Music Synthesizer Design with

Field-Programmable Analog Arrays (FPAAs)

ECE 4012 Senior Design Project

Electronics for Music Applications

Dr. Jennifer Hasler

Chris Walds, EE, cwalds3@gatech.edu

Harrison Zhang, EE, hzhang462@gatech.edu

Jongheon Park, CmpE, jpark802@gatech.edu

Justin Kelley, CmpE, jkelley1@gatech.edu

Kristyn DiGiovanni, CmpE, kdigiovanni3@gatech.edu

Sriram Pulavarty, EE, spulavarty3@gatech.edu

Yewon Kim, EE, ykim728@gatech.edu

Submitted

April 29th, 2020

Table of Contents

Executive Summary

Analog audio synthesizers are the dominant tools of music production. However, these devices come with downsides, such as a lack of portability, low-programmability, and high-power dissipation. The usage of a Field-Programmable Analog Array (FPAA) in a synthesizer design can help to improve on those drawbacks. As a result, the Electronics for Music Applications Team has designed an ultra-low-power, real-time analog music synthesizer using the FPAA that takes in external hardware controls to produce analog audio outputs. The system consists of a proprietary FPAA board, external hardware control devices, and a portable speaker. External push buttons can be used to control the harmonics produced by the FPAA, which uses three major blocks to synthesize the audio output to the speaker. Those blocks are an array of transmission gate selectable Voltage-Controlled Oscillators (VCOs), which generates selectable initial signals of various frequencies; a Voltage-Controlled Amplifier (VCA), which creates an amplitude envelope for each selected signal; and a Voltage Summer, which combines the selected audio signals into a final audio waveform. The outcome of this project is a prototype for a portable analog synthesizer that can produce clear audio signals that compare to those produced from common market devices but with the benefits of fitting into a small, portable package and low power utilization. Many high-end synthesizers cost more than $2,000. However, with the usage of the FPAA, this synthesizer design is projected to cost approximately $1,600.

Nomenclature

- ADSR - Attack, Decay, Sustain, and Release
- FPAA - Field-Programmable Analog Array
- LFO - Low-Frequency Oscillator
- PLD - Programmable Logic Device
- VCA - Voltage-Controlled Amplifier
- VCF - Voltage-Controlled Filter
- VCO - Voltage-Controlled Oscillator

Music Synthesizer Design w/ FPAAs

1. Introduction

The Electronics for Music Applications Team has built a prototype for an ultra-low power, real-time analog music synthesizer using the Field Programmable Analog Array (FPAA). The team is requesting $108 to develop a working prototype of this system and its peripherals.

1.1 Objective

This team's goal was to design and build a working prototype of an analog music synthesizer using the FPAA. This circuit would be able to receive many configurations of bias currents that would precisely adjust the character of a given note to produce a wide variety of sounds as audio output. Pushbuttons would be used as an input to control audio being produced while the FPAA would create and modify signals in real time, and the resulting synthesized signal would be routed to a speaker which will play the sound.

1.2 Motivation

Although synthesizers have existed for a while in many styles and forms, analog music synthesizers are preferred by many due to their ability to produce subtle distortions and variations in waveform shape, frequency, and amplitude [1]. Analog synthesizers, however, are often expensive, and sacrifice low-level programmability for predefined user configurations [2]. The FPAA provides an opportunity to eliminate these downsides while still producing an excellent sound quality that is standard in analog music synthesizers. The FPAA's programmability allows for the testing of many different circuit configurations in a cheap and efficient manner, and its ultra-low power consumption will allow the FPAA to perform all functions of a music synthesizer at a much higher energy efficiency [3]. The FPAA will allow for the development of a custom-built synthesizer with operation standards superior in many aspects to music synthesizers in the modern industry. Most high-end commercial synthesizers sell for around $2,000, but this system can be commercially developed for $1,600. This project would most benefit musicians interested in the fields of music technology and signal processing, as this product would allow for the tuning of virtually every aspect of the synthesized sound. The synthesizer's basic components could be tweaked by musical experimentalists to produce many new kinds of sounds.

1.3 Background

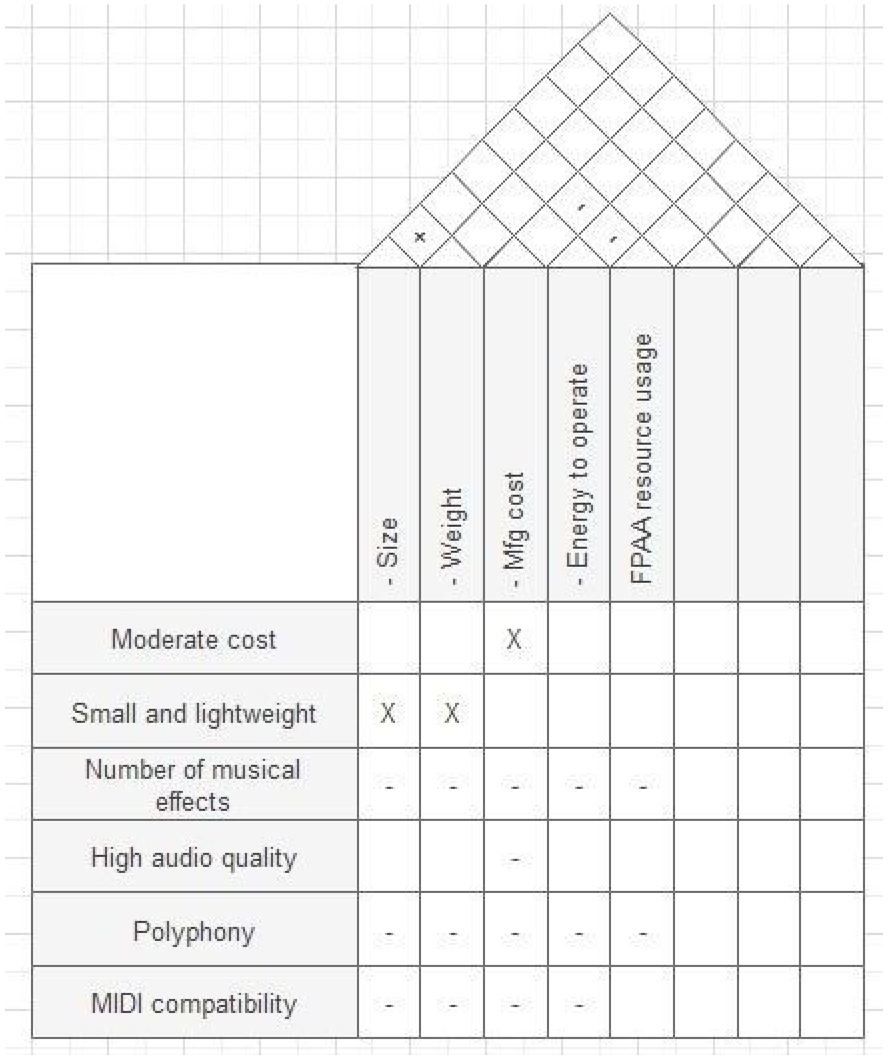
Research into analog computation has significantly increased in recent years, and one device that utilizes this technology is the FPAA. The FPAA is a system-on-chip that integrates analog and digital configurable components into a single fabric controlled by a 16-bit MSP430 microprocessor. It allows for operation at up to 1000x the energy efficiency of its digital counterparts, and applications for the device range from signal processing to neuromorphic computation [3]. A large number of input/output pins are also available for interfacing with peripherals.

An open-source toolset named RASP Tools has been developed for compiling and programming circuit designs for the FPAA using configurable block diagrams. This tool allows a user to abstract certain circuit elements and build custom subunits for operation on the device. RASP Tools comes with a number of analog and mixed-signal components in its default library [4].

Investigation into music synthesis has already begun with the FPAA, and preliminary testing has been performed on a number of synthesizer components implemented on the FPAA. In particular, a voltage-controlled oscillator, amplifier, and filter have all shown promising operation. While testing of music synthesis on the FPAA has only been conducted to show proof-of-concept, this project will realize a full implementation of the synthesizer, complete with hardware controls and an audio output [5].

2. Project Description, Customer Requirements, and Goals

The Electronics for Music Applications Team designed a monophonic music synthesizer with a compact form factor. The system was intended to consist of a synthesis engine and a microcontroller. The synthesis engine is the circuit that produces the sound and was designed by the team to be implemented on an FPAA board using the FPAA programming environment. The microcontroller processes user inputs from pushbuttons. The synthesis engine consists of an array of VCOs, a VCA, and a Voltage Summer. The array of VCOs is a bank of voltage controlled oscillators that produces sine, square, and triangle waveforms of different frequencies. These waveforms will be modified by the rest of the components. The VCA is a voltage controlled amplifier that creates the envelope of the internal signal to control volume and change certain sound characteristics. The Voltage Summer is a circuit that combines selected waveforms of different pitches and volume levels to create a single output audio signal. Once the internal signal passes through the components of the synthesis engine, the output waveform is played by a speaker. While the engine itself has not been fully put together with the microcontroller to produce a synthesizer, the design and testing of the synthesis engine's building blocks was successful.



	- Size	- Weight	- Mfg cost	- Energy to operate	FPAA resource usage			
Moderate cost			X					
Small and lightweight	X	X						
Number of musical effects	-	-	-	-	-			
High audio quality			-					
Polyphony	-	-	-	-	-			
MIDI compatibility	-	-	-	-				

Figure 1-1. QFD Diagram for the design of a FPAA synthesizer.

3. Technical Specifications

Important specifications from all major components of the project are shown below. Many specifications of this project fall in line with the product specifications. However, there are certain standards that are intended for the synthesizer system.

