ECEN 215 Lab Kit

Luis Diaz-Santini
Ryan Freed
Yusuf Hossain
Peter Zhang

CONCEPT OF OPERATIONS

REVISION – Final
2 December 2024

Concept of Operations
for
ECEN 215 Lab Kit

Team 30

Approved by:

_____
Project Leader Date

_____
Prof. Kalafatis Date

_____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
-	2/5/2024	ECEN 215 Lab Kit		Draft Release
	2/9/2024	ECEN 215 Lab Kit		Draft Editing
	2/10/2024	ECEN 215 Lab Kit		Review for First Submission
	12/2/2024	ECEN 215 Lab Kit		Updating and Review for 404 Submission

Table of Contents

List of Tables

List of Figures

1. Executive Summary

The Principles of Electrical Engineering course (ECEN 215) at Texas A&M is a second year course offered at the university that serves to introduce engineering students outside of electrical engineering to the principles of electrical engineering. The ECEN 215 Lab Kit project addresses the need for an affordable and convenient solution for Texas A&M students enrolled in ECEN 215. The primary objective is to develop a small-scale device that enables students to perform laboratory experiments from the comfort of their homes, with the goal of offering this course 100% online. The key differentiator lies in the cost-effectiveness of the ECEN 215 Lab Kit, offering students a more economic option compared to existing alternatives such as the Analog Discovery 2 (AD2).

Our team aims to develop this device with a student-centric approach that caters specifically to the lab assignments in ECEN 215. The device will incorporate essential circuit analysis tools, such as an oscilloscope, waveform generator, and multimeter. In keeping the design of this device simple, and only implementing necessary functions for ECEN 215 we can ensure minimal confusion and maximum efficiency among students when performing laboratory experiments.

In the spirit of simplicity, interacting with the device will be through a smartphone app that connects via Bluetooth. Using a smartphone app allows the user to easily interact with the device with a tool they are immensely familiar with. The incorporation of a smartphone app, again, reduces the overall cost of the product and simplifies the design as well. The app will be used to set output voltage and signals, interact with and view waveforms, measure outputs, and more. Overall, the ECEN 215 Lab Kit project strives to revolutionize the accessibility and affordability of laboratory equipment for ECEN 215 students, achieving an enriching educational experience in the field of electrical engineering.

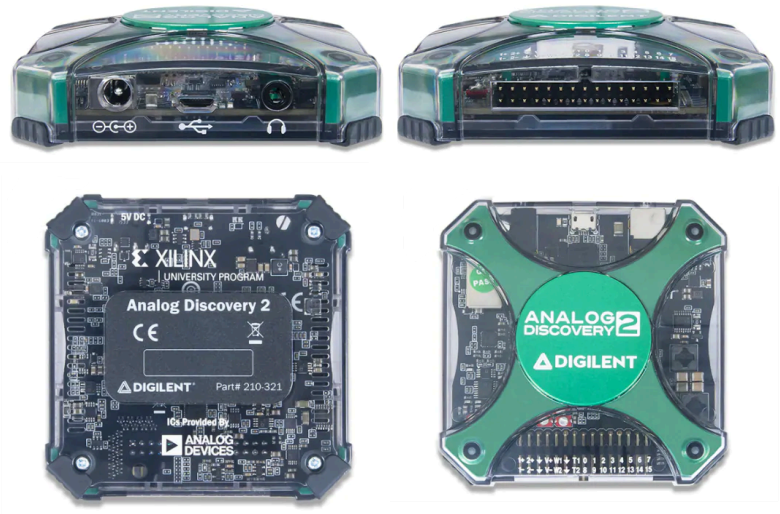


Figure 1: Analog Discovery 2

2. Introduction

Texas A&M University is known for producing top-tier engineers, and this would not be possible without developing students through top-tier coursework, including Principles of Electrical Engineering (ECEN 215). ECEN 215 is a second year engineering course offered to students in disciplines outside of electrical engineering. It introduces students to the fundamentals of circuit analysis and electronics. The course also includes a laboratory component which allows students to apply what they are learning within the classroom to practical applications. Assignments in the lab include using circuit analysis tools like an oscilloscope and waveform generator to build and analyze DC and AC circuits systems. For one to complete the lab assignments successfully they must have a device that can do everything that the lab manual requires.

Usually the students are provided with an Analog Discovery 2 (AD2) to accomplish assigned tasks; however, with the goal of transitioning ECEN 215 to an online course, using an AD2 is not reasonable. It is impractical to expect students to spend upwards of $300 for a device they would most likely use once in their careers. With this being the case, this team has set out to provide an alternate solution. Our goal is to develop and build a device that will function as a miniature oscilloscope, waveform generator, multimeter, and power supply. The device will also be accompanied by a smartphone app that communicates via Bluetooth and will allow users to be able to set output voltage and signals, interact with and view waveforms, measure outputs, etc. Upon completion of this project the ECEN 215 lab will be able to be done from home at an affordable cost.

2.1. Background

The Analog Discovery 2 is a portable test and measurement device that is used in all ECEN 215 labs. When the course begins at the start of the semester these devices are handed out to students to facilitate their completion of the laboratory assignments. The AD2 includes a two-channel oscilloscope that can capture and display analog signals, a waveform generator that can produce various signal outputs, a power supply to provide adjustable power to a circuit, a multimeter to measure current, voltage, resistance, and more. Although the AD2 is a useful and well-made device that has all the features necessary to perform accurate circuit analysis, it is very expensive and does more than what is needed for the ECEN 215 lab. With this in mind, the ECEN 215 Lab Kit project aims to replace the AD2 within the ECEN 215 laboratories.

The ECEN 215 Lab Kit will be designed specifically for the ECEN 215 lab. This means the device will have no unnecessary or overcomplicated features that are beyond what users need to complete their lab assignments. Another reason for replacing the AD2 is to make ECEN 215 a remote course. If ECEN 215 is going to be a remote course the students need a way to be able to perform the lab remotely, and it is unrealistic to expect students that are not studying electrical engineering to purchase a $300 electrical analysis device like the AD2. Therefore, our product offers another solution, affordability. Our device aims to be priced at $50, 1/6th of the price of the AD2.

2.2. Overview

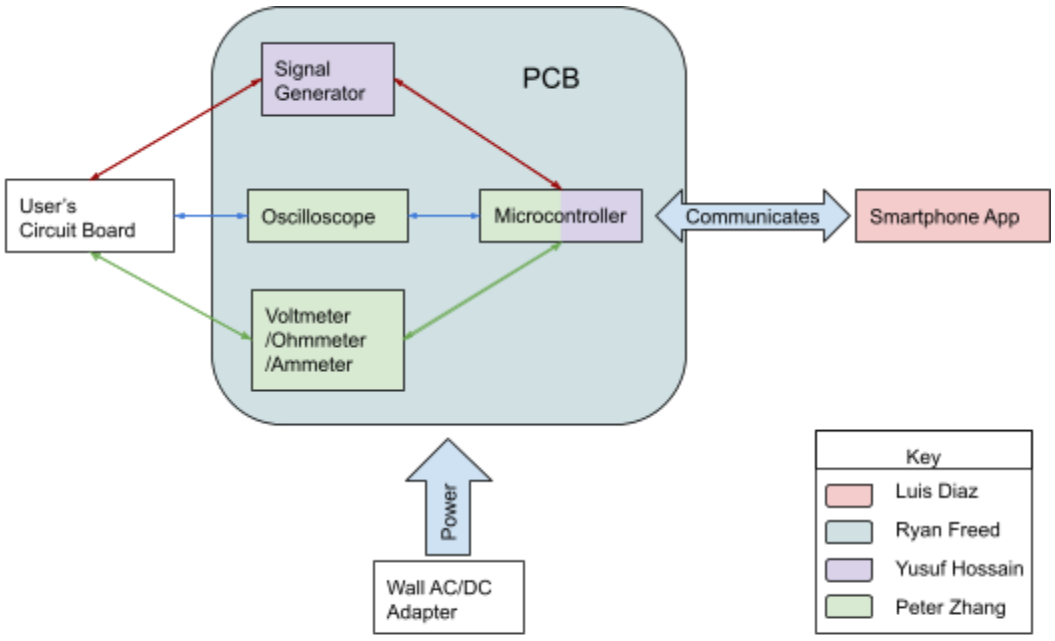


Figure 2: Project Block Diagram

The ECEN 215 Lab Kit will consist of everything needed to ensure user success when performing the assignments given in the laboratory for this course. The main elements our device will consist of are a two channel oscilloscope, a waveform/signal generator, a multimeter, and a power supply.

The two channel oscilloscope will allow the device to read analog signals from the users' circuits and display the signals via Bluetooth communication with the mobile app. Once the oscilloscope reads and sends this data to the app, the user will be able to make use of cursors and other tools to further analyze the data. Users will have the ability to use the waveform generator to choose from square, triangle, or sine waves. The multimeter within the device will be able to measure voltage, amperage, and resistance. All values measured by the multimeter will be displayed on screen within the mobile app. Finally, the ECEN 215 Lab Kit will have the ability to produce -5 to 5 volts.

The mobile application will serve as a digital interface, displaying measurements to users either through numerical values or graphs, depending on the chosen operation. Users can select their intended operation and adjust input signals via different menus and read output values. Additionally, the application will offer an option to enable cursors for more detailed information from any graphs.

2.3. Referenced Documents and Standards

- https://chat.openai.com/
- https://www.youtube.com/watch?v=EnfjYwe2A0w
- https://www.youtube.com/watch?v=g4BvbAKNQ90
- https://www.youtube.com/watch?v=xqgu3VveUrA
- Analog Discovery 2 Hardware Design Guide - Digilent Reference
- A High-Performance Open Source Oscilloscope: development log & future ideas
- What is an Oscilloscope » Electronics Notes
- What is a DMM, Digital Multimeter » Electronics Notes
- Analog Discovery 2 Specifications - Digilent Reference

3. Operating Concept

3.1. Scope

This ECEN 215 portable circuit analysis tool will be a cheap and user-friendly alternative to the Analog Discovery 2. It will be able to generate all of the relevant waveform shapes, frequencies, and amplitudes required for the ECEN 215 class, at an affordable price for students at the entry level. Device deliverables for the proposed device are listed below.

- Multimeter (Voltmeter, Ammeter, Ohmmeter)
- Waveform Generator (Produces sin, square, and triangular waves at 1, 20, and 30 Hz, with amplitudes of 0.5 V, 1 V, and 2 V, and an offset of at least 1 V)
- Power Supply (provides at least + and - 5 V to power user's circuit)
- App (connects to phone via Bluetooth Low Energy (BLE) to allow adjustments to input amplitude and frequency, as well as display output readings)
- Intended to be below $50

3.2. Operational Description and Constraints

This device is intended for the ECEN 215 laboratory at Texas A&M University. It will provide the necessary functions of a portable oscilloscope, and will be powered by a standard American wall outlet. The device requires a compatible smart device (to read waveforms and adjust input values) and a breadboard to operate. The device will provide limited flexibility for input waveforms, as it is tailored specifically for ECEN 215 lab to keep the device as affordable as possible. The device is desired to be at, or below, $50.

3.3. System Description

User Circuit Board: Standard breadboard provided by Texas A&M University in ECEN 215 Lab Kits. Used to create simple circuits that will be powered and monitored by the portable oscilloscope.

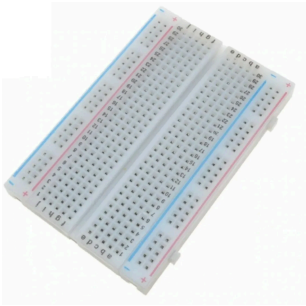


Figure 3: User Circuit Board

Wall AC Adapter: Standard adapter for American wall outlets that will supply power to the device.

Signal Generator: Communicates with the microcontroller to adjust signals and generate waveforms with the following minimum capabilities:

- AC Signals - sine, triangular, and square waves
- AC Input voltage amplitudes - 0.5, 1, and 2 Volts
- AC Frequencies - 1, 20, and 30 Hz
- Power supplies - Provides ± 5 Volts DC

Multimeter: Measures instantaneous voltages, currents, and resistances. Communicates with the microcontroller to display measurement readings.

Oscilloscope: Reads AC input and output waveforms and communicates with the microcontroller to display waveform signals as a function of time.

Microcontroller: Intakes voltages, currents, and resistances from the signal generator, multimeter, and oscilloscope, and “translates them” to the app to display measurements via a graph or number. Intakes signals from the app via Bluetooth and “translates” to the signal generator to increase or decrease voltage signals as necessary. Essentially the microcontroller is a digital interpreter between the system and app to modify and display signals.

App: Digital interface that will display measurements to the user via a time-scaled graph or simple number depending on the selected operation. Will have a menu to select the intended operation (i.e. transient waveform, voltmeter, ohmmeter, etc.). Will have another menu to adjust input signals and read output values. Will have a button to enable cursors, to read x and y values on time scaled graphs.

3.4. Modes of Operations

There are three major modes of operation for this project:

- First, the multimeter mode captures either current, resistance, or voltage and displays the measurement on an app.
- Second, the generation mode produces a DC or AC signal through probes. The mode will be capable of producing a square, sine, and triangle wave along with a DC voltage ranging from -5 to +5 V.
- Finally, the oscilloscope mode reads and displays the voltage as a function of time as a graph through the app.

3.5. Users

Students taking the ECEN 215 course will use the ECEN 215 Lab Kit to complete laboratory assignments. The goal of this project is to provide students working on experiments remotely, or outside of the laboratory, with a cheaper alternative to the nearly $300 AD2. ECEN 215 is an introductory course for non-electrical engineering students, therefore, only the lecture and lab manual are required to operate the device. Additional information such as installing the app and operating each instrument will be included in an electronic user manual. The manual will consist of visual aids to simplify instructions.

3.6. Support

Support for the ECEN 215 Lab Kit will be provided on the project’s GitHub providing information on device setup, app installation, and usage. The GitHub will describe the location of each tool within the app and instructions for each tool’s operation.

4. Scenario

4.1. ECEN 215 Lab Facilitator

The ECEN 215 Lab Kit is intended to only be used by students taking ECEN 215. Therefore the kit's instruments and specifications are designed to be compatible with labs conducted in ECEN 215. In addition, the ECEN 215 Lab Kit simplifies the process of completing each assignment. Before the user would require a separate power supply, multimeter, waveform generator, and oscilloscope to complete every lab. Even if the user had a device which combined the instruments above, such a device would cost about $300, and with the lab kit, these are combined into a single device expected to cost about $50.

5. Analysis

5.1. Summary of Proposed Improvements

- The system will be much cheaper than the previously used AD2
    - Instead of roughly $300, it will ideally be around $50 per unit
- Extra kits will be available in case of unforeseen damages without much extra cost
- The kits will be tailored for ECEN 215 labs with preset values for each lab (where applicable), leaving little room for error when inputting and extracting data
- There will be an app to accompany the kit that displays graphs and values on any compatible smart device with Bluetooth capability, as opposed to needing a physical connection to a laptop

5.2. Disadvantages and Limitations

- The system is optimized for ECEN 215 so any modification to the lab curriculum could lead to incomplete or unusable data
- Higher latency from a Bluetooth connection, as opposed to a wired connection, means delayed response time between the kit and the device
- There are no physical dials or buttons on the device that could be used for the fine-tuning of values
- An external brick will be used as the AC/DC converter possibly taking up more workspace

5.3. Alternatives

- Breaking down the kit into its core components (Oscilloscope, Wave Generator, Multimeter, Power Supply) and using them separately
    - While the measurements would be more precise and accurate, it would cost significantly more and there would be at least four different devices to worry about instead of one
    - There would be no app where the data could be stored and extracted from, only physical values would be available

- Using an AD2
    - Measurements would be more accurate but the device would be more expensive
    - Students in ECEN 215 would not need a majority of the features offered by the AD2 increasing the likelihood of making errors when looking for certain features

5.4. Impact

- The ECEN 215 Lab Kit improves the course's accessibility for remote learning
- The low cost of the device allows students to afford the device for personal use or further academic use
- The low manufacturing cost and simplified lab kit reduces the number of different components used in the device

ECEN 215 Lab Kit

Luis Diaz-Santini
Ryan Freed
Yusuf Hossain
Peter Zhang

FUNCTIONAL SYSTEM REQUIREMENTS

REVISION – Final
2 December 2024

Functional System Requirements
for
ECEN 215 Lab Kit

Prepared by:

_____
Author Date

Approved by:

_____
Project Leader Date

_____
John Lusher, P.E. Date

_____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
-	2/12/23	Luis Diaz-Santini, Peter Zhang		Beginning Draft
	2/19/24	Luis Diaz-Santini, Peter Zhang		Edits
	2/23/24	ECEN 215 Lab Kit		Final Edits before Midterm Submission
	12/2/2024	ECEN 215 Lab Kit		Update and Review for 404 Submission

Table of Contents

List of Tables

List of Figures

1.Introduction

1.1. Purpose and Scope

At Texas A&M University, Principles of Electrical Engineering (ECEN 215) is taught to engineering students outside of the electrical and computer engineering program. The course teaches the fundamentals of circuit analysis and electronics, which is accompanied by a laboratory (lab) component where students physically build and analyze circuits. However, distance learning students face challenges in completing ECEN 215's lab portion due to the required instruments' high cost. The ECEN 215 Lab Kit aims to provide a cost-effective alternative to tools traditionally used in these laboratory activities. It will measure and transmit current, resistance, and voltage values to a microcontroller which are subsequently processed and displayed to a smartphone application. For accessibility, the device should cost around $50 and be no larger than a cellular phone. A portable wall charger will power the device. The lab kit will be capable of completing laboratory assignments 1 through 8 by operating as an oscilloscope, a waveform generator, and a multimeter. Figure 1 shows a representative integration of the project in the proposed CONOPS. The verification requirements for the project are contained in a separate Verification and Validation Plan.

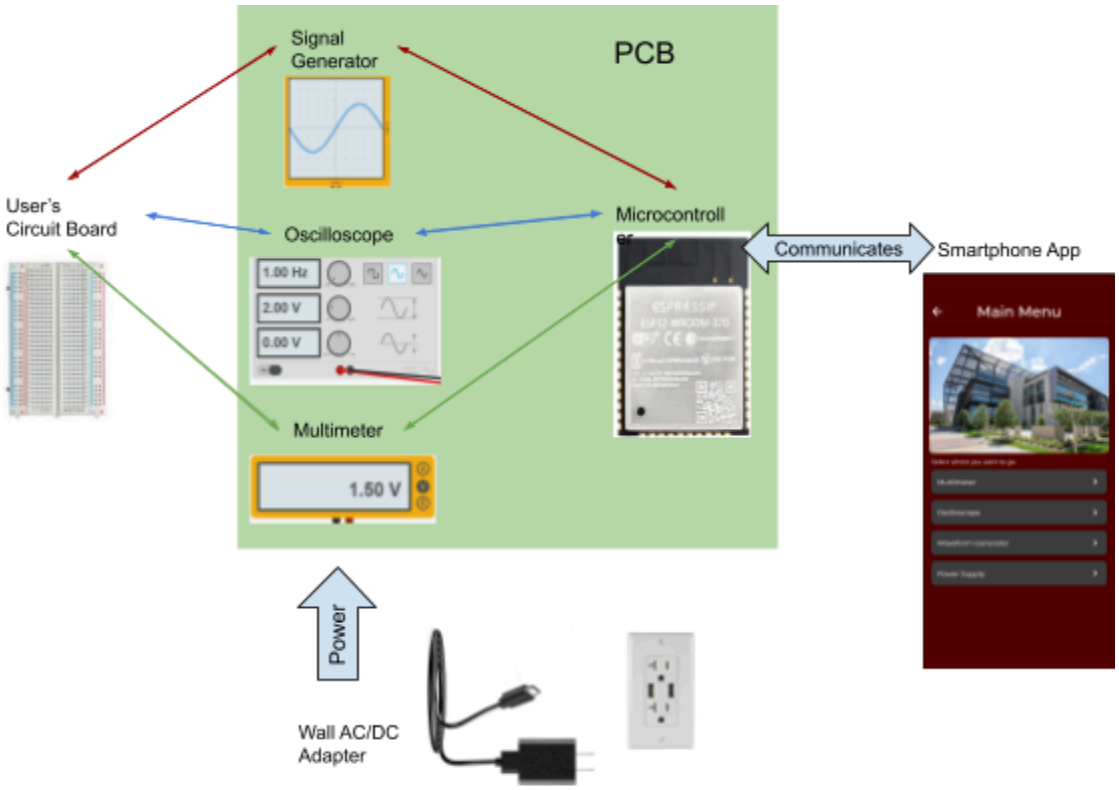


Figure 1: Project Conceptual Image

1.2. Responsibility and Change Authority

The team leader, Peter Zhang, is responsible for guaranteeing all requirements of the project are met. If any changes are made to these requirements, they must first be approved by the client, Dr. John Lusher, and the team leader. The following table identifies each member and the subsystem each is responsible for.

Subsystem	Responsible Member
Mobile Application	Luis Diaz-Santini
Oscilloscope and Voltmeter	Peter Zhang
PCB Design	Ryan Freed
Waveform Generator	Yusuf Hossain
Microcontroller	Yusuf Hossain/Peter Zhang

Table 1: Member Subsystem Responsibilities

2. Applicable and Reference Documents

2.1. Applicable Documents

The following documents, of the exact issue and revision shown, were referenced in the design and manufacturing of this product:

- ChatGPT
- Electronics Basics ADC
- Analog Discovery 2 Hardware Design Guide - Digilent Reference
- A High-Performance Open Source Oscilloscope: development log & future ideas
- Analog Discovery 2 Specifications - Digilent Reference
- Creating an App for Interacting with IoT Devices using BLE and FlutterFlow
- IEEE 802.15.1 Standards

2.2. Order of Precedence

In the event of a conflict between the text of this specification and an applicable document cited herein, the text of this specification takes precedence without any exceptions.

All specifications, standards, exhibits, drawings or other documents that are invoked as “applicable” in this specification are incorporated as cited. All documents that are referred to within an applicable report are considered to be for guidance and information only, except ICDs that have their relevant documents considered to be incorporated as cited.

3. Requirements

3.1. System Definition

The ECEN 215 Lab Kit is a portable circuit analyzer capable of functioning as an oscilloscope, multimeter, waveform generator, and power supply to facilitate ECEN 215's laboratory component. This device is divided into four primary subsystems, as shown in Figure 2. These subsystems include the multimeter and oscilloscope, waveform generator, PCB and power, and mobile application. The microcontroller is a minor subsystem primarily shared between the oscilloscope, waveform generator, and mobile application.

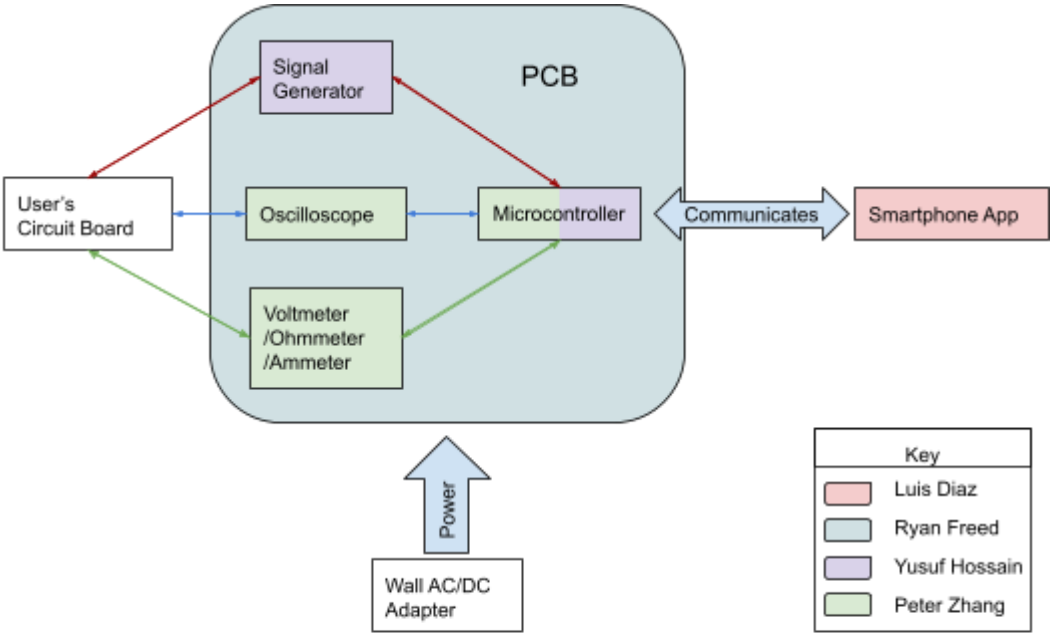


Figure 2: Block Diagram of System

The multimeter and oscilloscope subsystems operate in their respective modes. In the oscilloscope mode, the device obtains a signal that displays a transient waveform on the user's mobile device. The multimeter mode will measure either the current, resistance, or voltage and display the reading on the app. Any processing or transmission of data is handled through the microcontroller. The waveform generator subsystem allows users to output signals such as a DC voltage or a sine, rectangle, and triangle wave, communication to the subsystem and app is provided by the microcontroller. To utilize each instrument, the user will select, on the mobile application, which operation the lab kit will perform. The app wirelessly communicates the chosen function to the microcontroller. Additionally, the app subsystem designs the user interface. Finally, the PCB is responsible for integrating the multimeter, oscilloscope, power supply, and microcontroller into a small and portable device.

