyboard, the application creates a string value that is sent to the Bluetooth module. This string is received and is sent via UART to the ATMega328. From there microcontroller determines whether or not the note is an analog note or a digital instrument. If it has an instrument associated with it, that sound is generated on the ATMega328 board and sent through a filter using five of the six pulse width modulation pins for sound output. If the instrument is analog, control voltages are generated using the final pulse width modulation pin and this drives the VCO.

5.2.1 Android Application

Starting with the application, this is the starting point of the entire design. The application is where the user inputs the notes and uses the keyboard interface. The Graphical User Interface is written in XML and it is stored in a sub-folder the res folder in the Android project. This is also where the icon and the splash screen for the application is placed. A picture of the splash screen is shown below (Figure 5.2.1.1).

Figure 5.2.1.1: The Hybrid Synthesizer splash screen. Image owned by the Hybrid Synthesizer Team

The splash screen is the first thing that the user sees. It is a picture of a synthesizer that was created in photoshop. This picture appears on the application for 5 seconds. The words "Hybrid Synthesizer" are typed across the image. It is horizontal because the entire application is set to this mode. This is done using the Main.XML file which controls the interface of the application. Normally, an application only switched to a horizontal orientation when the user turns the phone sideways, but in our case, the application is permanently in a horizontal orientation. This is to create the most real estate on the screen to show our piano interface (Fig. 5.2.1.2).

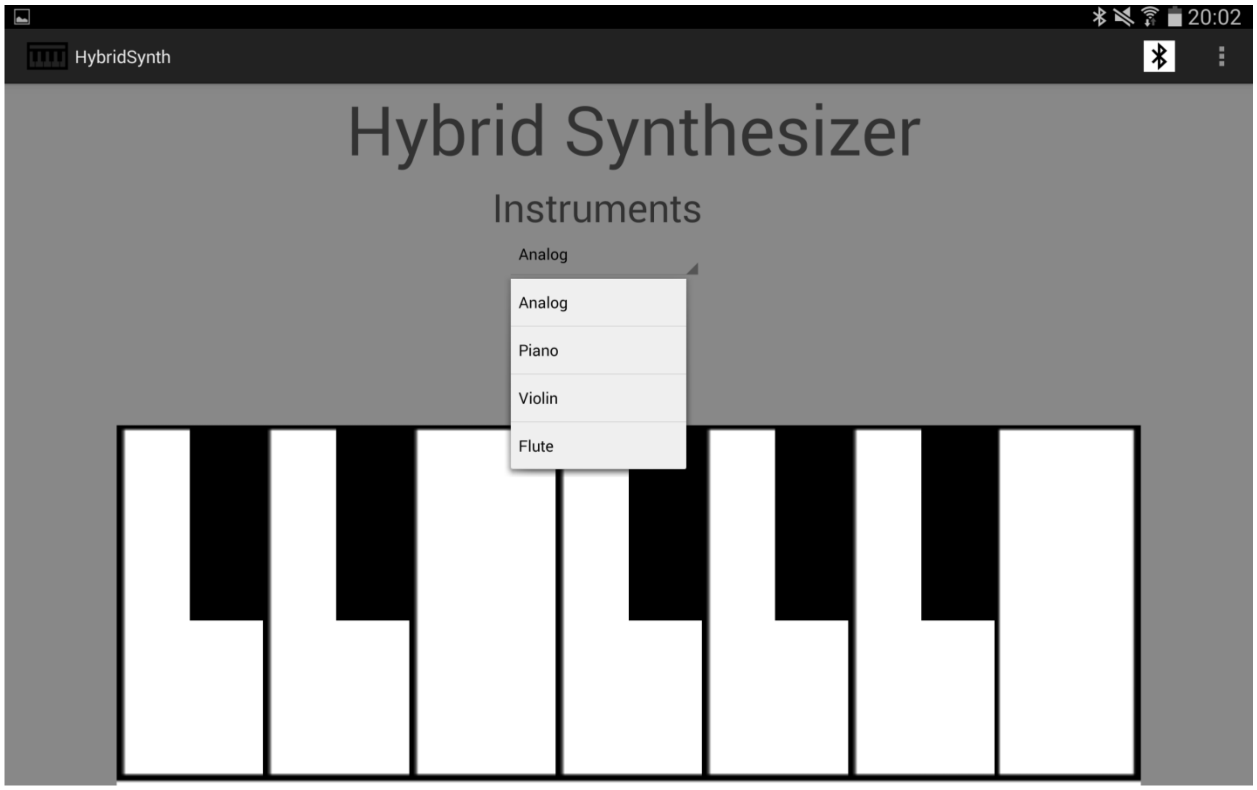


Figure 5.2.1.2: Keyboard interface screen. Image owned by the Hybrid Synthesizer Team

The layout of the piano created in Java in order to allow the buttons to be placed on top of each other. This layout references the keys which will be placed in the res folder and ultimately in the drawable folder. When a key is pressed, the key will change color. This different key is another image that is also put into the drawable folder. These pictures are saved as .PNG images and they have different resolutions which can be used in different versions of the applications.

This application is created using only two active classes, the Main Activity and the Splash Activity. There was also going to be a separate Bluetooth Activity, but for the sake of time, the Bluetooth implementation had to be changed. The Main Activity is the home page of the application. This contains 12 buttons that represent the piano keys, an octave button, and an instrument drop down. The different notes are identified by their names going from C to B and their accidentals, which are what makes a note sharp or flat, is also generated. The GUI also has an octave button which allows the note to be the same name but just a higher pitch.

Figure 5.2.1.3 demonstrates some of the classes that were going to be used in the software implementation. An instrument class has also been created to let the user choose the type of instrument that they would like to hear. This is an added feature in addition to creating the piano sound which is synthesized. These sounds are also generated by lookup tables and the string has an instrument portion. The instrument sound is sent to speakers instead of to the synthesizer hardware.

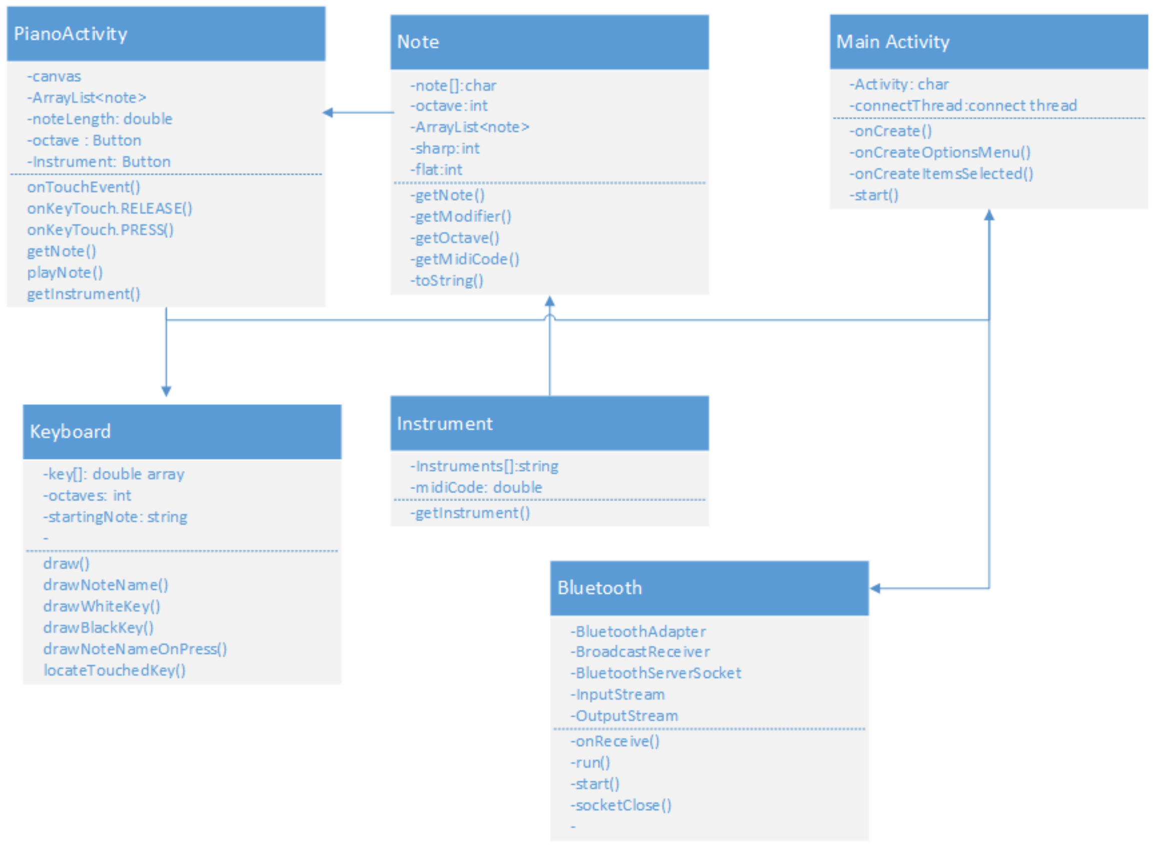


Figure 5.2.1.3: Class Diagram of the Android Application. Image owned by the Hybrid Synthesizer Team

The Keyboard class will be what draws the keyboard and it will reference the keyboard pictures that are placed into the drawable folders. There will be listeners used to determine how long the key was pressed so that the program knows how long to change the color. This information will also be shared with the Piano Activity class.

The Piano Activity class will draw the canvas for the application. It will also listen to the button and see if the Octave button has been pressed. If it has, then the notes will go up in pitch. It will also take in the information from the instrument menu and see what the user chose for the instrument. It will also change the color of the key if it is pressed along with causing a vibration. This will give the user feedback so that they know which note they pressed.

The Main Activity of the application is where everything starts. When the application is opened, it will start with the oncreate() method to perform all of the startup logic in the activity. This should also define the user interface. When onCreate() is finished, the system should call onStart and onResume as well. The user only starts to see the application when it is in the onStart() stage. The method onResume() is also necessary to any android application because it will pause the applications activity so that if the user has to exit the application for any reason, such as a phone call, they can come back and pick right back where they left off. The last method of the main activity should be onDestroy() because this method will stop all background threads and remove the application from clogging up the system memory.

The Integrated Development Environment that was used to develop this project is Eclipse Juno with the Android Development Tools (ADT) and it was used with the Android SDK. The version that the design team used for this project will be Android 4.4 API 19 (KitKat). This is because most devices are not compatible with Android 5.0 (Lollipop) just yet, so using the next version down seems like the ideal solution.

5.2.2 Arduino/ATMega328

The design team used a reference design from Arduino called the Arduino Uno Development Board. This is designed to take in input data from any Bluetooth device, and output it the circuitry which is attached to speakers. This board incorporates the AtMega328 which is the Microprocessor that the team used for the Hybrid Synthesizer project. It simplifies our design because all of the pieces that the team needed to include were able to be controlled via the AtMega328 chip on this development board included the VCO control voltages, the instrumental note voices, the muting circuit, the low and highpass filters, and the Bluetooth module circuit. The Arduino Uno also served as a local ground for the ground supernode.

The data was transferred from the Bluetooth Module to the Arudiono using UART capabilities. In Figure 5.2.2.1 it shows the Block Diagram courtesy of Arduino. It is a very simplified version of the Arduino Uno which shows how the connection from the Bluetooth to the Arduino is connected. A power supply for this design is typically a battery.

The Arduino code was developed in the Arduino Software. The majority was written in C code but the IDE was capable of compiling Assembly code as well. The synthesizer code was quickly developed in C language and proved to be much quicker when compared to assembly step by step code. The team also felt more comfortable with the language of programming. The code has a collective serial interface that reads, sorts, and compares the Bluetooth module data, but only after it has been appropriately installed to match the baud rate or the Android terminal, and the Receiver and Transmitter pins have been placed correctly. The deciphered string reads in data for either analog control voltages or digital instruments. Pin 11, PWM pin 5 is set to control voltages which get sent to the VCO while the other five PWM pins are sent through a high order filter (Figure 5.2.2.1) which creates instrumental sounds.