Table 1. Product Specifications	
Synthesizer Specifications	
Feature	**Specification**

Input-to-Sound Latency Time	0.2 seconds
Frequency Control	Yes, with VCO
Volume Control	Yes, with VCA
Envelope Generation	No
Waveform Selection	Yes, with Voltage Summer
FPAA Specifications	
Processor Speed	55 ns (Cycle Period)
SRAM Size	16k x16 (Program), 16k x 16 (Data)
Power (Active)	300 mW
Voltage	3.3 V (+/- 0.3 V)

4. Design Approach and Details

4.1 Design Approach

System Overview

The portable music synthesizer system consists of a synthesis engine programmed on a proprietary FPAA. Input control voltages are sent to the synthesizer via GPIO pins before programmed on-chip components process them and output an audio signal, which can be played by a speaker. Figure 2-1 shows a block diagram of the inputs and outputs of the music synthesizer system.

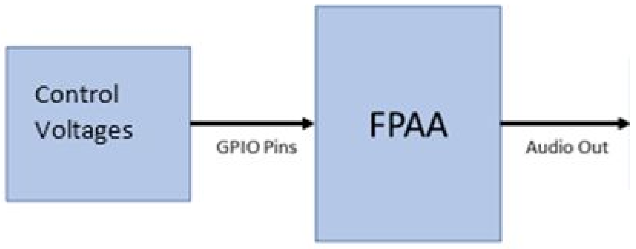


Figure 2-1. Component-level block diagram for the music synthesizer system.

FPAA Board

The FPAA board consists of a "fabric" of programmable digital and analog blocks, a 16-bit MSP430 open core processor, a register file, a series of SPI ports and GPIO pins, and a programming (DAC and ADC) infrastructure [3]. Figure 3-1 shows the layout of the FPAA SoC, while Figure 3-2 provides a photo of the FPAA die.

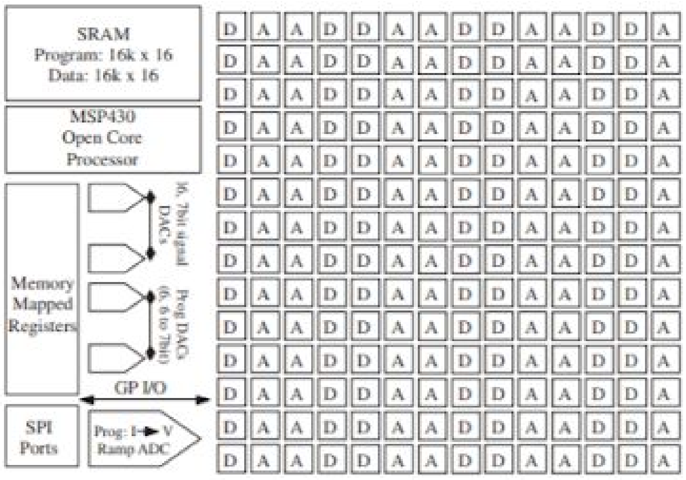


Figure 3-1. Layout of the FPAA SoC, including digital and analog block array, MSP430 open core processor, register file, SPI ports, GPIO pins, DACs, and ADCs [3].

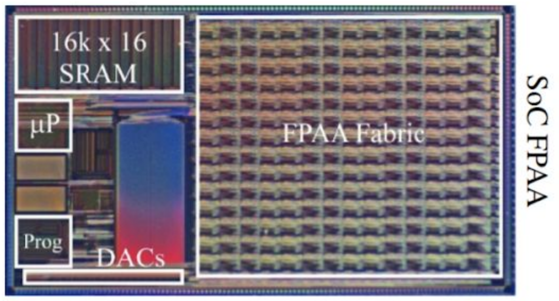


Figure 3-2. Die photo of the FPAA SoC, which measures 12 mm × 7 mm and is manufactured on a 350-nm CMOS process [3].

Music synthesis in the FPAA begins by accessing input control voltages from on-board GPIO pins. From there, the signals are used in three main voltage controlled components to generate audio signals via analog audio synthesis: an array of voltage controlled oscillators (VCOs), the voltage controlled amplifier (VCA), and the voltage summer [5]-[6]. Figure 4-1 shows how these three components interact with each other in addition to their inputs and outputs.

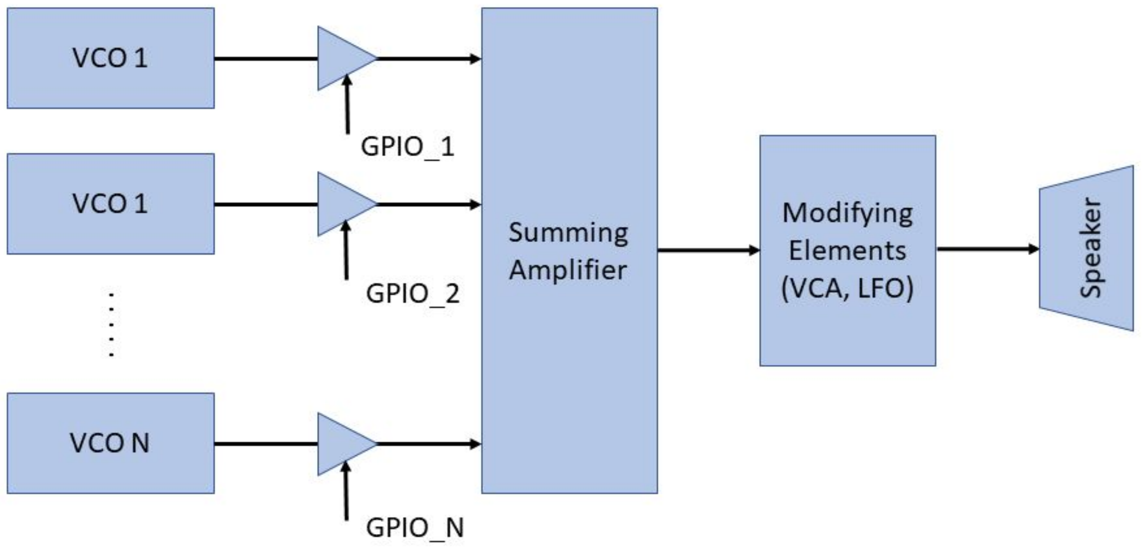


Figure 4-1. Block diagram of audio synthesis with high-level architecture.

In the ideal design, a bank of signals of different frequencies is generated using several VCOs. Using the input control voltages via GPIO, we select which signals to sum through transmission gates in series with each VCO, effectively generating unique harmonics based on input selections. These signals are then combined with a voltage summer to produce a single output audio waveform. The summed signal is further modified via a VCA to effect changes in volume and sound character before being output to the speaker. While in theory this design should produce a proper wound wave, in practice, the design was unable to compile and be programmed into the FPAA. However, each individual component (the VCO, the VCA, and the summer) of the synthesizer was able to compile and exhibit functional behavior. Each component is described in further detail below.

In the VCO, a control voltage (CV) is taken in to produce an output analog waveform of a desired frequency. This waveform serves as the basis of the overall sound wave that is ultimately sent to the speakers [5]. Figure 5-1 shows how a VCO can be created from operational amplifiers and current starved inverters. Figure 5-2 shows the transistor layout for the VCO's current-starved inverters, which are slew-rate limited inverters that can be used to convert square waves to triangle waves. Figure 5-3 shows the VCO design that the team implemented for the project. Figure 5-4 shows the output waveform of the VCO, which can be controlled by its control voltages.

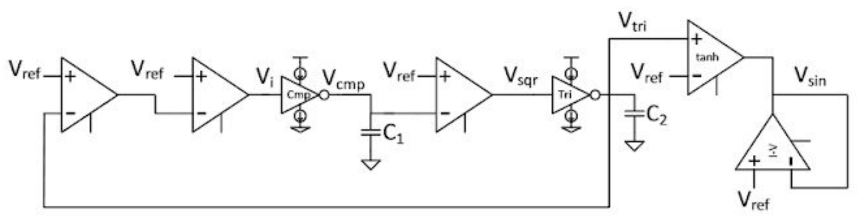


Figure 5-1. Component-level schematic of a VCO made from CSIs (current starved inverters) and operational amplifiers. The reference voltages control the high and low voltages of internal square waves, while the CSIs' bias voltages and the value of the capacitor determine the slopes of internal triangle waves so that the frequency of the output waveform can be controlled [5].

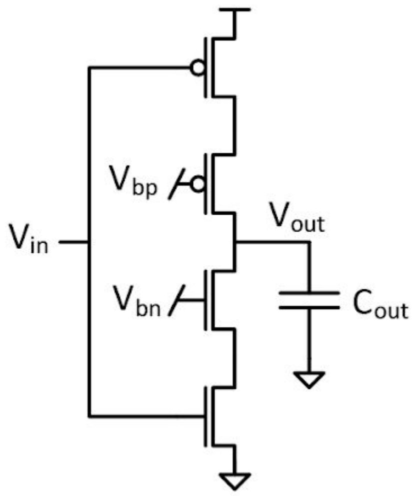


Figure 5-2. Transistor layout of a CSI (current starved inverter). An additional PMOS is added to the pull up network and an additional NMOS is added to the pull down network so that the output current and, therefore, the slew rate of the output may be limited through the application of bias voltages V_{bp} and V_{bn}. This allows input square waves to be converted to triangle waves [5].