3.2. Characteristics

3.2.1. Functional / Performance Requirements

3.2.1.1. Mobile Application

This project aims to provide the end user with an attractive and responsive app that will serve as the user's interface with the ECEN 215 Lab Kit. The mobile application will communicate with the device via bluetooth low energy. The user will be able to navigate to multiple menus to assist in their circuit analysis. Currently the application will have four menus: oscilloscope, multimeter, waveform generator, and power supply. At any time the user can navigate to and between these menus to accomplish the tasks assigned by the lab manual. Within each menu different options will be presented to the user to assist them in the analysis of their system.

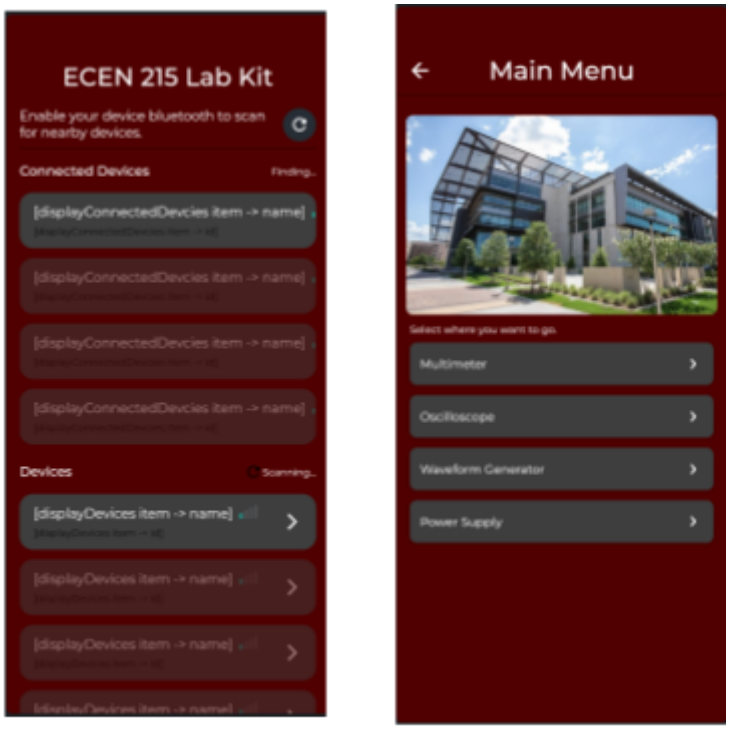


Figure 3: Application Menus

3.2.1.2. Oscilloscope

The ECEN 215 Lab Kit will be equipped with the ability to read voltage from the user's circuit and display that data as a function of time to produce a waveform graph. The graph will be displayed via the application on the user's smartphone. When viewing the waveform the user will have the option to acquire several parameters of data to further analyze the signal being received.

3.2.1.3. Signal Generator

Within the application the user can set parameters to output a certain signal to their circuit. The device will have the capability of generating any sine, square, or triangle signals with the frequency range of 1Hz, 20Hz, and 30Hz.

> *Rationale: The ECEN 215 lab manual states that only square, sine, and triangle signals need to be generated. The only frequencies needed to be generated are 1Hz, 20Hz, and 30Hz.*

3.2.1.4. Multimeter

The multimeter within the device will be able to read any voltage, resistance, and current that the user's circuit is outputting. The reading will be displayed on the application in the appropriate menu, and it is up to the user to tell the device what parameter is being measured in order to obtain an accurate reading.

3.2.1.5. Power Supply

The power supply will have two voltage outputs -5V and 5V .

> *Rationale: The ECEN 215 lab manual only requests a supply of either -5V or 5V.*

3.2.2. Physical Characteristics

3.2.2.1. Mass

The ECEN 215 Lab Kit is aiming to be less than or equal to 8.5 ounces.

> *Rationale: The AD2 weighs 8.5 ounces, we aim to be lighter since the lab kit will be a simpler device.*

3.2.2.2. Volume Envelope

The volume envelope of the ECEN 215 Lab Kit aims to be less than or equal to 2 inches in height, 3 inches in width, and 6 inches in length.

> *Rationale: One of the goals in making this device is to make it extremely portable, and at this size the device would be the size of a smartphone.*

3.2.2.3. Device Enclosure

Since the actual device will be a PCB with different components mounted onto it, the device will be enclosed in a protective casing. The casing will be a lightweight but durable plastic perfect for everyday use. The design will ensure quick access to all I/O, while the rounded edges will provide a comfortable grip.

3.2.3. Electrical Characteristics

3.2.3.1. Inputs

3.2.3.1.1 Power Consumption

The maximum peak power of the system shall not exceed 36W.

3.2.3.1.2 Input Voltage Level

The input voltage level for the ECEN 215 Lab Kit will be +12V.

Rationale: To power the device an AC to DC converter that takes 120 VAC and converts it to 12VDC will be used.

3.2.3.1.3 Application Input

The signal generation menu of the application will take in user input to output the requested signal generation. Within the power supply menu the user's input will determine what voltage is supplied. Within the multimeter menu the user's input will determine what parameter the device measures. Within the signal generator menu the user's input will determine what type of signal is output to the circuit through the aforementioned probes.

3.2.3.1.4 Oscilloscope and Multimeter

The ECEN 215 Lab Kit will measure circuit parameters using probes and display this input on the oscilloscope menu as a waveform or as a measured value within the multimeter menu. Listed below are the parameters that can be measured using the multimeter.

- **Voltage**
- **Amperage**
- **Resistance**

3.2.3.1.5 Signal Generator

As stated before the signal generation is based on the user input within the app. Below are all the listed inputs that the applications requests of the user to be able to generate a desired signal.

- **Signal Type**
- **Signal Amplitude**
- **Signal Frequency**
- **Signal Phase Shift**
- **Signal Offset**

Rationale: All parameters listed are what is required to generate an output signal.

3.2.3.2. Outputs

3.2.3.2.1 Oscilloscope Waveform

As the lab kit takes in the voltage from the user's circuit, a waveform graph will be output on the application. With the information given by the oscilloscope the user will be able to calculate the time constant of the graph. Along with displaying the waveform, the user has the options to view the parameters listed below. For the user to obtain and output these parameters the data received from their circuit will be analyzed by the system and output back out.

- **Amplitude**
- **Frequency**

Rationale: The ECEN 215 lab only requires the student to measure the parameters listed above.

3.2.3.2.2 Measured Values from Multimeter

Depending on user input the device will connect to and read these parameters from the device then output these values to the application. Below are the parameters that the device will be able to output.

- **Voltage**
- **Amperage**
- **Resistance**

Rationale: The ECEN 215 lab only requires the student to measure the parameters listed above.

3.2.3.2.3 Signal Generated

The ECEN 215 Lab Kit will be equipped with the ability to output certain signals, depending on the user's input in the application. The signal generated will be supplied to the user's circuit through connections between the device and their circuit.

3.2.3.3. Connectors

3.2.3.3.1 Wall Power

The ECEN 215 Lab Kit will be powered by household AC power. The device will be furnished with an AC to DC adapter that will be able to connect to any 120VAC US outlet.

3.2.3.3.2 Leads

The ECEN 215 Lab Kit will come with several leads, of different colors. These leads will be used to communicate and transfer any voltage, signals, or circuit information between the device and the user's circuit board.

3.2.3.4. Wiring

The only wiring that will be present on the system will be the aforementioned leads, probes, and wall power cord. The internals of the system will be integrated on a PCB, meaning there is no need for internal wiring.

3.2.4. Environmental Requirements

The ECEN 215 Lab Kit shall be designed to operate in the causal environments and laboratory settings where there is not much external strain introduced to the device.

Rationale: Students will use this device in a casual, household setting.

3.2.4.1. Thermal

No components of the ECEN 215 Lab Kit will be consuming enough power that would lead the device to overheat. That being said the heat being output by each component in the device will not be enough that would cause the device to run hotter than room temperature.

3.2.4.2. Weather

The ECEN 215 Lab Kit is intended for indoor use only and will not be rated to withstand water damage. The device will be designed to withstand temperatures from -40 degrees celsius to 105 degrees celsius.

3.2.5. Failure Propagation

Since the ECEN 215 Lab Kit is a small scale device there is not much to be done besides restarting the system if it were to crash or become unresponsive. In the case of a crash the device will have a RESET button that the user can press at any time to power cycle the system.

4. Support Requirements

The ECEN 215 Lab Kit requires a Bluetooth connection to display measured values. Users must provide a device capable of both Bluetooth connectivity and downloading mobile applications. The application will support both Android and IOS devices. Support regarding installing or connecting with the mobile application will be included in the project's GitHub. The lab kit consists of an oscilloscope, an ammeter, an ohmmeter, a voltmeter, a waveform generator, an AC adapter, and a microcontroller. While an AC adapter is included in the kit, continuous access to a wall outlet is required for the device to operate.

Appendix A: Acronyms and Abbreviations

AC	Alternating Current
AD2	Analog Discovery 2
DC	Direct Current
ECEN	Electrical and Computer Engineering
Hz	Hertz
I/O	Input/Output
PCB	Printed Circuit Board
V	Measure Voltage
VAC	Measured Alternating Current Voltage
VDC	Measured Direct Current Voltage
W	Watts

Appendix B: Definition of Terms

No terms need to be defined at this time.

ECEN 215 Lab Kit

Luis Diaz-Santini
Ryan Freed
Yusuf Hossain
Peter Zhang

INTERFACE CONTROL DOCUMENT

REVISION – Final
2 December 2024

Interface Control Document
for
ECEN 215 Lab Kit

Prepared by:

_____
Author Date

Approved by:

_____
Project Leader Date

_____
John Lusher II, P.E. Date

_____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
-	2/12/23	Ryan Freed, Yusuf Hossain		Beginning Draft
	2/19/24	Ryan Freed, Yusuf Hossain		Edits
	2/23/24	ECEN 215 Lab Kit		Final Edits before Midterm Submission
	12/2/2024	ECEN 215 Lab Kit		Update and Review for 404 Submission

Table of Contents

List of Tables

List of Figures

1. Overview

The Interface Control Document for the ECEN 215 Lab Kit will cover the physical description and characteristics of each component used, the connectivity between each subsystem, power requirements for the whole kit, and communication between the user and the board.

2. References and Definitions

2.1. References

The following documents/resources, of the exact issue and revision shown, were referenced in the design and manufacturing of this product:

- ChatGPT
- Electronic Basics #27: ADC (Analog to Digital Converter)
- Analog Discovery 2 Hardware Design Guide - Digilent Reference
- A High-Performance Open Source Oscilloscope: development log & future ideas
- Analog Discovery 2 Specifications - Digilent Reference
- Creating an App for Interacting with IoT Devices using BLE and FlutterFlow
- https://standards.ieee.org/ieee/802.15.1/1180/

3. Physical Interface

3.1. Weight

One of the key design features of the ECEN 215 Lab Kit is to ease the process of completing the labs at home. This means that the device must be portable and comfortable to carry around. With a small and light frame both of these can be accomplished.

System	Weight
2 ADCs (MCP3201-CI/SN)	0.0381oz
ESP32-WROOM-32D	0.257 oz
ADC (ADC141S626CIMM/NOPB)	0.005051 oz

Table 1: Weight Specifications

3.2. Dimensions

Making the lab kit as compact as possible without cutting corners in any of the subsystems is the goal. Ideally, the whole system would fit on a PCB and frame no bigger than the average smartphone, these dimensions are 152mm x 76mm.

System	Length	Width
ADC (MCP3201-CI/SN)	4.9mm	3.9mm
ESP32-WROOM-32D	18mm	25.5mm
ADC (ADC141S626CIMM/NOPB)	3mm	3mm

Table 2: Dimension Specifications

3.2.1. Dimension of Power System

Having the AD/DC converter on the PCB could cause unwanted noise and can generate excess heat damaging neighboring components, because of this it is best to have it as an external component. The dimensions for the external power brick are 114.3mm x 50.8mm.

3.3. Mounting Locations

The entire circuit will be mounted onto a PCB. The lab kit will also contain ports that can be used for wire, leads, or probes. There will also be a port for the power connector. The designed PCB and all the ports that are accounted for will be placed in a shell that has yet to be designed.

4. Thermal Interface

There are no plans for a thermal interface at this time, however, in the future, we may include air vents on the casing of the board.

5. Electrical Interface

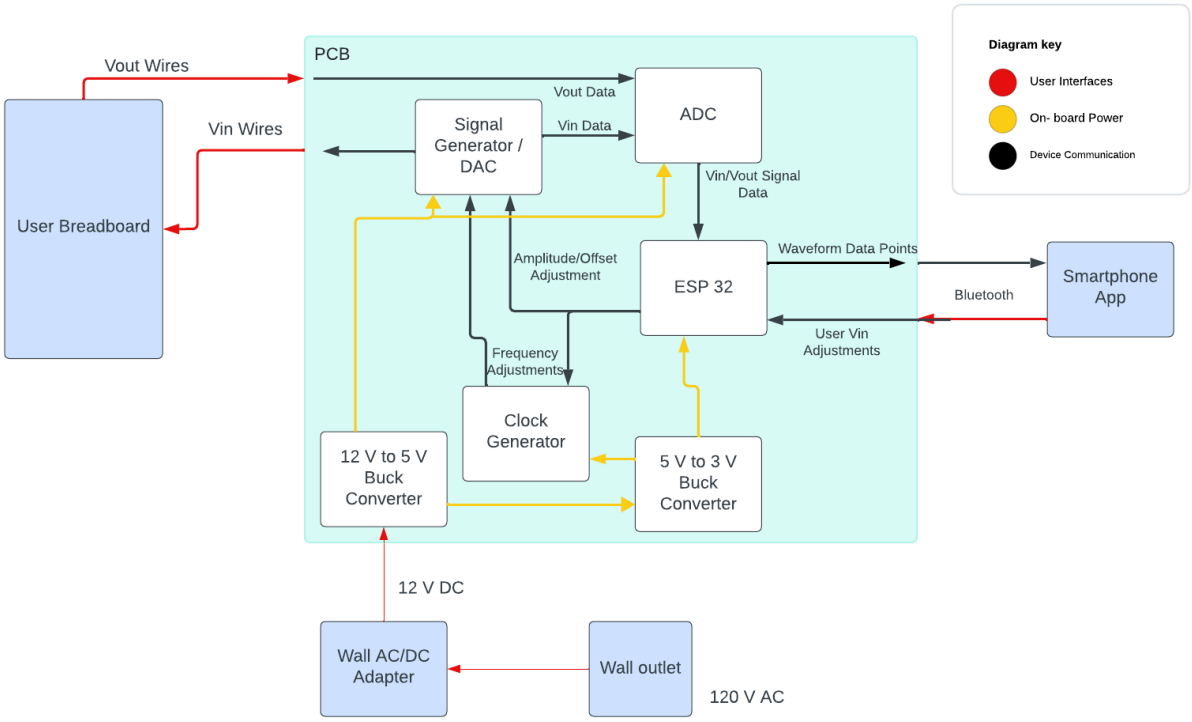


Figure 1: Electrical Systems Block Diagram

5.1. Primary Input Power

For user convenience, the kit will come with an AC/DC wall adapter for standard American wall outlets that will directly plug into the kit. The adapter will be rated for 12V/3A. The 12V taken at input will run through a buck converter that will drop down the voltage to both 5 and 3 V for use in the ESP32, Signal generator, Clock, and ADC.

5.2. Voltage and Current Levels

System	Input Voltage Range	Input Current (idle)	Input Current (max)
ADC (MCP3201-CI/SN)	2.7-5.5 V	500 nA	0.4 mA
ESP32-WROOM-32D	-0.3-3.6 V	0.5 A	1.1 A
ADC (ADC141S626CIMM/NOPB)	4.75-5.25 V	2 uA	50 mA
Converter (LM2611BMFX)	2.7-14 V	1 uA	2 A

Table 3: Voltage and Current Levels

5.3. Signal Interfaces

All chips will be mounted to a PCB and will be controlled via signals set from pins on the microcontroller..

5.4. User Control Interface

The user will control this device through a mobile application. The app will give access to input voltage waveform settings such as amplitude, frequency, and shape, and will allow the user switch between modes of operation such as, scope, waveform generator, multimeter, and power supply.

- Scope will allow the user to view the transient response of the input and output waveforms, as well as measure the time and frequency during the response.
- Waveform generator will allow the modification of the input waveform attributes.
- Multimeter will allow the measurement of voltage, current, and resistance.
- Power supply will allow the user to enable the +5 and -5 volt supply lines.

6. Communications / Device Interface Protocols

6.1. Bluetooth

A mobile phone app will be responsible for accepting user selections through a user interface and will transmit the signal to the ESP32 microcontroller (see Section 6.2) via Bluetooth (BQB Certified). User operation modes will include, power supply, oscilloscope, voltmeter, ammeter, ohmmeter, and waveform generator (User operation modes further detailed in Section 5.5).

6.2. Microcontroller Input and Output

The ESP32 microcontroller will be responsible for all internal communications. The ESP32 will receive signals via Bluetooth (see section 6.1) and transmit signals to the appropriate systems to manipulate the associated voltage signal

ECEN 215 Lab Kit

Luis Diaz-Santini
Ryan Freed
Yusuf Hossain
Peter Zhang

Schedule and Validation

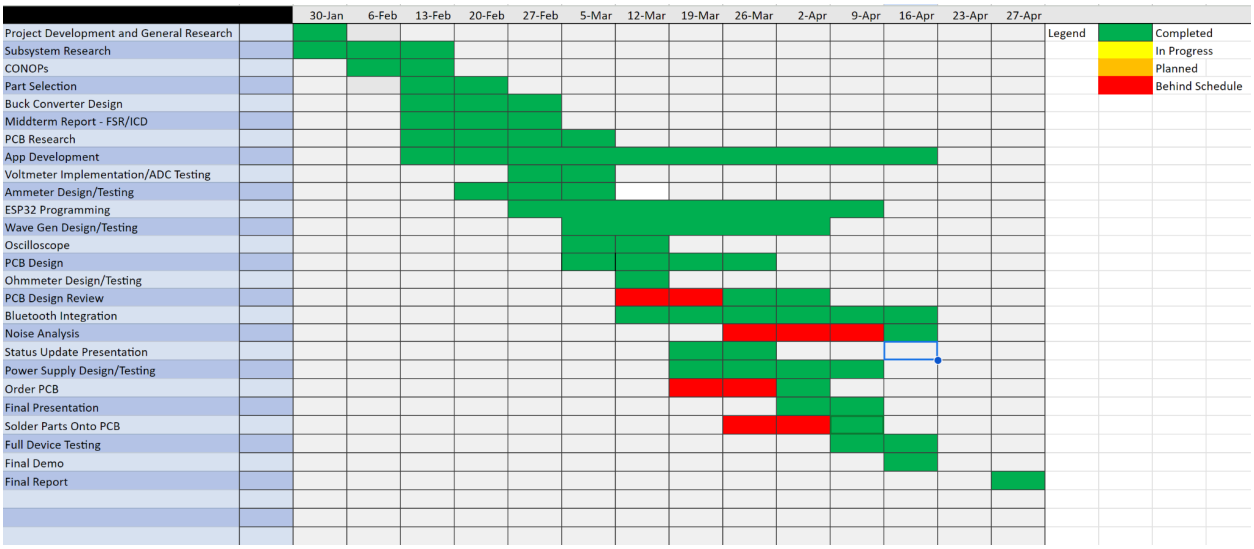
Validation Plan

Test	FSR Paragraph #	Owner	Status	Success Criterion	Methodology
Buck Converter 12 -> 5 V	3.2.3.3.1	RF	Succ…	Intakes 12 V DC outputs 5 V DC	Measured with multimeter
Buck Converter 5 -> 3 V	3.2.3.3.1	RF	Succ…	Intakes 12 V DC outputs 3 V DC	Measured with multimeter
Buck Converter 5 -> -5 V	3.2.3.3.1	RF	Succ…	Intakes 5 V DC outputs -5 V DC	Measured with multimeter
Wave Gen Powers On	3.2.3.1.5	YH	Succ…	Outputs voltage when provided 5 V supply power	Measured with multimeter
Wave Gen Amplitude	3.2.3.1.5	YH	Succ…	Outputs 0.5, 1, and 2 V successfully	Measured with multimeter
Wave Gen Frequency	3.2.1.3	YH	Succ…	Outputs frequencies 1, 20, and 30 Hz	Measured with oscilloscope
Wave Gen Output Shape	3.2.1.3	YH	Succ…	Produces sine, square, and triangular waves at all 3 amplitudes and frequencies	Measured with oscilloscope
Wave Gen Noise	3.2.3.2.1	YH	Succ…	Output signal is moderately accurate	Measured with oscilloscope
Ammeter	3.2.1.4	PZ	Succ…	Able to read 0.5 A of current	Measured with multimeter
Ohmmeter	3.2.1.4	PZ	Succ…	Able to read 100 ohms and 10k ohms	Measured with multimeter

Duty Cycle	3.2.3.1.5	YH	Succ…	Can produce a square wave with 50% duty cycle	Measured with oscilloscope
App Connects to Bluetooth	3.2.1.1	LDS	Succ…	App connects to ESP32	Observation from app and code terminal
App Receives/Sends Data from/to ESP32	3.2.1.1	LDS	Succ…	ESP32 reacts to commands sent from the app / App displays message from ESP32	Observation from app and code terminal
App - Multimeter Function	3.2.3.1.3	LDS + PZ	Succ…	App displays correct parameter with correct value	Observation from app and code terminal
App - Oscilloscope Function	3.2.3.1.3	LDS + PZ	Succ…	App displays accurate graph and can display requested parameters	Compare to actual oscilloscope
App - Waveform Gen Function	3.2.3.1.3	LDS + YH	Succ…	App takes in appropriate settings and communicates with ESP32 to output correct signal	Observation from app, code terminal, and oscilloscope
App - Power Supply Function	3.2.3.1.3	LDS + RF	Succ…	App triggers Lab Kit to output power	Measured with multimeter and observed from app
Switching Menus Within App	3.2.1.1	LDS	Succ…	Can switch to a different menu within app while not disabling previous action	Observation from app and ESP32
App Works on IOS	3.2.1.1	LDS	Succ…	App is downloaded and operates with all functionality	Observation from phone
App Works on Android	3.2.1.1	LDS	Succ…	App is downloaded and operates with all functionality	Observation from phone
ADC Power	3.2.1.2	PZ	Succ…	ADC powers on and interprets analog signal into binary output correctly	Observation from

ADC Speed	3.2.1.2	PZ	Succ…	ADC reads voltages fast enough to produce a smooth sine wave of at least 60 Hz	Observed through use of application and real oscilloscope
ADC Amplitude	3.2.1.2	PZ	Succ…	ADC can read between 0, 2.5 and 5 V	Observation from app and code terminal
ESP32 Power	3.2.3.1.1	RF	Succ…	Turns on with correct input signals	Observation code terminal
ESP32 Communicates With App	3.2.1.1	Tea m	Succ…	ESP32 communicates via Bluetooth and sends/receives data from the app	Observation from app and code terminal
ESP32 Communicates With Wave Gen	3.2.3.1.5	Tea m	Succ…	ESP32 sends correct information to waveform generator	Observation from app, code terminal, and oscilloscope
ADC Communicates With ESP32	3.2.1.2	Tea m	Succ…	ESP32 correctly reads digital input from ADC	Observation from code terminal
ESP32 Communicates With Power Supply	3.2.1.5	Tea m	Succ…	ESP32 triggers circuit to output power	Observation from multimeter and terminal
Kit runs at max current/voltage for 30 minutes	3.2.4	Tea m	Succ…	Kit can run at max outputs for extended period of time without catastrophic device failure or inappropriately excess noise	Validated by validating everything else

Gantt Chart



ECEN 403 Gantt Chart

ECEN 404 Gantt Chart

ECEN 215 Lab Kit

Luis Diaz-Santini
Ryan Freed
Yusuf Hossain
Peter Zhang

SUBSYSTEM REPORTS

REVISION – Final
2 December 2024

Subsystem Reports
for
ECEN 215 Lab Kit

Prepared by:

_____
Author Date

Approved by:

_____
Project Leader Date

_____
John Lusher II, P.E. Date

_____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
-	4/20/23	ECEN 215 Lab Kit		Beginning Draft
	4/24/24	ECEN 215 Lab Kit		Edits
	4/26/24	ECEN 215 Lab Kit		Edits
	4/27/24	ECEN 215 Lab Kit		Final Edits before Final Submission
	12/2/2024	ECEN 215 Lab Kit		Update and Review for 404 Submission

Table of Contents

List of Tables

List of Figures

1. Introduction

The ECEN 215 Lab Kit is a portable and inexpensive device for students taking the course Principles of Electrical Engineering (ECEN 215). The device will perform as an oscilloscope, a multimeter, and a waveform generator. It will also be accompanied by a smartphone app that communicates via Bluetooth for users to be able to operate the waveform generator or measure outputs. The system is broken down into the following subsystems: board design and power, mobile Bluetooth application, waveform generator and MCU, and multimeter and oscilloscope. Since each subsystem was validated to be working properly and fulfilling all requirements, there is a clear path to integration for these subsystems into the full system specified in the Concept of Operations, Functional System Requirements, and Interface Control Document.

