Hybrid Synthesizer

Marlena Brammer, Josef Hirmann, and Michael Smith

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — The Hybrid Synthesizer combines traditional forms of hardware and software synthesis into one package that makes use of modern technology to make a portable, easy to control , flexible synthesizer for the budding musician. An android keyboard application connects to the synthesizer wirelessly via Bluetooth, allowing the musician to play freely on stage via their phone or tablet's touchscreen capability. The user can select between traditional analog waveforms that can be mixed real-time via faceplate control knobs or choose to play one of several software simulations of well-known orchestral instruments. The Hybrid Synthesizer is lightweight, easy to handle, and plugs into the wall via a traditional wall-wart power brick, making it easy to work with in almost any setting.**

***Index Terms* — Android application, Bluetooth, control voltages, music synthesis, oscillator, pulse-width modulation.**

I. Introduction

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, however they are often prohibitively expensive to new musicians (entry level synthesizers tend to run for $1000 or more). This inspired the design team to create an affordable alternative product for a new musician.

The Hybrid Synthesizer utilizes both analog circuitry to create classic electronic sounds as well as a software synthesis infrastructure that can simulate various well-known orchestral instruments. 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 is connected to the synth wirelessly via Bluetooth, allowing the user to play real-time 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 Hybrid Synthesizer is 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 has a 1/8" headphone line out to be plugged into an audio interface for recording or a speaker/headphones for live playback.

II. Voltage Controlled Oscillator

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 (VCOs) in synthesizers commercially produced follows CV/Gate implementation, which is a standard that requires designing the oscillator to scale at a rate of 1V/Octave.

TABLE I
EXAMPLE CV/GATE VOLTAGE AND FREQUENCY VALUES

Note	**A2**	**A3**	**B3**	**C4**	**D4**	**A4**
CV	2	3	3.167	3.250	3.417	4
F (Hz)	110	220	247	261	294	440

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 Pulse Width Modulation (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, as seen in Table 1. 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. 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 Control Voltage (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

