SONOMA STATE UNIVERSITY | ENGINEERING SCIENCE

Senior Design Project Progress Report
EE 493 Senior Design Project Planning

R.E.A.C.H.

By:

Jair Pacheco
Kamryn Shigemoto
Juan Jimenez

Spring 2026

Faculty Advisor: Dr. Farid Farahmand, Sonoma State University
Industry Advisor: Mr. Noah Mervine,
Client: Mr. Zac Robinson, Husch Vineyards

Project Website: https://juanjimenez.io/Senior%20Design/reach_seniordesign.html#team
Project GitHub Page: https://github.com/kamrynshigemoto/REACH_SeniorDesign

Abstract

This senior project is designed to address common problems experienced in remote locations, specifically in wineries and vineyards. With the understanding that access to power and reliable connectivity in remote areas is hard to come by, REACH was designed to be an external module that enhances existing remote power systems. By focusing on working with existing solar power systems, REACH offers power management over connected low-power devices, remote monitoring and control, and connectivity via LTE. A user-friendly website dashboard is used to control REACH's outlets through scheduling features, priority assignment, and simple on and off commands. Data analysis is conducted over time so users can understand how the REACH system is operating. The combination of features that REACH offers reduces device downtime, maintenance trips, and operational uncertainties to improve customer satisfaction.

Table of Contents

List of Figures

List of Tables

Section 1: Introduction

1.1 Problem Statement

The problem that REACH aims to address is that there are many remote and rural locations, such as ranches, farms, vineyards, and nature preserves, that rely on field devices like sensors, actuators, and cameras. To be more specific on wineries they use sensors like soil moisture monitors and actuators like automated valves. Another device of note is the frost alarms that need to communicate in order to prevent large losses in their crop yield. Although power generators and remote power systems exist, they lack reliable connectivity and remote power management over the devices. Some of the current methods that are used would be power coming directly off the grid, Power over ethernet (which will be referred to as PoE in this report) and Batteries. As a result, it is difficult to manage these field devices and monitor their status or health. All of these methods deliver power but only 1 of them, PoE, allows for communication. The result of the lack of good solutions comes out in the form of high system down time and regularly needing to go on maintenance trips in order to ensure the system is still operating as intended. This lack of better solutions is where reach was born and aims to address the lack of connectivity and power management that the current systems don't consider.

1.2 Introduction

Wineries along with many other businesses that operate in remote locations are currently dependent on many field devices and sensors in order to ensure efficient operation of their businesses. This may include monitoring crop or plant health through the use of sensors in the field. The use of irrigation systems in order to water crops or simply just opening a gate to allow livestock to roam when they need to. As modern technology advances more and more, many of these tasks are shifting more towards automation and less human input needed in order to do the same tasks quicker, some issues may arise. Some of these devices are so advanced that they can do their tasks completely remotely but when shifting to remote locations, the use of these devices can cause issues.This is where the previously mentioned devices are used and where REACH is looking into assisting. Not only do these devices need to be powered in some way, but they need to be able to receive data from the user or data from the internet and they need to be able to send this data back to the user. Since these devices are being used in the middle of nowhere, getting an internet connection can be difficult and managing these devices is just as obnoxious to do. So what reach is aiming to do, is take the current issues with device management and connectivity and address both at the same time. Providing wifi to those smart devices would allow for the connectivity issues to be solved and our load control would allow for selective control of the field devices. Load management would allow for the system to be integrated into an off the grid power generation system, such as solar panel charging, and do so efficiently. The Wifi would not only provide the connectivity for the sensors but it can also be used by workers who are out in those fields. This would help everyone on the Vineyard to stay connected and informed as to what is happening at any given time.

1.3 Literature Review and Previous Works

Remote power monitoring and control systems have become increasingly important as farms, vineyards, ranches, and other rural operations depend more on electronic field devices. These devices include soil sensors, irrigation controllers, cameras, frost alarms, and other low-power electronics that may be installed far from grid power or reliable internet access. Because many of these systems operate from solar panels and batteries, the main challenge is not only producing power, but also managing how that power is used. Existing products and previous research show that solar monitoring, IoT communication, and load control are all active areas of development. However, most solutions focus on only one part of the problem, while REACH combines remote monitoring, LTE communication, outlet-level control, scheduling, and load prioritization into one portable system.