2. Board Design and Power Report

2.1. *Subsystem Introduction*

The board design and power subsystem is responsible for uniting all physical aspects of the device, including ensuring proper communication between devices, powering on all board components, maintaining size restrictions, maintaining a portion of the budget, and finally assembly of all board components onto the PCB. The PCB is powered primarily by a standard 12 V, 3 A, barrel jack, AC adapter that plugs directly from a wall outlet and into the board. The AC adapter then leads to 3 separate converters that provide 5 V, 3.3 V, and -5 V. The power system was tested for reliability and consistency as well as to test the efficacy of the design and assembly of the board.

2.2. *Subsystem Details*

A block diagram of the proposed board design and interactions is included below.

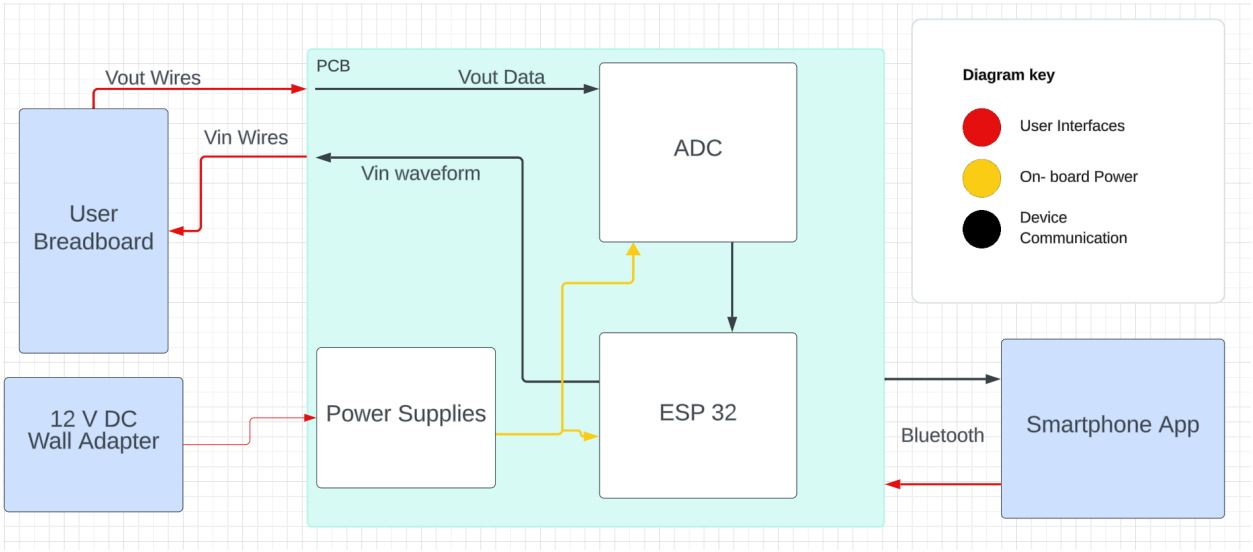


Figure 1: Simplified Diagram of PCB Make-up and Power

Additional tasks associated with this subsystem include Altium design and soldering, selecting all small-scale intermediate components (i.e. resistors, capacitors, inductors, etc.), determining the PCB I/O methods and components to facilitate these interactions, designing a 3D printed case in Fusion 360, and troubleshooting assistance for all physical implementations of the systems, and design of a switching system for a +5 and -5 I/O wire.

2.3. *Board Design*

All PCB design was performed using Altium Designer. The schematic level design of each system is included below for a detailed view of the device makeup.

Many challenges that occurred to the previous board were, as expected, issues with PCB assembly, as a result of the small size needed to accommodate mobility and convenience. The issues that were shorting the buck converters and damaging the board were confirmed

to be the results of solder bridging and were promptly fixed. However, many revisions were still necessary for the original design of the PCB, including size reductions, a complete redesign of the multimeter/oscilloscope and waveform generator system, and the introduction of safety devices to protect the onboard ICs from overcurrent. As a result of this, 3 models of the PCB were created. The first model verified the IC powering systems, which were still larger than requested. The second design was made to reduce the size to within specifications and remove the waveform generator chip entirely, as the ESPs onboard DAC were decided to be suitable for applications in the lab and would reduce the overall price of the product. The third and final design introduced a complete redesign of the multimeter/oscilloscope system, buffer amplifiers to protect the onboard IC's, and an amplifier for the waveform generator system to increase the input signal's amplitude above the ESP's possible 3.3 volts. The below figures highlight the design of the final implementation of the PCB.

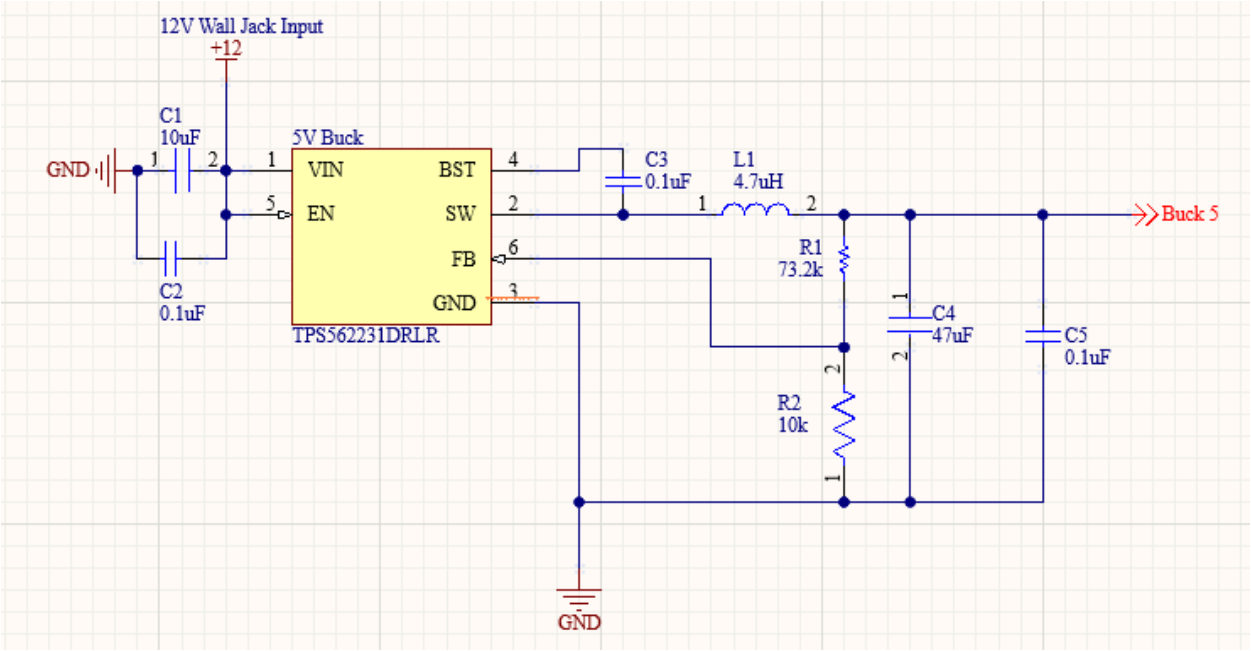


Figure 2: Schematic Level Design of 5 V Buck

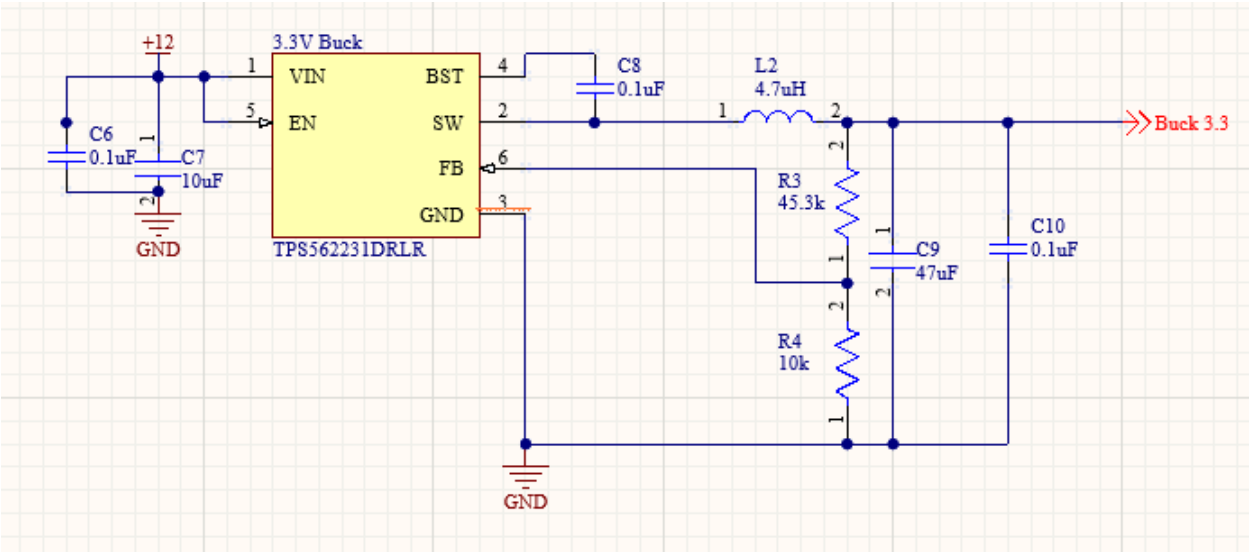


Figure 3: Schematic Level Design of 3.3 V Buck

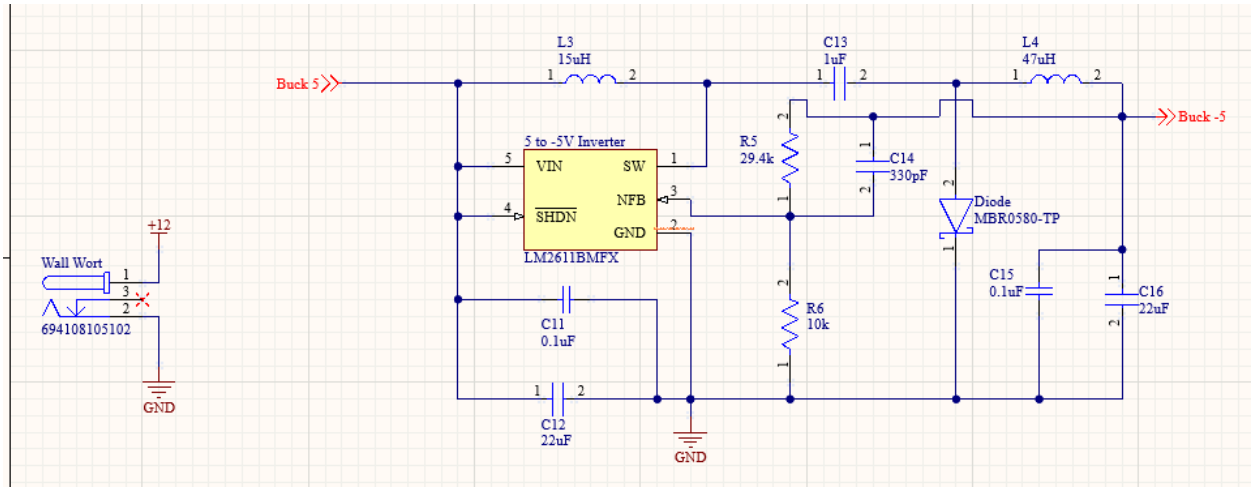


Figure 4: Schematic Level Design of -5 V Converter and Power Jack

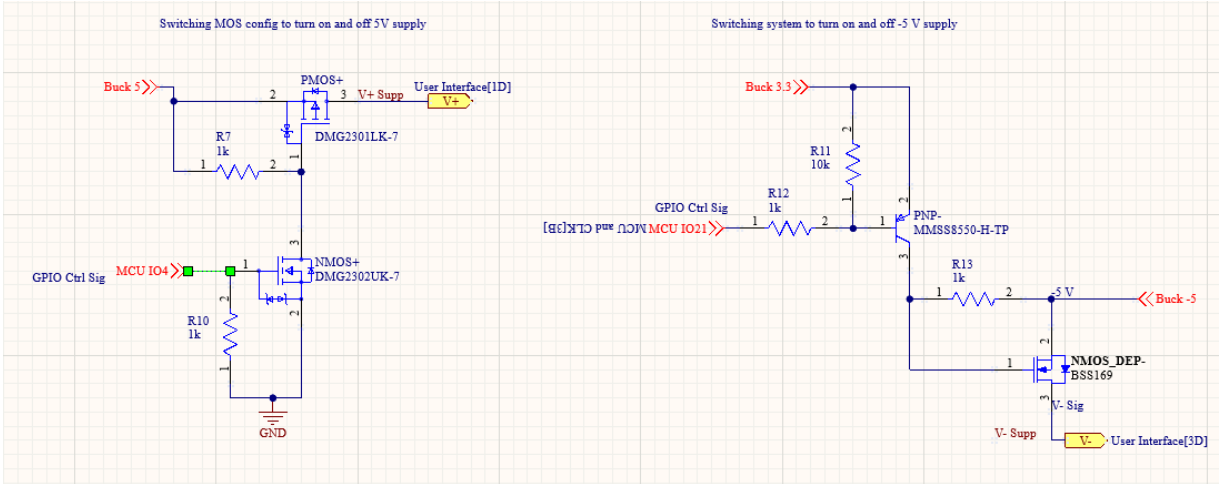


Figure 5: Schematic Level Design of +5 and -5 V Power Supply Switches

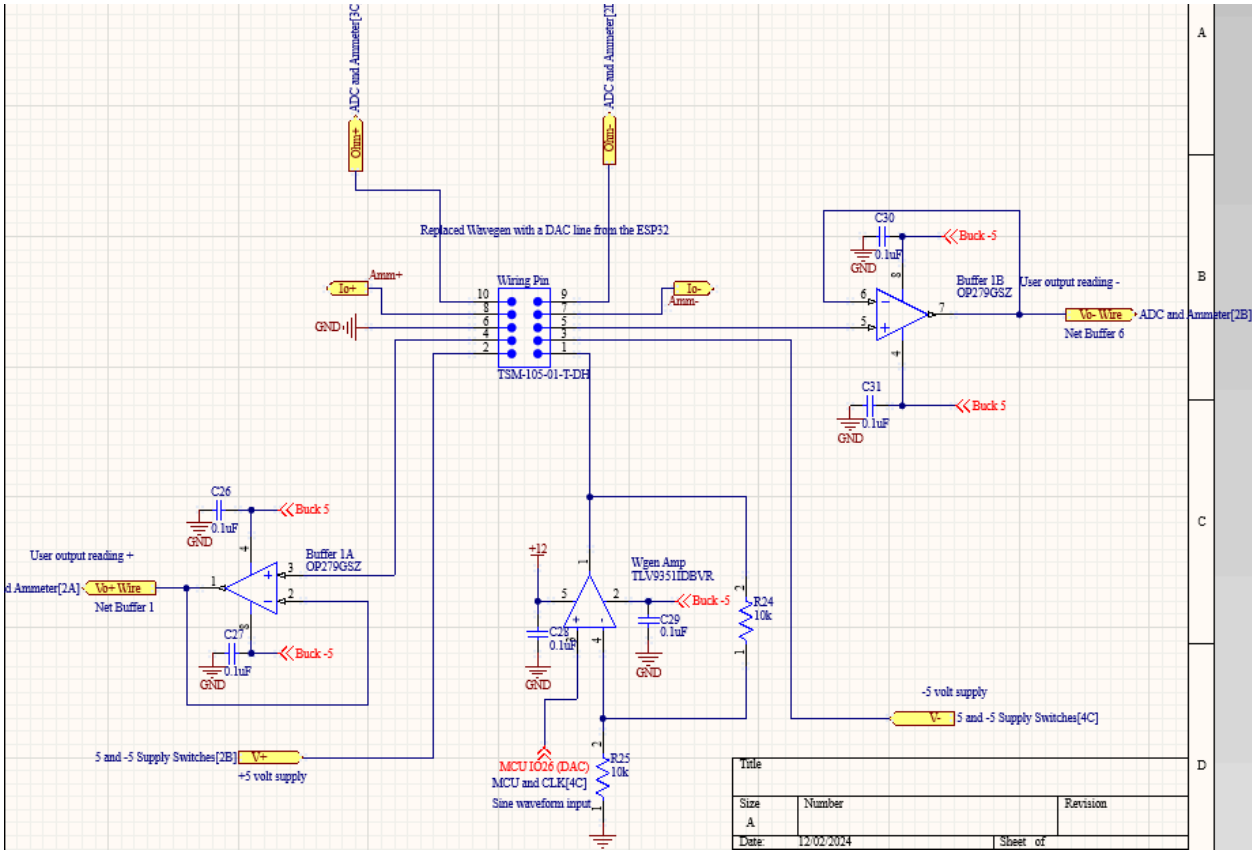


Figure 6: Schematic Level Design of Wiring Pin and Protection Circuitry

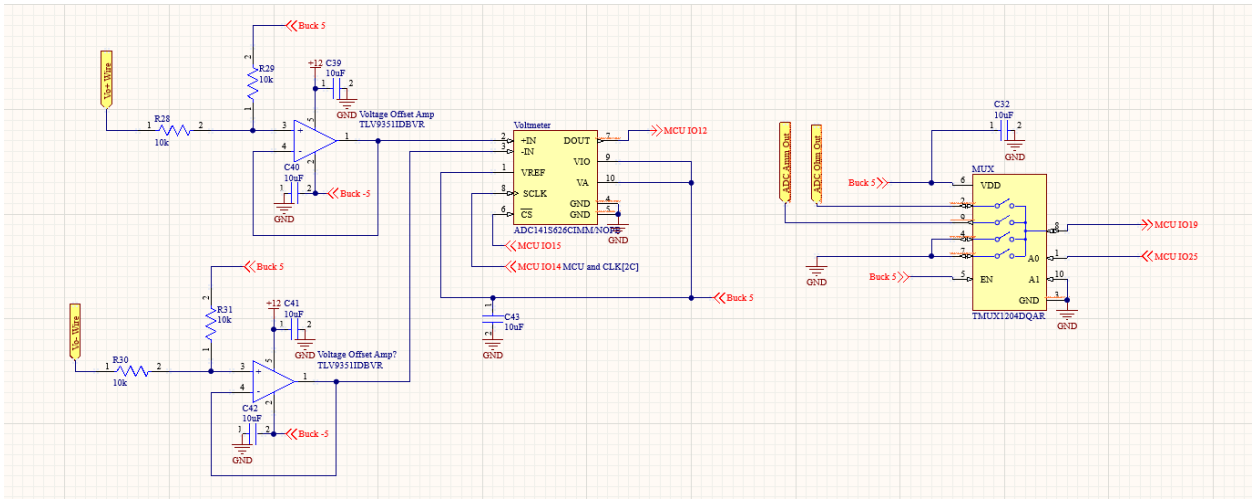


Figure 7: Schematic Level Design of Voltmeter and Multimeter Control MUX

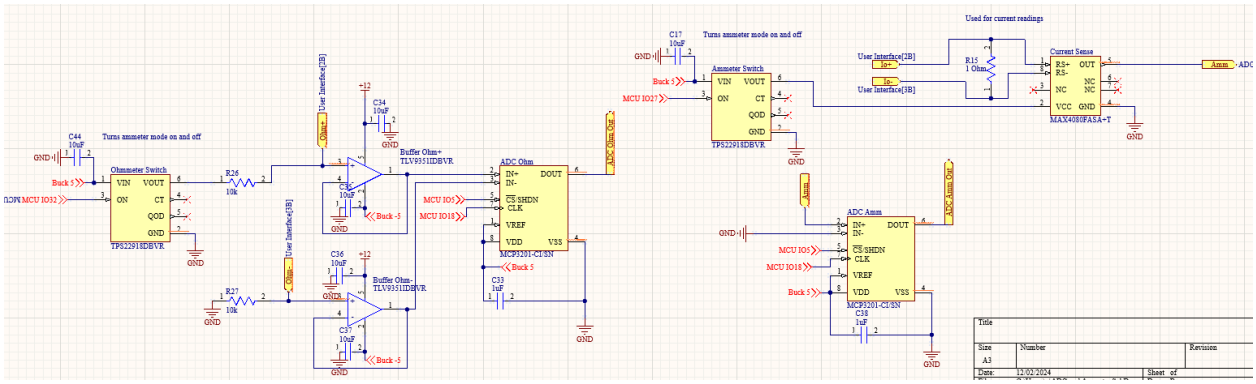


Figure 8: Schematic Level Design of Ohmmeter (left) and Ammeter (right)