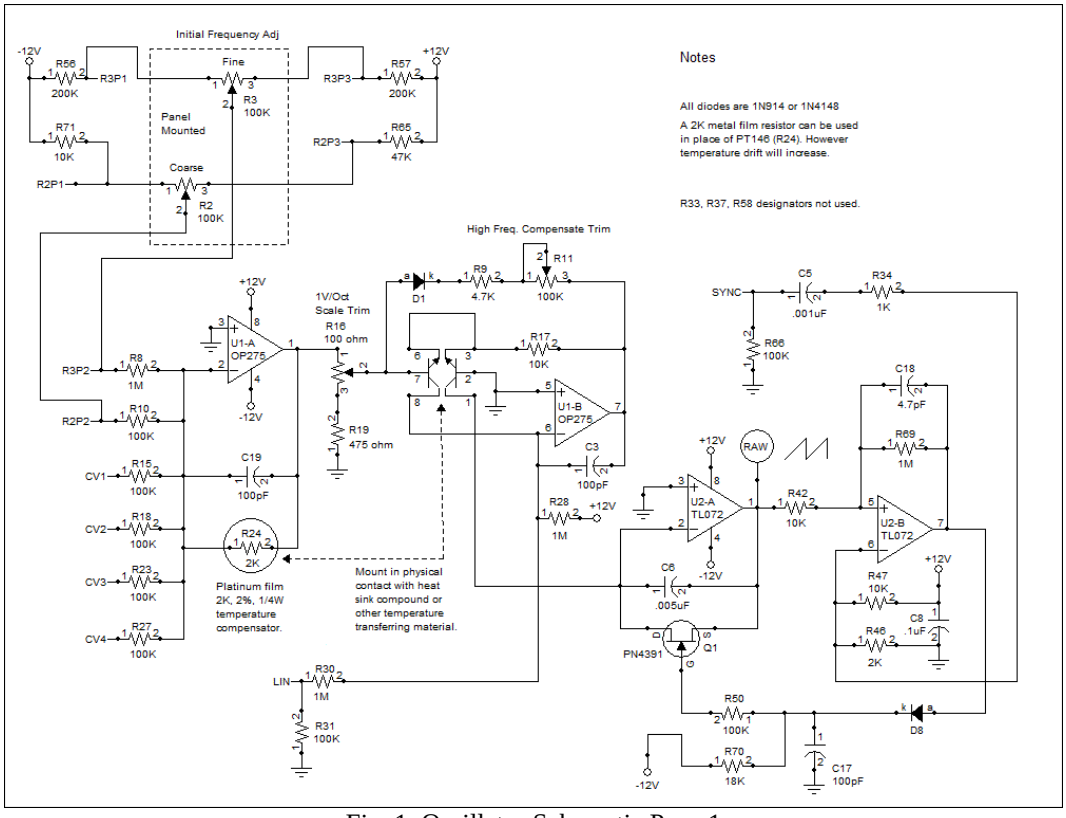

Fig. 1. Oscillator Schematic Page 1

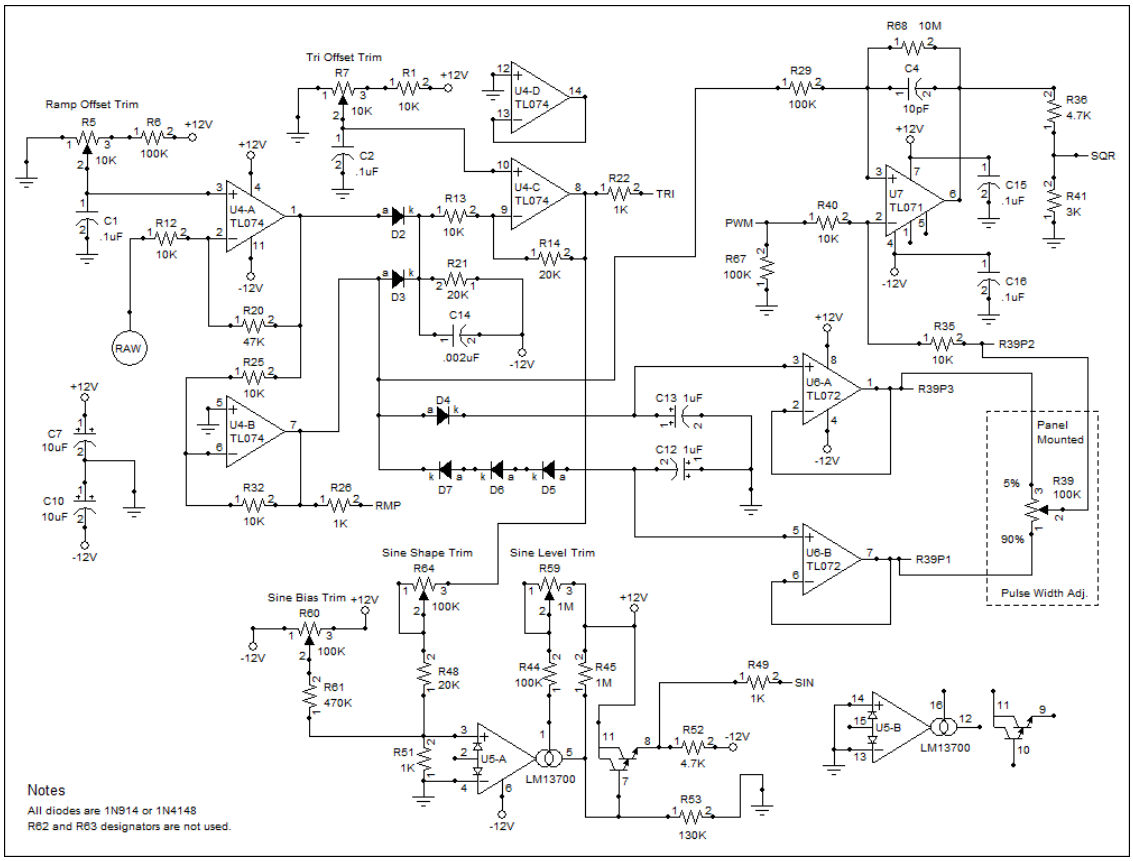

Fig. 2. Oscillator Schematic Page 2

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. 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 3.