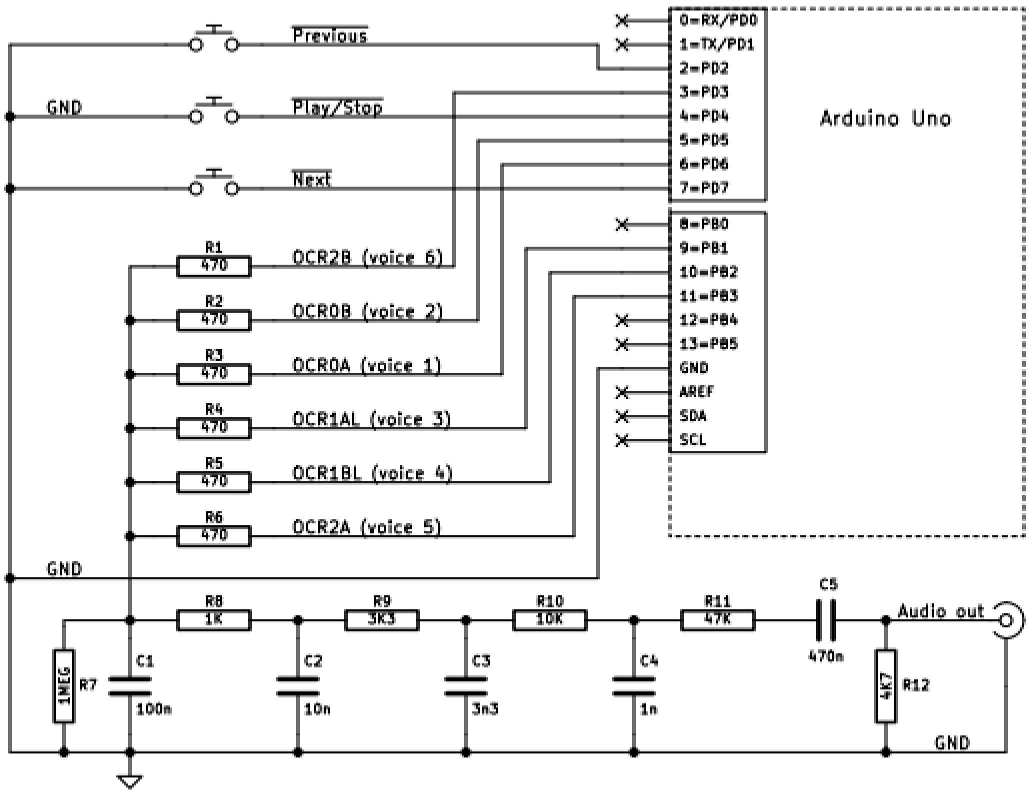


Figure 5.2.2.1: Arduino Uno and High Order Filter Reference Design, used with permission from Arduino

The CPU clock frequency for the AtMega328is 16 MHz and the clocking cycle for the Pulse Width Modulation pins are set at 62.5KHz. The pulses are synchronized with Serial Data comparators so that the correct note flagged, located, and output through the mixing filter. The player function is set at half of the Pulse Width Modulation frequency to check the instrument wave sampling, the tempo divider and to upload the note from the music interpreter function to have a note ready to play twice as fast as each Pulsed Clock Cycle for the PWM pins. The whole while these clock cycles are occurring, the ADSR clock is generating instrument wavetables envelopes constantly at a 1.3KHz logic time.

Figure 5.2.2.2 shows how the data transfer occurs between the devices. There is data passed from the Android application to the Bluetooth Module and then finally to the ATMega328. The data will be passed into the ATMega328 through the UART from the Bluetooth module. They are connected in hardware using the RX and TX pins. RX is the receiver pin and TX is the transmitter pin. These are how the bits are communicated serially between the two devices. Since the ATMega328 will not be writing anything back to the Android Application, the Bluetooth module does not need to send anything back via the TX but the RX will be used.

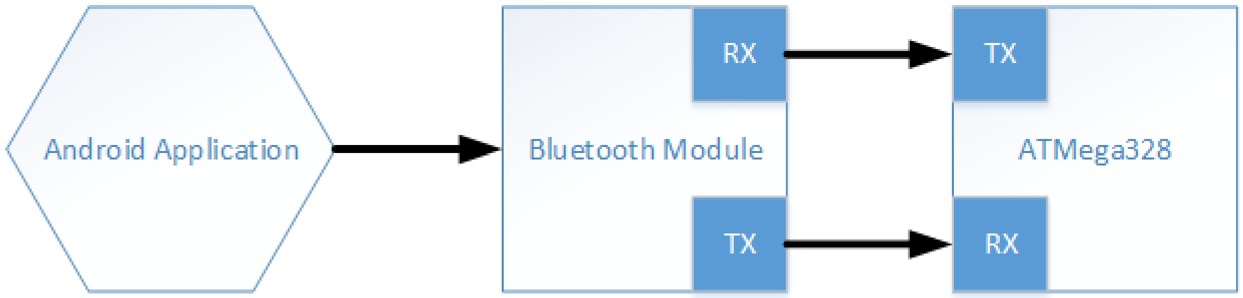


Figure 5.2.2.2: A diagram of Bluetooth to Synthesizer communication. Image owned by the Hybrid Synthesizer team

The UART which stands for universal asynchronous receiver/transmitter is simply a block on circuitry that controls both the parallel and serial communication interfaces. They are generally already implemented on microcontrollers and they are usually a standard communication tool.

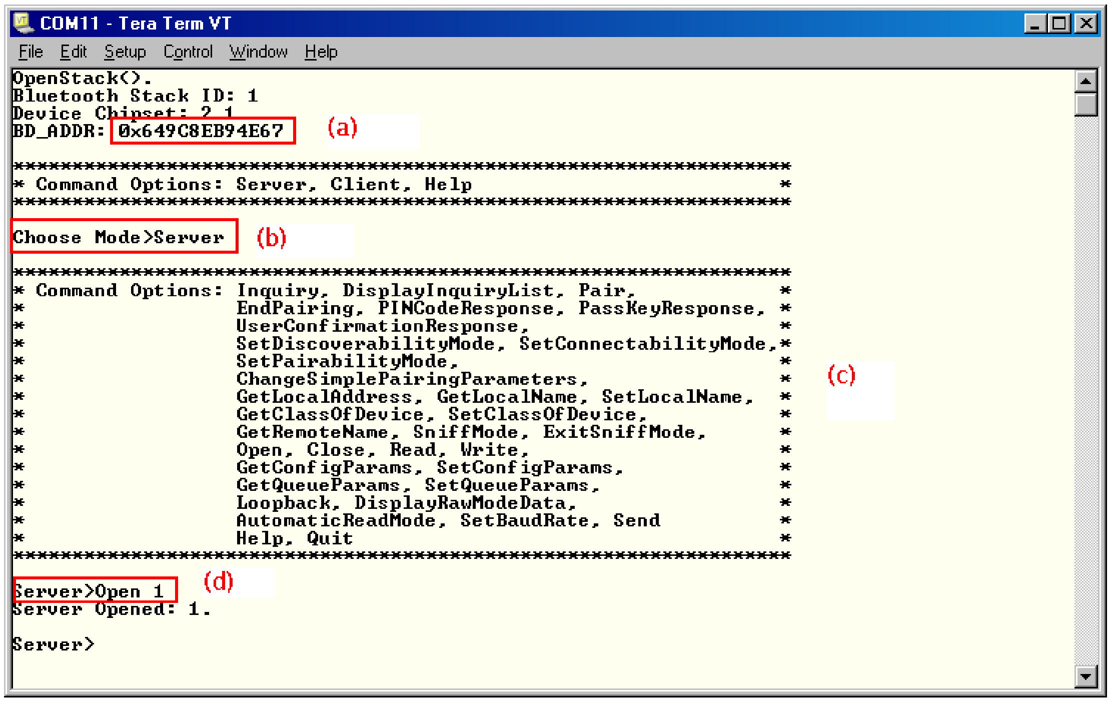
Once the data has been passed from the RX port, the developers are ready to get to the meat of the program. The output of the Arduino PWM pins are hooked up to different pieces of hardware on the analog side, specifically a Low-Pass Filter which sends the control voltages to the VCO. The hardware is what will generate the notes and create the different waveforms. These PWM voltages in the program so that the code is formatted nicely and the rest are used for the digital functions as needed.

5.2.3 Bluetooth Module Configuration

The Bluetooth module that the team ended up choosing was the HC-05. This was a Bluetooth Module that already had a Bluetooth Stack installed. This allowed for ‘plug and play’ functionality with minor set up. In order to have this module interface with the Arduino, we needed to create a voltage divider to step the voltage down from 5v to 3.3v which is all that was needed to power the HC-05. This module was connected using the RX, TX, VCC and GND pins. The team was able to power on the Bluetooth module simply by giving it power. Then the AT commands were used to make sure the module was set into slave mode and to get the MAC address of this specific Bluetooth module, as each one is different. Once the developers flashed the device, they needed to use a terminal program such a TeraTerm or Hyperterminal in order to send commands to the Bluetooth module (Figure 5.2.3.1).

The Bluetooth device was configured using the terminal and the UART capabilities. The UART, which stands for Universal Asynchronous Receiver/Transmitter is how the information is transferred from the Android device. The design team used the UART_TX (transfer) and the UART_RX ports. The other two ports UART_UTS and UART_CTS were not used. This is useful because there needs to be a continuous flow of data since the notes can be played immediately one after the other.

In order to pair the Bluetooth to the phone you need to use a passcode. For this module the passcode default is '1234'. This should make the phone connect to the Bluetooth module. An LED will be triggered that told the team whether or not the Bluetooth is paired with the Android device. If the LED is on and blinking slowly, then the devices are synched, and if the LED is off, then the HC-05 is off. The design team can configure the module so that if the Android device's Bluetooth is turned on, they will automatically connect. The parameters that will be passed to the device will be a Baud rate of 9600, 8 n, 1 and no flow control. This gets the terminal set up to create a connection**.**



```
OpenStack().
Bluetooth Stack ID: 1
Device Chipset: 2.1
BD_ADDR: 0x649C8EB94E67

******************************************************************
* Command Options: Server, Client, Help                          *
******************************************************************

Choose Mode>Server

******************************************************************
* Command Options: Inquiry, DisplayInquiryList, Pair,            *
*                  EndPairing, PINCodeResponse, PassKeyResponse, *
*                  UserConfirmationResponse,                     *
*                  SetDiscoverabilityMode, SetConnectabilityMode,*
*                  SetPairabilityMode,                           *
*                  ChangeSimplePairingParameters,                *
*                  GetLocalAddress, GetLocalName, SetLocalName,  *
*                  GetClassOfDevice, SetClassOfDevice,           *
*                  GetRemoteName, SniffMode, ExitSniffMode,      *
*                  Open, Close, Read, Write,                     *
*                  GetConfigParams, SetConfigParams,             *
*                  GetQueueParams, SetQueueParams,               *
*                  Loopback, DisplayRawModeData,                 *
*                  AutomaticReadMode, SetBaudRate, Send          *
*                  Help, Quit                                    *
******************************************************************

Server>Open 1
Server Opened: 1.

Server>
```

Figure 5.2.3.1 : A screenshot of the Terminal Setup from the Server perspective. Image owned by the Hybrid Synthesizer team

5.2.4 Android Application Bluetooth Implementation

Once the module is set-up, the code for the Android application can be implemented. This is done by adding a new class to our existing Android Project. The first thing that needs to be done is the Bluetooth adapter needs to be turned on and you need to have an android device that has a Bluetooth module already built in.

Before writing any code in the android project to get Bluetooth up and running, the Bluetooth permission needs to be declared in the application manifest file. There is also a way to make the application initiate device discovery when the application opens up. This could be useful in this project because it is completely dependent on having a Bluetooth connection in order for the Hybrid Synthesizer to be functional. The code is placed in the AndroidManifest.XML file and it will look like the following:

```
<manifest ... >
      <uses-permission android:name="android.permission.BLUETOOTH" />
      <uses-permission
android:name=”android.permission.BLUETOOTH_ADMIN/>
</manifest>
```

The Bluetooth permission enables the use of Bluetooth in the application entirely and the Bluetooth Admin permission allows the application to override the settings on the Android device so that the user can automatically discover other local devices.

After this, the team created the majority of the Bluetooth code. This is how the developer gets the Java in the android application to pair the devices together. The code is put into the main function of the android application. First it checks to see if the device has a Bluetooth adapter and if the adapter is not found the rest of the code will not work. If there is a Bluetooth adapter, then the code will ask to turn the adapter on if it is not already on. This will also allow the user to enable Bluetooth using the isEnabled() method. A dialog box will appear and the user will be able to select Yes and the system will start enabling Bluetooth automatically.