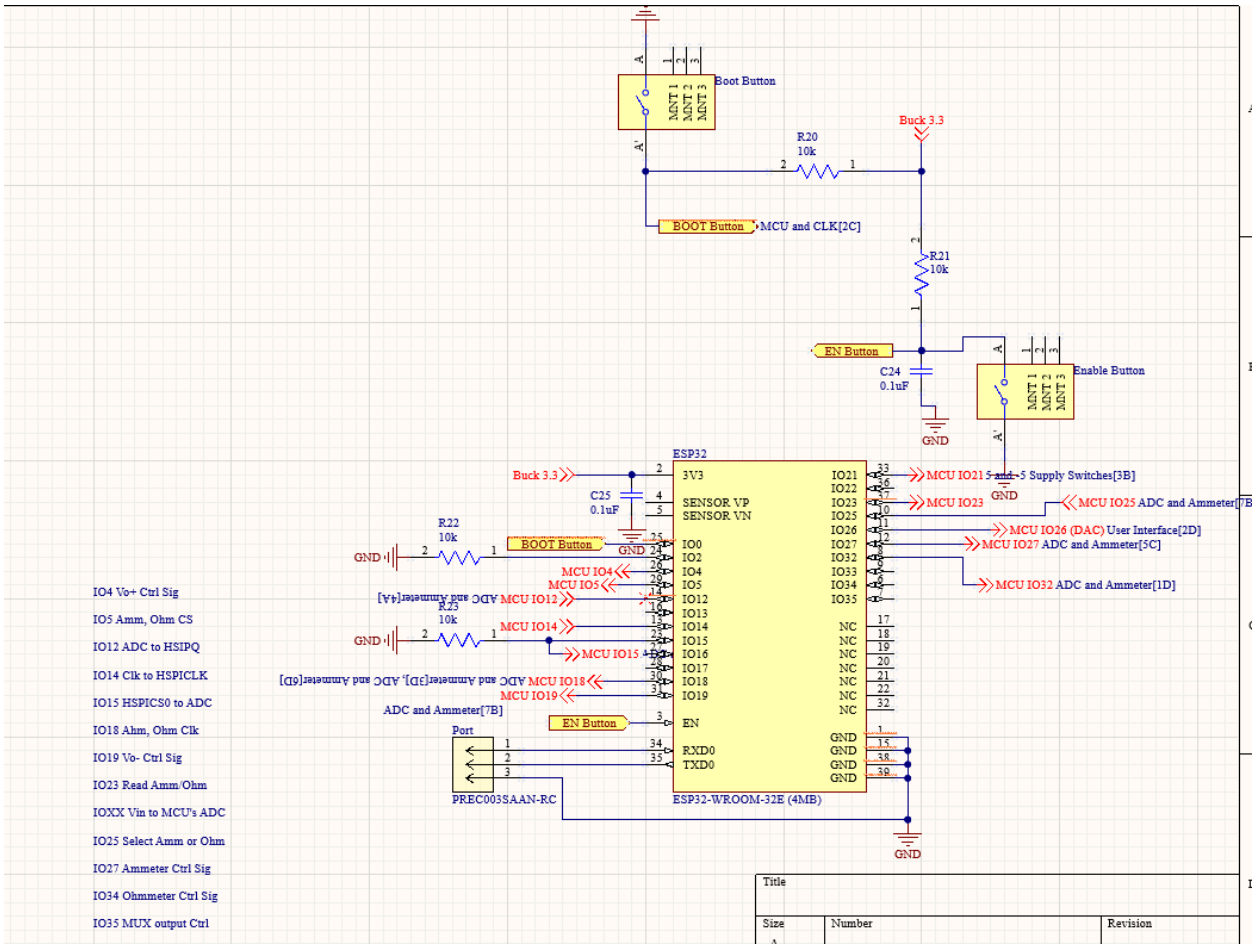


Figure 9: Schematic Level Design of MCU

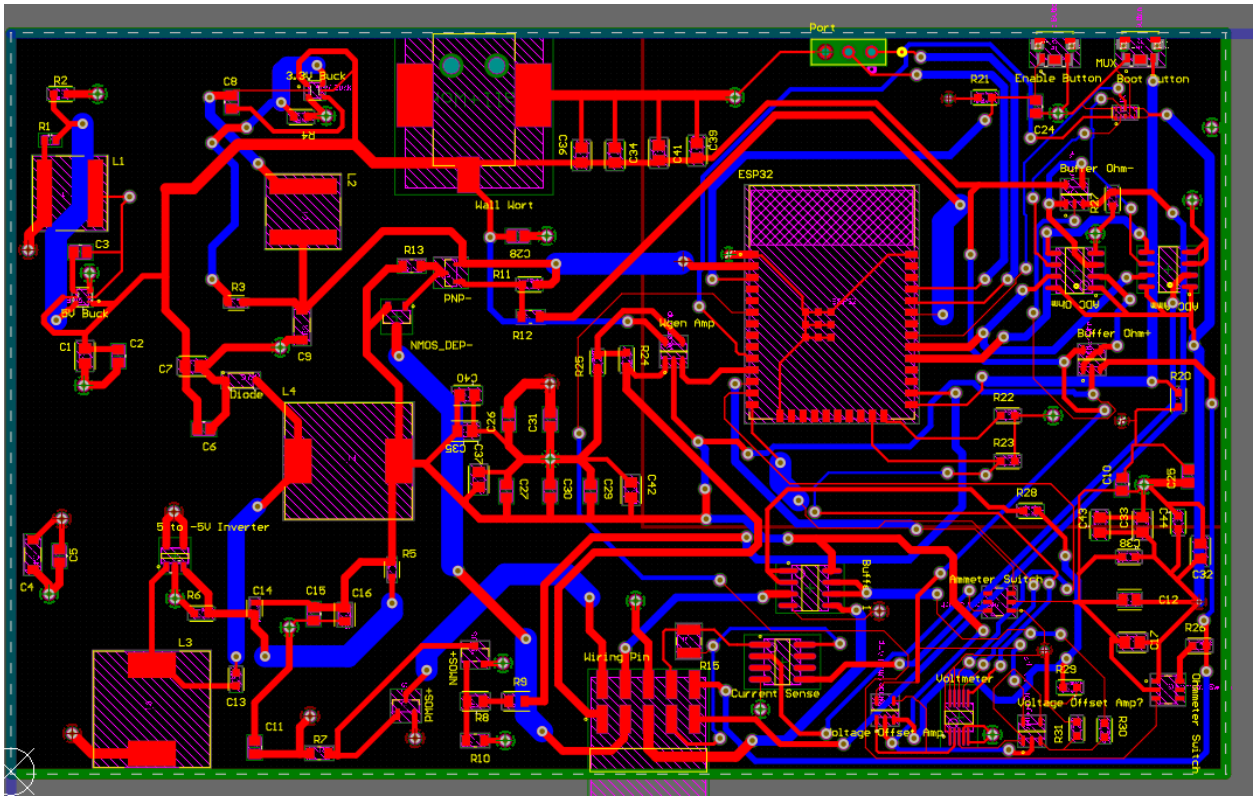


Figure 10: Design of Final PCB

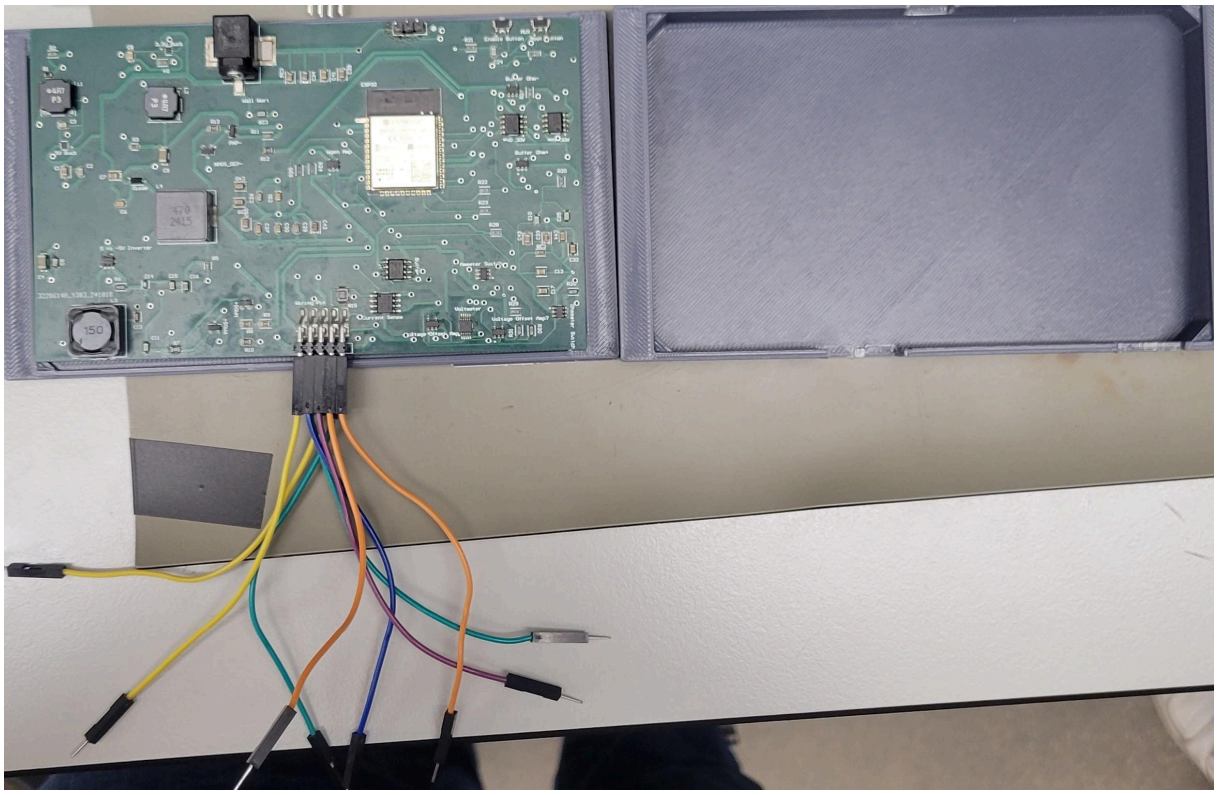


Figure 11: Final PCB and Opened Casing Unit

2.4. *Power*

As previously mentioned in 2.3 Board Design, the buck converter issues previously encountered were found to be shorting issues produced by solder bridging. The buck converters developed an odd “charging” issue that occurred later on in the development of the product. The powering systems have a few brief moments upon being plugged in where the power planes appear to be shorting, however, this issue quickly resolves itself and the power systems behave as expected. This “charging” issue was determined to be a result of a large capacitance on the board, and as the charging is fairly quick, was determined to not be a concern. The final results of the power system can be seen below.

Power Systems Result	Expected	Actual
System 1	5 V	4.987 V
System 2	3.3 V	3.319 V
System 3	-5 V	-4.976 V

Table 1: IC Power Results

As depicted above we can see that all devices delivering power on the board are within 30 mV of the desired result. Below are the results for the +5 and -5 V supply switches.

Switching System Results	ESP Low	ESP High
+5 V supply	0.019 mV	4.995 V
-5 V supply	-4.975 V	-4.973 V

Table 2: Power Switch Results

As we can see above the +5 V supply switch performs as desired, however, the -5 V supply does not switch off as expected. This is likely a very simple issue to troubleshoot and address, however, troubleshooting assistance for the other more critical systems was determined to be a higher priority and this issue was not fully addressed as it does not restrict any functionality to the ECEN 215 labs.

2.5. *Casing*

A simple casing was produced to protect the PCB and provide a more complete look to the device. The casing is a simple 2-piece 3D printed design that is secured together using size 6, ¾” screws, encapsulating the PCB. The design has not been drop tested and is not waterproof, however will still protect the internal circuitry from any external interference as well as any dirt and dust.

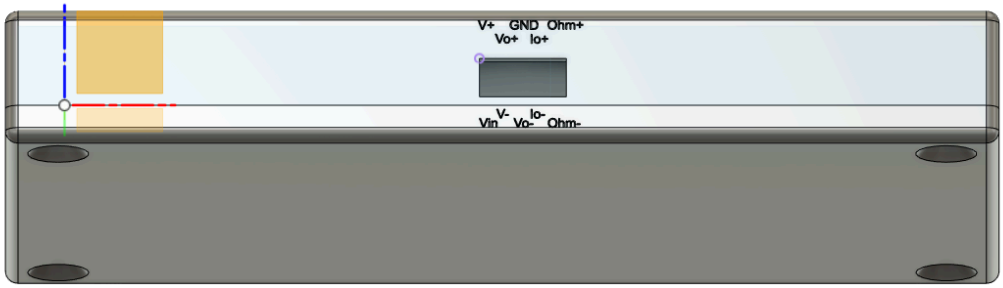


Figure 12: Casing Unit - Front and Bottom View

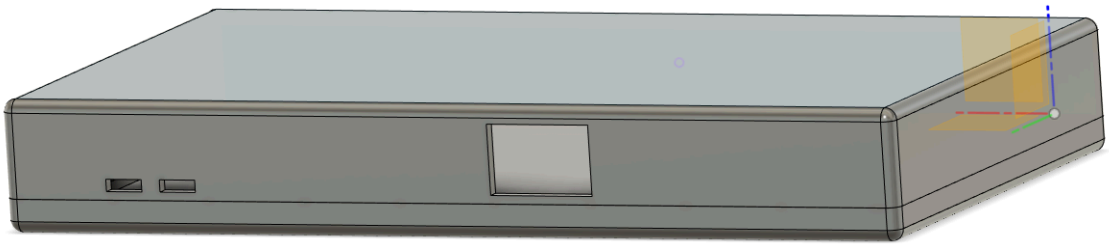


Figure 13: Casing Unit - Back and Top View

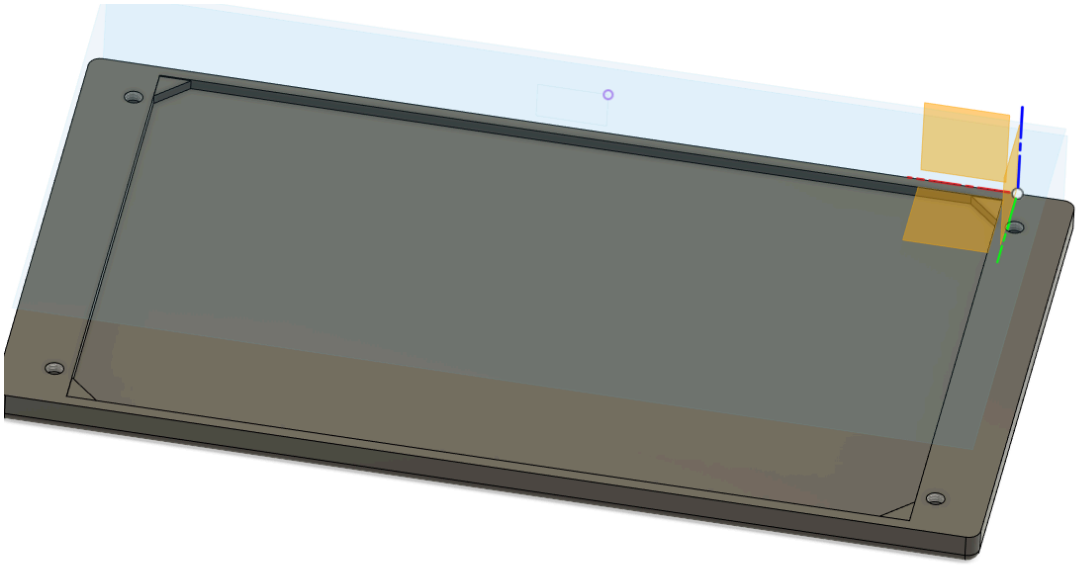


Figure 14: Casing Unit - Bottom Half Inside View

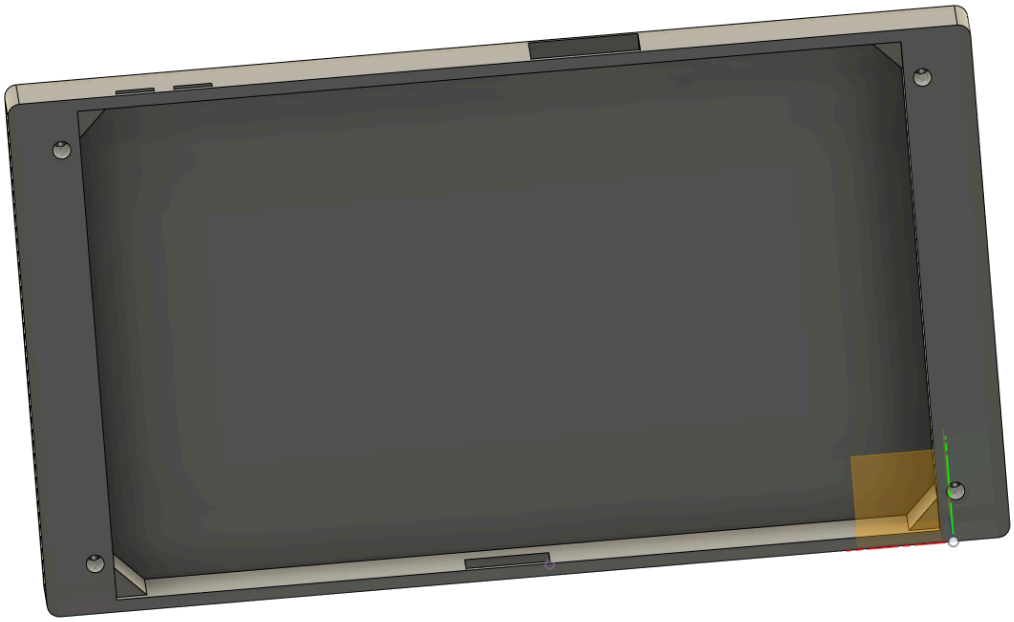


Figure 15: Casing Unit - Top Half Inside View

2.6. Size and Price

The overall size and price requested for the device was that the product be approximately the size of an average mobile phone and about $50. The final results yielded a casing size of 156.7x85.6x21.7mm which is about 24% larger than the average phone. While this increase may seem rather sizable, the overall purpose of the request was still achieved as the product is still small, mobile, and pocket-sized. The pricing, after including the wall adapter and assembly costs, totals to approximately $57.20 per board. The cost estimate breakdown is detailed below.

- Altium Price Analysis: $44.57
- JLC Price Analysis per 500: $3.80
- Wall Adapter: $8.83
- Overall Estimate: $57.20 per board
- Case Size: 156.7x85.6x21.7mm
- Area increase vs. phone: 24%

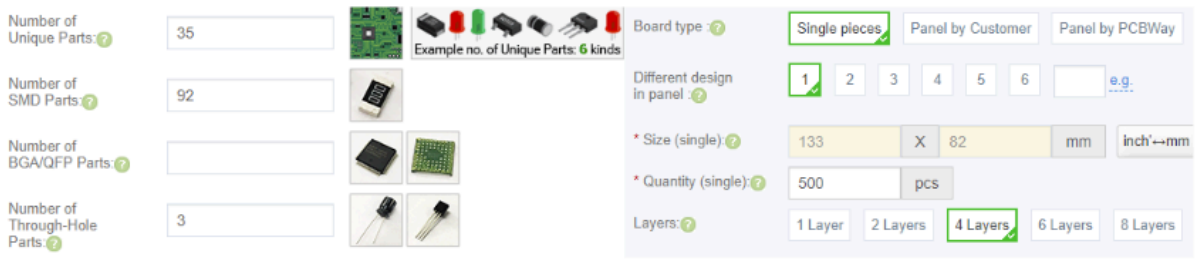


Figure 16: Cost and Size Analysis

2.7. Subsystem Conclusion

The PCB assembly and power were largely successful, with a few small yet easily addressable concerns. All converters on the board deliver power to the ICs successfully, after a small, yet negligible, “charging” time, with no greater than a 30 mV error. The +5 V supply works as expected and can be turned on and off through a simple button press in the app. Although the -5 V supply wire does not deactivate, it still delivers power via the wiring pin and can be used to power the ECEN 215 labs. The size of the product in its case is 24% larger than the average phone, however, is still mobile and pocket-sized. The overall estimated price is $7.20 over the requested price, at a total of $57.20, including IC pricing, PCB fabrication, assembly costs, PCB shipping, and including the price of a 12 V, 5 A wall adapter.

3. Mobile Bluetooth Application

3.1. Subsystem Introduction

The ECEN 215 Lab Kit requires an external interface for use, and to satisfy this requirement a Bluetooth mobile application tailored specifically for the ECEN 215 Lab Kit was developed. This application serves to function as a versatile tool that will guide students through all ECEN 215 laboratory exercises while enabling students to obtain real-time data and control circuit parameters. The solution presented in this report showcases a user-friendly and intuitive application that is compatible with IOS and Android systems and connects to the main device via Bluetooth low energy.

3.2. Creating Application

3.2.1. Application Pages

FlutterFlow, a “little-code” programming software was used to facilitate the creation of this application. Using FlutterFlow not only eased the learning curve that came with app development, but it also provided a simple click-and-drag interface that greatly assisted in creating the pages for this application. Using the resources provided by FlutterFlow each page was equipped with the appropriate selections that allow for ease of app navigation.

The first page of the application is just a brief pop-up page that serves as a loading screen for the application. The purpose of this page is to push Bluetooth permissions, which will officially show up on the next page. Next, the application will load into the device discovery page. It shows a list of currently connected devices up top and the lower half of the screen lists discoverable devices that the user can click on and establish a connection. Once the connection is established, the application leads to the main menu. It is here where the user can choose to navigate to the multimeter menu, oscilloscope menu, waveform generator menu, or power supply menu. Within the multimeter menu, the user has an option to choose between measuring voltage, amperage, or resistance. Selecting either option will navigate the user to a page where the appropriate measurement is displayed. If the user selected the oscilloscope menu then the user would be navigated to a page that displays a start button, a stop button, and an empty graph. Once the user interacts with the start button the graph will start populating according to whatever is connected to the device. Measurements are listed on the bottom of this page. The next option the user has within the menu is navigating

to the waveform generator. Once this option is selected the user will be navigated to a page that lists three options to select from, square, sine, and triangle. These three options represent the signals that the user would want to see applied to their circuit. Upon selecting one of the three options the user will be guided to another page that lists out the specific signals one would want to generate throughout this lab. The final menu option takes the user to the power supply page. Here the user can select between activating the 5V supply or the -5V supply. The button turns green whenever the supply has been activated and returns to its original color when deactivated. An image of the application pages can be seen below.

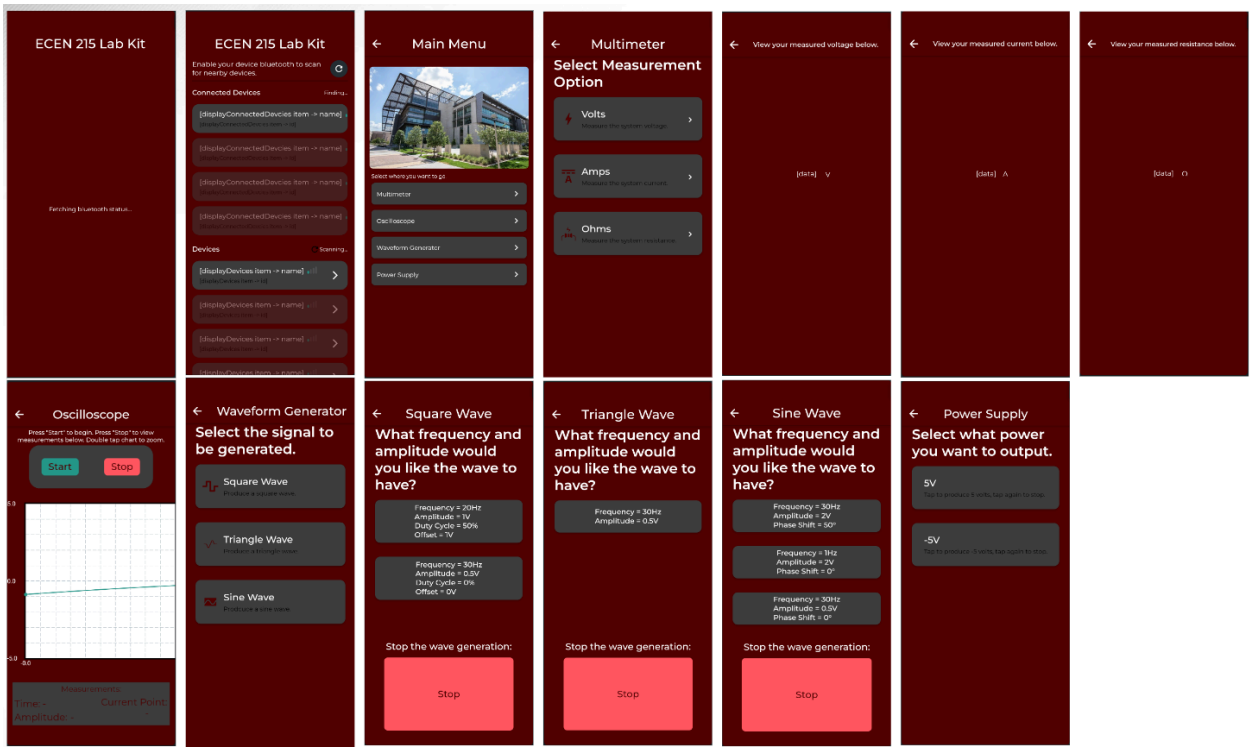


Figure 17: Application Pages

3.2.2. Emulating Application

Once the application pages and functionality had been created and set in FlutterFlow the project was pushed to GitHub and opened in VSCode. A snapshot of the project converted to code can be seen below. Specifically, the figure shows one of the custom actions developed for the application.