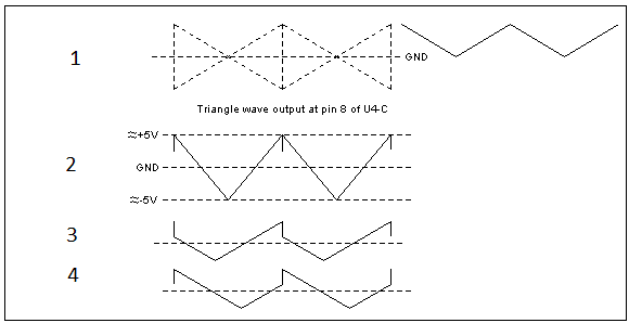

Fig. 3. Triangle Wave Generation and Adjustment

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 its 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.

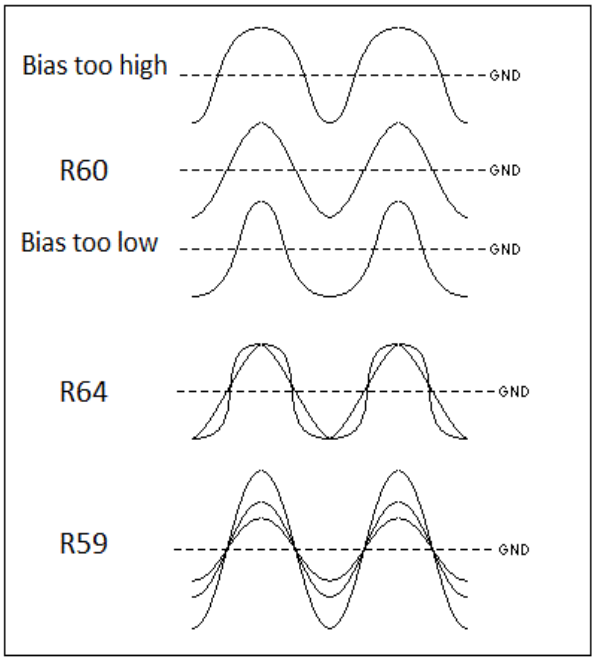

Fig. 4. Sine Wave Adjustment

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 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.

A. Parts Selection

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.

III. AUDIO MIXER

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 takes the incoming synthesizer voltages and applies a gain of $1/50^{th}$ 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 falls 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 is then 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.

IV. ANDROID APPLICATION

The application interface is how the user will input the note choices into the Hybrid Synthesizer. It is downloaded onto a Samsung Tab Pro tablet and it is accessible by clicking on an icon. The user will first see a splash screen that has our Hybrid Synthesizer logo on it. This Splash Screen is activated for five seconds, and its gives the application time to run background threads that connect the device to the Bluetooth device that is already paired. If for some reason, the Tablet's Bluetooth is not activated, the application will ask the user if they wish to turn on Bluetooth, and if the user selects 'Yes', then at that time, the Bluetooth will turn on and connect to the HC-05 Bluetooth Module.

On the android side, the connection takes place using a few steps [1]. First, the BluetoothAdapter within the android Device must be declared, along with a BluetoothSocket and BluetoothDevice and an OutputStream. In order to create a connection with the Module, you must also declare a UUID which is short for Universally Unique Identifier. The base UUID for Bluetooth communication is 00001101-0000-1000-8000-00805F9B34FB. We used this to connect the BluetoothSocket because it matched the UUID in the Bluetooth Module. In this application, we also declared the MAC Address of the HC-05 Bluetooth Module to make the pairing easier, so it is configured to connect to the HC-05 only.