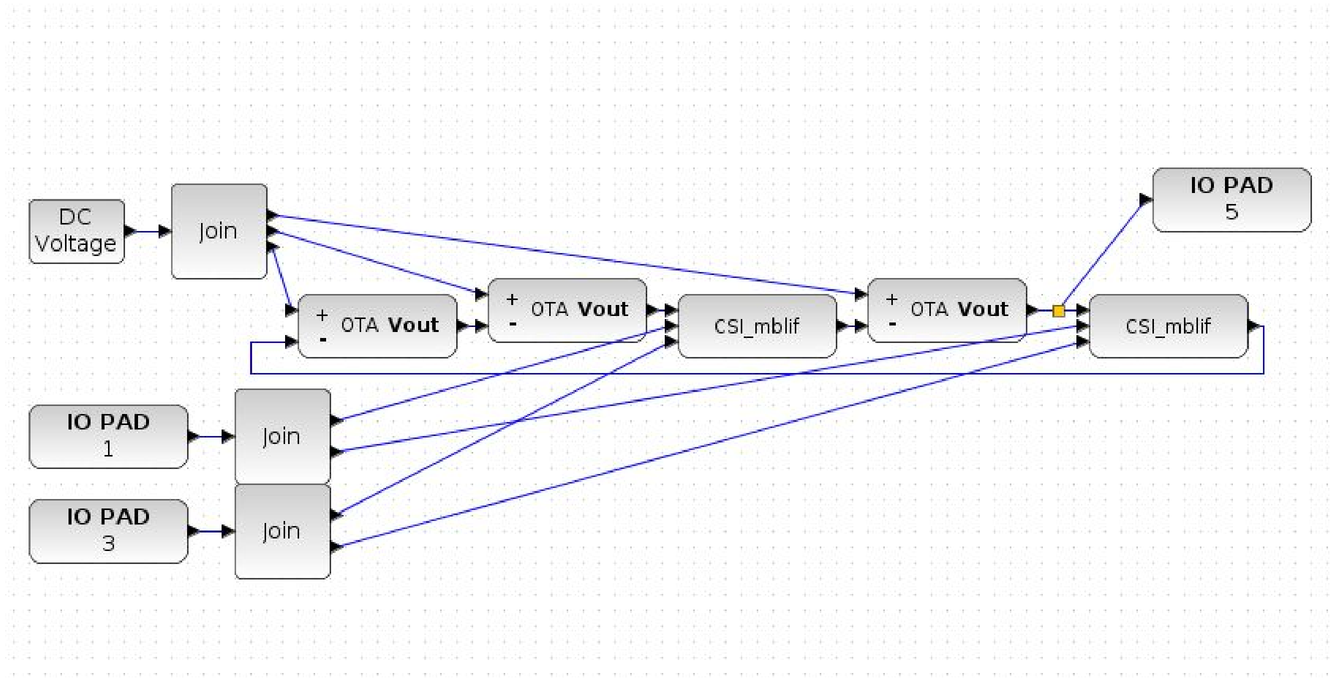


Figure 5-3. The Electronics for Music Applications Team's implementation of a square wave VCO for an analog music synthesizer. This implementation is similar to the one shown in Figure

5-1 but without a low pass filter at the end to produce a sine wave. The VCO is controlled by a DC voltage that serves as the reference voltage described in Figure 5-1 as well as V_{bp} and V_{bn} bias voltages (IO1 and IO3) for the CSIs.

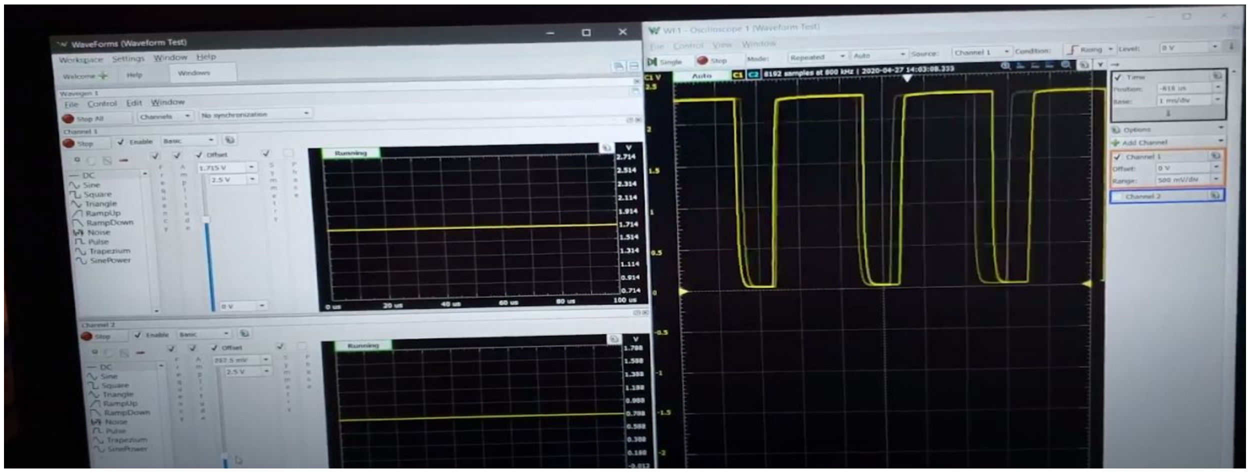


Figure 5-4. Oscillation of the Electronics for Music Applications Team's VCO at V_{ref} = 1 V as V_{bp} and V_{bn} are varied. At this point, V_{bp} = 1.715 V and V_{bn} = 252.5 mV. The output appears to be a square wave.

In the voltage summer, selected VCO signals are combined into a single audio output waveform which can later be modified by the VCA and directed to the system's speakers. Figure 5-5 shows a potential voltage summer design based on OTAs (operational transconductance amplifiers). Each addend waveform is converted from a voltage to a current by an OTA before being summed at a common node and output through an OTA resistor [6]. Figure 5-6 shows the Electronics for Music Applications Team's implementation of a voltage summer. Figure 5-7 shows the summer exhibiting functional behavior.

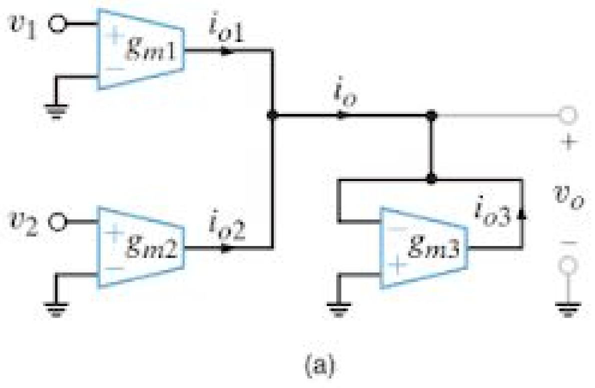


Figure 5-5. Component-level schematic of a voltage summer. OTAs' (operational transconductance amplifiers') output currents are proportional to their differential input voltages, allowing each addend to be summed by Kirchhoff's Current Law. The current sum is then passed through an OTA resistor to create an output voltage [6].

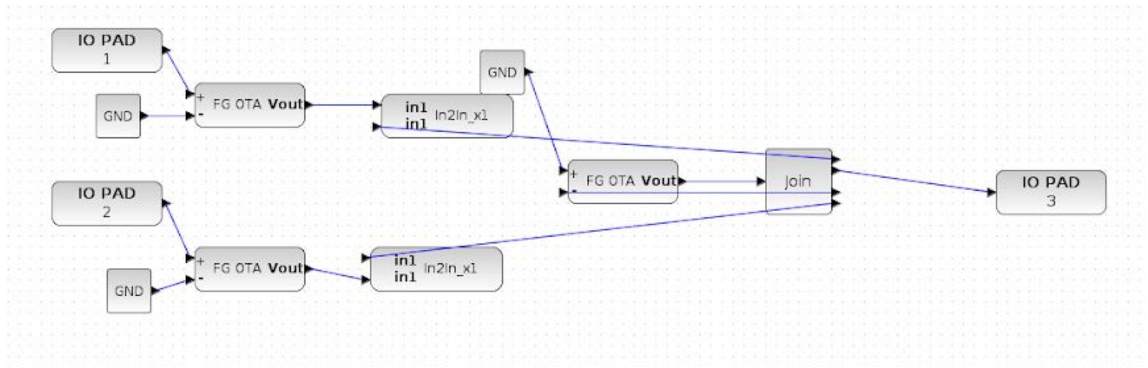


Figure 5-6. The Electronics for Music Applications Team's implementation of a voltage summer for an analog music synthesizer. This implementation is the same as the one shown in Figure 5-5. IO1 and IO2 are input voltages to be summed.

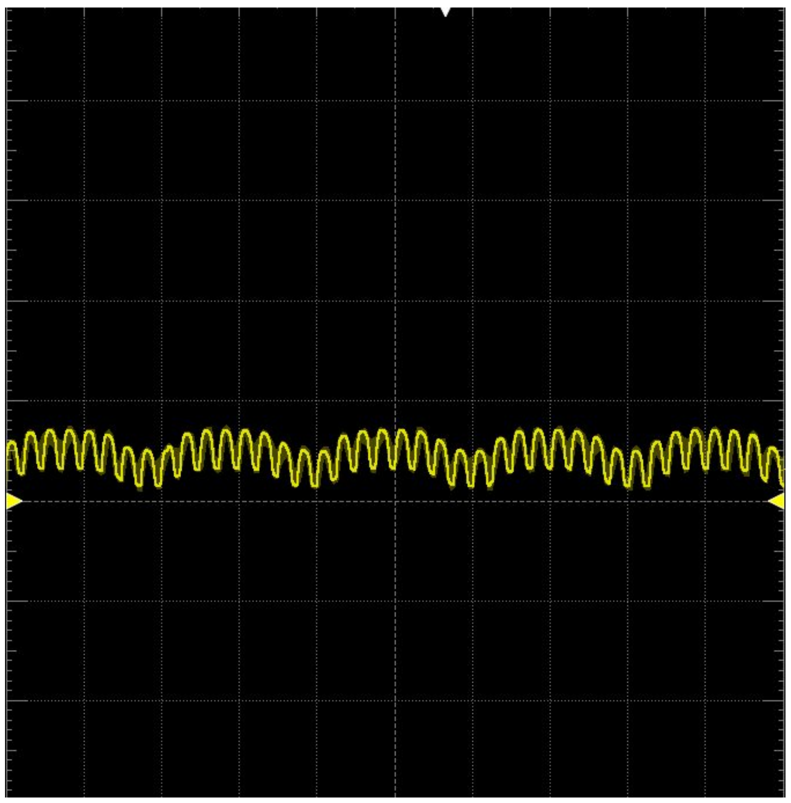


Figure 5-7. Output of the Electronics for Music Applications Team's summer. The summer is adding two sine waves with all floating gate OTA bias currents set to 5e-7 A.