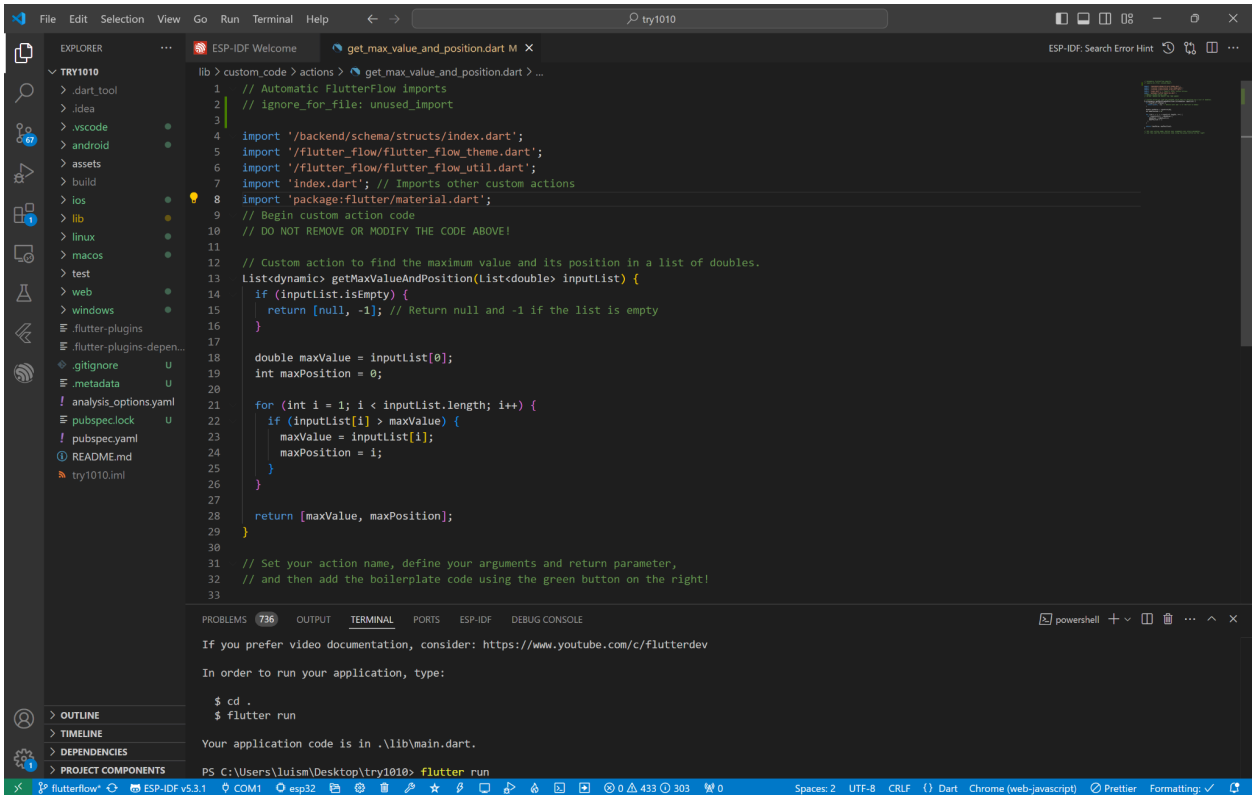


Figure 18: Application Code in Visual Studio Code

With the project loaded in VSCode it can now be run on an Android phone emulator. Using an emulator allows for confirmation of page design and navigation; however, because emulation is not a physical device, the Bluetooth capabilities of the project cannot be tested. The Bluetooth testing will be covered in section **3.3**. The emulated application can be seen in the figure below.

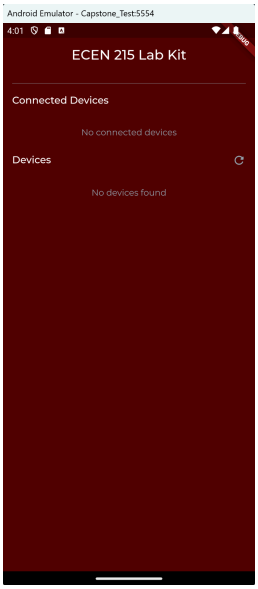


Figure 19: Application Running on Android Emulator

3.3. Implementation of Bluetooth Low Energy

Flutter does not provide Bluetooth functionality on its own, so a custom plugin is required to implement this feature. The plugin used for creating the Bluetooth functionality within this application is flutter_blue_plus. This plugin will allow for Bluetooth low-energy functionality.

3.3.1. Custom Code

In order to make use of the flutter_blue_plus plugin custom code needs to be created and implemented within the FlutterFlow project. The first step in creating custom code is to establish what functions and features are desired from the application. Does the application only need to make a connection, does it need to send a message to a server, and more are questions all asked during the development process. A useful resource for discovering what functions are needed to correctly implement this plugin and Bluetooth into the application is the plugin's documentation. This documentation covered all the necessary functions that are required to be written and implemented in order for Bluetooth to function as intended. In referring to this document it was discovered that the required functions are *connectDevice*, *disconnectDevice*, *findDevices*, *getConnectedDevices*, *getRssi*, *isBluetoothEnabled*, *receiveData*, and *sendData*. The functionality of these functions is self-explanatory by the function names, except for *getRssi*. Received signal strength indicator (Rssi) is a measure of the Bluetooth signal's strength. After following the plugin documentation and creating the code it was put into the FlutterFlow project in the custom code section. The list of the functions that make up the project's custom code can be seen below.

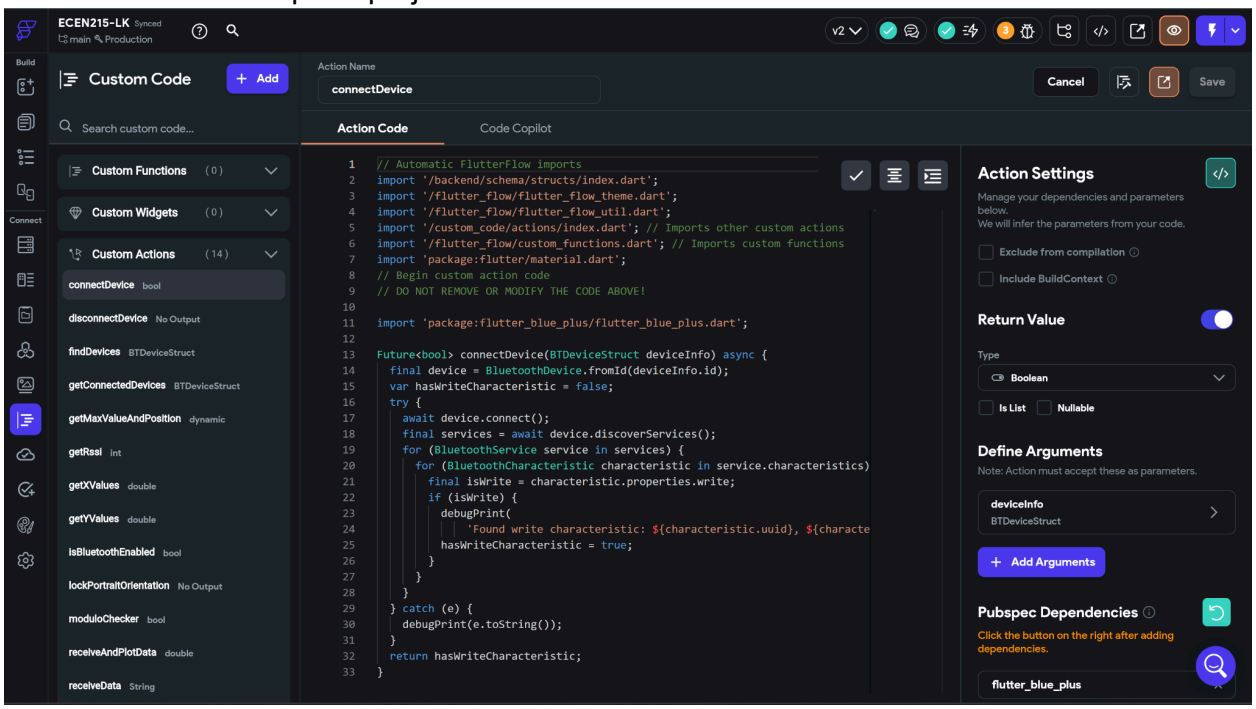


Figure 20: Custom Code within FlutterFlow Project

In addition to the functions mentioned above the project necessitated more custom code to be able to implement functionality such as a live oscilloscope graph. The *recieveAndPlotData* and the *getMaxValueAndPosition* functions were created to handle the oscilloscope graph creation. The first code takes in the appropriate data received by the lab

kit and plots it in real time on the mobile application. The second code records the amplitude of the graph and displays that and the time on the screen.

Once these functions are all in the project they can be implemented in the application through action menus and settings. One example of a function being implemented can be seen below in the figure showing an action menu for device connection. In the figure, the *connectDevice* function is called and its parameters and output(s) are defined on the right. This output can be used anywhere in the application to assist with performing other actions. Then the next action updates the page before the app navigates to the preceding page.

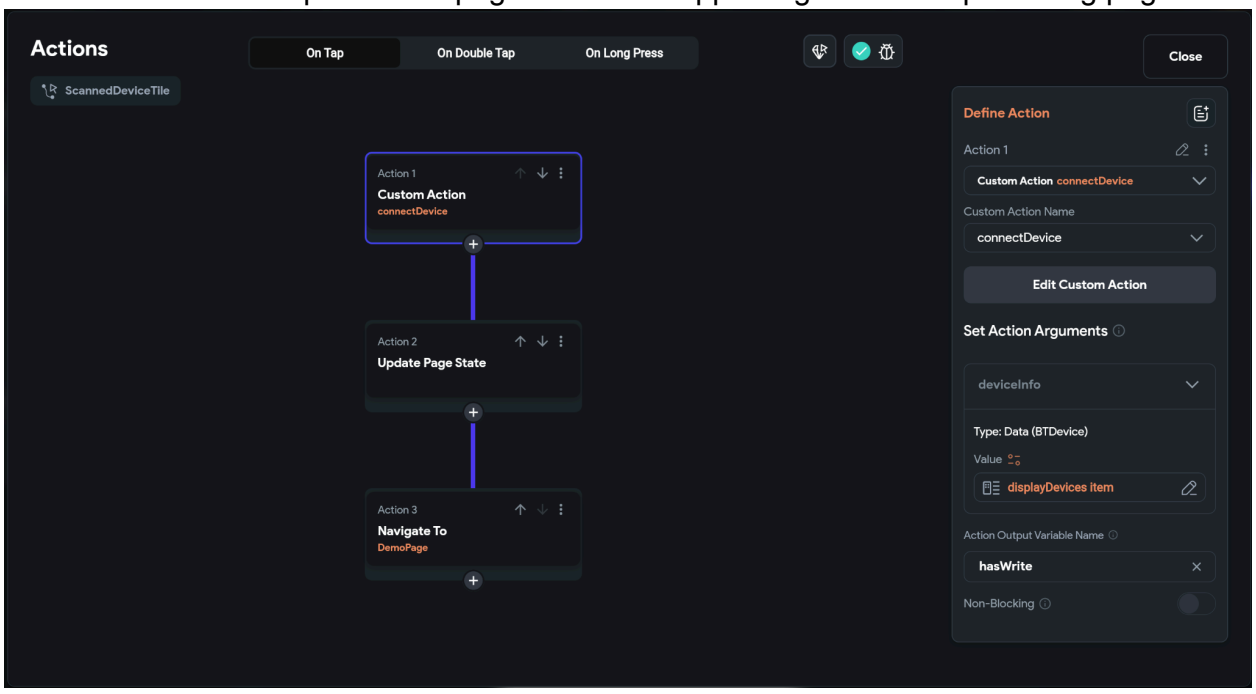


Figure 21: Custom Code Implementation

3.3.2. Debugging

Due to the flutter_blue_plus plugin's great documentation, there were not many problems to debug after implementing the custom code with Bluetooth functionality. The first few issues were just simple errors, such as typos and missing semicolons. The major problem that needed to be debugged was that none of the Bluetooth functionality would work initially. The debug console kept throwing errors that did not make sense. What led to debugging this problem was deeper research on the plugin. Further research revealed that using the plugin within FlutterFlow could lead to errors if not using the correct version that is compatible with FlutterFlow. In all, the resolution to this first big problem was switching the plugin's version from 1.32.4 to 1.7.5. Although this may seem like a huge rollback, this older version offers all the functionality that is offered in the most recent version of the plugin. Once the older version was implemented into the custom FlutterFlow code the Bluetooth functionality on the application worked as intended.

Another big issue encountered in the development of this mobile application was the implementation of the live oscilloscope graph. While creating this code, many problems arose. The first issues to debug were syntax errors, such as missing brackets, incorrect variable names, or missing imports. These were relatively straightforward to resolve using

error messages and code tools. However, the serious challenge was ensuring the oscilloscope graphed the data correctly and in real time.

The best way to debug this was with a systematic approach. First accuracy of the data retrieved from the source had to be verified and formatted correctly. This was done by logging raw data values to check for incorrect types or unexpected values. Aligning the data retrieval and graph update frequencies was the next step, as mismatched intervals caused choppy or incorrect plotting. Debugging this required fine-tuning the frequency at which the data was sent from the device to the app and synchronizing it with the graph's refresh rate. Lastly, issues regarding the graph's rendering such as improper scaling and missing points were addressed by adding debugging statements to track the data passed to the graph.

3.3.3. Testing Communication

A Motorola 6CS was used to test the Bluetooth functionality of the app. After the application had been installed on a device that had Bluetooth capabilities, the ESP32 that the app would be communicating with had to be set up. For testing purposes only, the ESP32 was configured within the Arduino IDE. Using Bluetooth libraries from Arduino a simple code was set up to send and receive messages to and from the ESP32. In order to facilitate simpler testing and debugging a demo page was temporarily added to the application. In the figures below the Motorola can be seen sending a message from the demo page to the ESP32 and the ESP32 can be seen receiving a message and sending one back within the Arduino IDE. Overall, the application was able to send messages to the ESP32 successfully, while also receiving messages from the ESP32 successfully. Once the application's functionalities had been verified, every button and action within the application was modified to send a message to the ESP32 when activated.

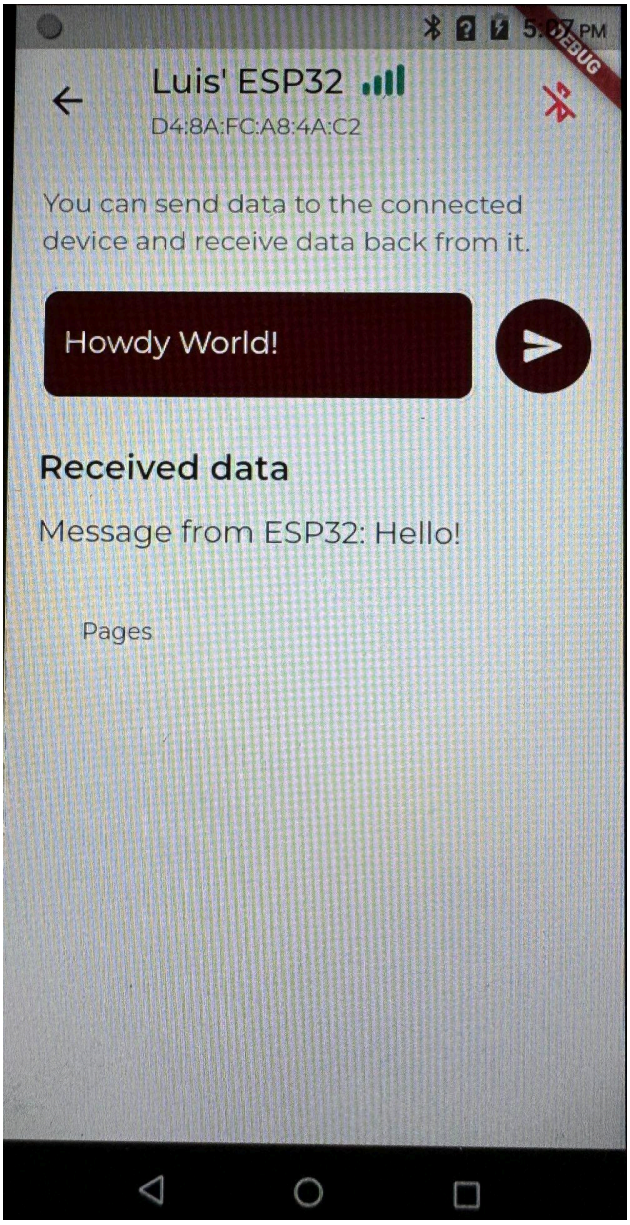


Figure 22: Application Demo Page Sending and Receiving Messages

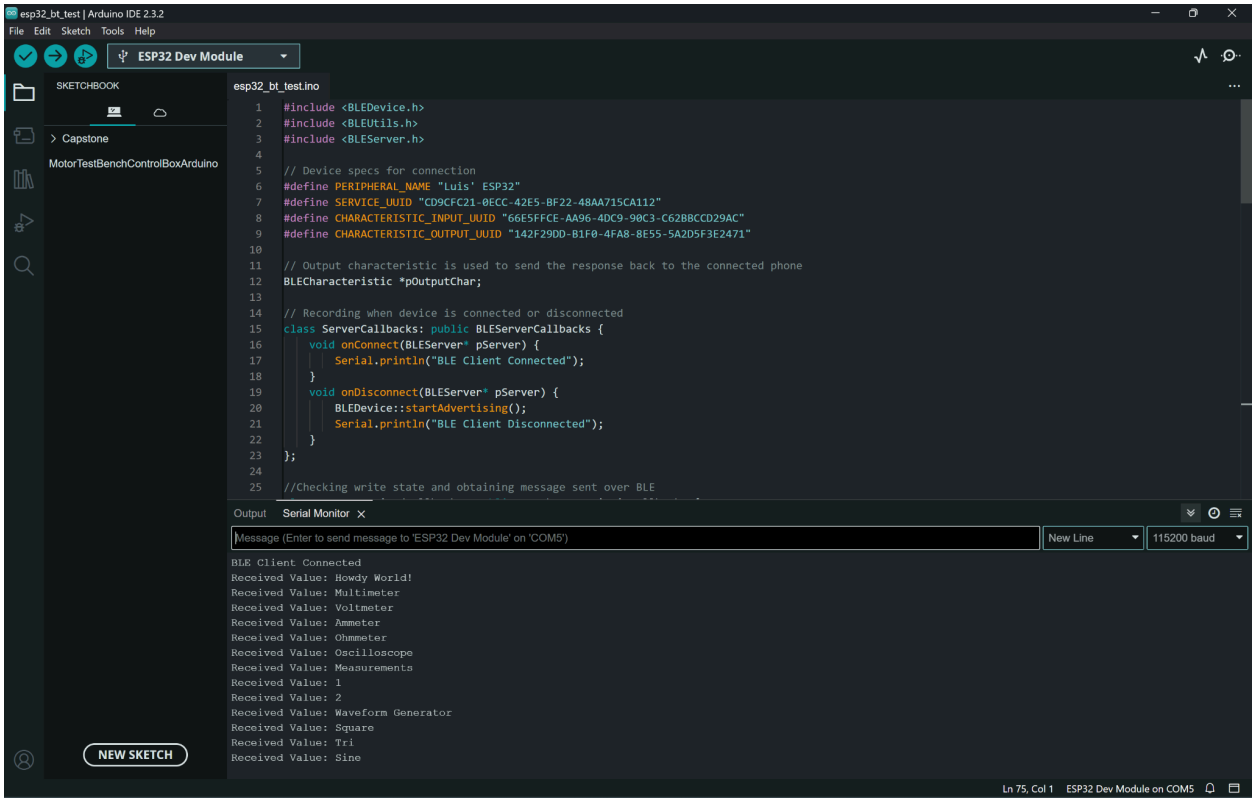


Figure 23: ESP32 Sending and Receiving Messages within Arduino IDE

After confirming Bluetooth functionality using the Motorola device and Arduino IDE, the code handling Bluetooth communication was migrated to an Espressif IDF environment in VSCode and a newer phone was obtained. The new phone was a Motorola Moto G Pure and was used to facilitate the mobile app's integration with the other subsystems. Integration was the driving force behind using Espressif IDF for Bluetooth communication as well. The ***Systems Report*** dives deeper into the integration process and provides the process taken to ensure smooth overall system functionality.

3.4. Validation

When testing and validating the mobile application it proved to be a robust and efficient system. The system demonstrates efficient device discovery, with devices being identified within four seconds. Connection stability has been verified, ensuring a stable link to the device for at least 30 minutes while idle. The app supports immediate reconnection after disconnecting from a device. Read and write operations have been successfully tested, with the app sending and receiving messages verified through the VSCode terminal and system functionality. Connection latency is minimal, with the app establishing connections consistently within 1.5 seconds. Command response times are excellent, with data sent to the app within 0.25 seconds after a request. The app has proven device compatibility, functioning on multiple devices ranging from 2 to 10 years old. Additionally, the app maintains a stable connection even in environments with abundant Bluetooth traffic, demonstrating resilience to interference.

3.5. Subsystem Conclusion

To conclude, the ECEN 215 Lab Kit requires a user interface that will allow students to perform their lab exercises from the comfort of their homes. Through the use of the FlutterFlow program, the custom flutter_blue_plus plugin with custom functions, and meticulous testing and debugging such an application has been developed. This application succeeds in performing all functions that have been desired since the conception of the overall ECEN 215 Lab Kit project. The application had been developed with the purpose of providing a responsive, attractive, and functional user interface to allow students in the ECEN 215 course to develop their skills in electrical engineering by manipulating circuit parameters, monitoring real-time data, and analyzing results with ease. This application will undergo implementation with the other subsystems within this project and be put through rigorous testing to ensure the proper functionality.

4. Wave Generator and MCU

4.1. Subsystem Introduction

For multiple labs in ECEN 215, there needs to be the generation and manipulation of multiple waveforms. The wave gen feature in our system is capable of generating 3 different shapes(sine, triangle, and square) along with a square wave at a half-duty cycle. The frequencies and amplitudes can also be altered for each lab. The frequency has 3 preset values of 1, 20, and 30Hz. The amplitude of each of the waves can be adjusted to 0.5, 1, and 2 Volts. These graphs can then be displayed on a smart device for further inspection.

4.2. Firmware and Hardware

4.2.1 ESP32-WROOM 32E-N4

The firmware for the wavegen feature and Bluetooth connection are implemented using this ESP32-WROOM-32E-N4. Originally an AD9833 wave-gen chip connected via serial peripheral interface (SPI) was also included in the project design but it was later removed because of the available DAC channels on the ESP32. Dac channels 1 and 2(GPIO25 and GPIO26) were then used to output the needed voltages. To implement code onto the ESP32 a UART connection was made between devices.

4.2.2 AC-Coupling and Amplification

The ESP32 DAC channels generate waveforms with a DC offset because it can only output voltages in the range of 0 - 3.3V. To account for the negative values needed for each of the waveforms a capacitor was connected in series(AC coupling) to allow the wave to oscillate around 0v. The capacitor blocks the DC components of the wave only allowing the AC values to pass through. This averages the voltages out, eventually centering the wave around 0v. The capacitor size we selected was 10uF, this seemed to be the sweet spot that allowed for the AC coupling to occur relatively fast, zeroing out the voltages in less than 10 seconds without any noise. While bigger capacitors allowed for marginally smoother outputs the time it took to generate the negative values was much longer. To allow for higher voltages to be outputted in case it was needed, a buffer in the form of a non-inverting amplifier with a gain of 2 was added.

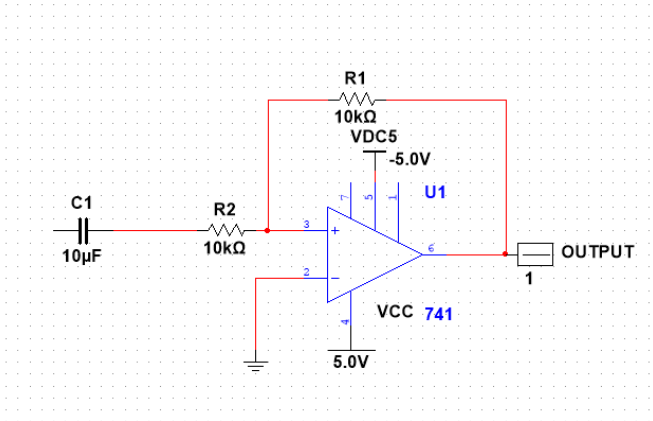


Figure 24: Non-Inverting Op-amp and AC Coupling

4.3. Software

The code for all the firmware was done in C using ESP-IDF(Espressif IoT Development Framework) to include all the necessary libraries to control the ESP32s features. The project was created using PlatformIO which is an Open-Source Ecosystem that supports systems like the ESP32. All the code was stored on a GitHub repository.

4.4. Wavegen

4.4.1 Operation

The GATT server code contains all the necessary code to fully operate the wave generation function. Operation begins when the wavegen feature for a certain lab is triggered on the mobile app after the connection is made with the ESP32. Once a specific wave shape and lab are selected the corresponding wave will be outputted from the DAC channel and fed through the capacitor and amplifier.