The first step of creating a Bluetooth connection is to access the Android device's internal Bluetooth Adapter. The application then checks to see if Bluetooth is turned on, and if not it will prompt the user to turn on their Bluetooth Adapter. Then, the Bluetooth Adapter uses the MAC address and UUID values to find the external device. After it is finished discovering, it is time to connect. If the BluetoothSocket is not connected to anything else, it will try to connect to the HC-05. After a connection has been established, the OutputStream will be created to send the data from the Application to the module.

Fig. 5. Hybrid Synthesizer User Interface

The rest of the interface consists of a Piano Keyboard, which contains one octave of notes ranging from C to B, as seen in Fig. 5. It also has a Spinner dropdown that allows the user to select the instrument that they wish to play, and an octave button, that lets the user have a range of two octaves.

Each of these choices corresponds to a number that will be sent to the Arduino. They will be sent in a comma-delimited string that will allow the Arduino to use the parseInt() function to distinguish the different variables. The instruments will each be assigned a number for identification by the microcontroller. Analog Waveforms are assigned to 1, with piano, violin, and flute assigned to

2, 3, and 4 respectively. The octaves will be either 1 or 2, and the notes will be listed 1-12.

V. BLUETOOTH COMMUNICATION

The Android Application on the Tablet will connect to the microcontroller using Bluetooth communication. The Bluetooth Module that the design team chose is the HC-05. It is compatible with the Arduino, and after further testing, it has very fast transmission from the Android Application, so latency at this step wouldn't be a problem.

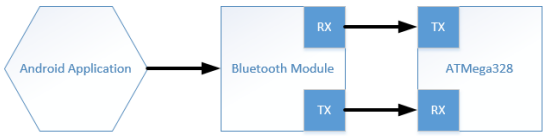

Fig. 6. Bluetooth Communication Flow

The Bluetooth module is connected to the board by connecting the ground, VCC, RX and TX pins to the correct pins on the Arduino, as seen in Fig. 6. The RX and TX pins must be crossed in order for communication to occur. The HC-05 must be in Slave mode in order to receive the bytes from the Android Application [2].

VI. MICROCONTROLLER DETAIL

The design team decided to use the ATMega328 Microcontroller to power our project. This was chosen because of familiarity with the Arduino framework, and it had all of the specifications that were needed.

A. Hardware

The ATMega328 microcontroller has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, and the board has a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button [3]. It has an operating voltage of 5V which was important because the design team needed that to send an accurate range of control voltages to the PCB. The chip has 32K Bytes of Flash memory, 1K Bytes of EEPROM and 2K Bytes of Flash Memory. There needed to be enough Serial Pins to take in data from the Bluetooth Module (RX and TX), enough analog output pins to power the PCB and pins that supported Pulse Width Modulation to output the software instrument voices. For the design team's prototype, they are going to power the Arduino using a 9v battery that is plugged into a battery switch clip. This will allow for easy battery replacement should it get low during use.

B. Software

The software coding in this project traverses a multitude of devices and interfaces. The successful creation of audio from the microprocessor board is dependent on the incoming Bluetooth data and the outgoing reference signals. The inner processing separates the analog section of the project from the digital wavetable synthesis section of the project by a series of flags and command. Four sections are defined within the program memory, to create the instrument sounds desired by the user. Each title is self-explanatory but for clarity purposes "Analog" will access the voltage controlled oscillator while the "Piano, Violin, and Flute" will each output their respective timbre through a filter and out of the speakers.

The software controlling this system is written in C code with multiple libraries. The microcontroller software system is based on a state machine which is separated into two main function calls with respective software subsystems. The first being the analog audio output, which connects the microprocessor board to the printed circuit board to generate and play analog voltage notes via a pulse width modulation method; the latter being the digital audio output, which also uses a pulse width modulation method in accordance with program stored wavetables multiplied onto an ADSR array and passed through a filter.