In the VCA, the audio output waveform and an envelope waveform are taken in to produce an amplified signal with a new envelope. Various types of envelopes can be chosen to produce different kinds of sounds, but one common envelope waveform is called ADSR (attack-decay-sustain-release). The ADSR waveform consists of a rapid rise from an initial voltage level (attack); a shorter, rapid fall (decay); a constant voltage value (sustain); and a rapid fall back to the initial voltage level (release). It is notable for giving notes a "plucked string" sound [5]. Figure 5-8 shows how a VCA can be implemented using a Gilbert Multiplier, a circuit whose output is proportional to the product of its input voltages. Figure 5-9 shows the transistor layout of an ADSR Generator, which provides a possible input envelope for the VCA, though it was unused in the team's design. Figure 5-10 shows the

Electronics for Music Applications Team's implementation of a wide range Gilbert Multiplier with PMOS and NMOS current mirrors. Figure 5-11 shows the Electronics for Music Applications Team's implementations of PMOS and NMOS current mirrors for the VCA in CAB blocks. Figure 5-12 shows a snapshot of the output of the team's VCA as it multiplies two input sine waves of varied frequencies.

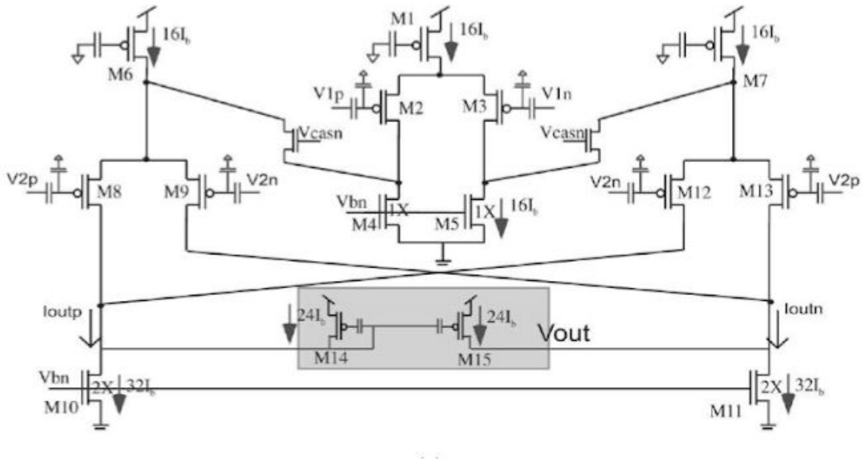


Figure 5-8. Transistor layout of a VCA made from a Gilbert Multiplier. The bias voltages V_{bn} and V_{bp} can be used to control the magnitude of the output voltage [5].

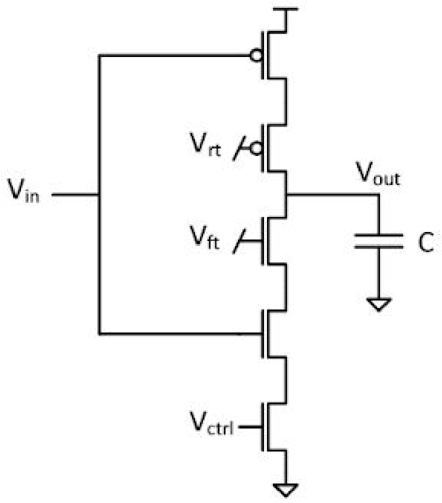


Figure 5-9. Transistor layout of an ADSR Generator made from a modified CSI. An additional NMOS is added to the pull down network to cut off current completely during the "sustain" phase of the signal. V_{rt} and V_{ft} can be used to control the rise and fall of the signal [5].

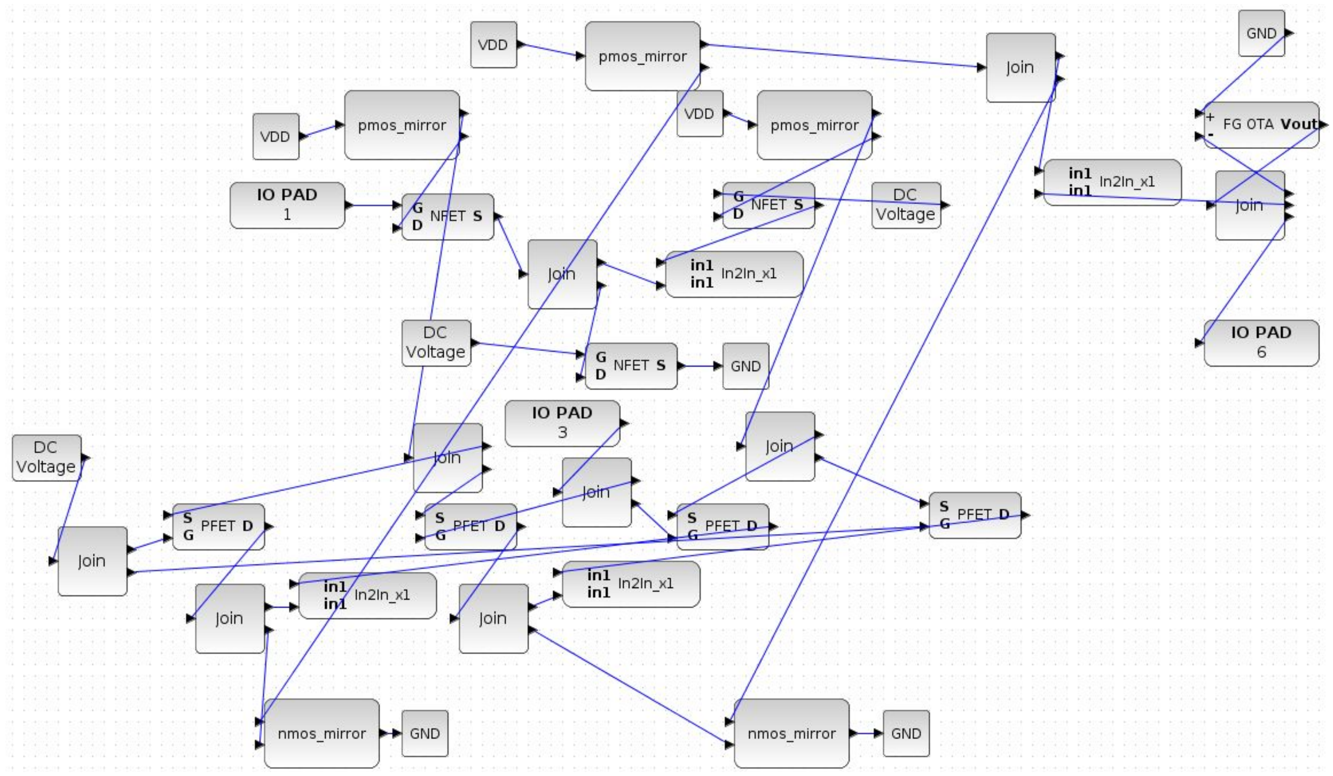
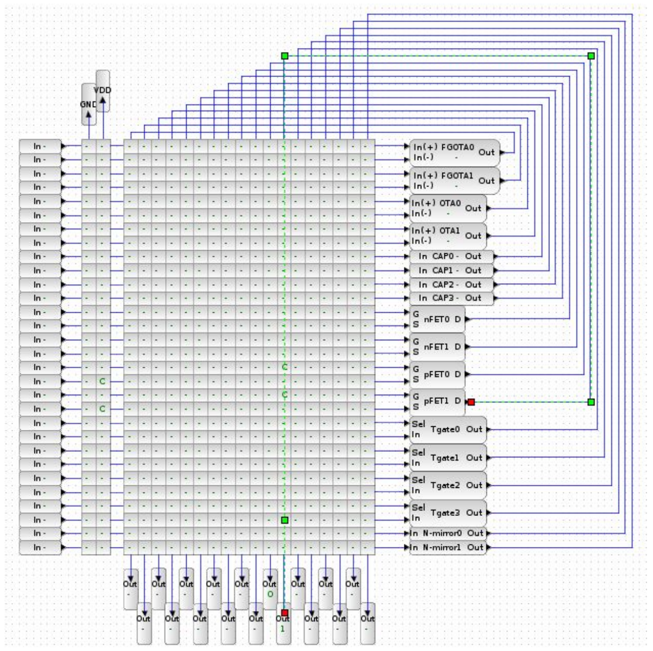
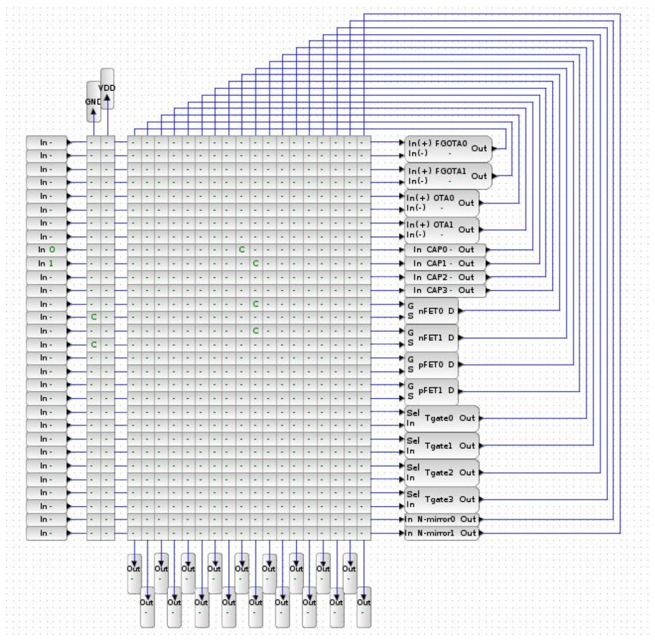


Figure 5-10. The Electronics for Music Applications Team's implementation of a wide range VCA with transistors and PMOS and NMOS current mirrors. IO1 and IO3 are the multiplicand voltages V1 and V2, and the NFET gate's DC Voltage is Vb, which controls the amplitude of the output IO6.