4.4.1.2 Table Generation

To prevent the ESP32 from potentially being overloaded with information, pre-computed values were used for the sine table that were calculated based on the desired amplitude. This method was used instead of calculating the data points in real-time which is how a real wave generation function works, but for the purpose of ECEN 215, this method works fine. Using the <math.h> library included in ESP-IDF, a table is filled with values corresponding to table_size. It calculates each of these values using the appropriate equation that takes into account, table size, amplitude, and DC offset.

```
529
530 static uint8_t sine_table[TABLE_SIZE];
531 static uint8_t sine_table2[TABLE_SIZE2];
532 static uint8_t sine_table3[TABLE_SIZE3];
533
534
535 void generate_sine_table() {
536     for (int i = 0; i < TABLE_SIZE; i++) {
537         // sine_table[i] = (uint8_t)(SINE_AMPLITUDE * sin(2 * M_PI * i / TABLE_SIZE) + SINE_AMPLITUDE);
538         sine_table[i] = (uint8_t)((SINE_AMPLITUDE / 2) * sin(2 * M_PI * i / TABLE_SIZE) + 128);
539     }
540 }
541
542 void generate_sine_table2() {
543     for (int i = 0; i < TABLE_SIZE2; i++) {
544         sine_table2[i] = (uint8_t)(SINE_AMPLITUDE2 * sin(2 * M_PI * i / TABLE_SIZE2) + SINE_AMPLITUDE2);
545     }
546 }
547
548 void generate_sine_table3() {
549     for (int i = 0; i < TABLE_SIZE3; i++) {
550         sine_table3[i] = (uint8_t)(SINE_AMPLITUDE3 * sin(2 * M_PI * i / TABLE_SIZE3) + SINE_AMPLITUDE3);
551     }
552 }
553
```

Figure 25: Sine Table Generation

4.4.1.3 Generation Loop

The ESP32 is only capable of producing waveforms with a certain frequency range. For the SIne wave that range is 16Hz - 500KHz, for the square wave it is a range of 1Hz - 40MHz and for the triangle wave it is 150 Hz - 150 KHz. To bypass this a generation loop was implemented. Inside the loop, each value from the table generation is outputted to the DAC channel on the ESP32 using ‘dac_output_voltage()’. Then a delay is added to replicate the desired frequency. This process is repeated indefinitely until the program is stopped.

```
24
25 void app_main(void)
26 {
27     // Generate sine wave table
28     generate_sine_table();
29
30     // Configure DAC
31     dac_output_enable(DAC_CHANNEL);
32
33     // Calculate the delay period in ticks to achieve the desired sample rate
34     TickType_t delay_ticks = pdMS_TO_TICKS(1000 / SAMPLE_RATE);
35
36     // Infinite loop for generating sine wave
37     while (1) {
38         for (int i = 0; i < TABLE_SIZE; i++) {
39             // Output the value from the sine wave table to the DAC channel
40             dac_output_voltage(DAC_CHANNEL, sine_table[i]);
41
42             // Delay to maintain the desired sample rate
43             vTaskDelay(delay_ticks);
44         }
45     }
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
```

Figure 26: ESP32 SINE Generation Code

4.4.2 Validation

The code was validated and tested by using an oscilloscope connected to the dac channels of the ESP32 along with the local ground of the ESP32. The accuracy of the wavegen function was verified by the built in frequency and amplitude measuring tools already installed on the oscilloscope.

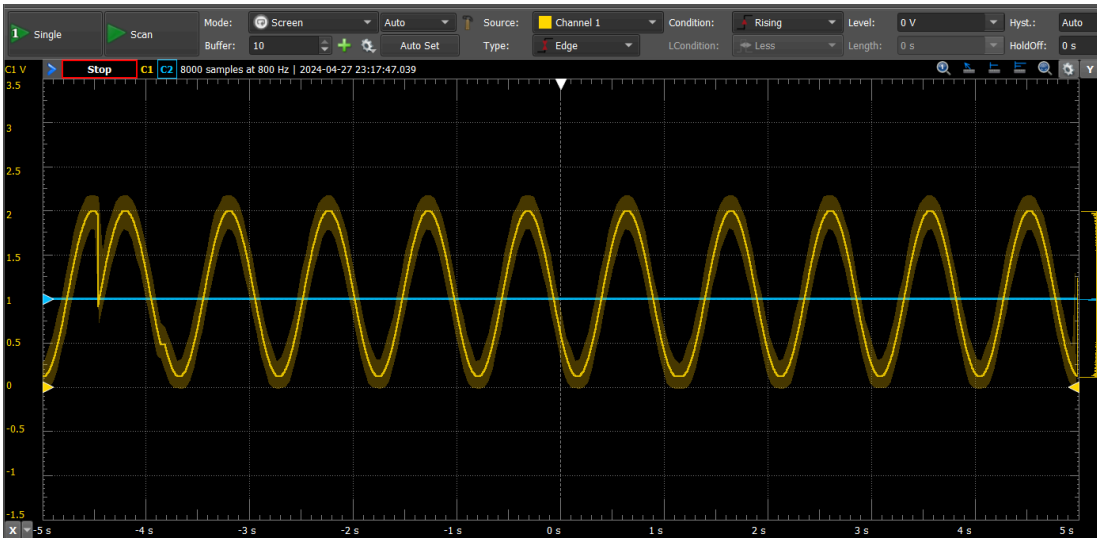


Figure 27: SINE Wave with amplitude of 1V and Frequency of 1Hz

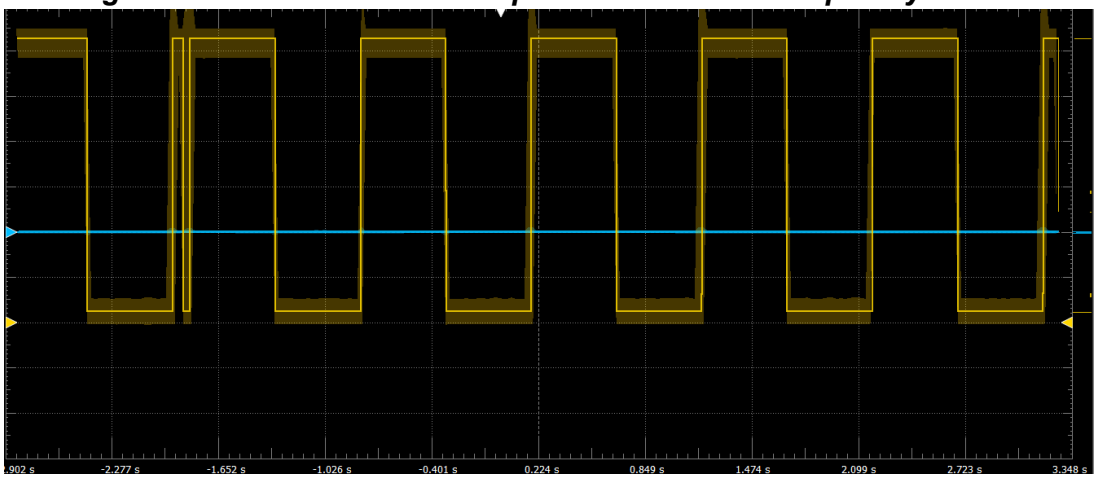


Figure 28: Square Wave with amplitude of 1.5V and Frequency of 1Hz

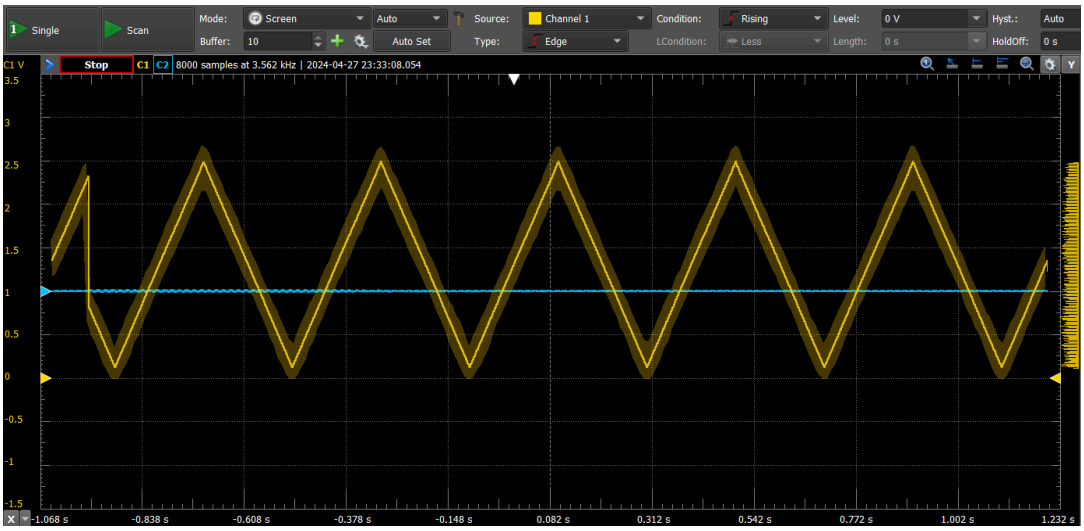


Figure 29: Triangle Wave with amplitude of 1.5V and Frequency of 1Hz

4.5. Communication

4.5.1 Operation

To connect the ESP32 with a smart device a BLE server is generated that allows for communication between devices, this can be used to display the wave gen function and adjust the desired parameters. The GATT Server code demonstrates the implementation of a Bluetooth Low Energy (BLE) Generic Attribute Profile (GATT) server on the ESP32. The code manages BLE communication using application profiles, GATT services, and event handlers, allowing the device to interact with BLE clients, like our mobile application. Buttons on the app correspond with certain characters that will trigger the functionality of the different waveforms. For example, selecting the button “Lab 4 Square” would call the xTaskCreate(Square, "square_wave_task", 2048, NULL, 5, &waveform_task_handle); function.

```
void process_received_data(uint8_t *received_data, uint16_t received_data_len)
{
    // Define the string to compare against
    const char *expected_string = "Square";
    const char *expected_string_ = "Sine";
    const char *expected_string__ = "Tri";
    // Add underscores to change which one is called
    // EX: if (received_data_len == strlen(****expected_string****) && strncmp((const char *)received_data, ****expected_string****, received_data_len) == 0)
    // call specific funcitons like square 1 and Swuare 2

    const char *expected_string_lab4sq = "Lab 4 square";
    const char *expected_string_lab5sine = "Lab 5 sine";
    const char *expected_string_lab6sine = "Lab 6 sine";
    const char *expected_string_lab7sine = "Lab 7 sine";
    const char *expected_string_lab7square = "Lab 7 square";
    const char *expected_string_lab7sinetri = "Lab 7 tri";

    const char *expected_stringy = "Voltmeter";
    const char *expected_stringy_ = "Ammeter";
    const char *expected_stringy__ = "Ohmmeter";
    const char *expected_string_stop = "Off";


    const char *expected_stringyy = "Oscilloscope";


    const char *expected_stringyyy = "5V";
    const char *expected_stringyyy_ = "Neg5V";
```

```
// Compare received_data to the expected string
if (received_data_len == strlen(expected_string_lab4sq) && strncmp((const char *)received_data, expected_string_lab4sq, received_data_len) == 0)
{
    //Square
    stop_current_waveform();  // Stop any existing task
    printf("Received data matches the string 'Square'\n");
    ESP_LOGI(TAG, "DAC square wave example start");
    ESP_LOGI(TAG, "--------------------------------------");
    xTaskCreate(Square, "square_wave_task", 2048, NULL, 5, &waveform_task_handle);
}

else if (received_data_len == strlen(expected_string_lab5sine) && strncmp((const char *)received_data, expected_string_lab5sine, received_data_len) == 0)
{
    //Sine
    stop_current_waveform();  // Stop any existing task
    printf("Received data matches the string 'Sine'\n");
    // Add your code to handle the matching case
    // Generate sine wave table
    ESP_LOGI(TAG, "DAC sine wave from lab 5 example start");
    ESP_LOGI(TAG, "--------------------------------------");
    xTaskCreate(Curvy, "sine_wave_task", 2048, NULL, 5, &waveform_task_handle);

}

else if (received_data_len == strlen(expected_string_lab6sine) && strncmp((const char *)received_data, expected_string_lab6sine, received_data_len) == 0)
{
    //Triangle
    stop_current_waveform();  // Stop any existing task
    printf("Received data matches the string 'lab 6 sine'\n");
    // Add your code to handle the matching case
```

Figure 30: Application Selection

4.5.2 Validation

To validate the connection between ESP32 and the application an Android device was used to verify that the Bluetooth server was available for a successful connection.

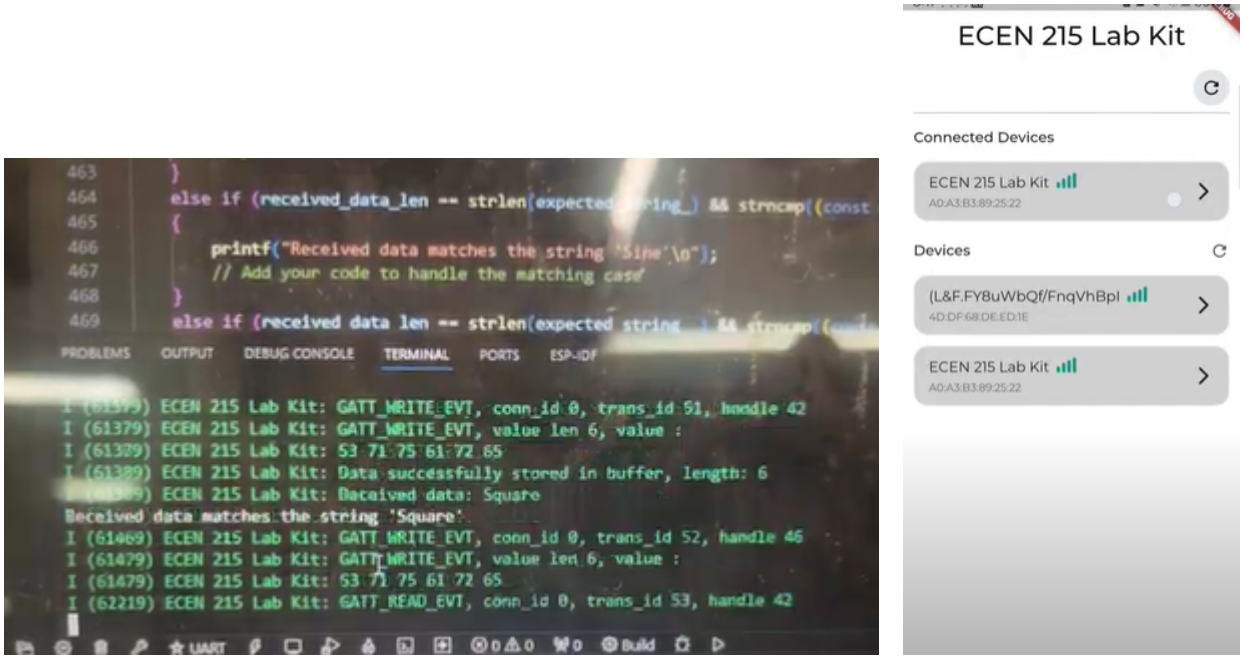


Figure 31: Bluetooth Connection

4.6. Conclusion

To conclude, the wave gen feature of the lab kit is capable of generating all the desired waveforms needed to complete the ECEN 215 Lab Kit. The minimal noise and high accuracy allow for the correct readings needed to complete any calculations. To reduce the amount of RAM and files used in the wavegen subsystem, optimization of storage is needed on both the software and hardware sides of things. The wave generation function is an integral part of the completion of almost all ECEN 215 labs. Pairing this with the mobile application, this feature can be used for many other applications too!

5. Oscilloscope and Multimeter

5.1. Subsystem Introduction

The multimeter subsystem is capable of reading current, resistance, and voltage while the oscilloscope subsystem is capable of reading signals up to 60 Hz. External hardware measures and transmits data to the ESP32 microcontroller which will process and communicate via Bluetooth to the mobile application.

5.2. Firmware and Hardware

5.2.1. ESP32-Wroom-32E-N4

For firmware, both the oscilloscope and multimeter utilized the ESP32 WROOM 32E. The microcontroller contains 4 MB of flash, operates on Bluetooth 4.2, and uses the communication protocols I2C, I2S, PWM, SDIO, SPI, and UART. The microcontroller's maximum transfer rate is 150MHz. SPI was the connection protocol used for the hardware. In the oscilloscope and voltmeter, general-purpose input/output (GPIO) 15 is chip select (CS), 14 is serial clock (SCLK), 12 is VSPIQ or MISO. For the ohmmeter and ammeter, GPIO 5 is CS, 18 is SCLK, and 19 is VSPIQ or MISO. As both outputs from the ammeter and ohmmeter are shared, a mux controls their respective output using GPIO 25. Additionally, switches to turn on and off the ammeter and ohmmeter used GPIO 27 and 32 respectively.

5.2.2. ADC141S626CIMM

The ADC141S626CIMM is an analog-to-digital converter (ADC) that converts analog voltages into digital values. It is a differential SAR ADC with both an analog supply and digital voltage of 2.7 V to 5.5 V. In our device, we set the analog and digital supply to 5V. It has a sampling rate from 50 KSPS to 250 KSPS, a 14-bit resolution, a signal-to-noise ratio (SNR) of 84.3 dB, and communicates using a 3-wire SPI. The output is given in 2's complement and outputs 16 bits in a transaction with the most significant bit (MSB) first. The 16 bits are usually split up into 8-bit chunks in SPI communication. Data is converted when CS goes high and output when CS goes low. The data is transmitted on the SCLK's falling edge with the first two bits containing dummy data.

5.2.3. MCP3201

MCP3201 is a successive approximation 12-bit ADC that converts analog voltage to digital values. It has a supply voltage of 2.7 to 5 V with a maximum sampling rate of 50 ksps and 100 ksps at the respective voltages. It is capable of a voltage reference of 0.25 V to the supply voltage, but within our circuit, we used 5 V for its supply and voltage reference. The ADC has a signal to noise and distortion (SINAD) of 86 dB and communicates using a 3-wire SPI. Data is converted when CS goes high and output when CS goes low while data is transmitted on the SCLK's falling edge. Data is transmitted in 16 bits MSB where the first 3 bits and last bit are discarded.

5.2.4. MAX4080FASA+T

The MAX4080FASA+T is a current sense amplifier that measures current across a resistor and outputs a voltage using the following formula:

Vout = Iload * Av * R,

where Vout is the output voltage, Iload is the current across the load, Av is the gain, and R is the external resistor. The amplifier's voltage supply ranges from 4.5 V to 76 V along with

a gain of 20 V/V and an accuracy of 0.1%. A 1-ohm shunt resistor was used in our circuit while 5 V is supplied to the part.

5.2.5. TPS22918DBVR

The TPS22918DBVR is a power switch IC to turn on and off the ammeter and ohmmeter. Input voltages range from 1 V to 5.5 V and the output follows the input. GPIO 27 and 32 are connected to the input of two TPS22918DBVR.

5.2.6. TLV9351IDBVR

TPS22918DBVR is a single-sided low-power operational amplifier with a 3.5 MHz gain bandwidth product, 350uV input offset voltage, voltage supply range from 4.5V to 40V, and a 20 V/us slew rate.

5.3. Software

Firmware code was programmed in Espressif IoT Development Framework (ESP-IDF) on the independent development environment (IDE) Visual Studio.

5.4. Oscilloscope

5.4.1. Operation

The oscilloscope would read a signal from the ADC and convert these measurements into a pair of time and voltage which would be graphed. The oscilloscope's SCLK is set at a frequency of 4 MHz which is in accordance with the ADC's specifications. To operate the oscilloscope, the user will first place the positive and negative oscilloscope probes in the circuit being measured. Then, they will power the ECEN 215 Lab Kit and connect to the mobile App. Once on the mobile App, the user will press on the button "Oscilloscope" and then "Start" to start recording and "Stop" to stop recording.

5.4.2. Input Voltages

The oscilloscope would read a signal from the ADC and convert these measurements into a pair of time and voltage which can be graphed. The input voltages are scaled between 0 and 5 V as negative voltages would damage the ADC. This is done by placing a summing amplifier at each of the ADC141S626CIMM's inputs and directing the input voltage into the summing amplifier. The circuit is given below:

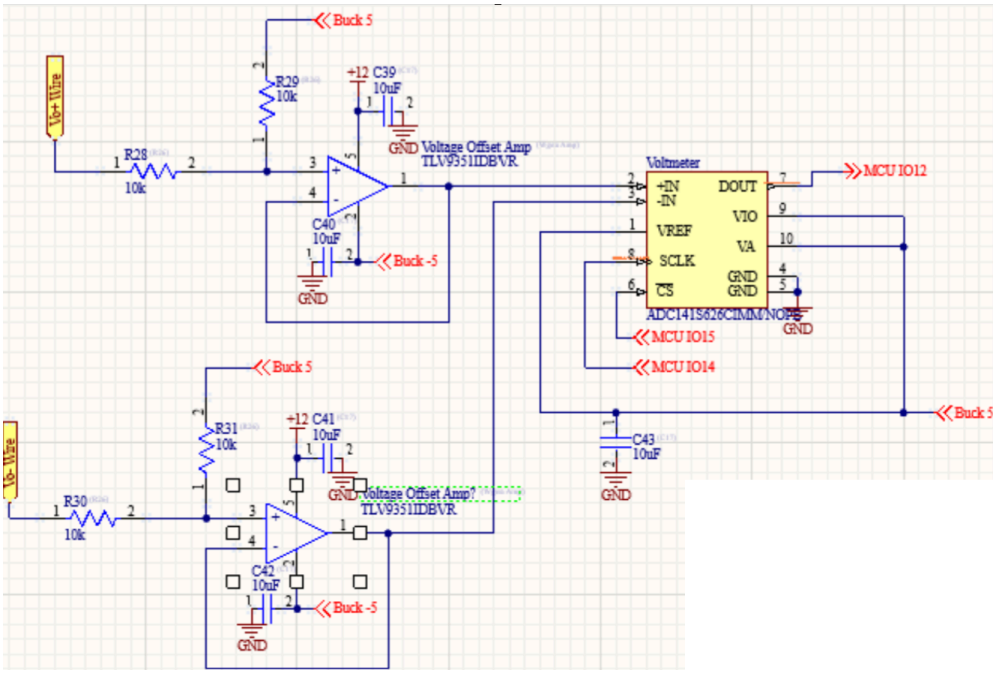


Figure 32: Oscilloscope and Voltmeter Circuit

To scale the voltage, the inverting input of the TPS22918DBVR was connected to the output. Then, the input voltage is connected in series with two 10K ohm resistors and 5 V. Finally, the OP Amp's non-inverting input is connected between the two 10K ohm resistors. Voltage supplies for the TPS22918DBVR are +12 V and -5 V. The circuit divides the voltage by 2 and then adds 2.5 V. This scheme results in an input voltage of -5 V to be scaled to 0 V, 0 V to 2.5 V and 5V to 5V.

5.4.3. Software

As stated before, the ESP32 translates and transmits the data in ESPIDF. Since the oscilloscope and voltmeter share the same ADC, their code is similar.