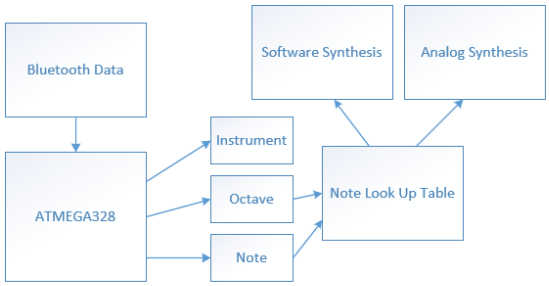

Fig. 7. Software Signal Flow

The Arduino will first receive the input from the Android application in the form of Bluetooth data, as seen in Fig. 7. This part of the code is written in C. This is done by checking to see if the HC-05 is sending any data, and if so, parsing integers from a comma delimited string. The parseInt() function is used to read each number into the corresponding variable name which will be Instrument, Octave and Note. The octave and note will be sent to a separate function, getNoteValue() where they will be matched up to their corresponding value. This function will return CVnote and InsNote variables with the value

TABLE II
CONTROL VOLTAGE VALUES FOR OCTAVES 1 & 2

Note Octave 1	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
Voltage (V)	3.16	3.24	3.32	3.40	3.48	3.56	3.64	3.72	3.80	3.88	3.96	4.04
Pin value (0-255)	161.1	165.2	169.3	173.4	177.4	181.5	185.6	189.7	193.8	197.8	201.9	206.0

Note Octave 2	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
Voltage (V)	4.12	4.20	4.28	4.36	4.44	4.52	4.60	4.68	4.76	4.84	4.92	5.0
Pin Value (0-255)	210.12	214.2	218.28	222.36	226.44	230.52	234.60	238.68	242.76	246.84	250.92	255

of the voltage that will be sent to the VCO and the note frequency.

If the user sends data to the Microcontroller using the Analog instrument it will send a '1'. Once getNotevalue() returns the CVNote, this will be passed into the ControlVoltage() function we will output this value using the Analog Out pin. The instrument data is set aside for the time being while the octave and note variables get sent in to the Control voltage function. By comparing the values, a pulse width modulation technique is utilized by altering the duty cycle of output pulse signals.

The PWM cycle can create voltages either by altering the internal timers in the microprocessor or by altering the duty cycle of the square waveform signals. The hybrid synthesizer uses the later technique by focusing on the 8 bits of resolution granted by the on board clock, and dividing the 256 binary values evenly across the 24 notes as seen in Table II. To create a control voltage from these PWM square waves, a pin directory is associated with the square wave and altered through a function called analogWrite(). After a call to analogWrite(), the pin generates a steady square wave at a frequency of 490 Hz. The following formula created the appropriate value and sets the duty cycle accordingly.

$$Pin\ Value\ (0-255) = (255\ bits * \frac{Output\ Analog\ Voltage}{Total\ Possible\ Voltage\ (5V)}) \quad (1)$$

An example of this process shown in (1) is analogWrite is (0) = 0% duty cycle, analogWrite(64) = 25% duty cycle, analogWrite(127) = 50% duty cycle , analogWrite(191) = 75% duty cycle etc. Adding a simple low-pass filter, with a capacitor of 1.0 µF and a resistor of 15 kΩ, to the output pin converts the PWM signal into a voltage corresponding to the percentage of the PWM Square waveform. In turn, this essentially provides a Digital-to-Analog converter.

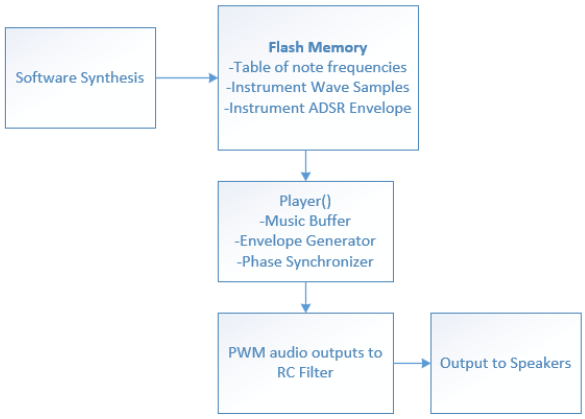

Fig. 8. Software Synthesis Signal Flow

If the user chooses a software synthesis instrument, the Arduino will read in that instrument's code along with the octave and note. It will take the InsNote variable and use the instrument name to match this data with the corresponding Instrument Wave Samples and Instrument ADSR Envelopes, as seen in Fig. 8. This will return the frequency value that will be used to play each note. All of the different instruments will be played using the different harmonics that will be generated using a stored sine table. This function will populate an array with values that will

be used to create the points along a sine function. The data will also send this data through an Envelope Generator to properly simulate the timbre of each instrument.