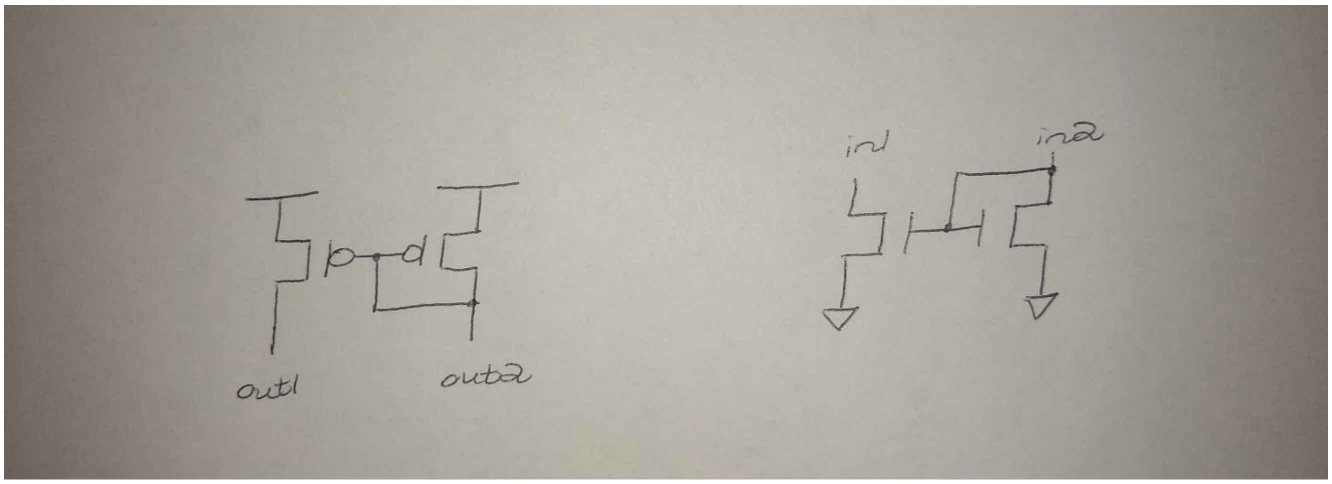


Figure 5-11. The Electronics for Music Applications Team's implementation of PMOS (top left) and NMOS (top right) current mirrors in CABs (Computing Analog Blocks) as subcomponents for the VCA. Below each implementation is the hand-drawn circuit diagram for the block.

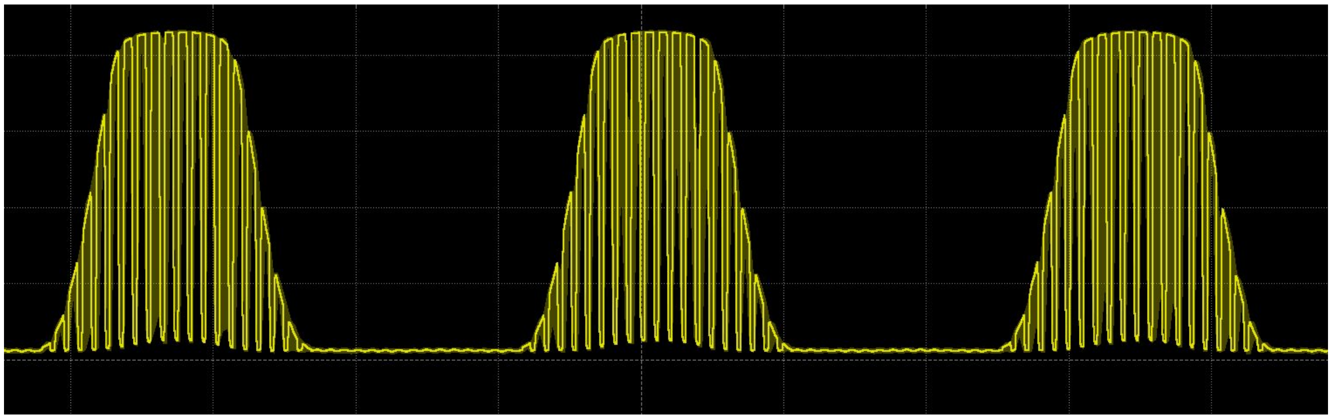


Figure 5-12. Output waveform of the Electronics for Music Applications Team's VCA. The waveform shows the amplitude modulation of two input sine waves. For this simulation, Vb = 2.5 V and all floating gate OTA bias currents are set to 5e-7 A.

FPAA Programming Environment

While the FPAA programming environment is not an explicit component of our music synthesizer system, it is still crucial, for it allows the VCO, VCA, and VCF circuitry described in the previous section to be implemented. The FPAA programming environment is called RASP, and it consists of many layers. The user interacts directly with a high end graphical design environment built in Xcos that compiles into distinct analog, digital, and assembly components. The analog and digital components are converted into netlists with blif (Berkeley Library Interface), which are then compiled to a switch list for the FPAA's program unit with an open source tool called x2c. Meanwhile, the assembly component of the software is compiled into hex code, where it is also delivered to the FPAA's program unit [4]. Figure 6-1 shows a block diagram of the programming interface's different layers.

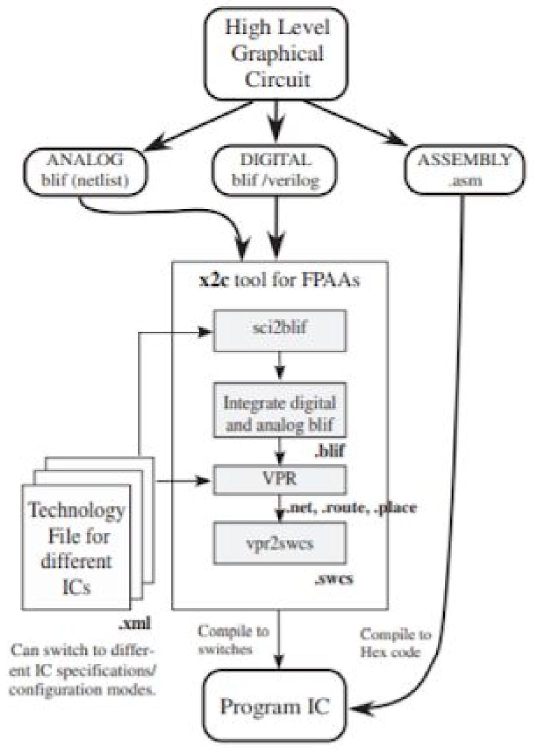


Figure 6-1. A block diagram of the different layers of the FPAA board's programming interface. The programming environment consists of a palette of graphical design blocks, which compile to lower analog, digital, and assembly software levels from the Xcos environment [4].

To program the FPAA board, a user simply drags pre-made graphical design blocks from a palette onto a blank design window. The premade design blocks range in complexity from simple PFET blocks to entire analog circuits, such as operational amplifiers. From the design window, the user can press a button to drag wires between the input and output ports of different design blocks before compiling the circuitry to the FPAA board. Programmed circuitry can be reloaded for subsequent uses of the FPAA board so that the locations of important cells do not change.

4.2 Codes and Standards

1. Universal serial bus (USB) is utilized to program the FPAA microcontroller. USB is also used in serial communication from the Android application to the microcontroller. USB features:
    - 480 Mbps data transfer rate.
    - Versatility in peripheral connections [7].
2. General Purpose Input/Output (GPIO) is a communication protocol used to connect the FPAA microcontroller with additional peripherals. GPIO can be used to provide control voltages to the combined synthesizer.
3. The FPAA board also features a 3.5 mm headphone jack, which can be routed to a speaker in the final audio output [3].

4.3 Constraints, Alternatives, and Trade-Offs

Alternatives

The main alternative to an analog music synthesizer system is a digital synthesizer system. Digital synthesizers are often portable and easy to interface with other devices. However, analog music synthesizers are often considered to produce more natural, aurally pleasing sounds than their digital counterparts, for they have a tendency to produce distortion as well as subtle variations in waveform shape, frequency, and amplitude [1]. As a result, an FPAA board implementation of a music synthesizer seems to strike the best balance between portability and sound quality.

Constraints

In order for the music synthesizer system to be effective, it must be portable, energy-efficient, and fast. These constraints will enable the system to provide unique benefits to users who are unsatisfied with the large but rich-sounding analog synthesizers that are on the market today. This goal can be achieved by combining the FPAA board and a portable speaker onto a single package.