The team first set up the Bluetooth module to be discoverable. After doing this, once the module is powered on, it should broadcast a signal that can be picked up by the android devices’ Bluetooth scanning capabilities. Next, the Android Device and the Bluetooth module need to discover each other. The adapter on the phone will query the surrounding devices and the devices will share some information such as the unique MAC Address so that the devices can identify which Bluetooth module we need to connect to. This is done using the startDiscovery() method. This method returns a Boolean value that will say whether or not discovery has started, after that android device will process the information sent by the Bluetooth module like the MAC Address.

Once the MAC address for the device is found the Bluetooth module can now connect. Since that was all that was needed from the discovery process, the developer can stop the discovery method by using the method cancelDiscovery(). This will free up valuable resources in the application and it also frees up bandwidth that will be needed to send information to the module.

Since the devices are now aware of each other, the next step is connecting the two devices. In the Bluetooth realm, a client-server relationship must be initiated, this is also sometimes referred to as a master-slave relationship. In this case, the Android device will be the master, and the Bluetooth module will be the slave. This implementation is done by having the Device open up a server socket and the Module will initiate the connection. There needs to be a unique code that is shared between each of the devices and this is called the UUID. It is a 128 bit string that is used to uniquely identify the device and module. The UUID that was used was `00001101-0000-1000-8000-00805F9B34FB`.

Now that there is a MAC Address associated, a new class will be created to make the connection between the Android device and the Bluetooth Module. The developers need to now set up a new BluetoothServerSocket and this will be done by calling the listenUsingRfcommWithServiceRecord(String, UUID) method. The UUIDs must match up between the two devices. Then the accept() method which will accept the connection only from the device that matches. Then the close() method will close off the socket that was opened to make this connection.

The Main Activity will be used and that is where the developers will implement the accept() and close() methods, because while the application is pairing, there can be no other activity on the application. The following code will also be added to the onCreate() method in the main function which will officially have our application connected.

This will tell the Android device which socket is open for communication and it will start the connection thread. Next the developers created a new thread that is completely dedicated to the transfer of data. This is one of the most crucial pieces of the project. This thread will take in a BluetoothSocket as a parameter and it creates an InputStream and OutputStream. These methods are called getInputStream() and getOutputStream(). This class will also have to write data, there is also an option to read data but since the project will only be communicating from the Device to the Bluetooth module there is no need to worry about the read portion. This read method is implemented by read(byte[]) which transfers data in the form of an array.

Now the connection should be established and the developers should be able to send the string to the Bluetooth module and ultimately the ATMega328. This string will have the note name that was pressed and the instrument that the user chooses.

6.0 Final Coding Plan

The final coding plan took what was in the Android application and sent a string to the Bluetooth Module for each note that was pressed. The ATMega328 was able to determine from that string whether or not the user wanted to play an analog or digital note. Analog notes sent control voltages to the VCO, while digital notes were generated using a wavetable for each instrument, an ADSR envelope and a note value. Each of these note values were stored into a music array with each note having a corresponding index. These notes are played using a music buffer that locates the notes in the array. Each note is hardcoded to play for the length of an eighth note. The analog notes are set up in such a way that the control voltage will be sent as long as the key on the android application is pressed. When the key is released, then the application will send an additional character to turn off the voltage and set the value to zero. This is detailed in Figure 6.0.1 below.

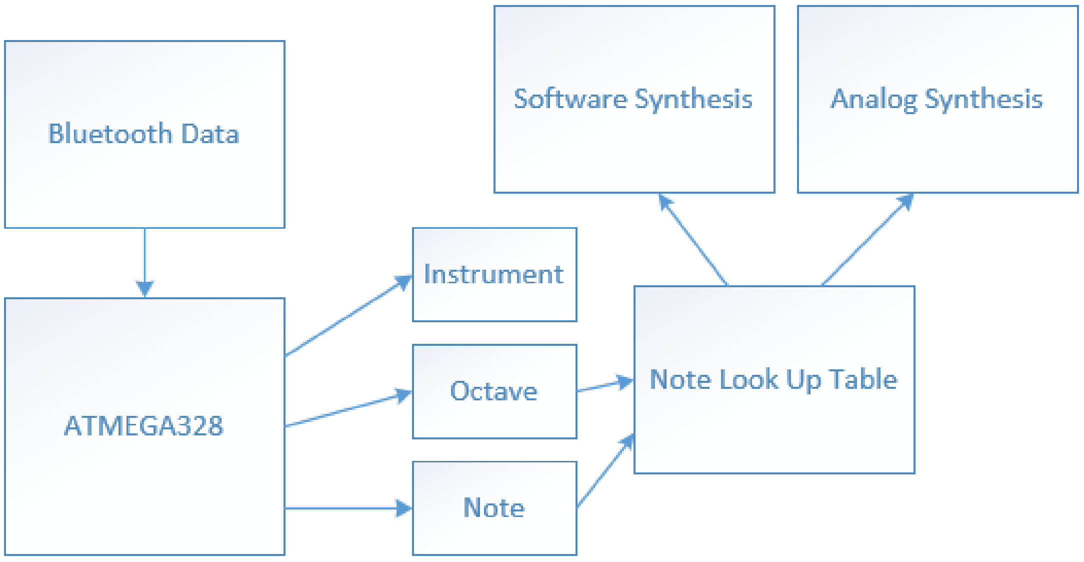


Figure 6.0.1: A diagram mapping the final coding plan. Image owned by the Hybrid Synthesizer team

Testing of the code occurred while it was being written in the IDEs, and it was all flashed onto the devices for further testing. The goal is to begin the integration process in small steps. First the developers sent and received strings using the android device and the Bluetooth module. After that, the developers sent the string to the ATMega328 using the terminal to format the I/O. The developers then integrated the entire software process and perform testing on the whole software system. Finally, we will output the control voltages to the VCO hardware and this will create the Hybrid Synthesizer.

6.1 PCB Vendor and Assembly

The schematics for the Voltage Controlled Oscillator (Figure 6.1.1) and the Power Supply Unit (Figure 6.1.2) of the Analog components were laid out onto PCB format and printed into circuit boards. Before any assembly, it was appropriate to check the board for over etching, shorts between tracks, or any other etching faults Adding on the diodes, electrolytic capacitors, LEDs, and regulators had to be checked for polarity. The PCB designs, confirmed from private parties, will be reasonable representation of the level of design necessary for VCO and Power Supply PCB boards for the Hybrid Synthesizer.

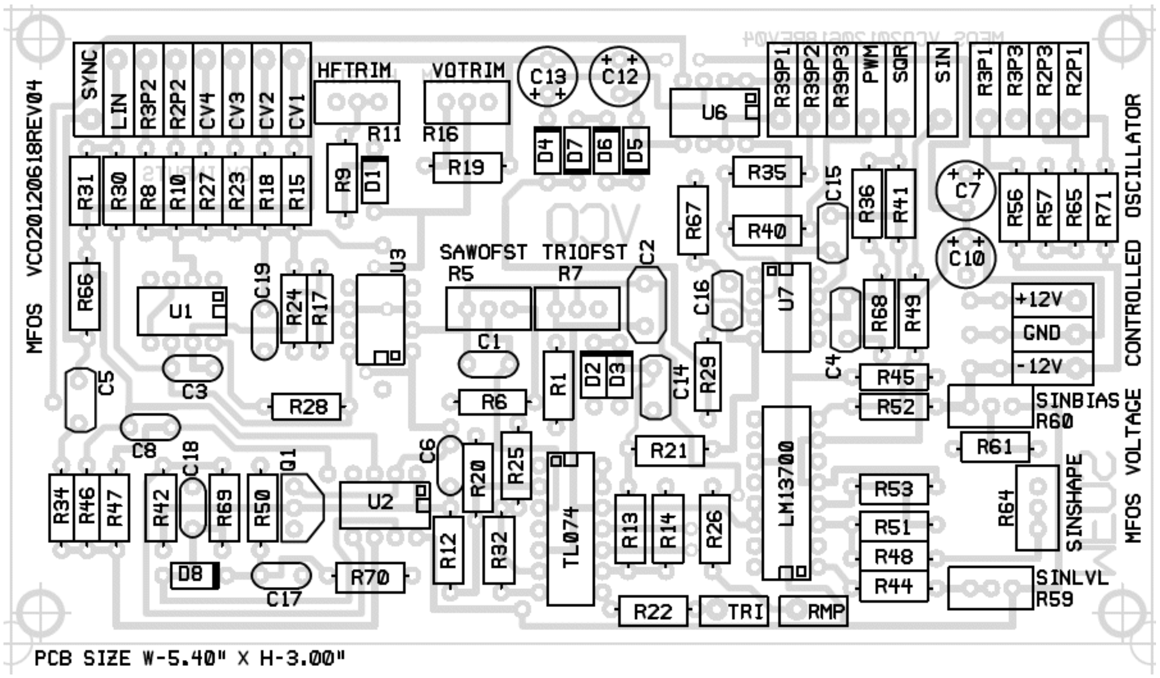

MFOS VCO20120618REV04
MFOS VOLTAGE CONTROLLED OSCILLATOR
PCB SIZE W-5.40" X H-3.00"

Figure 6.1.1: VCO PCB Designator View. Image owned by Ray Wilson, reprinted with displayed permission

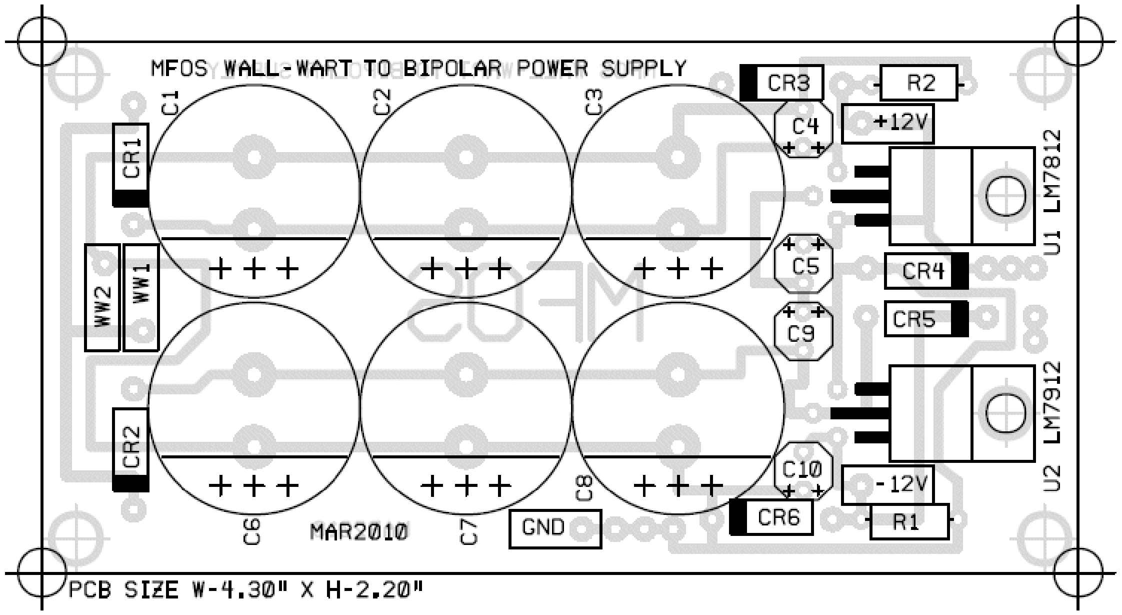

MFOS WALL-WART TO BIPOLAR POWER SUPPLY
MAR2010
PCB SIZE W-4.30" X H-2.20"

Figure 6.1.2: Power Supply PCB Designator View. Image owned by Ray Wilson, reprinted with displayed permission

7.0 Project Prototype Testing

The project underwent testing as it was developed, in order to assure quality work and that the designs will meet the project goals. Because the system is inherently modular, each divisible module was individually tested to ensure its functionality is sound. Then, once each module was verified, the design team moved on to the second stage of testing.

In the second stage of testing, the design team began to iteratively integrate the modules with each other, starting at their sources. These modules were then retested to ensure their expected functionality and confirm successful integration. Then once each subsystem was fully integrated with itself (all analog modules are running successfully with each other, all stages of software synthesis are functioning properly), another iteration of integration testing were done between the hardware systems. These tests isolated any problems with hardware communication from the ATMega to the analog board and the android device to the ATMega.