After this, the board will use another Pulse Width Modulation pin to output these to another Low Pass RC filter that will take the average of the duty-cycle to get a steady DC voltage output.

VII. POWER SUPPLY

For convenience and portability, the design team decided to create a power supply that would be compatible with a standard 120V to 12VAC wall wart, to allow ease of replacement should the wall wart ever fail and to protect the design team from having to design a supply from the ground up, which can be very dangerous. The VCO and subsequent circuits were designed expecting reference rails of +/- 12V DC, so a power supply would have to be designed to regulate the voltage sent from the wall wart down to a clean and reliable bipolar power supply.

The supply the design team implemented utilizes half way rectification to produce positive and negative DC voltages. The 12VAC output of the wall wart is an RMS value, meaning the actual voltage value supplied to the circuit is +/- 17VAC. Six 1N4004 general purpose rectifier diodes are used, two of which rectify the peaks and valleys as shown above. The other four are utilized as protection diodes for the regulators and protect the pins from undesirable current paths that can be caused under unusual conditions. 1µF tantalum capacitors are used near the regulator pins for operational stability.

Six hefty 3300µF capacitors are used to smooth the AC voltage supply into a consistent +/- 17VDC voltage supply, which is then fed to the regulator pair to produce a clean, stable +/- 12V of regulated DC output voltage for use as power rails on all subsequent boards.

The design team was torn between using the LM7812/7912 regulator pair and the LM78L12/79L12 regulator pair for the power supply's voltage regulators. While the oscillator circuit alone is only pulling 26mA at +12V and 28mA at -12V, the team decided to leave the power supply open to modular additions to the synthesizer and therefore selected the LM7812/7912 regulator pair for their ability to supply up to 1A of current, as opposed to the LM78L12/79L12 pair's supply of only 100 mA.

A Longprofit UL approved wall wart was selected to supply the regulation board. It takes in 120VAC and supplies 12VAC out with a maximum current draw of 500mA, which the design team decided would be more than adequate for the scope of the project.

THE DESIGN TEAM

Marlena Brammer is a senior at the University of Central Florida majoring in Computer Engineering. After graduation in Spring 2015, she will continue to work at Raytheon Company in Orlando, FL

Josef Hirmann is a senior at the University of Central Florida majoring in Electrical Engineering. After graduation in Spring 2015, he plans to pursue a career at a major firm in Orlando, FL

Michael Smith is a senior at the University of Central Florida majoring in Electrical Engineering. After graduation in Spring 2015, he will continue to work at Rockwell Collins in Melbourne, FL

ACKNOWLEDGEMENT

The authors wish to acknowledge the assistance of Dr. Samuel Richie, David Douglas, Ray Wilson, Enrico Colombini, Quality Manufacturing Services of Lake Mary, and Alan Muise.

Figs 1-4 are reprinted with permission from Ray Wilson, owner of Music From Outer Space LLC.

REFERENCES

[1] Bluetooth (n.d.) Retrieved From: http://developer.android.com/guide/topics/connectivity/bluetooth.html

[2] HC Serial Bluetooth Products (n.d.) Retrieved From: http://www.rcscomponents.kiev.ua/datasheets/hc_hc-05-user-instructions-bluetooth.pdf

[3] Arduino – ArduinoBoardUno (Arduino – ArduinoBoard Uno) (n.d.) Retrieved From: http://arduino.cc/en/Main/ArduinoBoardUno