Trade-Offs

Configurability traded off with ease of use in this project. The values of various control voltages (CVs) for the VCO and VCA components of the music synthesizer could have been chosen as user-defined inputs to the circuit or as predetermined values. For example, the VCA could have used the ADSR envelope detailed above to produce a "plucked string" sound, or it could have used other user-defined envelopes. The VCO could have used another oscillator to drive its CSI biases and produce unique vibrating sounds. Allowing users to define specific CVs allows for greater low-level control, for these CVs can impact the types of sounds that can be synthesized. Using predetermined values for some CVs (such as only allowing voltages from GPIO pins), however, would limit the types of sounds that could be synthesized but would result in a smaller learning curve for new users if the music synthesizer system entered the market. For this reason, the team chose to allow set GPIO inputs to control the synthesizer.

4.4 Engineering Analyses and Experiment

The individual components built on the FPAA were tested through its various GPIO pins. One such device that interfaces with these pins is the Analog Discovery 2. This product, which communicates directly with the software *Waveforms* on a computer, is an all-in-one USB oscilloscope and instrumentation system which can generate input waveforms for the FPAA and read resulting output waveforms [11]. This device allows for quantitative and efficient testing of each circuit component. For example, testing the VCO involve feeding the unit a biases for its CSIs and its OTA reference voltages and observing the output waveforms. Similar trials were run with the VCA and voltage summer, with bias currents adjusted as needed along the way. Overall, our tests showed that the individual components behaved as expected for the most part, although they were unable to be combined into a single synthesizer.

5. Project Demonstration

Because the design team was unable to compile a single working synthesizer out of mostly functional VCO, VCA, and voltage summer blocks, a different approach to the project demonstration was chosen. The design team assembled testable program files for the VCO, the VCA, and the voltage summer so that future teams could use these designs to produce a working synthesizer. The team also compiled a video with instructions on how to use RASP Tools to program an FPAA and how to use and understand each component (VCO, VCA, and voltage summer) of the synthesizer design. This demonstration was integrated into the final presentation for the team so that future teams could easily access needed information in a single file.

6. Schedule, Tasks, and Milestones

The design team designed, tested, and implemented the music synthesizer over the semester until the major milestones in the last week of April, although the outbreak of COVID-19 altered the design approach, as specified in the team's modified proposal. Appendix A is the comprehensive Gantt chart of the project. A partial Gantt Chart is shown in Appendix B; Appendix C contains the related Pert chart outlining the expected duration for each task.

PERT Chart Critical Path

Path	**Time**
ABEFH~P	1.17+2.33+2.33+4.67+51.16 = 61.66

ABEGH~P	1.17+2.33+1.17+4.67+51.16 = 60.5
ACEFH~P	4.17+2.33+2.33+4.67+51.16 = 64.66
ACEGH~P	4.17+2.33+1.17+4.67+51.16 = 63.5
ADEFH~P	5.17+2.33+2.33+4.67+51.16 = 65.66
ADEGH~P	5.17+2.33+1.17+4.67+51.16 = 64.5

The estimated time of the critical path is 65.66 days. This estimate is excluding holidays and weekends of the number of days between the first day of class (1/6/20) and the Expo (4/10/20). The probability of finishing one week prior to the Senior Design Capstone Expo is 99.92 %.

7. Marketing and Cost Analysis

Cost Estimate (Year 1)

A. Employee Services (12 hr/week)

a.	Electrical/Computer Engineer - 7 @ $40/hr	$280.00
b.	Project Manager - 1 @ $40/hr	$40.00
c.	Fringe Benefits (25% of total salary)	$14,400
d.	Total (1 year)	$72,000

B. Materials/Supplies

a.	FPAA Components (per unit)	$94.69
b.	Supporting Components (per unit)	$107.01
c.	Total (10 testing units)	$2,017

C. Miscellaneous Costs

a. Spare parts (per month)	$500.00
b. Overtime - 8 @ $45/hr (~2 hr/week)	$720.00
c. Total (1 year)	$43,440
D. Total Cost	$117,457

Employee salaries were estimated based on the average salary for an electrical/computer engineer in the Atlanta area. Hours worked per week were based on ECE4012 credit hours and allocated lab hours (3 credit class * 4 hours = 12 hours/week).

Product cost breakdown below. Miscellaneous costs based on spare parts (in case of defective components) and need for overtime as required throughout the semester.

FPAA Breakdown

Component	Units	Cost
TI-MSP430 (Microprocessor) [12]	1	$5.01
70V261L25PFG (SRAM) [1	1	$44.38
Custom Analog/Digital PL	1	~$25.00
TMS320C5515 (SPI I/O) [1	1	$5.30
Manufacturing	N/A	~$15.00

Supporting Components

Component	Units	Cost
Standard USB 3.0 Cable	1	$10.00
LPC1768 (Microcontroller) [16]	1	$52.06
Speaker	1	$12.00
Pushbuttons	8-10	$15.00

Estimates for the custom PLD cost were determined by researching component costs for commonly used FPGA PLD chips from popular electronic part shopping sites. The estimate for the manufacturing cost was determined by researching custom PCB costs with the sizes/parameters for the current FPAA model we intend to use.

Development Cost

Parts	$2,017
Labor (~180 hr/semester, 8 employees)	$57,600
Fringe Benefits (25% total salary)	$14,400
Subtotal	**$74,017**
Overhead (120% of subtotal)	$88,820
Total	**$162,837**

Determination of Selling Price

Parts Cost (1 unit)	$201.70
Assembly Labor (Manufacturing)	$15.00
Testing Labor	$135.00
Total Labor	$150.00
Fringe Benefits (25% of labor)	$37.50
Subtotal	**$539.20**
Overhead (120% of subtotal)	$647.04
Subtotal (Input Costs)	**$1,186.24**
Sales Expense (10% selling price)	$160.00
Subtotal (All Costs)	**$1,346.24**
Profit	$253.76
Selling Price	**$1,600**

Assuming a selling period of 5 years and an approximate selling capacity of 120 units per year, to amortize the development cost, each unit must be sold at: [5 * (120 * (selling price - unit cost)) = $162,837]. The calculated unit price comes out to be about $1,600. Each unit provides a $253.76 profit, or 15.9% of selling price profit.

8. Conclusion

Although the group was unable to create a working synthesizer by the end of the semester, the three major components of the synthesizer (VCO, VCA, and voltage summer) were successfully compiled and exhibited functional behavior. Issues with compiling a combined design and generating subcircuit blocks on the FPAA hindered the team's performance. However, the created components should serve as a strong foundation for future groups to continue the development of a portable, energy efficient synthesizer. In particular, an array of VCOs (with additional oscillators to control their biases) could be further developed to ensure that a user can exert control over the frequency of his/her sound waves. Additionally, an ADSR envelope generator - along with other envelope generators - could help to expand the utility of the VCA to not only control volume but also impact the shape and quality of produced sound waves in the future. Regardless of the outcome, the design of a music synthesizer on an FPAA proved to be a valuable learning experience and emphasized the importance of quickly becoming acquainted with a new technology. The team's experience led to the modification of its product demonstration to reinforce this concept so that future teams can quickly jump into the finer details of design and continue to explore the applications of FPAAs in music synthesis in the coming years.

9. Leadership Roles

1. Chris Walds: Expo Coordinator
    a. Manages content to be presented at the Capstone Expo
2. Kristyn DiGiovanni: Webmaster
    a. Maintains and updates the team website and Google Drive

3. Jongheon Park: Design Coordinator
    a. Oversees design progress against schedule
    b. Works with Dr. Hasler to implement corrective measures for identified issues
4. Justin Kelley: Team Coordinator
    a. Arranges team meetings with Dr. Hasler
5. Harrison Zhang: Embedded Systems Specialist
    a. Reviews embedded system logistics for synthesizer
6. Sriram Pulavarty: Documentation Coordinator
    a. Creates and updates database with detailed explanations on functionality of synthesizer and design choices made
7. Yewon Kim: Hardware Specialist
    a. Assists in the installation of the synthesizer
    b. Analyzes and adjusts any hardware reconfigurations

10. References

[1] M. Doty, "What's Behind The Resurgence of Analog Synthesizers?", Performer Mag, July 14, 2016. [Online]. Available: https://performermag.com/best-instruments/best-music-keyboards-synth/resurgence-of-analog-synthesizers/. [Accessed Nov. 18, 2019]

[2] V. Lazzarini and J. Timoney, "The Analogue Computer as a Voltage-Controlled Synthesiser", Maynooth University. April 25, 2019. [Online]. Available: https://arxiv.org/pdf/1904.10763v2.pdf [Accessed Nov. 18, 2019]

[3] S. George, S. Kim, S. Shah, J. Hasler, M. Collins, F. Adil, R. Wunderlich, S. Nease, and S. Ramakrishnan "A Programmable and Configurable Mixed-Mode FPAA SOC," IEEE Transactions on VLSI, January 2016. [Online]. Available: http://hasler.ece.gatech.edu/SoCFPAA/RASP30_SoCFPAA.pdf [Accessed Nov. 18, 2019]

[4] M. Collins, J. Hasler, and S. George, "An Open-Source Toolset Enabling Analog–Digital–Software Codesign," *Journal of Low Power Electronics Applications*, January 2016. [Online]. Available: http://hasler.ece.gatech.edu/Published_papers/FPAA_Papers/Highlevel_tools_ jlpea_February2016.pdf. [Accessed Nov. 18, 2019].

[5] S. Nease. "Neural and Analog Computation on Reconfigurable Mixed-Signal Platforms", Ph. D. dissertation, Georgia Institute of Technology, GA, 2014. [Online]. Available: https://smartech.gatech.edu/handle/1853/53999. [Accessed Nov. 18, 2019].