First, SPI communication is initialized with GPIO 15 as CS, 14 as SCLK, and 12 as VSPIQ. The SPI device's clock speed is set to 4 MHz, SPI mode 0, and at half duplex mode. Then, the SPI transaction is set with a length of 16 bits, rx_ buffer's length of 16, rx_buffer and tx_buffer set to null, and set to SPI_TRANS_USE_RXDATA so it only uses an internal RX buffer. Changing these settings would cause errors to occur. Next, in the function read_adc(), spi_device_transmit begins transmission and the data is stored 8 bits at a time into rx_data[0] and rx_data[1]. These two are combined into a single value by shifting the first buffer 8 bits and then performing boolean or between them. The value is converted from 2's complement to a float and then scaled by dividing by 8192 and then multiplying it by its reference voltage 5.

```c
// 14 bit ADC ADC141S626 used for voltmeter and oscilloscope
#define MAX_MEASUREMENTS 300 // Maximum number of measurements

// Struct hold voltage and time
typedef struct
{
    float voltage;
    float timestamp;
} Measurement;


Measurement measurements[MAX_MEASUREMENTS];
int num_measurements = 0;
int osc_cnt = 0; // Count what is displayed on the oscilloscope
int curr_cnt = 0; // Current number that is being measured
int64_t start_time = 0;


#define ADC_SPI_HOST HSPI_HOST
#define ADC_CS_GPIO 15   // Chip Select for 14 bit ADC
#define ADC_SCLK_GPIO 14 // Serial Clock for 14 bit ADC
#define ADC_DOUT_GPIO 12 // Data Out for 14 bit ADC

// Start timer
void start_timer()
{
    start_time = xTaskGetTickCount();
}

// Determine time since start_timer()
int64_t stop_timer()
{
    int64_t elapsed_time = xTaskGetTickCount() - start_time;
    return elapsed_time;
};


spi_device_handle_t adc_handle; // SPI handle for 14 bit ADC


void init_spi_osc_volt()
{
    // Configure the SPI bus
    spi_bus_config_t buscfg = {
        .mosi_io_num = -1,           // No MOSI pin
        .miso_io_num = ADC_DOUT_GPIO, // MISO pin to 14 bit ADC data out
        .sclk_io_num = ADC_SCLK_GPIO, // Clock pin connected to 14 bit ADC clock
        .quadwp_io_num = -1,
        .quadhd_io_num = -1,
        .max_transfer_sz = 2, // Transferring 2 bytes or 16 bits
    };


    // Configure the SPI device
    spi_device_interface_config_t devcfg = {
        .clock_speed_hz = 4000000,   // 4 MHz. Can be between 0.9 - 4.5 MHz
        .mode = 0,                   // SPI mode 0
        .spics_io_num = ADC_CS_GPIO, // CS pin
        .queue_size = 1,
        .flags = SPI_DEVICE_HALFDUPLEX // 14 bit ADC read only
    };


    // Initialize the SPI bus and attach the device
    ESP_ERROR_CHECK(spi_bus_initialize(ADC_SPI_HOST, &buscfg, SPI_DMA_CH_AUTO));
    ESP_ERROR_CHECK(spi_bus_add_device(ADC_SPI_HOST, &devcfg, &adc_handle));
}
```

Figure 33: Oscilloscope and Voltmeter Initialization SPI Code

```c
int16_t convert_data(uint16_t raw_adc) {
    // Only use lower 14 bits
    uint16_t masked_adc = raw_adc & 0x3FFF; // 0x3FFF = 0011 1111 1111 1111

    // 2s compliment so check first bit for sign
    if (masked_adc & 0x2000) { // 0x2000 = 0010 0000 0000 0000
        // Negative
        return (int16_t)(masked_adc | 0xC000); // 0xC000 = 1100 0000 0000 0000
    } else {
        // Positive
        return (int16_t)masked_adc;
    }
}
```

Figure 34: 2's Complement Conversion Code

Using the value from read_adc(), the function voltmeter() removes the scaling mentioned in the section 5.4.3 and returns the voltage read. The oscilloscope calls the voltmeter function and stores it along with the time which the measurement was taken in an array of structs. The struct contains 300 measurements and will be continuously replaced once all measurements are sent to the app. Finally, if the app sends a signal to trigger the oscilloscope and collects the maximum number of data points before a pair of voltage and time are transmitted to be displayed on the app.

```c
float read_adc()
{
    uint8_t rx_data[2] = {0}; // Buffer store output

    spi_transaction_t trans = {
        .flags = SPI_TRANS_USE_RXDATA, // Internal RX buffer
        .length = 16,                  // Transfer 16 bits
        .rxlength = 16,                // Receive 16 bits
        .tx_buffer = NULL,             // Not transmit
        .rx_buffer = NULL,             // Make rx_buffer null or else error SPI_TRANS_USE_RXDATA
    };

    // Check SPI transaction
    esp_err_t ret = spi_device_transmit(adc_handle, &trans);
    if (ret != ESP_OK)
    {
        ESP_LOGE(TAG, "SPI transaction failed 14 bit ADC: %s", esp_err_to_name(ret));
        return 0xFFFF;
    }

    // Convert raw data into voltage value
    uint16_t raw_data = (trans.rx_data[0] << 8) | trans.rx_data[1]; // Combine rx_data buffers to 16-bit value
    int16_t unvolt = convert_data(raw_data);    // Convert bits from 2s complement
    float convolt = (float)unvolt;
    convolt = (convolt / 8192.0f) * 5.0f; // Reference voltage 5, convert to voltage

    return convolt;
}

// Returns voltage read from voltmeter
float voltmeter()
{
    float voltage = 2 * read_adc(); // Perform scaling to voltage
    // float voltage = (read_adc() - 2.5) * 2;  // Perform scaling to voltage
    return voltage;
}

// Reads voltage and places to measurements struct which oscilloscope reads
void oscilloscope(){
    // if (timer_1_cnt == 0){
    //     measurements[0].voltage = voltmeter();
    //     measurements[0].timestamp = 0;
    //     timer_1_cnt++;
    // } else {
    //     measurements[0].voltage = voltmeter();
    //     measurements[0].timestamp = (float) (stop_timer()) / configTICK_RATE_HZ;

        // }

    if (timer_1_cnt == 0){
        measurements[curr_cnt].voltage = voltmeter();
        measurements[curr_cnt].timestamp = 0;
        timer_1_cnt++;
    } else {
        measurements[curr_cnt].voltage = voltmeter();
        measurements[curr_cnt].timestamp = (float) (stop_timer()) / configTICK_RATE_HZ; // in seconds

    }

    curr_cnt = (curr_cnt + 1) % MAX_MEASUREMENTS;
    if (num_measurements < MAX_MEASUREMENTS){
        num_measurements++;
    }
}
```

Figure 35: ADC, Oscilloscope, Voltmeter Function Code

```
if (oscilloscope_mode)
    {
    // Send oscilloscope data to app

    // Read oscilloscope values
    if (osc_cnt == 0){
        for (int i = 0; i < MAX_MEASUREMENTS; i++){
            oscilloscope();
            vTaskDelay(1);  // wait 1 milisecond
        }
    }
    oscilloscope();
    // If the oscilloscope reaches end of max measurements, it will reset
    if (osc_cnt >= MAX_MEASUREMENTS)
    {
        osc_cnt = 0;
    }

    // Send data
    char data[512];
    int offset = 0;
    offset += snprintf(data + offset, sizeof(data) - offset, "%.2f,%.2f", measurements[osc_cnt].timestamp, measurements[osc_cnt].voltage);
    osc_cnt++;              // Increase count
    esp_gatt_rsp_t rsp;
    memset(&rsp, 0, sizeof(esp_gatt_rsp_t));
    rsp.attr_value.handle = param->read.handle;
    rsp.attr_value.len = strlen(data);
    memcpy(rsp.attr_value.value, data, rsp.attr_value.len);
    esp_ble_gatts_send_response(gatts_if, param->read.conn_id, param->read.trans_id, ESP_GATT_OK, &rsp);
```

Figure 36: Oscilloscope BLE Read to App Code

5.4.4. Validation

The accuracy of the oscilloscope was validated by comparing output string with a 60Hz sine wave. By comparing the first period, it is possible to estimate the accuracy of the system. The current point displays the time in seconds and voltage. Despite satisfying the validation, at higher frequencies, some measurements are displayed as sine waves as the points are not able to be plotted fast enough.

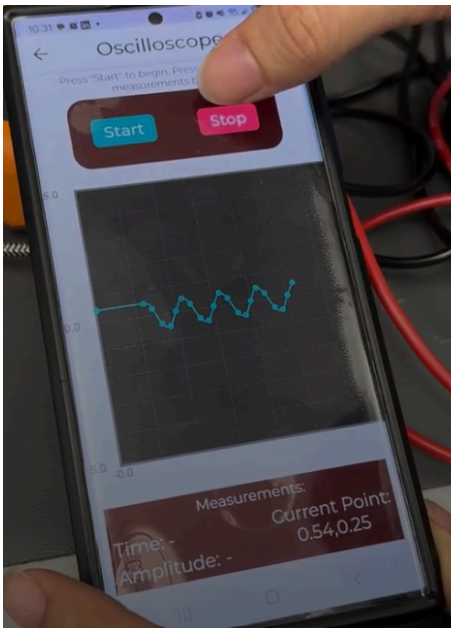


Figure 37: 60 Hz Sine Wave on Mobile App

The following are 1 Hz sine, square, and triangle waves. Stopping the oscilloscope displays the most recent point's time in seconds and amplitude in volts.

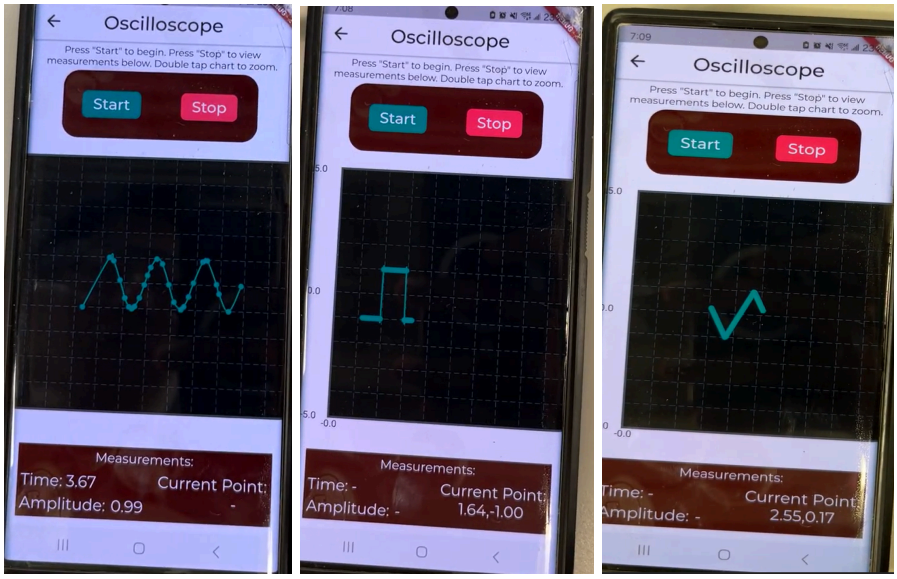


Figure 38: Sine wave, Square Wave, and Triangle Wave on App

5.5. Multimeter

5.5.1. Operation

The multimeter utilizes two ADCs to measure the current, resistance, and voltage. To operate the multimeter, the user will first connect the respective device's positive and negative probes to the circuit being analyzed. Next, the user will power the lab kit and connect it with the app. Upon reaching the app's main screen, the user will click the "Multimeter" button which will bring them to press either the ammeter, ohmmeter, or voltmeter.

As stated above, the oscilloscope and voltmeter operate share the same ADC (ADC141S626CIMM), so details regarding the voltmeter are located in section 5.4. The ammeter and ohmmeter similarly share the same ADC (MCP3201). For the ammeter, the current flows through the current sense amplifier which then generates a voltage. An ADC reads this voltage and the ESP32 converts it into a current value. To measure the resistance, current passes from a voltage source through resistors in series with the resistor under test. The voltage across the resistor under test is read by an ADC, and the ESP32 converts this into a resistance.

5.5.2. Ammeter

The circuit of the ammeter was changed from the previous iteration with the ammeter having an individual ADC. First, the user will turn on the ammeter by turning on the switch

via GPIO 27 which powers the current sense amplifier. Secondly, the current enters the ammeter's positive probe (Io+) and runs across the 1 ohm shunt resistor and exits the negative probe (Io-) back to the user's test circuit. The current across the shunt resistor triggers the MAX4080FASA+T to output a voltage, Vout, using the equation stated in section 5.2.4. This voltage is read by the MCP3201 and then sent to the ESP32. By inverting the equation from 5.2.4, the equation:

Iload = Vout / (Av * R)

where Vout is voltage read by the ADC, Iload is the current across the load, Av is the gain of 20, and R is a shunt resistor of 1.

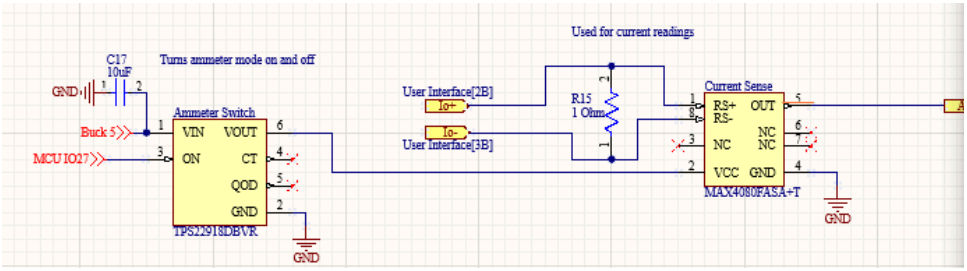


Figure 39: Ammeter Circuit, Power Switch and MAX4080FASA+T

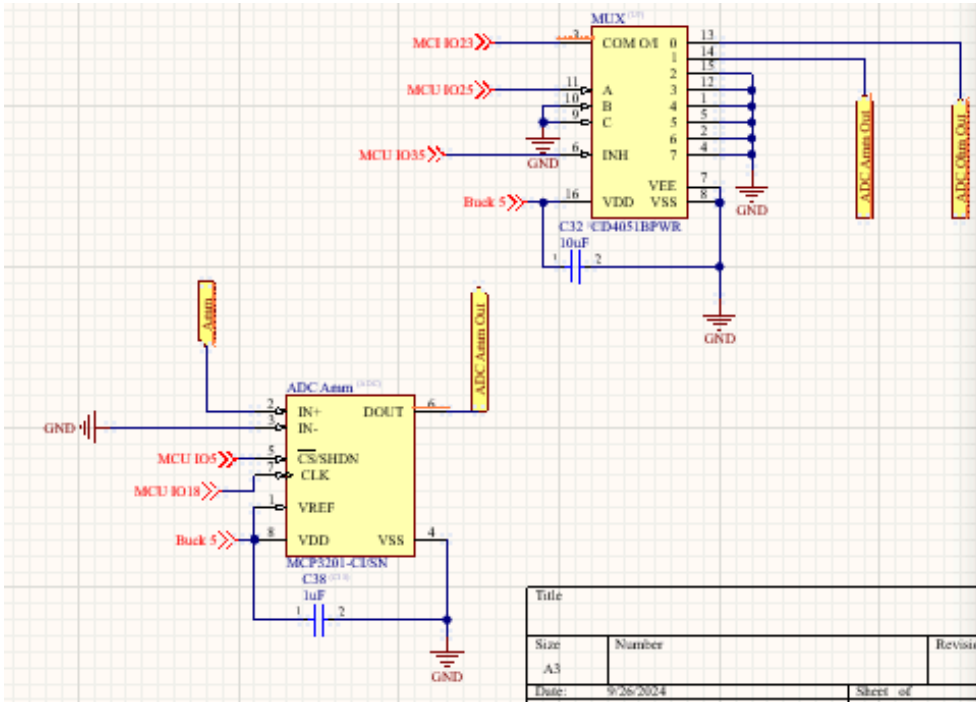


Figure 40: Ammeter Circuit, MCP3201 ADC and MUX

Due to the ESP32 Wroom 32E N4's limited GPIO's, the ammeter and ohmmeter's ADC's share CS, CLK, and VSPIQ pins. Each ADC's output enters a MUX which receives the

ohmmeter's output via GPIO 19 when GPIO 25 is high and the ammeter’s output when it is low.

Since the ohmmeter and ammeter share the same ADC, the initialization and how each ADC reads is the same.

First, SPI communication is initialized with GPIO 5 as CS, 18 as SCLK, and 19 as VSPIQ. The SPI device’s clock speed is set to 1.6 MHz, SPI mode 0, and at half duplex mode. In a similar fashion to the oscilloscope and voltmeter, the SPI transaction is set with a length of 16 bits, rx_ buffer's length of 16, rx_buffer and tx_buffer set to null, and set to SPI_TRANS_USE_RXDATA so it only uses an internal RX buffer. Then, the GPIOs to power the switches and mux are set along with the VSPIQ GPIO set to pull up mode. When the function read_adc_amm_ohm() is called, spi_device_transmit begins transmission and the data is stored 8 bits at a time into rx_data[0] and rx_data[1]. Rx_data[0] is shifted 7 bits after removing its first 3 bits in and is logically and with rx_data[1] once its least significant bit is removed.

Within the function ammeter(), the voltage from the ADC is multiplied by the voltage reference, 5 V, and divided by 12 bits or 4095. This voltage is plugged into the equation above which derives the load current that is sent to the mobile application.

```c
#define PIN_OHM_AMM_MISO 19 // Output for 12 bit ADC
#define PIN_OHM_AMM_CLK 18  // Serial Clock for 12 bit ADC
#define PIN_OHM_AM_CS 5     // Clock Select for 12 bit ADC


#define PIN_AMM_SW 27       // Switch for ammeter
#define PIN_OHM_SW 32       // Switch for ohmmeter
#define PIN_MUX 25          // Mux: 0 ohm and 1 amm output

spi_device_handle_t spi_amm_ohm;


void init_spi_amm_ohm()
{
    spi_bus_config_t buscfg_amm_ohm = {
        .miso_io_num = PIN_OHM_AMM_MISO,    // 12 bit ADC output
        .mosi_io_num = -1,                  // Not used
        .sclk_io_num = PIN_OHM_AMM_CLK,     // 12 bit ADC Serial Clock
        .quadwp_io_num = -1,                // Not used
        .quadhd_io_num = -1,                // Not used
    };


    spi_device_interface_config_t devcfg_amm_ohm = {
        .clock_speed_hz = 1600000,      // 1.6 MHz
        .mode = 0,                      // SPI mode 0
        .spics_io_num = PIN_OHM_AM_CS,  // 12 bit ADC Clock Select
        .queue_size = 1,                // 1 transaction each
        .flags = SPI_DEVICE_HALFDUPLEX, // 14 bit ADC read only
    };


    // Initialize the SPI bus
    ESP_ERROR_CHECK(spi_bus_initialize(VSPI_HOST, &buscfg_amm_ohm, SPI_DMA_CH_AUTO));
    // Attach the MCP3201 to the SPI bus
    ESP_ERROR_CHECK(spi_bus_add_device(VSPI_HOST, &devcfg_amm_ohm, &spi_amm_ohm));

    // set up GPIO
    esp_rom_gpio_pad_select_gpio(PIN_AMM_SW);           // Ammeter
    gpio_set_direction(PIN_AMM_SW, GPIO_MODE_OUTPUT);   // Ammeter


    esp_rom_gpio_pad_select_gpio(PIN_OHM_SW);           // Ohmmeter
    gpio_set_direction(PIN_OHM_SW, GPIO_MODE_OUTPUT);   // Ohmmeter


    esp_rom_gpio_pad_select_gpio(PIN_MUX);              // Mux
    gpio_set_direction(PIN_MUX, GPIO_MODE_OUTPUT);      // Mux
}
```

Figure 41: Ammeter and Ohmmeter Initialization SPI Code

```c
uint16_t read_adc_amm_ohm()
{
    uint8_t rx_data[2] = {0};
    spi_transaction_t t = {
        .flags = SPI_TRANS_USE_RXDATA,  // Internal RX buffer
        .length = 16,                   // Transfer 16 bits
        .rxlength = 16,                 // Receive 16 bits
        .tx_buffer = NULL,              // Not transmit
        .rx_buffer = NULL,              // Make rx_buffer null or else error SPI_TRANS_USE_RXDATA
    };

    // Check SPI transaction
    esp_err_t ret = spi_device_transmit(spi_amm_ohm, &t);
    if (ret != ESP_OK)
    {
        ESP_LOGE(TAG, "SPI transaction failed 12 bit ADC: %s", esp_err_to_name(ret));
        return 0xFFFF;
    }

    // Pullup for MISO or Output
    gpio_set_direction(PIN_OHM_AMM_MISO, GPIO_MODE_INPUT);
    gpio_set_pull_mode(PIN_OHM_AMM_MISO, GPIO_PULLUP_ONLY);

    // Convert raw data into voltage
    uint16_t raw_data = ((t.rx_data[0] & 0x1F) << 7) | (t.rx_data[1] >> 1); // Turn 16 bits int 12 bits from datasheet
    return raw_data;
}
```

Figure 42: Ammeter and Ohmmeter ADC Reading Code

```
// Ammeter returns current.
float ammeter()
{
    // Convert raw data from ADC to current

    gpio_set_level(PIN_AMM_SW, 1);      // Turn on ammeter
    gpio_set_level(PIN_MUX, 1);         // Mux to ammeter

    vTaskDelay(pdMS_TO_TICKS(50)); // Delay for 50 ms

    float Vout = ((float)read_adc_amm_ohm() * 5.0) / 4095;  // Current
    printf("Voltage from ammeter: %3f \n", Vout);

    vTaskDelay(pdMS_TO_TICKS(50)); // Delay for 50 ms

    gpio_set_level(PIN_AMM_SW, 0);      // Turn off ohmmeter
    gpio_set_level(PIN_MUX, 0);         // Mux to default or 0

    float amm_val = Vout / (20 * 1);     // Calculate current from MAX4080 gain 20 shunt 1 ohm

    return amm_val;
};
```

Figure 43: Ammeter Function Code

```
else if (voltmeter_mode)
{
    // Send voltmeter data to app
    // Send "12" as the response when voltmeter mode is enabled
    float temp = voltmeter();   // Get voltage

    // Send data
    char data[512];
    int offset = 0;
    offset += snprintf(data + offset, sizeof(data) - offset, "%.3f V", temp);
    esp_gatt_rsp_t rsp;
    memset(&rsp, 0, sizeof(esp_gatt_rsp_t));
    rsp.attr_value.handle = param->read.handle;
    rsp.attr_value.len = strlen(data);
    memcpy(rsp.attr_value.value, data, rsp.attr_value.len);
    esp_ble_gatts_send_response(gatts_if, param->read.conn_id, param->read.trans_id, ESP_GATT_OK, &rsp);
}
```

Figure 44: Ammeter BLE Read to App Code

5.5.3. Ohmmeter

Similar to the ammeter, the ohmmeter's circuit was altered by providing it with an individual ADC. First, the user will turn on a switch which provides the ohmmeter 5 V by raising GPIO 32 to high. Then, the mux will be set to the ohmmeter's ADC after setting GPIO 25 low. Next, the user will connect the resistor being tested in series with the ohmmeter's positive and negative probe. The positive and negative probes are each in series with a 10k ohm resistor. Once the circuit is closed, the voltage across the test resistor is measured by the MCP3201 and configured on the ESP32 for the measured resistance.