Once all subsystems have been integrated and are functioning properly with each other, a series of system tests will be performed to ensure the product functions as the user would intend it to. This is primarily to ensure the interfaces are intuitive and easy to use, as well as gives a more natural picture of the device functioning in a randomized environment, ensuring its stability and robustness.

7.1 Hardware Testing Environment

The analog components of this design will need to be robust and effective, as repair on the level of the circuit board is well out of reach for the average consumer. As a result, great care must be taken to ensure each module functions as intended independently and when put together. To ensure this, the hardware tests described in the proceeding sections will require the following peripherals:

- Breadboard for part placement and real-time adjustment
- Oscilloscope for monitoring output signals and waveforms
- Spectrum Analyzer for monitoring output signals and waveforms
- Multimeter for circuit troubleshooting should any components be performing or faulting in an unexpected fashion
- Variable Power Supply for supplying any IC’s with power and providing DC control signals
- Function generator for simulating oscillation inputs on modules other than the oscillators

Because this is a musical instrument, it is assumed that it will be kept within a “normal” range of operating temperatures and will not move around significantly while in use. While significant motion should not be a problem, as the only mechanical components within this design are potentiometers, the designers will assume that the device will not be used in scenarios where the synthesizer would be bumped around for more than a few moments or dropped.

7.2 Hardware Specific Testing

The testing for the major hardware components can be seen below in Tables 7.2.1, 7.2.2 and 7.2.3.

Table 7.2.1: Oscillators Testing

Step	**Procedure**	**Expected Results**
Waveform Generation	Power oscillators via power supply Apply arbitrary control voltage to input of oscillator circuit Monitor output of circuit via oscilloscope	Oscillator outputs stable waveform of appropriate shape, keeping a consistent resonating frequency Repeat for all oscillators
Pitch Modulation	Power oscillators via power supply Apply a series of different control voltages, each for a period of at least three seconds Monitor output of circuit via oscilloscope	Oscillator outputs waveforms at corresponding resonating frequencies, with minimal noise or distortion between pitch shifts Repeat for all oscillators
Duty Cycle Modulation	Apply arbitrary control voltage to input of oscillator circuit If applicable, slowly alter duty cycle of oscillator over governing potentiometer’s entire range of motion Monitor output of circuit via oscilloscope	Duty cycle for applicable waveforms translates smoothly in real time alongside manual adjustment

Table 7.2.2: Mixer Testing

Step	**Procedure**	**Expected Results**
Check Gain	Apply sine wave to one channel input via function generator Slowly alter gain of mixer channel over governing potentiometer's entire range of motion Monitor output of circuit via oscilloscope	Output waveform goes through expected gain and is amplified or diminished by an expected amount Repeat for all channels
Check Relative Gain	Apply sine waves of unique frequencies to all channel inputs simultaneously via function generator Slowly alter gain of mixer channel over governing potentiometer's entire range of motion Monitor output of circuit via oscilloscope and spectrum analyzer	Output waveform demonstrates appropriate levels of each wave present in the overall waveform based on the adjustments made via the mixing knobs

Table 7.2.3: Audio Out Testing

Step	**Procedure**	**Expected Results**
Verify Signal Out	Apply sine wave to circuit Monitor output jack via oscilloscope	Applied waveform is passed through unaltered
Audio Verification	Apply sine wave to circuit Plug cable into output jack and into speaker/headphones Monitor output jack via oscilloscope and use tuner to verify frequency of audible signal	Applied waveform is passed through and reproduced in the speaker unaltered

The above tests can be performed iteratively. Once each module has been deemed functional via these tests, they should be performed again from the beginning, except substituting any power supplies or function generators with the preceding circuit modules. Then each test will be run through again to ensure the modules behave as expected together. Once all modules have been verified in this manner, running the tests a third time with the analog portion of the design receiving control voltages from the ATMega will be required, to verify the analog circuitry will properly respond to commands the android application gives via the ATMega.

7.3 Software Test Environment

7.3.1 Testing on the Android Device

The Android application is a crucial part of the Synthesizer. Testing will be happening throughout the course of the creation of the application, to ensure a high quality product. Whenever a new activity is introduced to the interface there will be a small unit test. There will be regression tests to make sure that when new functionality is added, the existing functionality does not break.

The testing environment will be performed using an Android device and the Eclipse Software running the Android IDE. The Android IDE comes with an emulator called the Android Virtual Device. It is customizable to different screen sizes, such as a phone, tablet and even a screen size for Android Wear. This is useful if you would like to simulate how the application would look, without having to purchase the different devices.

Usually, developers try to stay away from using the Android Virtual Device unless it is absolutely necessary because loading the code on the virtual device can take a very long time. Fortunately, there is a workaround and the IDE will allow the developer to upload the code to a device directly. This will be especially useful when we are testing the Bluetooth functionality because the Virtual Device does not have support to test Bluetooth. Once the application is loaded onto the Android Device the developers can test the application user interface along with the discovery, pairing and communication.

This application will be able to be run on any device that is running Android JellyBean all the way through KitKat. In the development environment, the developers will load the code onto the device by clicking the Run button and then choosing Run as Android Application. After a few seconds the application will install on the device and it will show up as an icon under the application screen. From here the developers can open the application and go through the test procedures that are listed in section 7.4.

7.3.2 Testing on the ATMega328

Testing of the Arduino began with the IDE Arduino Software and compiling the initial source code for pulse width modulation to a single pin. By setting a specific PWM clocking frequency and using the function analogWrite(), which accesses Pulse Width Modulation within the Arduino, a pulsed output was set for a specifying a pin. This output voltage between 0 and 5 was obtained from the following formula.

$$Output\ PWM\ Voltage = (Total\ Voltage\ Available) * \left(\frac{Number\ of\ bits}{Total\ Resolution\ bits}\right)$$

Rearranging this formula appropriately, the "Number of bits" required to put into the analogWrite function was found and an example of source code looks like this:

$$analogWrite(\ voltpin, 128);Which\ creates\ a\ 2.5\ volt\ PWM\ outpin\ on\ the\ voltpin$$

By testing various values, the range of 24 values was reached and applied to a full 2 octaves or 24 notes that was sent to a high pass filter and into the voltage controlled oscillator.

To test the notes of the instruments, the Arduino had to play a specified note, within two octaves, and output the audio note. After the lua52 music was built and stored into program memory, each note was coded to correspond with a specific character in ASCII. To test this, a set of if-else if statements were created to access the program memory built in music array and to choose the desired note. These notes were sent through a high order filter which connects to an audio output jack. Playing the character would output the notes appropriately and served as a testing platform to hear if each halfstep increased appropriately with the correct octave or not.

The other form of testing required Serial commands, a terminal (Tera Terminal, hyperterminal or the built in Arduino Terminal) and aided in making sure that any serial data being sent successfully made it through the Bluetooth module and into the Arduino. This test was completed by using a breadboard which had a 1:2 ratio voltage divider placed on the rx pin of the Bluetooth module to make sure the pin was not destroyed. From there on, the Rx open of the Bluetooth went to match the Tx pin of the Arduino, the Tx of the Bluetooth to the Rx of the Arduino, the Voltage In pin went to the 5 Volt pin of the Arduino and finally the device ground was sent to a ground pin of the Arduino. Once powered on, the hyperterminal read in data over initially over the Serial Bus to make sure the code was storing variables correctly and reading them back on the screen. Once that feat was achieved, the Bluetooth Incoming Com Port was used to read in data and was tested in a similar manner.

After Data was read in to the Arduino, the control voltages were turned on or an instrument note was played. The audio output jack served as a central connection to the circuitry and the audio speakers. Whether a note was output with sound or not demonstrated if the devices were working appropriately and also tested whether the right pitch, timbre, and volume were correct.

The way to test the audio output from the Arduino is to double check the connection between the audio output node and a headphone or loudspeaker. Double checking component values, jumpers, and Arduino wires are necessary, especially the Ground Wires near the 100nF capacitor.

Testing of the ATMega328 for all digital sound synthesis will begin with opening the Arduino IDE and opening our project. This file has multiple C files and header files so keeping it clean is going to be important. In order to put the code onto the board the team will need to plug the ATMega328 board into the USB port of the PC using a micro to USB cord. Now, click the debug button which is the same as the run button in most cases. It will take the user to another screen and from there click compile and run. It usually takes a few minutes to fully transfer the code to the board. Once it is on the board we can use the test cases in section 7.4 to test the functionality of the board.

7.4 Software Specific Testing

The below software tests detailed in Tables 7.4.1 – 7.4.7 were used to verify functionality of all software related implementation.

Table 7.4.1: Loading the Application onto the phone

This is to verify the application loads in the time expected and with the proper screens.

Step	Test Description
1	Open the Android IDE with the Hybrid Synthesizer Source Code Loaded
2	Install the Application on an Android Tablet
3	See that the icon appears on the Device's home screen
4	Open the application and see that the application has installed properly

Table 7.4.2: Application User Interface Functionality

This is to ensure the user interface is behaving as expected.

1	Open the application
2	See that the home activity loads
3	If Bluetooth does not turn on, a message should display telling the user to change their Bluetooth Settings
4	The display should automatically rotate into landscape mode
5	A piano interface should appear
6	When a key is pressed a note should sound
7	The key should also change color
8	See that you also have the ability to change instruments
9	After switching to a different instrument, see if the timbre of the note changes

Table 7.4.3: Pairing With the Bluetooth Module

This is to ensure wireless communication is connecting and performing as expected.

1	Open the application
2	Make sure that Bluetooth on the device is turned on
3	Make sure that the Bluetooth module has 3.3v of power in order to turn it on and make it discoverable
4	Find the device using the Bluetooth setting on the phone
5	Pair the device and connect using the correct passcode
6	When the devices are paired, an LED should display signaling that the devices are ready to communicate

Table 7.4.4: Testing the Bluetooth Range

The purpose of this test is to determine how far you can take the android tablet before the Bluetooth disconnects.

1	Pair the Bluetooth Module with the Android Device
2	Measure out 100m or approximately 300 feet from the ATMega328/Bluetooth module
3	Take the Android Device to that location and see if the pairing disconnects
4	The pairing will still be connected if the LED remains lit
5	Go until the LED turns off and measure the distance

Table 7.4.5: Testing UART using RealTerm

The purpose of this test is to make sure that the string is sending from the Android Application to the Bluetooth module and is being utilized by the UART that is integrated with the MSP430. This will ensure that the MSP430 code is ready for testing.

1	Open the terminal program RealTerm
2	Attach the Bluetooth module to a power supply
3	Pair the android device with the Bluetooth Module
4	Send a Note using the Android application graphical user interface
5	The Terminal program will receive the string that contains the note name
6	It should display the note on the terminal

Table 7.4.6: Testing Sending Musical Notes

This is to verify note messages are being sent, decoded, and resulting in the correct outputs on the ATMega.

1	Run the android code on the Android Device
2	Open up the Hyperterminal on the computer
3	Plug the ATMega328 into the computer
4	Run the ATMega328 code in Code Composer Studio
5	Make sure the output is set to go to the Terminal program
6	Press a note on the keyboard
7	A string should be sent to the ATMega328 through the Bluetooth module
8	Check the output on the Terminal program
9	It should match the Android String

Table 7.4.7: Testing the ATMega Code

This is to test the throughput of the signals being sent and ensuring that an audio signal is being output for every input given by the user.

1	Compile and run the C code in Code composer studio
2	Make sure that the LED on the board is flashing
3	Press a note on the keyboard
4	A note should output to the speakers

Test 8: Testing the Entire Digital System

This is to verify the entire digital system runs properly as a full pipeline.

1	Run the android code on the Android Device
2	Plug the ATMega328 into the computer
3	Run the ATMega328 code in Code Composer Studio
4	Press a note on the keyboard
5	The keyboard note should change color
6	The keyboard note should provide vibrational feedback
7	The Bluetooth module should blink showing that there is a data transfer occurring
8	If an instrument is chosen, the note should go directly to the output speakers
9	If the regular piano sound is chosen, the output should go to the synthesizer

8.0 Administrative Content

8.1 Milestones

This project is broken down and quantified into a series of production goals and milestones to span two semesters. The milestones were used to gauge and inform the team of needed progress, as well as prepare the team for future workflow alongside classes and other responsibilities.