[6] J. D. Irwin and R. M. Nelms, "Filter Networks," in *Basic Engineering Circuit Analysis,* 10th ed. Hoboken, NJ: John Wiley & Sons, 2011, ch. 12, pp. 637–640.

[7] "USB 2.0, Hi-Speed USB FAQ," Everything USB, 20-Mar-2017. [Online]. Available: https://www.everythingusb.com/hi-speed-usb.html. [Accessed: Nov. 18, 2019].

[8] Kumar, Kumar, and D. Rajani, “UART Communication Protocol - How it works?,” Codrey Electronics, 26-Jul-2018. [Online]. Available: https://www.codrey.com/embedded-systems/uart-serial-communication-rs232/. [Accessed: 18-Nov-2019].

[9] “What is GPIO?,” Estimote Community Portal. [Online]. Available: https://community.estimote.com/hc/en-us/articles/217429867-What-is-GPIO-. [Accessed: 18-Nov-2019].

[10] C. S. Sapp, MIDI Communication Protocol. [Online]. Available: http://www.ccarh.org/courses/253/handout/midiprotocol/. [Accessed: 18-Nov-2019].

[11] “Analog Discovery 2 Specifications,” Analog Discovery 2 Specifications [Reference.Digilentinc]. [Online]. Available: https://reference.digilentinc.com/reference/instrumentation/analog-discovery-2/specifications. [Accessed: 18-Nov-2019].

[12] Texas Instruments. “TMS320C5515 Fixed-Point Digital Signal Processor,” Texas Instruments Incorporated, 2013. [Online]. Available: https://www.ti.com/lit/ds/symlink/tms320c5515.pdf [Accessed: 18-Nov-2019].

[13] Integrated Device Technology. “HIGH-SPEED 3.3V 16K x 16 DUAL-PORT STATIC RAM,” July 2019. [Online]. Available: https://www.idt.com/us/en/document/dst/70v26-data-sheet [Accessed: 18-Nov-2019].

[14] Texas Instruments. “16-Mb RADIATION-HARDENED SRAM,” Texas Instruments Incorporated, 2014. [Online]. Available: http://www.ti.com/lit/ds/symlink/smv512k32-sp.pdf [Accessed: 18-Nov-2019].

[15] Byron, J. “MIDI Shield Hookup Guide,” Sparkfun Electronics. [Online]. Available:

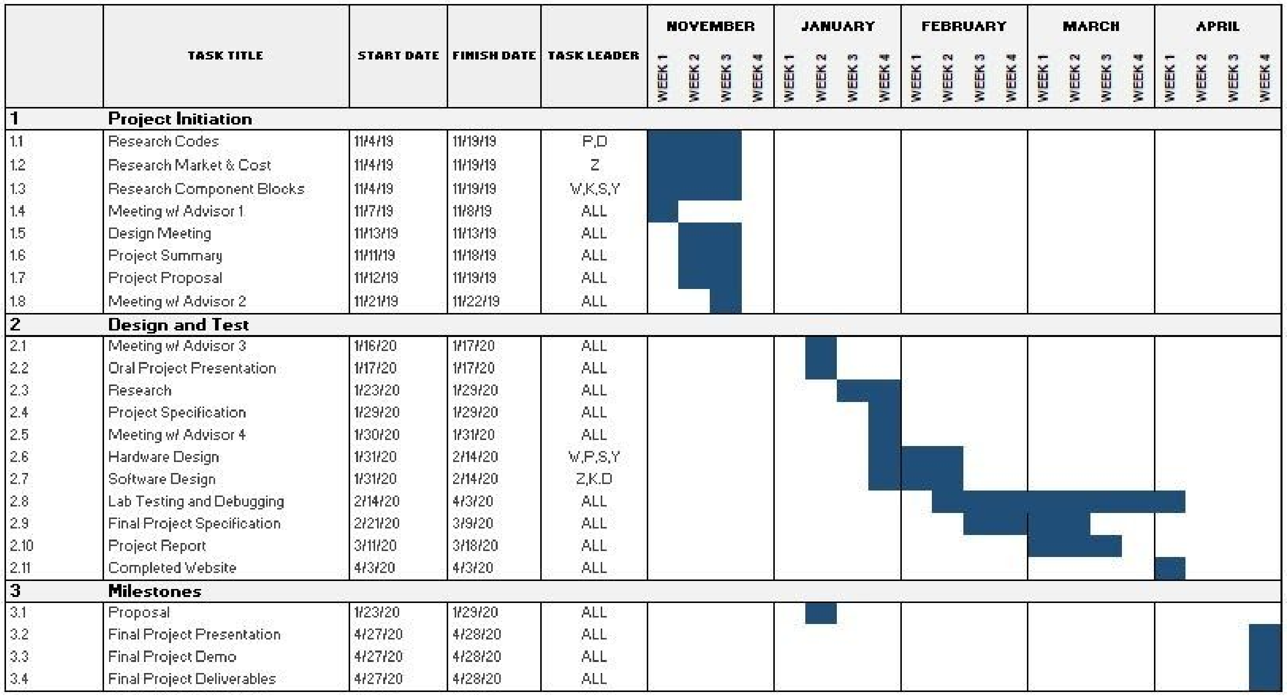
https://learn.sparkfun.com/tutorials/midi-shield-hookup-guide/all [Accessed: 18-Nov-2019].

[16] NXP Semiconductors. "LPC1769/68/67/66/65/64/63," May 4 2014. [Online]. Available: https://www.nxp.com/docs/en/data-sheet/LPC1769_68_67_66_65_64_63.pdf [Accessed: 18-Nov-2019].