The ohmmeter is configured such that if the voltage across the test resistor is known, it is possible to calculate the current across the circuit and thus the resistance of the unknown resistor. The equation is given as: Rtest = (Vs - Vr) / 20K ohm

where Rtest is the test resistance, Vs is the voltage from the switch (5 V), and Vr is the voltage measured by the ADC,

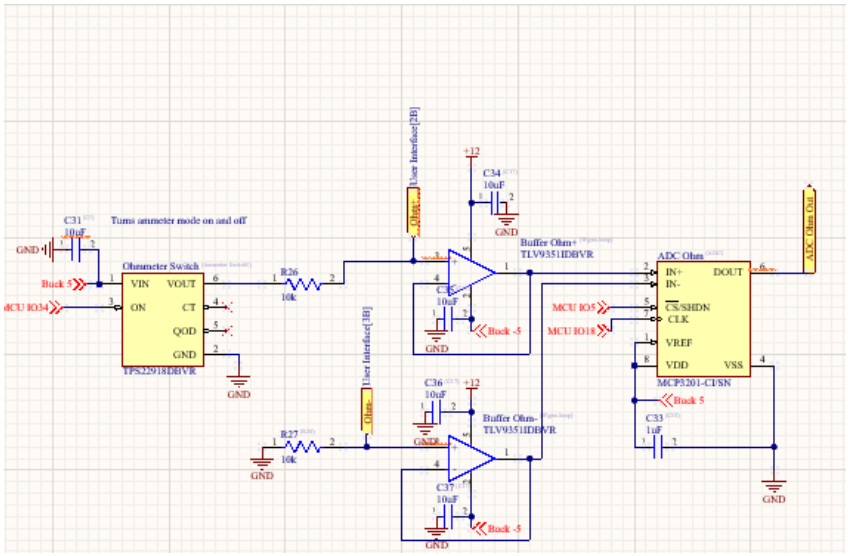


Figure 45: Ohmmeter Circuit, Power Switch and ADC

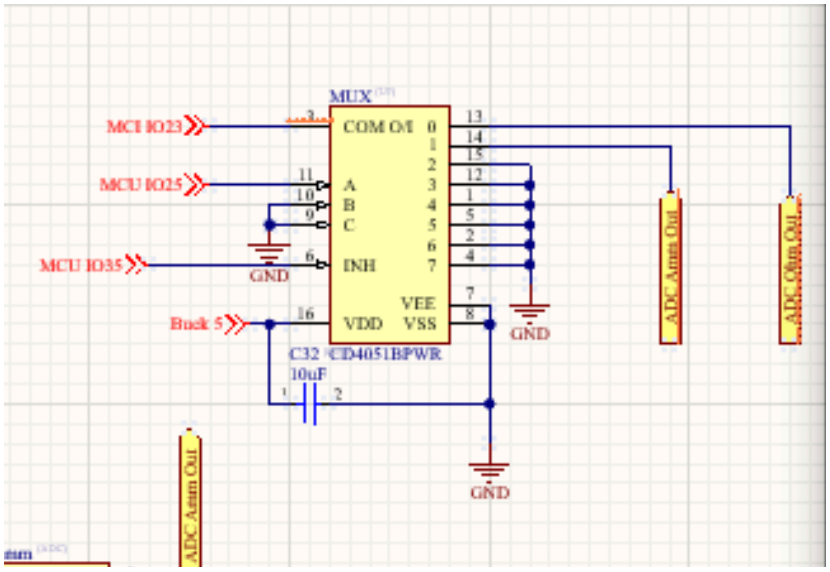


Figure 46: Oscilloscope Circuit, Mux

The setup and output of the ADC is described in 5.5.2 as stated above. The difference between the ammeter, and ohmmeter is after receiving the voltage across the resistor, the equation for Rtest is applied before sending the resistor data to the user. In addition, given an open circuit has infinite current, there is a safeguard which limits the maximum voltage read to 1 M ohm.

```
// Ohmmeter returns resistance.
float ohmmeter()
{
    // Convert raw data from ADC and into resistance

    gpio_set_level(PIN_OHM_SW, 1);      // Turn on ohmmeter
    gpio_set_level(PIN_MUX, 0);         // Mux to ohmmeter

    vTaskDelay(pdMS_TO_TICKS(50));      // Delay for 50 ms

    float Vres = (read_adc_amm_ohm() / 4095.0) *5;  // Voltage across resistor

    vTaskDelay(pdMS_TO_TICKS(50));      // Delay for 50 ms

    gpio_set_level(PIN_OHM_SW, 0);      // Turn off ohmmeter

    float Itotal = (5.0 - Vres) / 20000;    // Calculate current through resistor

    // Make sure if is open circuit or resistance is too high, it isnt infinity
    float ohm_val;
    if (Itotal > 0.0001){
        ohm_val = Vres / Itotal;    // Calculte resistance (Vres / Ires)
    } else {
        ohm_val = 1000000;  // Capped at 1 Mohm
    }

    return ohm_val;
};
```

Figure 47: Ohmmeter Function Code

```
else if (ohmmeter_mode)
{
    // Send ohmmeter data to app
    // Send "7" as the response when ohmmeter mode is enabled
    char data[512];
    float temp = ohmmeter();
    if (ohmmeter >= 1000000){
        ESP_LOGI(TAG, "Very large resistance");
    }
    int offset = 0;
    offset += snprintf(data + offset, sizeof(data) - offset, "%.3f Ohms", temp);
    esp_gatt_rsp_t rsp;
    memset(&rsp, 0, sizeof(esp_gatt_rsp_t));
    rsp.attr_value.handle = param->read.handle;
    rsp.attr_value.len = strlen(data);
    memcpy(rsp.attr_value.value, data, rsp.attr_value.len);
    esp_ble_gatts_send_response(gatts_if, param->read.conn_id, param->read.trans_id, ESP_GATT_OK, &rsp);
}
```

Figure 48: Ohmmeter BLE Read to App Code

5.5.4. Voltmeter

Most of the operation is displayed in section 5.4.1, 5.4.2, and 5.4.3 as the oscilloscope and voltmeter share the same ADC. The only difference is that the voltmeter has a separate function to deliver data to the app as shown below:

```
else if (voltmeter_mode)
{
    // Send voltmeter data to app
    // Send "12" as the response when voltmeter mode is enabled
    float temp = voltmeter();   // Get voltage

    // Send data
    char data[512];
    int offset = 0;
    offset += snprintf(data + offset, sizeof(data) - offset, "%.3f V", temp);
    esp_gatt_rsp_t rsp;
    memset(&rsp, 0, sizeof(esp_gatt_rsp_t));
    rsp.attr_value.handle = param->read.handle;
    rsp.attr_value.len = strlen(data);
    memcpy(rsp.attr_value.value, data, rsp.attr_value.len);
    esp_ble_gatts_send_response(gatts_if, param->read.conn_id, param->read.trans_id, ESP_GATT_OK, &rsp);
}
```

Figure 49: Voltmeter Function Code

5.5.5. Validation

To validate the multimeter, the ammeter measured 0.5 A and 0.005 A, the ohmmeter measured 10k ohm, 1k ohm, and 100 ohm, and voltmeter measured 5 V, 2.5 V, and 0 V. The ammeter was the most inaccurate and had trouble working. This is due to the small currents being measured and ,relative to most, a large shunt resistance. Additional issues may be attributed to buffers interfering with the circuit.

Voltage		
Expected (V)	Actual (V)	Percent Error
5	4.96	0.8%
2.5	2.435	2.6%
0	.024	~~~

Table 3: Voltage Validation Results

Resistance		
Expected (ohm)	Actual (ohm)	Percent Error
10000	9814.343	1.89%
1000	1005.385	0.09%
100	108.28	7.6%

Table 4: Resistance Validation Results

Current		
Expected (A)	Actual (A)	Percent Error
0.5	0.470800	5.84%
0.005	0.005372	7.44%

Table 5: Current Validation Results

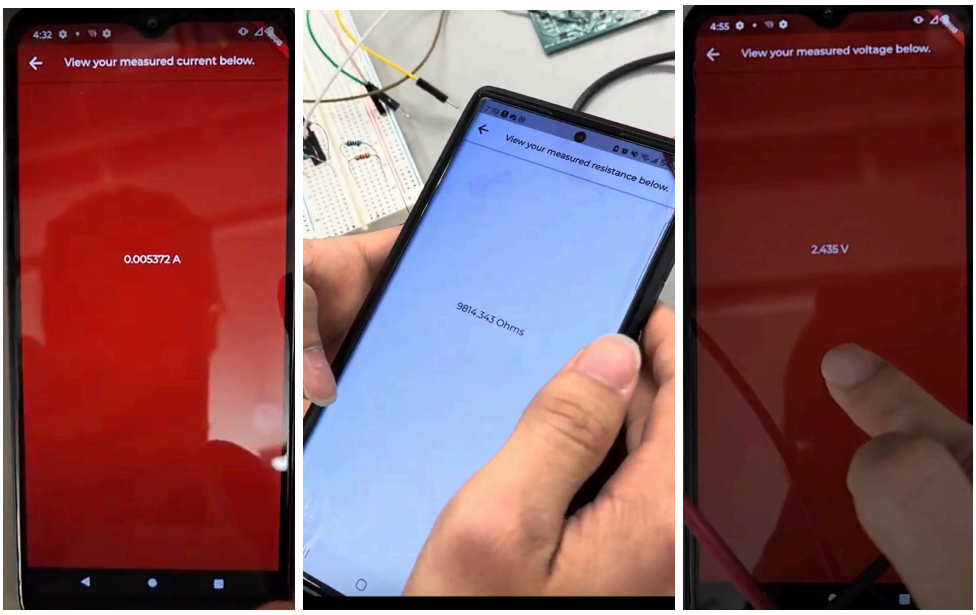


Figure 50: Ammeter, Ohmmeter, and Voltmeter App Readings

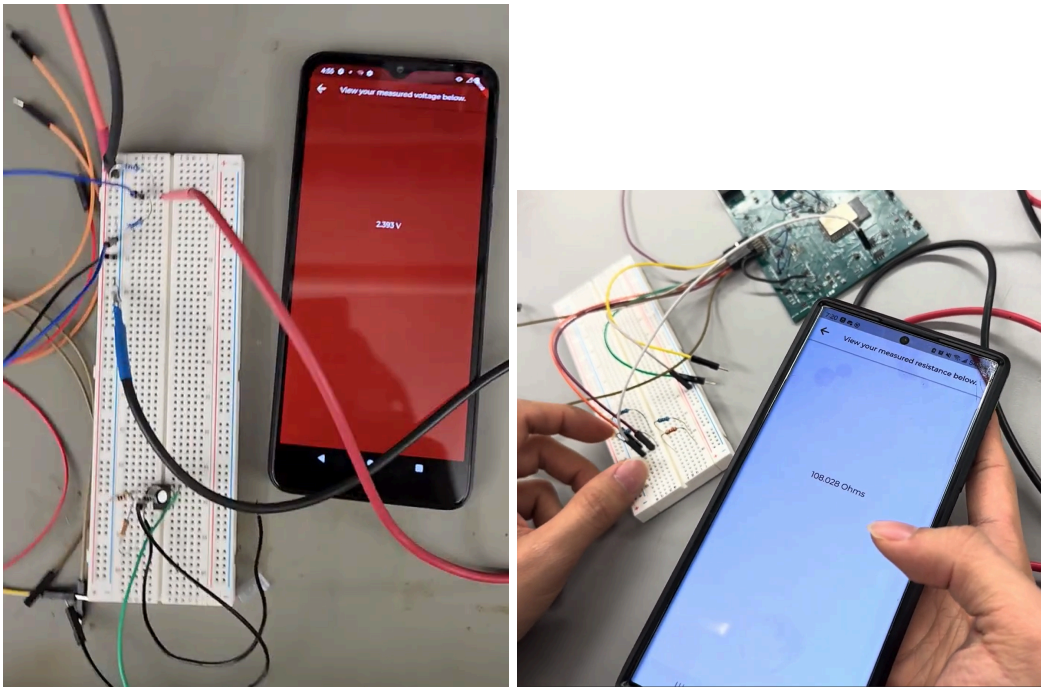


Figure 51: Voltmeter and Ohmmeter System

5.6. Hardware Communication

To demonstrate that the ESP32 was capable of communicating to a phone application, a BLE server would send data to a Bluetooth app. The mobile application would write a message which would trigger the read notification. Depending what was written, the

corresponding value would be sent to the app. More examples are demonstrated 5.4.4, 5.5.2, 5.5.3, and 5.5.4.

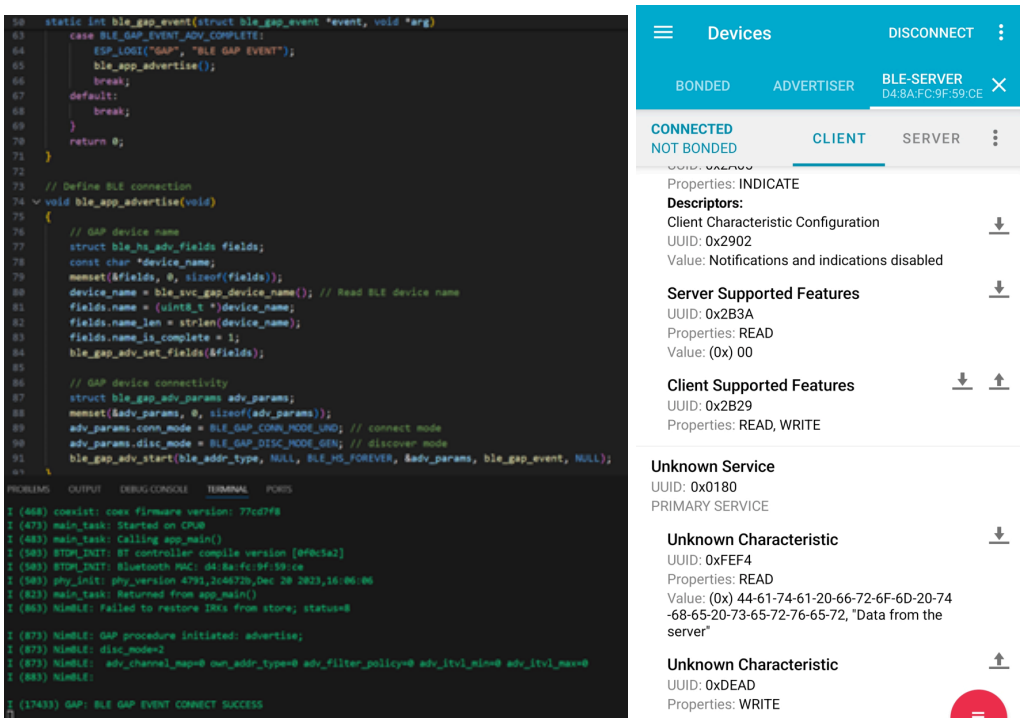


Figure 52: BLE Server and Communication to App

5.7. Conclusion

Both the oscilloscope and multimeter were shown to operate within the specified parameters in the validation. The voltmeter and ohmmeter's errors measured are acceptable when compared to the AD2's readings. In the oscilloscope, the values are a bit slower than expected due to the ESP32's memory limitations and BLE's transfer speed. The reliability issues from the ammeter are possibly due to noise from parasitic circuit elements from the PCB and unexpected behaviors from the buffering amplifier. However, the ESP32 and app's BLE connection is stable with successful read and write functionalities. The oscilloscope and multimeter are an integral part of the overall system as it is required to properly measure and analyze circuits completed in the ECEN 215 Labs that will be displayed on the mobile application.

ECEN 215 Lab Kit

Luis Diaz-Santini
Ryan Freed
Yusuf Hossain
Peter Zhang

SYSTEMS REPORT

REVISION – Final
2 December 2024

Systems Report
FOR
ECEN 215 Lab Kit

Prepared by:

_____
Author Date

Approved by:

_____
Project Leader Date

_____
John Lusher II, P.E. Date

_____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
-	12/2/2024	ECEN 215 Lab Kit		Begin creation
	12/2/2024	ECEN 215 Lab Kit		Update and Review for 404 Submission

Table of Contents

List of Tables

List of Figures

1. Overview

The Principles of Electrical Engineering (ECEN 215) course at Texas A&M University introduces non-electrical engineering majors to the fundamentals of circuit analysis and electronics. The goal of transitioning the course to an online format presents a significant challenge due to its lab component, which currently relies on the Analog Discovery 2 (AD2). The AD2 is a versatile but expensive piece of equipment that is difficult for the average student to be able to afford. Additionally, the AD2's wide range of features can overwhelm students who only need very few specific tools for ECEN 215's lab assignments.

The proposed ECEN 215 Lab Kit provides a solution to these issues by offering an affordable, simplified alternative designed specifically for the lab's needs. With an approximate price of $50 - $60, the Lab Kit streamlines the lab experience by focusing on the essential tools required for ECEN 215, enabling students to complete experiments conveniently from home. Acting as a power supply, waveform generator, multimeter, and oscilloscope, the Lab Kit eliminates the unnecessary tools that come with a device like the AD2 while maintaining the functionality needed to meet learning objectives.

2. Execution Plan

Table 1 shows the execution plan the team followed for the second capstone semester. The semester started with the debugging and testing of our respective completed subsystems from the end of semester 1. During these first couple of weeks, we ran into issues involving the multimeter and wave generator functions. The multimeter was redesigned because of the unreliable bipolar voltage capabilities of the ADC for the ammeter and ohmmeter. The wave generator function was also redesigned on the PCB because of the unnecessary usage of a wave gen IC. Due to these road bumps, subsystem integration was delayed by around 1 - 2 weeks. Once these adjustments were made the appropriate board was printed out and integration was able to begin.

Task	Member	Sem. 1	18-Sep	16-Oct	30-Oct	21-Nov
BLE communication with ESP and App Development	Luis					
Outputting Waveforms	Yusuf					
Working Oscilloscope and Multimeter	Peter					
Assembled First iteration of Lab Kit	Ryan					
Redesign of Multimeter	R + P					
3D printed Case Design	Ryan					
Wavegen and App integration	L + Y					
Multimeter and App integration	L + P					
Real-Time Graph Display on App	L + P					
Wavegen Adjustments on PCB	Y + R					
Full System Integration	Team					

Table 1: Second Semester Execution Plan

3. Final Product

3.1. Integration Process

Integrating this project from four distinct subsystems into a single efficient operating system was a complex and daunting task, but the integration was ultimately successful.

The first step in the integration process was connecting the mobile application with the waveform generator. This required updating the preliminary Bluetooth firmware to handle specific messages that would trigger the activation of particular waveforms. For example, the app sent the message “Lab 4 square” to the ESP32 via the established Bluetooth connection, prompting the ESP32 to output the corresponding square wave. The application was programmed to send six specific messages to the ESP32 depending on user input, each activating one of the required waveforms. The output was validated using a real oscilloscope, confirming that the frequency, amplitude, phase shift, and offset matched the expected values.

Next, the mobile application was integrated with the oscilloscope. Since the oscilloscope graph on the mobile app was already functional and ready to plot data, the primary task was ensuring that the ESP32 sent data in the correct format. Proper formatting was critical because the app could not plot the signals accurately if the data was invalid. The oscilloscope data was

organized as voltage readings at specific timestamps, formatted as “voltage, time” before being sent to the app. Additionally, the firmware was updated to ensure the oscilloscope data was transmitted only upon request. For instance, the app sent the message “Oscilloscope” to the ESP32, which then pushed the data only when this message was received.

The next stage involved integrating the multimeter. Since the multimeter subsystem was already capable of accurately reading values and the app was ready to display this data, the primary task was adjusting the firmware to ensure data transmission occurred only when requested by the app. When voltmeter data was needed, the app sent the message “Voltmeter” to the ESP32, which then transmitted the corresponding data. The same process was applied to the ammeter and ohmmeter functions.

The final step in the integration process was implementing all components on the custom PCB designed for this project. The firmware developed throughout the integration process was flashed onto the ESP32 onboard the PCB. One final adjustment was made to the firmware, enabling the GPIO pins controlling power supply activation to respond to application inputs. With this setup in place, the entire system was subjected to validation and testing, the results of which are detailed in the **Validation Data** section.

3.2. Operation

To operate this device, users must have a physical ECEN 215 Lab Kit and a Bluetooth-enabled mobile phone. If using a pre-configured lab kit, the user simply presses the "Enable" button on the case to activate the device and connect to it via the mobile application. The app can be downloaded from any app store on any Bluetooth-compatible device.

To utilize the Lab Kit’s functionalities, users connect the appropriate instrument wires as needed, and the device will display the measured data. The location of the wire connections are shown in the figure below.

Figure 1: Orientation of ECEN 215 Lab Kit I/O Pins

The wires are also color-coded for clarity: red for the +/- power supply, blue for the +/- voltmeter/oscilloscope, orange for the +/- ammeter, brown for the +/- ohmmeter, black for ground, and yellow for the waveform generator.

4. Validation Data

4.1. PCB and Power

The PCB and power system successfully powers all on-board ICs with up to a 30 mV error as seen in **Subsystem Reports Table 1**. The power systems have a small yet negligible charging time that will likely go unnoticed by consumers as a result of a large system capacitance. The +5 supply switch works as expected as seen in **Subsystem Reports Table 2** and delivers an accurate voltage to the external supply wire that turns on and off with a simple button press in the app. The -5 V supply delivers an accurate voltage to the negative supply, however, it does not turn off. A 3D printed casing has been developed to house the PCB that protects the unit from dust, dirt, and minor damages. The casing is about 24% larger than the average phone but is still mobile and pocket-sized. The overall estimated price after fabrication and assembly sits at $57.20 per board, wall adapter included as seen in **Subsystem Reports Figure 16**. For further details on the power system, price breakdown, and size results see **Subsystem Reports Section 2**.

4.2. Mobile Bluetooth Application

The Bluetooth app features a simple, easy to navigate interface (as seen in **Subsystem Reports Figure 17**), compatible with both iOS and Android devices. The app discovers devices within 4 seconds, constantly reconnects within 1.5 seconds, and receives data from the PCB within 0.25 seconds of the data request. The app has been successfully installed and operated on multiple devices with various software between two and 10 years old. The app offers a stable and consistent connection in environments with heavy Bluetooth traffic. It features a live oscilloscope graphing system with a simple zoom feature. The app allows users to switch between multimeter, ammeter, and ohmmeter functionality and provides users with the option to activate and deactivate the power supplies at the press of a button. In addition, the app provides a waveform generator control system that allows users to alter the shape, amplitude, and frequency of input voltage signals. For further details on the app interface and control systems see **Subsystem Reports Section 3**.

4.3. Waveform Generator and MCU

The waveform generator system provides a range of different signal shapes, amplitudes, and frequencies that users can configure through a control system on the mobile app. The signal shapes include a sine, square, and triangular waveform with amplitudes of 0.5 V, 1 V, and 2 V, and frequencies of 1 Hz, 20 Hz, and 30 Hz in accordance with the ECEN 215 lab requirements (see **Subsystem Reports Figures 25-27**). For further details on the waveform generator shape, signal, and amplitude results see **Subsystem Reports Section 4**.

4.4. Oscilloscope and Multimeter

The oscilloscope and multimeter system delivers a live signal graphing system, viewable through the use of the mobile application (as seen in **Subsystem Reports Figure 38**). This system features a simple and easy-to-understand (**Subsystem Reports Figure 50**) multimeter with capabilities such as a voltmeter with a percent error of 7.5% or lower, an ammeter with a percent error of approximately 6%, and an ohmmeter with a percent error of <10%. The voltmeter is capable of reading voltages between -5 to 5 V and the ammeter has capabilities of up to 0.5 A. For further details on the oscilloscope and multimeter system results see **Subsystem Reports Section 5**.

5. Key Decisions

There were a few changes between the design in ECEN 403 and 404. The first was dropping the clock oscillator as it was more convenient to use the ESP32's SCLK. The next one was utilizing Op Amps to scale the oscilloscope and voltmeter's inputs between -5 and 5 V. This method was recommended as the previous had mixed results. Another Op amp was added to the output of the Wave Gen function to act as a buffer in case of an accidental spike in the voltage. In addition to the Op Amps, the ammeter and ohmmeter were given their own ADC as having all the components share the ADC could potentially lead to short circuits. Then, buffering amps were placed whenever entering the ESP32 or at the wires to prevent reverse current as well as protecting the user and lab kit.

6. Learnings

Senior design taught the group valuable lessons such as communication, project development, public speaking, and teamwork. The final project met most of the planned goals and expectations.

As a group, we each learned technical skills such as building and designing circuits in Multisim, communication in Bluetooth, creating 3D structures in Fusion, mobile application development in Flutterflow, PCB design in Altium, soldering SMD and other parts, and utilizing microprocessors such as the ESP32.

A main lesson the group learned was the importance of time management and perseverance. We were usually behind in progress for the presentation throughout the semester, but after the Blitz, we were able to better utilize our time and continue to work until we finished the project.

7. Conclusions

7.1. Limitations

While the project has come a long way and offers many of the features requested, it still comes with a number of limitations such as a harshly limited waveform generator signal. To accommodate a low price and small device the design is configured to only cater to the current ECEN 215 lab manual specifications, which harshly limits the signal amplitude and frequency. Additionally, while this device does provide a reasonable degree of accuracy, it is not suited to conditions in which a precise measurement is required.

7.2. Impact

The achievement this device set out to accomplish is to provide a cheap and convenient alternative to the AD2 so that non-ECEN students could take ECEN 215 online. While some features require more time and polishing, this product still meets the requirements that we set out to achieve. This kit, in conjunction with the ECEN 215 course offered at Texas A&M, can increase educational access to students all over the world by providing a cheap and reasonable introduction to electrical engineering tools. Despite not being suited to high-precision fields, this kit offers a cheap, portable toolset suitable to familiarize students with multimeter operation.

8. Future Work

Overall, the current state of this project can be considered complete. The final product of the ECEN 215 Lab Kit achieves all the objectives outlined in its proposal, featuring a functioning multimeter, oscilloscope, waveform generator, and power supply, all of which can be interfaced through a mobile application via Bluetooth. That being said, there is still room for improvement.

8.1. Communication Latency

Communication latency is a big part of system functionality, so it's safe to say that although latency is at a good point for the system to operate well it can always improve. For instance, the real-time oscilloscope graph is configured to pull data every 250 milliseconds, as testing revealed this to be the optimal frequency for data communication. Increasing the frequency caused a significant lag in the mobile application while decreasing it resulted in the graph updating too slowly. This highlights that, with additional time, it would likely be possible to further optimize data communication to achieve the best latency outcome for the entire system.

8.2. Mobile Application Aesthetics

The mobile application functions exactly as intended, fulfilling all requirements for the system to operate efficiently. That said, app design is a complex process involving various factors such as color theory, accessibility, and user-friendliness. During development, the primary focus was on functionality and ease of use, with aesthetics addressed as the final step in the design process. Given more time to further develop the application and refine the project, the app’s appearance could be improved, potentially making it more visually appealing and engaging for users.

8.3. Code Efficiency

The firmware code for this project was developed with the same primary goal as previously mentioned: prioritizing functionality and operation. In that spirit, the team adopted a systematic approach when writing the firmware. For example, the initial focus was on ensuring that multimeter data could be transmitted accurately before optimizing the activation of the power supplies. However, this approach had the downside of potentially resulting in code that was somewhat disorganized. While the code functions well and no visible issues have been observed in the system, it could be further optimized to run more efficiently and consume less operating power during execution.