8.1.1 Research Phase

Most of the first semester of our Senior Design Process was allocated to researching and learning about all of the different aspects that go into building a hybrid synthesizer. The design team researched the algorithms and hardware needed to create a fully functional device. Frequently, groups will discover that certain aspects of their original design will not work or the parts are out of the budget scope. It is ideal to find these things out in advance so that the team does not purchase parts that are unnecessary.

The milestones for this semester are set by the group and also around our deadlines given to us by Dr. Richie. Our Initial Proposal was due on September 25th, 2014. This assignment was an overview of the project and a rough milestone and budget analysis. This analysis gave the team a starting point so after this, writing officially began.

8.1.2 Writing Phase

The Table of Contents assignment was due on October 9th, 2014. This assignment gave us a starting point for our paper and after that is when the writing portion began. Each group member took sections that pertained to the portion that they were responsible for and wrote on their specific topics. The team broke it out into 30 pages each so that each individual member could meet their Senior Design 1 requirement. The design team also conducted weekly meetings to discuss our findings and see how the research compared to the other group members. The team members broke our page count milestones into three different deadlines.

- October 15th ,2014 – 10 pages
- November 9th , 2014 – 20 pages
- November 21st , 2014 – 30 pages

8.1.3 Editing Phase

The editing phase began on November 22nd and the process took one week. This is when the design team added any final documentation and made sure that everything was cited and formatted correctly. The designers also made sure that all of the image permissions had been acquired or else the image was removed. The final document was due on December 4th, 2014. It was printed and bound so the editing was completed before to ensure that everything was turned in on time.

8.1.3.1 Senior Design 1 Timeline - Fall 2014

Below is a breakdown of the schedule and timeline for Senior Design 1.

	Task Name	Sept	Oct	Nov	Dec	Jan	Feb	Mar	Apr
1	Finalize Project Idea	🟩							
2	Get Project Approved	🟩							
3	Find a Sponsor	🟩							
4	Research analog waveforms	🟩	🟩	🟩					
5	Research Android Application	🟩	🟩	🟩					
6	Research Microprocessors	🟩	🟩	🟩					
7	Research Wireless Communication	🟩	🟩	🟩					
8	Research Integration between analog and digital components		🟩	🟩					
9	Research Parts and Pricing		🟩	🟩					
10	Finalize Design			🟩	🟩				
11	Write Report			🟩	🟩				

8.1.4 Senior Design 2 - Spring 2015

Senior Design 2 is where the magic happens. During this phase of the Senior Design year, the design team built the project until it was completed to the specifications and it was fully functional. The design team also completed the necessary documentation as well as the final presentation during this time. The milestone table below shows the timeline of when the group expected to complete each milestone for this phase of our design.

8.1.4.1 Final Research

The first task that was completed for Senior Design 2 was to do a design review and make sure that all of the design specifications are in order. The team reviewed all the parts before ordering to make sure that they were compatible with all of the other parts in our design. This will took place over winter break and in January. After the review was completed, this was when parts were ordered so that the team received them in with enough time to build and test.

8.1.4.2 Building Phase

The software building phase began during winter break because there are no parts involved with that. It was written using the Eclipse ADK and the application was done at the beginning of February. The ATMega software was built starting in January and finishing up in March. The analog system was built when the team received the parts. This happened in March, and some preliminary testing occurred before buying the parts in order to test the circuit designs. The Bluetooth module was integrated when the board shipped. The integration of the digital and analog systems happened in late March and it continued into April. This involves setting up the Bluetooth communication and sending data on the digital side. The PCB parts were also ordered so that they can be used in the later testing. In mid April, the design team worked on the final presentation and performed a full unit test with all of the parts integrated together.

8.1.4.3 Testing Phase

The testing phase happened at every stage of the project. The software was run multiple times on the computer before being flashed onto our different boards and devices. The analog system was tested at the beginning and throughout the project to make sure that the circuit output is displaying the correct waveforms. This is the most important step because the pieces need to work together in order to have a fully-functional design. At the end of April the Hybrid Synthesizer was fully functional and ready for demos.

8.1.4.4 Senior Design 2 Timeline

Below is a breakdown of the schedule and timeline for Senior Design 2.

	Task Name	Sept	Oct	Nov	Dec	Jan	Feb	Mar	Apr
1	Review Designs					🟦			
2	Order Parts					🟦			
3	Build Android Application					🟦			
4	Build and test Analog Circuits Designs					🟦	🟦	🟦	
5	Build and test MSP430 Code					🟦	🟦	🟦	
6	Integrate Application and Bluetooth Module						🟦	🟦	🟦
7	Combine all Modules						🟦	🟦	🟦
8	Perform Unit Testing						🟦	🟦	🟦
9	Perform System Testing						🟦	🟦	🟦
10	Finalize System Design							🟦	🟦
11	Prepare final presentation and documentation							🟦	🟦

8.2 Bill of Materials

Below is a listing of all parts and designators for the VCO (Table 8.2.1) and Power Supply (Table 8.2.2) circuits.

Table 8.2.1 VCO Parts List

Qty.	Description	Value	Designators
1	LM13700 Dual gm OpAmp	LM13700	U5
1	OP275 Dual BiFET Op Amp	OP275	U1
1	Matched Pair of Discrete 2N3904 NPN Transistors	2N3904	U3
1	TL071 Op Amp	TL071	U7
2	TL072 Dual Op Amp	TL072	U2, U6
1	TL074 Quad Op Amp	TL074	U4
1	PN4391 N Channel JFET	PN4391	Q1
8	High Speed Switching Diode	1N914	D1, D2, D3, D4, D5, D6, D7, D8
3	Linear Taper Potentiometer	100K	R2, R3, R39
2	Metal Film 1/4 Watt 1% Resistor	2K	R24, R46
2	Metal Film 1/4 Watt 1% Resistor	47K	R20, R65
1	Metal Film 1/4 Watt 1% Resistor	475 ohm	R19
1	Metal Film 1/4 Watt 1% Resistor	470K	R61
3	Metal Film 1/4 Watt 1% Resistor	4.7K	R9, R36, R52
1	Metal Film 1/4 Watt 1% Resistor	3K	R41
12	Metal Film 1/4 Watt 1% Resistor	100K	R6, R10, R15, R18, R23, R27, R29, R31, R44, R50, R66, R67
11	Metal Film 1/4 Watt 1% Resistor	10K	R1, R12, R13, R17, R25, R32, R35, R40, R42, R47, R71
1	Carbon Film 1/4 Watt 5% Resistor	10M	R68
1	Metal Film 1/4 Watt 1% Resistor	130K	R53
1	Metal Film 1/4 Watt 1% Resistor	18K	R70
5	Carbon Film 1/4 Watt 5% Resistor	1K	R22, R26, R34, R49, R51
5	Metal Film 1/4 Watt 1% Resistor	1M	R8, R28, R30, R45, R69
2	Metal Film 1/4 Watt 1% Resistor	200K	R56, R57
3	Metal Film 1/4 Watt 1% Resistor	20K	R14, R21, R48
1	Top Adjust Multi-Turn Trim Pot	100 ohm	R16
3	Top Adjust Multi-Turn Trim Pot	100K	R11, R60, R64
2	Top Adjust Multi-Turn Trim Pot	10K	R5, R7

1	Top Adjust Multi-Turn Trim Pot	1M	R59
1	Ceramic Capacitor	10pF	C4
3	Ceramic Capacitor	100pF	C3, C17, C19
5	Ceramic Capacitor	.1uF	C1, C2, C8, C15, C16
1	Ceramic Capacitor	.002uF	C14
1	Ceramic Capacitor	4.7pF	C18
1	Ceramic Capacitor	.001uF	C5
2	Electrolytic Capacitor	10uF 25V	C7, C10
1	Polystyrene Cap	.005uF	C6
2	Tantalum Capacitor	1uF 25V	C12, C13

Table 8.2.2 Power Supply Parts List

Qty.	Description	Value	Designators
1	LM7812 +12V Voltage Regulator	LM7812	U1
1	LM7912 Voltage Regulator	LM7912	U2
2	Resistor 1/4 Watt 5%	2.4K	R1, R2
6	1N4004 General Purpose Rectifier	1N4004	CR1, CR2, CR3, CR6, CR4, CR5
6	Capacitor Electrolytic	3300uF	C1, C2, C3, C6, C7, C8
4	Tantalum Electro (35V)	1uF	C4, C5, C9, C10

8.3 Standards

The Design Team's project adhered to the following standards.

8.3.1 Audio Standards

- IEC 60065 Audio, video and similar electronic apparatus – Safety requirements

This International Safety Standard applies to electronic apparatus designed to be fed from the MAINS, from a SUPPLY APPARATUS, from batteries or from REMOTE POWER FEEDING and intended for reception, generation, recording or reproduction respectively of audio, video and associated signals. This standard applies to apparatus with a rated supply voltage not exceeding 250 volts ac single phase or d.c. supply which is the case of the hybrid synthesizer’s power supply unit. The unit supplies +/- 12 volts dc and inputs 120 VAC which is in compliance of the Internation Safety Standards for Audio electronic apparatus’.

- IEC 60268-7 Ed. 3.0 en:2010: Sound system equipment - Part 7: Headphones and earphone

"IEC 60268-7:2010(E), is applicable to headphones, headsets, earphones and earsets, intended to be used on, or in, the human ear. It also applies to equipment, such as pre-amplifiers, passive networks and power supplies which form an integral part of the headphone system. It specifies the characteristics which should be included by the manufacturer in specifications, and relevant methods of measurement. It includes a classification of the different types of earphone, mainly characterized by the way in which the transducer is coupled acoustically to the ear, and a classification code which may also be used for marking, cancels and replaces the second edition published in 1996, and constitutes a technical revision. This edition contains the following changes: This standard relates to the headphone usage which connects to the audio output jack from the project. The relevant measurements are characterized in this document and help to maintain safe usage of headphones in accordance with their respective pre-amplifiers, passive networks, and power supplies like the ones in the project.

- IEC 60268-6 Ed. 1.0 b:1971: Sound system equipment. Part 6: Auxiliary passive elements.

Defines rated and normal working conditions. Specifies the essential characteristics and the relevant measuring methods. The working conditions of the project audio output, range in the millivolt with extremely low current which meets the sound equipment standard.

8.3.2 Bluetooth Standards

- IEEE Std 802.15.4a-2007: IEEE Standard for Information Technology - PART 15.4: Wireless MAC and PHY Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs): Amendment 1: Add Alternate PHY

This standard defines the protocol and compatible interconnection for data communication devices using low data rate, low power and low complexity, short-range radio frequency (RF) transmissions in a wireless personal area network (WPAN). Remarks: Amendment of IEEE Std 802.15.4-2006. Low data rate at a baud rate of 9600 is used in this project. The power source comes from the development board which is 5 volts and low complexity in the circuitry. Also, the range is well below a meter so this project falls under the supervision of this standard.

8.4 Design Constraints

The design constraints that were determined by the team members were considered for the duration of this project. Seeing as we were participating in a senior design class within an academic setting, this meant that our budget was limited as is any engineering project. This caused the team to have an economic restraint, which meant that we had to keep the cost of our project under $502, or we would have to pay for the materials out of pocket.

We also had an environmental constraint to have our device have low power. We only used a 12v wall wart and 9v battery to be able to power our entire system. Our device pulled less the 150mA overall from our wall wart.

We met ethical design constraints because we wanted to make sure that we used things that were available under an open source license, and will be providing all such material online and available for future researchers to make use of.

For health and safety, we will advise users of the Hybrid Synthesizer to not listen too loudly, as this could have a negative effect on their hearing over time. Future iterations of the project also would not have easy access to the internal components, thus protecting the user from being shocked by interacting with circuitry in an unexpected manner.

As far as manufacturability is concerned, we used readily available materials for our design, allowing it to have to possibility of being mass produced. All parts were standard through-hole design and can be easily manufactured and produced.

And lastly, for a social aspect, we designed our Hybrid Synthesizer to have a headphone output voltage of 400 mV, and this can be amplified to adapt to all hearing ranges. Additionally, the added mixing capabilities of the analog waveforms allow for a greater control of user self-expression.

8.5 Project Operation

The Hybrid Synthesizer has a handful of inputs and outputs that the user can expect to interact with. On the back of the unit the user will find a power jack and a 1/8" headphone jack. The included wall wart should be plugged in to the power jack and a pair of headphones or an auxiliary cable leading to a speaker may be plugged in to the 1/8" audio jack.