Appendix A: Comprehensive GANTT Chart

ECE4011/4012 COMPREHENSIVE GANTT CHART

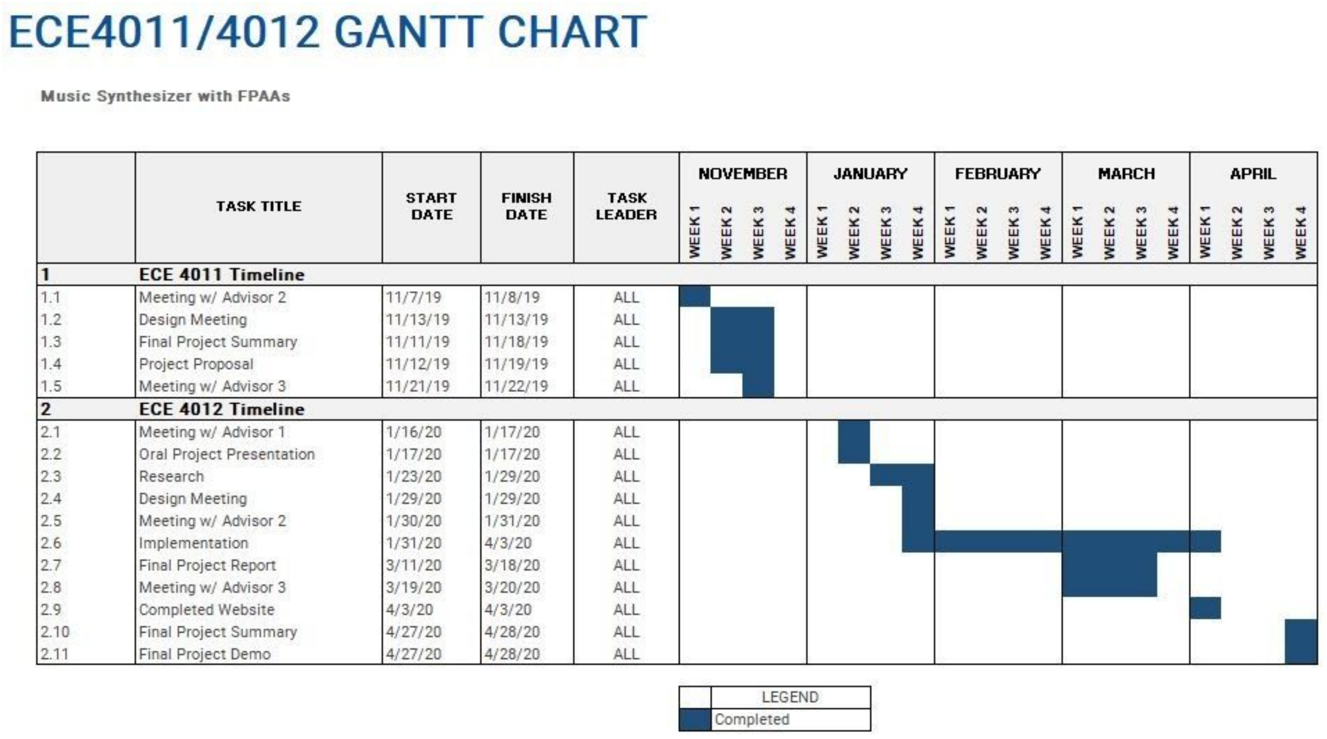
Music Synthesizer with FPAAs



	TASK TITLE	START DATE	FINISH DATE	TASK LEADER	NOVEMBER				JANUARY				FEBRUARY				MARCH				APRIL			
					WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2	WEEK 3	WEEK 4
1	**Project Initiation**																							
1.1	Research Codes	11/4/19	11/19/19	P,D	🟦	🟦	🟦																	
1.2	Research Market & Cost	11/4/19	11/19/19	Z	🟦	🟦	🟦																	
1.3	Research Component Blocks	11/4/19	11/19/19	W,K,S,Y	🟦	🟦	🟦																	
1.4	Meeting w/ Advisor 1	11/7/19	11/8/19	ALL	🟦																			
1.5	Design Meeting	11/13/19	11/13/19	ALL		🟦	🟦																	
1.6	Project Summary	11/11/19	11/18/19	ALL		🟦	🟦																	
1.7	Project Proposal	11/12/19	11/19/19	ALL		🟦	🟦																	
1.8	Meeting w/ Advisor 2	11/21/19	11/22/19	ALL			🟦																	
2	**Design and Test**																							
2.1	Meeting w/ Advisor 3	1/16/20	1/17/20	ALL						🟦														
2.2	Oral Project Presentation	1/17/20	1/17/20	ALL						🟦														
2.3	Research	1/23/20	1/29/20	ALL							🟦	🟦												
2.4	Project Specification	1/29/20	1/29/20	ALL								🟦												
2.5	Meeting w/ Advisor 4	1/30/20	1/31/20	ALL								🟦												
2.6	Hardware Design	1/31/20	2/14/20	W,P,S,Y								🟦	🟦	🟦										
2.7	Software Design	1/31/20	2/14/20	Z,K,D								🟦	🟦	🟦										
2.8	Lab Testing and Debugging	2/14/20	4/3/20	ALL										🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦			
2.9	Final Project Specification	2/21/20	3/9/20	ALL											🟦	🟦	🟦	🟦						
2.10	Project Report	3/11/20	3/18/20	ALL													🟦	🟦	🟦					
2.11	Completed Website	4/3/20	4/3/20	ALL																	🟦			
3	**Milestones**																							
3.1	Proposal	1/23/20	1/29/20	ALL						🟦														
3.2	Final Project Presentation	4/27/20	4/28/20	ALL																				🟦
3.3	Final Project Demo	4/27/20	4/28/20	ALL																				🟦
3.4	Final Project Deliverables	4/27/20	4/28/20	ALL																				🟦

	LEGEND
🟦	Completed
W	Chris Walds
Z	Harrison Zhang
P	Jongheon Park
K	Justin Kelley
D	Kristyn DiGiovanni
S	Sriram Pulavarty
Y	Yewon Kim

Appendix B: GANTT Chart



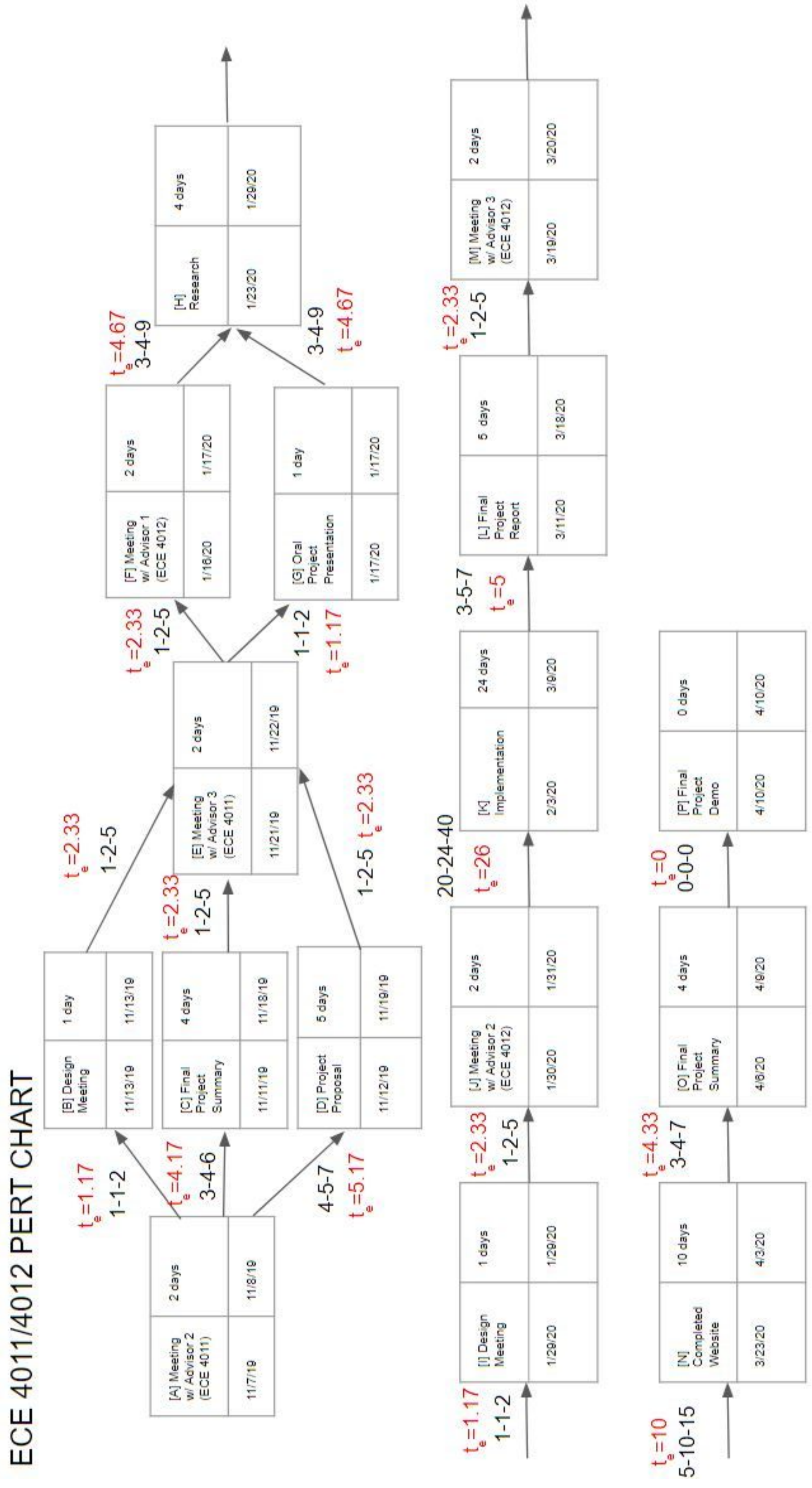
ECE4011/4012 GANTT CHART

Music Synthesizer with FPAAs

	TASK TITLE	START DATE	FINISH DATE	TASK LEADER	NOVEMBER				JANUARY				FEBRUARY				MARCH				APRIL			
					WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2	WEEK 3	WEEK 4	WEEK 1	WEEK 2	WEEK 3	WEEK 4
1	**ECE 4011 Timeline**																							
1.1	Meeting w/ Advisor 2	11/7/19	11/8/19	ALL	■																			
1.2	Design Meeting	11/13/19	11/13/19	ALL		■	■																	
1.3	Final Project Summary	11/11/19	11/18/19	ALL		■	■																	
1.4	Project Proposal	11/12/19	11/19/19	ALL		■	■																	
1.5	Meeting w/ Advisor 3	11/21/19	11/22/19	ALL			■																	
2	**ECE 4012 Timeline**																							
2.1	Meeting w/ Advisor 1	1/16/20	1/17/20	ALL						■														
2.2	Oral Project Presentation	1/17/20	1/17/20	ALL						■														
2.3	Research	1/23/20	1/29/20	ALL							■	■												
2.4	Design Meeting	1/29/20	1/29/20	ALL								■												
2.5	Meeting w/ Advisor 2	1/30/20	1/31/20	ALL								■												
2.6	Implementation	1/31/20	4/3/20	ALL								■	■	■	■	■	■	■	■	■	■			
2.7	Final Project Report	3/11/20	3/18/20	ALL													■	■	■					
2.8	Meeting w/ Advisor 3	3/19/20	3/20/20	ALL													■	■	■					
2.9	Completed Website	4/3/20	4/3/20	ALL																	■			
2.10	Final Project Summary	4/27/20	4/28/20	ALL																				■
2.11	Final Project Demo	4/27/20	4/28/20	ALL																				■

LEGEND	
■	Completed

Appendix C: PERT Chart


ECE 4011/4012 PERT CHART
[A] Meeting w/ Advisor 2 (ECE 4011)
2 days
11/7/19
11/8/19
[B] Design Meeting
1 day
11/13/19
11/13/19
[C] Final Project Summary
4 days
11/11/19
11/18/19
[D] Project Proposal
5 days
11/12/19
11/19/19
[E] Meeting w/ Advisor 3 (ECE 4011)
2 days
11/21/19
11/22/19
[F] Meeting w/ Advisor 1 (ECE 4012)
2 days
1/16/20
1/17/20
[G] Oral Project Presentation
1 day
1/17/20
1/17/20
[H] Research
4 days
1/23/20
1/29/20
[I] Design Meeting
1 days
1/29/20
1/29/20
[J] Meeting w/ Advisor 2 (ECE 4012)
2 days
1/30/20
1/31/20
[K] Implementation
24 days
2/3/20
3/9/20
[L] Final Project Report
5 days
3/11/20
3/18/20
[M] Meeting w/ Advisor 3 (ECE 4012)
2 days
3/19/20
3/20/20
[N] Completed Website
10 days
3/23/20
4/3/20
[O] Final Project Summary
4 days
4/6/20
4/9/20
[P] Final Project Demo
0 days
4/10/20
4/10/20
t_e=1.17 1-1-2
t_e=4.17 3-4-6
4-5-7 t_e=5.17
t_e=2.33 1-2-5
t_e=2.33 1-2-5
1-2-5 t_e=2.33
t_e=2.33 1-2-5
1-1-2 t_e=1.17
t_e=4.67 3-4-9
3-4-9 t_e=4.67
t_e=1.17 1-1-2
t_e=2.33 1-2-5
20-24-40 t_e=26
3-5-7 t_e=5
t_e=2.33 1-2-5
t_e=10 5-10-15
t_e=4.33 3-4-7
t_e=0 0-0-0