One common type of existing solution is the commercial solar monitoring platform. For example, OutBack Power's OpticsRE platform is designed to monitor and control off-grid solar systems through an internet-connected device. OpticsRE allows users to view system performance, monitor energy production and usage, verify system health, access historical graphs, receive alerts, and remotely control system parameters [1]. This makes OpticsRE useful for larger solar installations where the main concern is checking the health of inverters, batteries, and charge controllers. However, OpticsRE is mainly designed around OutBack's own ecosystem and higher-level solar power equipment. It is not designed as a portable add-on module that controls several small DC field devices individually. REACH is different because it focuses on the load side of the system. Instead of only monitoring the solar system, REACH allows the user to turn individual USB/DC outlets on or off, create schedules, and assign priority levels to connected loads.

Victron Energy's GX device family and VRM Portal are another example of a mature remote monitoring solution. Victron's GX devices act as the communication center of a system by connecting solar chargers, batteries, and inverter/chargers, while the VRM Portal provides local and remote monitoring, remote firmware updates, and remote setting changes [2]. Victron's VRM dashboard can also control certain device settings and relay switches remotely [3]. This approach is powerful because it connects many parts of a solar energy system into one monitoring platform. The main difference is that Victron's system is built for full energy-system integration, often involving inverters, charge controllers, and batteries from the Victron ecosystem. REACH is smaller and more focused. It is intended to work with typical existing 12 V solar setups and provide practical control over low-power field devices without requiring the user to replace the entire power system.

Morningstar Corporation provides another related category of products through its off-grid solar charge controllers and inverters. Morningstar states that its products are used in off-grid and mission-critical environments, which shows that reliability is a major concern in remote solar applications [4]. These systems are useful for battery charging, solar regulation, and long-term outdoor operation, but they do not directly solve the problem of controlling multiple external field devices through a user-friendly dashboard. In other words, a charge controller can help keep a battery charged and protected, but it does not necessarily allow a vineyard worker to remotely shut off a frost alarm, schedule an irrigation sensor, or prioritize one load over another

when the battery voltage drops. REACH addresses this gap by sitting between the customer's power system and the field devices.

Previous academic work has also explored IoT-based solar monitoring systems. Pitchaimuthu et al. proposed an IoT-based smart solar photovoltaic monitoring and control unit for remotely monitoring PV plant performance through web-based interfaces [5]. This is similar to REACH because both systems use internet connectivity to reduce the need for physical site visits. However, that work is mainly concerned with monitoring the PV system itself, such as performance evaluation and data logging. REACH includes monitoring, but its main contribution is using that information for outlet-level decisions, such as turning off low-priority loads when the battery voltage becomes too low.

Hamied et al. developed a low-cost IoT-based photovoltaic monitoring system for a greenhouse farm. Their system monitored PV voltage, PV current, solar irradiance, and cell temperature, displayed data through a web page, and could notify users through SMS [6]. This is closely related to agricultural applications because it shows how IoT monitoring can support remote farm operation. Still, the system is mostly centered on PV performance and greenhouse power supply. REACH is different because it is not only a data monitoring tool. It gives the user direct power control through four outlets, manual control commands, scheduling, load priority settings, and local display feedback.

Another related project is the IoT-based solar system monitoring and load management system for a small farm. That work used voltage and current sensing with cloud-based access so users could monitor renewable energy production and manage power consumption [7]. Compared to simple monitoring systems, this is closer to REACH because it includes load management. The difference is that REACH makes the load control more specific to remote field devices by using individual outlets, priority assignment, dashboard controls, and scheduled operation. This makes REACH more useful in cases where certain loads are more important than others. For example, a frost alarm or irrigation valve may need to remain powered while a lower-priority device can be temporarily shut off to preserve battery life.

More recent research has also focused on IoT-based battery and energy storage monitoring. Burgio et al. presented an IoT-based solution for monitoring and controlling battery storage systems independently from manufacturer-specific cloud platforms [8]. This is important because many commercial energy products lock users into a specific ecosystem. REACH follows a similar idea by being an external module that can connect to existing solar power systems instead of forcing the user to redesign the whole installation. However, REACH is not just a battery-monitoring system. It uses battery voltage and current measurements to support real-time decisions about connected loads.

Finally, the Particle Boron platform is relevant because it provides LTE-based connectivity for embedded IoT systems. Particle describes the Boron as a cellular-enabled development kit with LTE Cat M1 support, Bluetooth, GPIO, and built-in battery charging support, making it useful when WiFi is missing or unreliable [9]. This directly relates to REACH's communication design because remote vineyards and farms may not have dependable WiFi access. By using LTE communication, REACH can send telemetry data and receive dashboard commands even when the system is installed away from buildings or wired networks.