From here, the user must now start up the Hybrid Synthesizer by running the application on his or her android device. By tapping the icon, the program will begin running, and will connect automatically via Bluetooth (the user must ensure to have Bluetooth communication enabled on his or her device). Once at the main screen, the user is presented with a keyboard, an octave toggle button, and a drop down menu. The individual keys of the keyboard may be pressed to result in the Hybrid Synthesizer playing corresponding notes. The drop down menu may be used to select a different instrument. The instrument selected will be the one that plays when keys are pressed by the user. The octave toggle button is used to toggle between the two octaves the Hybrid Synthesizer can play along. The toggle button displays which octave is currently active, with the keyboard ranging from C3 to C5.

Should the user desire at any time to complete his or her use of the Hybrid Synthesizer, all he or she has to do is close down the app on his or her android device and unplug the Hybrid Synthesizer’s wall wart from the wall.

Appendix A - Works Cited

BU-106: Primary Batteries. (n.d.). Retrieved from http://batteryuniversity.com/learn/article/primary_batteries

CC256x Beginners Guide (2014, September 17) Retrieved from http://processors.wiki.ti.com/index.php/CC256x_Beginners_Guide

Lundkvist, A. (2008, December 2). How to Build an Analog Synthesizer from scratch. Retrieved from http://www.jiisuki.net/reports/howto-build-analog-synth.pdf

Microbrute Details (n.d.). Retrieved from http://www.arturia.com/microbrute/details

Moog Sub 37 Tribute Edition Analog Synthesizer (n.d.). Retrieved from http://www.sweetwater.com/store/detail/Sub37TE

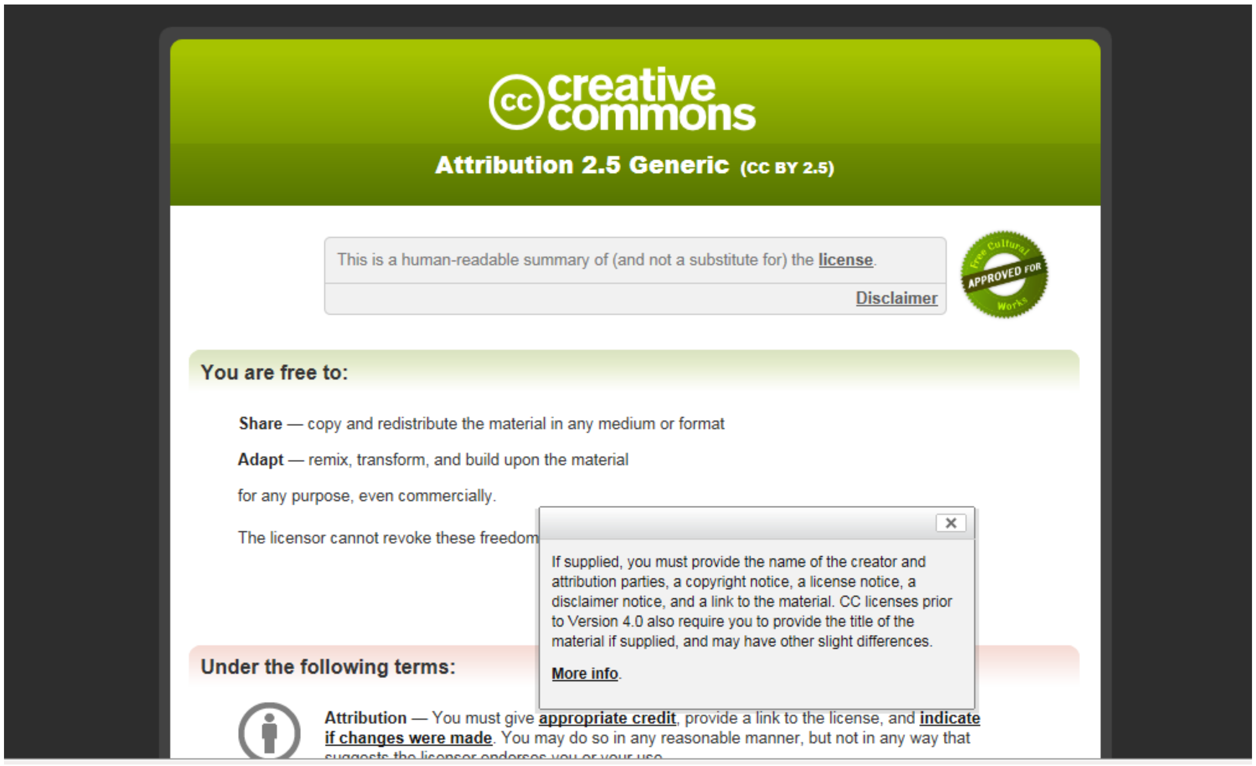
VST Little Phatty Editor for Windows (n.d.). Retrieved from http://www.moogmusic.com/products/apps/vst-little-phatty-editor-windows

Wirsing, B. (2014, March 26). *Sending and Receiving Data via Bluetooth with an Android Device.* Retrieved from http://www.egr.msu.edu/

Appendix B - Other Licenses

The information that is provided in the Android development website is available for use under the Apache 2.0 license and it can be found at http://www.apache.org/licenses/LICENSE-2.0

The information on TI's Website and Wikis is available for use under the creative commons license which can be found at http://creativecommons.org/licenses/by/2.5/



creative commons

Attribution 2.5 Generic (CC BY 2.5)

This is a human-readable summary of (and not a substitute for) the license.

Disclaimer

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material

for any purpose, even commercially.

The licensor cannot revoke these freedom

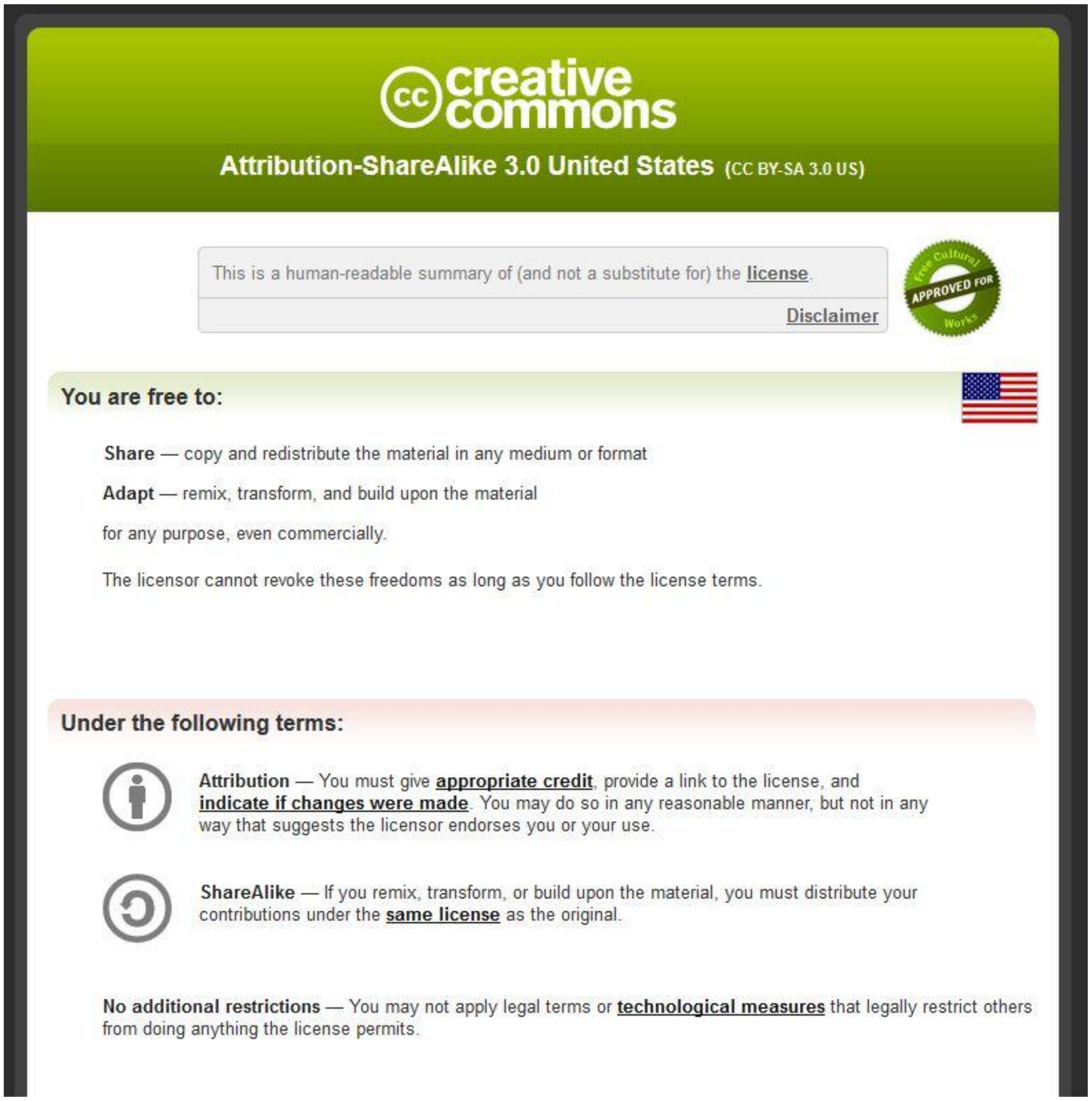
If supplied, you must provide the name of the creator and attribution parties, a copyright notice, a license notice, a disclaimer notice, and a link to the material. CC licenses prior to Version 4.0 also require you to provide the title of the material if supplied, and may have other slight differences.

More info.

Under the following terms:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that

All littleBits schematics, board layouts, and other designs are availab e for use under the creative commons share-alike 3.0 license which can be found at https://creativecommons.org/licenses/by-sa/3.0/us/



creative commons

Attribution-ShareAlike 3.0 United States (CC BY-SA 3.0 US)

This is a human-readable summary of (and not a substitute for) the **license**.

Disclaimer

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material

for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

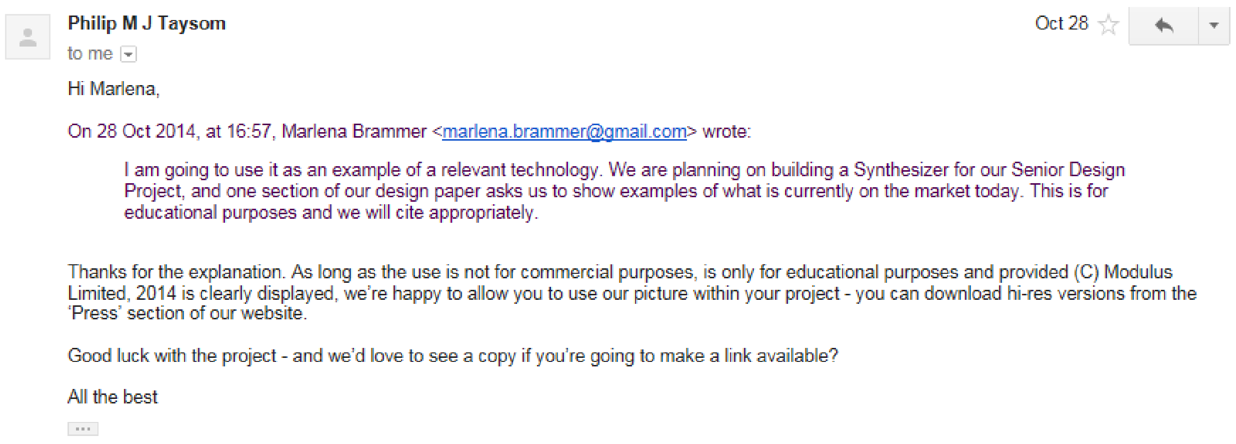
Under the following terms:

Attribution — You must give **appropriate credit**, provide a link to the license, and **indicate if changes were made**. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the **same license** as the original.

No additional restrictions — You may not apply legal terms or **technological measures** that legally restrict others from doing anything the license permits.

Appendix C - Image Permissions



Philip M J Taysom
to me

Oct 28

Hi Marlena,

On 28 Oct 2014, at 16:57, Marlena Brammer <marlena.brammer@gmail.com> wrote:

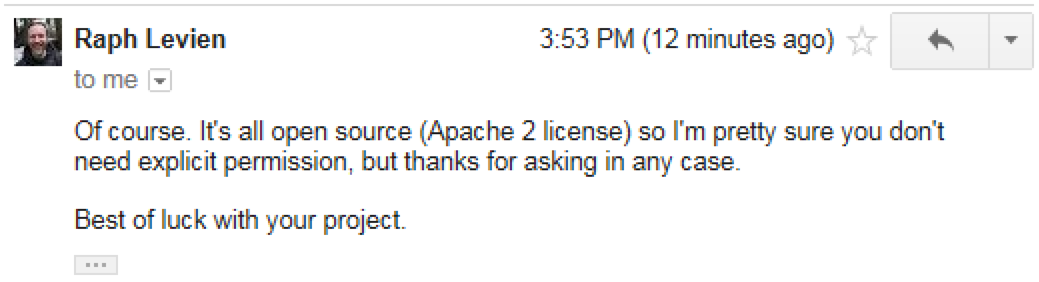
> I am going to use it as an example of a relevant technology. We are planning on building a Synthesizer for our Senior Design Project, and one section of our design paper asks us to show examples of what is currently on the market today. This is for educational purposes and we will cite appropriately.

Thanks for the explanation. As long as the use is not for commercial purposes, is only for educational purposes and provided (C) Modulus Limited, 2014 is clearly displayed, we're happy to allow you to use our picture within your project - you can download hi-res versions from the 'Press' section of our website.

Good luck with the project - and we'd love to see a copy if you're going to make a link available?

All the best

Image permission for the Modulus 002 Image



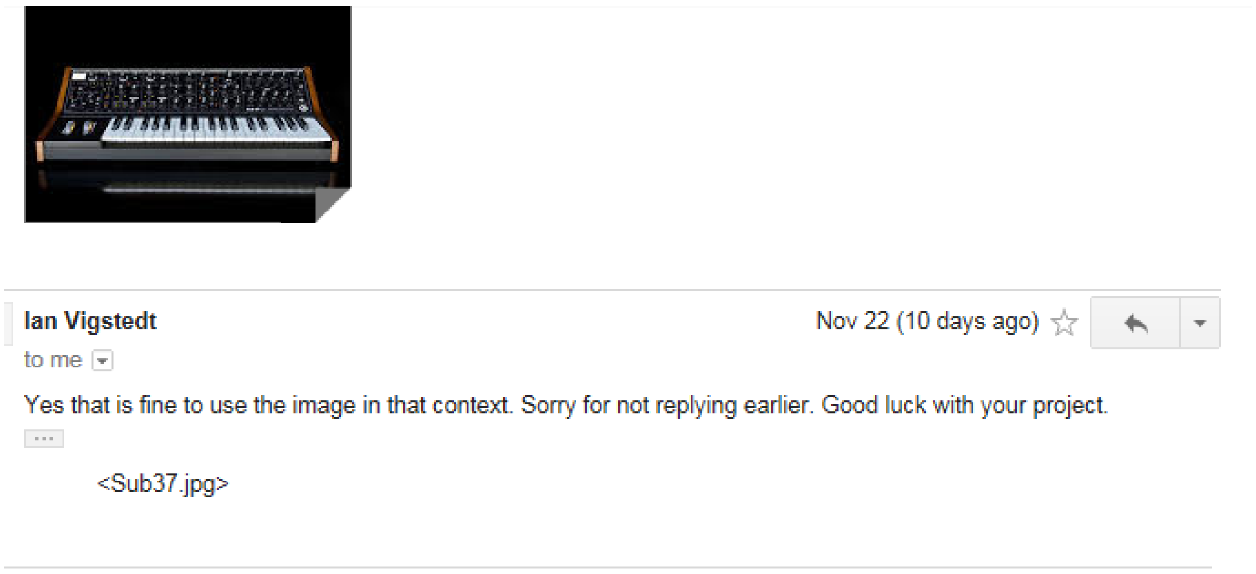
Raph Levien
to me

3:53 PM (12 minutes ago)

Of course. It's all open source (Apache 2 license) so I'm pretty sure you don't need explicit permission, but thanks for asking in any case.

Best of luck with your project.

Image permission for the Android Music Synthesizer Image



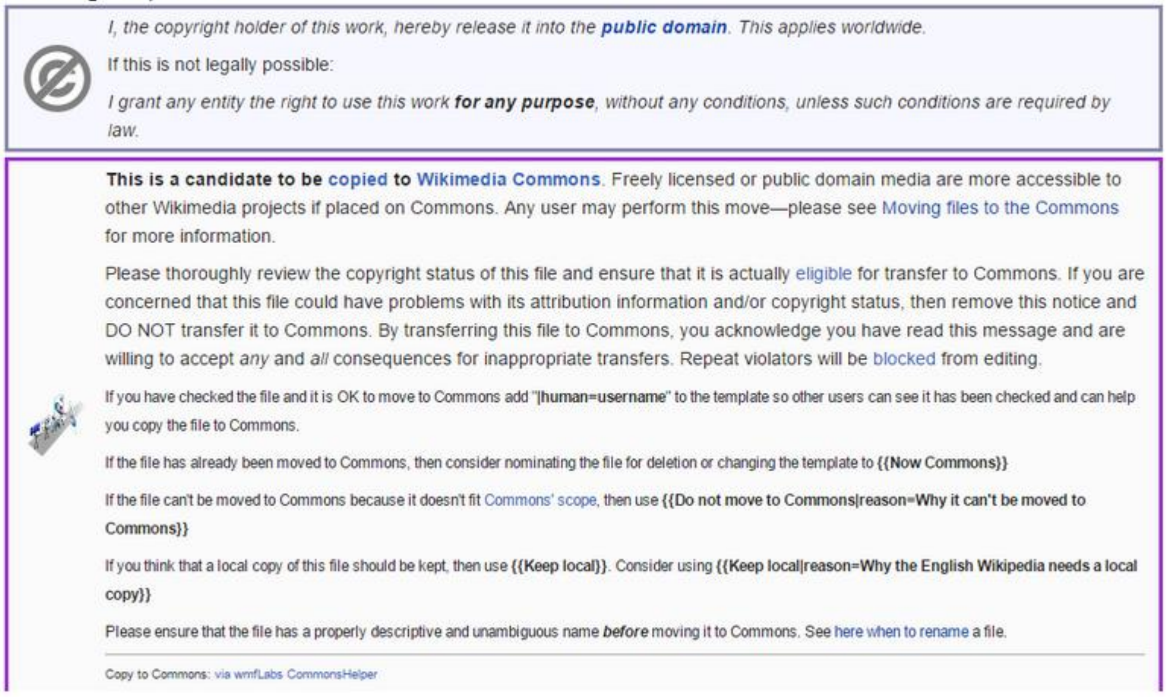
Ian Vigstedt
to me

Nov 22 (10 days ago)

Yes that is fine to use the image in that context. Sorry for not replying earlier. Good luck with your project.

<Sub37.jpg>

Image permission for the Sub 37 by Moog



I, the copyright holder of this work, hereby release it into the ***public domain****. This applies worldwide.*

If this is not legally possible:

I grant any entity the right to use this work ***for any purpose****, without any conditions, unless such conditions are required by law.*

This is a candidate to be copied to Wikimedia Commons. Freely licensed or public domain media are more accessible to other Wikimedia projects if placed on Commons. Any user may perform this move—please see Moving files to the Commons for more information.

Please thoroughly review the copyright status of this file and ensure that it is actually eligible for transfer to Commons. If you are concerned that this file could have problems with its attribution information and/or copyright status, then remove this notice and DO NOT transfer it to Commons. By transferring this file to Commons, you acknowledge you have read this message and are willing to accept *any* and *all* consequences for inappropriate transfers. Repeat violators will be blocked from editing.

If you have checked the file and it is OK to move to Commons add "**|human=username**" to the template so other users can see it has been checked and can help you copy the file to Commons.

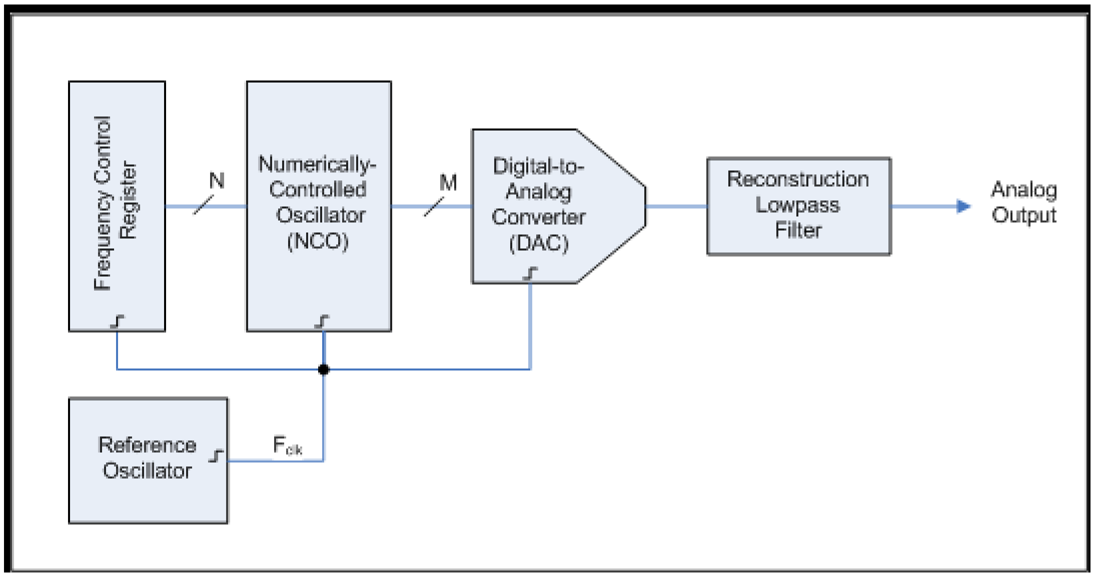
If the file has already been moved to Commons, then consider nominating the file for deletion or changing the template to **{{Now Commons}}**

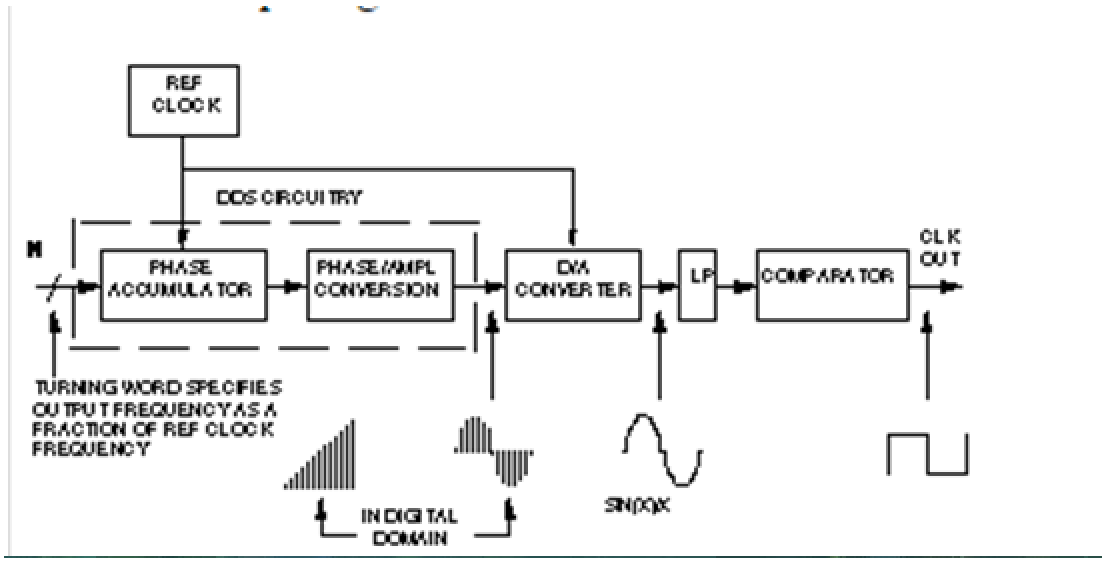
If the file can't be moved to Commons because it doesn't fit Commons' scope, then use **{{Do not move to Commons|reason=Why it can't be moved to Commons}}**

If you think that a local copy of this file should be kept, then use **{{Keep local}}**. Consider using **{{Keep local|reason=Why the English Wikipedia needs a local copy}}**

Please ensure that the file has a properly descriptive and unambiguous name ***before*** moving it to Commons. See here when to rename a file.

Copy to Commons: via wmfLabs CommonsHelper


Frequency Control Register
N
Numerically-Controlled Oscillator (NCO)
M
Digital-to-Analog Converter (DAC)
Reconstruction Lowpass Filter
Analog Output
Reference Oscillator
Fclk


REF CLOCK
DDS CIRCUITRY
N
PHASE ACCUMULATOR
PHASE/AMPL CONVERSION
D/A CONVERTER
LP
COMPARATOR
CLK OUT
TURNING WORD SPECIFIES OUTPUT FREQUENCY AS A FRACTION OF REF CLOCK FREQUENCY
IN DIGITAL DOMAIN
SIN(X)/X

To:
HJ@knights.ucf.edu;
Hi Josef,

Yes, you may use the material as requested. Please cite sources where appropriate.

Regards,

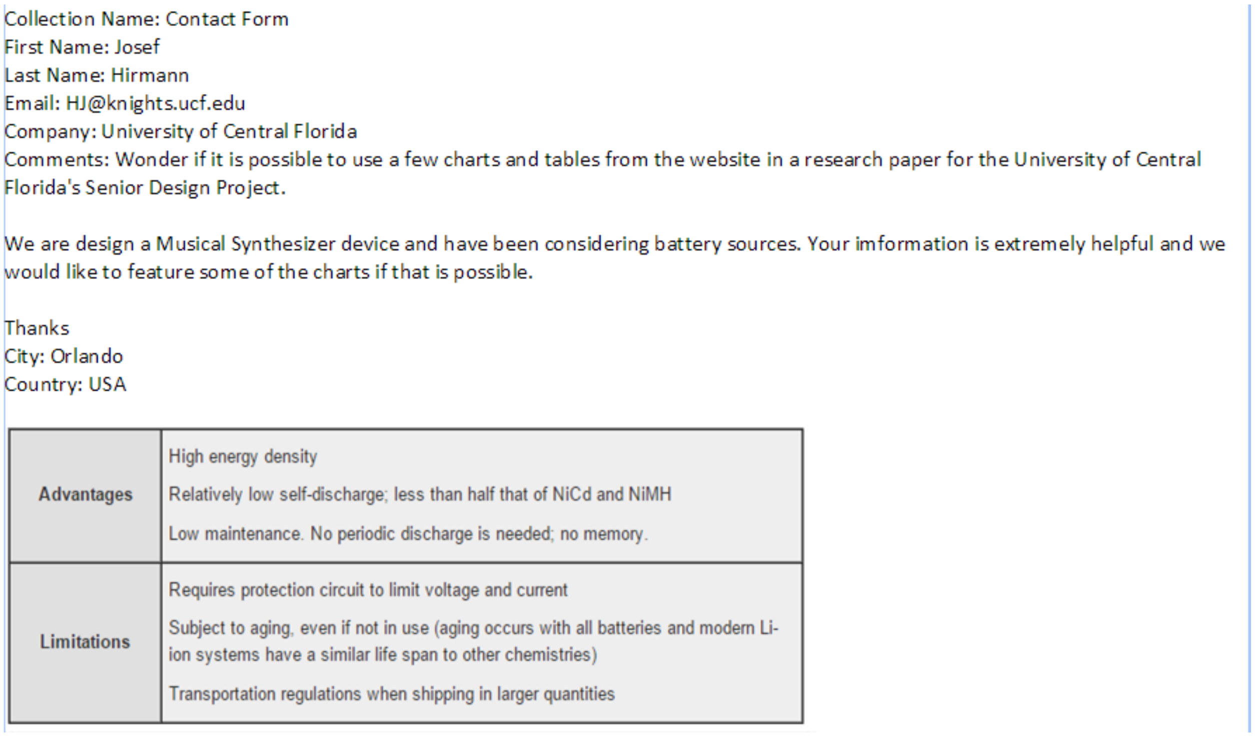
John Bradshaw - Marketing Communications Manager
Cadex Electronics Inc. | www.cadex.com
Vancouver | Minneapolis | Frankfurt
Tel: +1 604 231-7777 x319 | Toll Free: 1-800 565-5228

Follow us on Twitter: twitter.com/cadexelectronic
Join us on Facebook: facebook.com/cadexelectronics
Add us on Google+: plus.google.com/+Cadex

>>> "Battery University" <web@batteryuniversity.com> 12/1/2014 9:26 AM >>>
Someone has submitted a Battery University contact form.

Here are the details:

Entry Date: 2014-12-01 10:26 AM

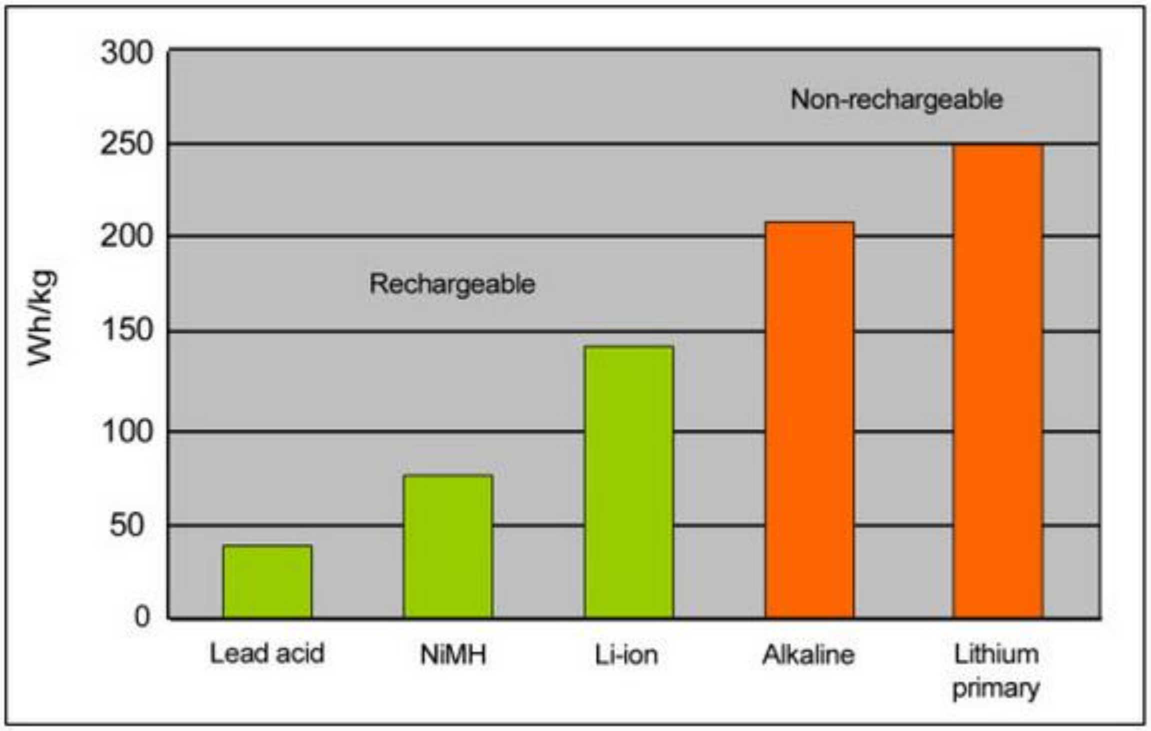


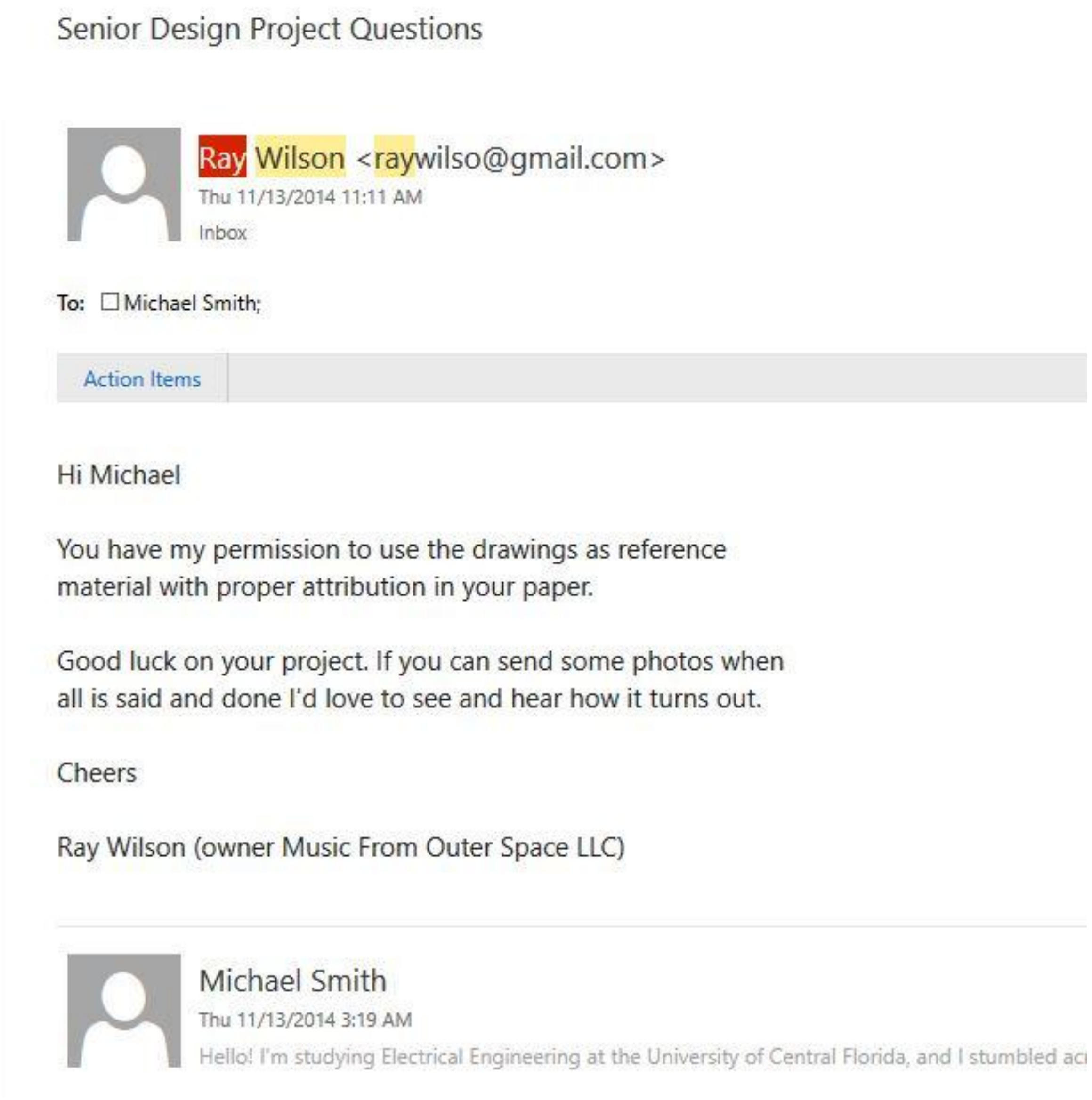
Collection Name: Contact Form
First Name: Josef
Last Name: Hirmann
Email: HJ@knights.ucf.edu
Company: University of Central Florida
Comments: Wonder if it is possible to use a few charts and tables from the website in a research paper for the University of Central Florida's Senior Design Project.

We are design a Musical Synthesizer device and have been considering battery sources. Your imformation is extremely helpful and we would like to feature some of the charts if that is possible.

Thanks
City: Orlando
Country: USA

Advantages	High energy density Relatively low self-discharge; less than half that of NiCd and NiMH Low maintenance. No periodic discharge is needed; no memory.
Limitations	Requires protection circuit to limit voltage and current Subject to aging, even if not in use (aging occurs with all batteries and modern Li-ion systems have a similar life span to other chemistries) Transportation regulations when shipping in larger quantities




Senior Design Project Questions

Ray Wilson <raywilso@gmail.com>
Thu 11/13/2014 11:11 AM
Inbox

To: ☐ Michael Smith;

Action Items

Hi Michael

You have my permission to use the drawings as reference material with proper attribution in your paper.

Good luck on your project. If you can send some photos when all is said and done I'd love to see and hear how it turns out.

Cheers

Ray Wilson (owner Music From Outer Space LLC)

Michael Smith
Thu 11/13/2014 3:19 AM
Hello! I'm studying Electrical Engineering at the University of Central Florida, and I stumbled ac