Overall, existing commercial systems such as OpticsRE and Victron VRM are strong for monitoring full solar energy systems, while academic IoT projects show that low-cost sensors and cloud dashboards can be effective for PV and battery monitoring. However, many of these solutions either depend on a specific manufacturer ecosystem, focus mainly on solar performance, or do not provide practical outlet-level control for small field devices. REACH is different because it combines several features into one portable add-on system: LTE communication, local display, dashboard monitoring, manual outlet control, scheduling, and load prioritization. This makes REACH better suited for remote agricultural environments where the user needs to manage several low-power devices while protecting limited battery energy.

Section 2: Methodology

2.1 Methodology

In order to solve the problems that were mentioned previously, we plan to start with researching existing solutions and seeing how they can be improved while also comparing their functionality with what REACH plans to provide. It's also important that we maintain constant communication with industry advisors, academic advisors, and our client to keep everyone informed of the progress made and receive feedback on changes that need to be made along the way. Keeping that window of communication open also helps us identify what kinds of challenges and risks we should be aware of when it comes to putting REACH together. Utilizing their expertise will help shape how research, testing, and construction is conducted in order to work efficiently and achieve the goals that we've set in place. It's important that we put a lot of time into creating the diagrams used for the hardware and software sides of the project. There are a lot of little details that cannot go unnoticed and unaddressed and it's important to keep them updated with changes we make along the way. A series of tests will need to be conducted in order to make sure that the components we select are sufficient for the senior design standards. The functionality and system tests are also important for making sure that we satisfy the engineering requirements set in place by the group and approved by our advisor and client. These tests will determine whether the progress we've made so far is sufficient or considered a failure that needs to be addressed with further testing and improvement. The immense workload of this yearlong project will be laid out in a detailed schedule so that we can make sure to accomplish things on time and don't spend too much time on unnecessary sections. The division of labor between the group members is essential for maximizing productivity and accountability.

While conducting several rounds of functionality and system tests, some key things that we will be looking at are the power consumption of REACH, voltage and current sensor readings, and the range of connectivity that we provide. Testing the remote control aspect of REACH will also address the key feature that we are hoping to provide. Selection of components, troubleshooting, and calibration will take a while to make sure we finetune the system to the best of our ability and minimize power consumption. Making sure that REACH has necessary circuit protection will also be important so that we can enforce a level of reliability that the customer's devices will be operational in these remote locations. Testing will also be done in order to make sure that REACH is able to operate in remote locations that are exposed to various environmental conditions such as water, wind, and dust.

2.2 Challenges and Risks

There are plenty of challenges and risks that have been discovered and discussed as a team so far. One of these challenges, the largest one we have encountered, being the possibility that there will be days in which the solar charging system that we will connect to will not be generating any power or generating so little that it's completely negligible. This is a very real challenge that many solar panel systems encounter and because of it we had to take action. We have run and created such scenarios and tested system life time in the worst conditions our system could have. It is very little and such further measures had to be taken to ensure that we can extend system lifetime to the most it can be. This is where our power and load management system come in. In utilizing load management via a priority system we can minimize the impact of low power situations and last long enough to reach the next period of sun availability.

Another challenge that has been mentioned and discussed is the risk that the reach module should be placed in a place with either poor connection to an LTE tower or in a place where LTE is simply not available. This scenario, while unlikely, is possible so a potential solution my team has discussed is looking at different antenna types that can be used in order to reduce the possibility that there is not an antenna in range of the module. Something else that could be used instead is that the current operating mode that was previously sent to the module can be saved and used in order to continue normal operation until either the connections strengthen or the user can move the device to a better location for it to be used. This way the whole system doesn’t break upon losing its connection.

2.3 Project Requirements

The following project requirements are the guidelines used to define what is being accomplished and how REACH will appeal to potential customers. The marketing requirements add more definition to the key features that REACH provides and the engineering requirements provide quantifiable goals that will be met through function and system testing.

2.4 Marketing Requirements (MR)

MR1. The system shall provide control over remote field devices using multiple DC power outlets.
MR2. The system shall allow remote scheduling and load management to optimize power usage.
MR3. The system shall support LTE connectivity for remote monitoring and control via a cloud-based dashboard.
MR4. The system shall include a local display that presents clear, easy-to-understand information.
MR5. The system shall easily connect to typical existing solar power systems, including off-the-shelf solar panels, solar charge controllers, and standard DC batteries.
MR6. The system shall monitor and log voltage, current, and power metrics for solar input, battery state, and output ports.
MR7. The system shall include appropriate circuit protection before delivering power to USB/DC outlets.
MR8. The system shall be lightweight, transportable, and easy to install in field locations.

MR9. The system shall be water resistant for outdoor deployment.
MR10. The system shall provide wireless connectivity in remote locations through a built-in WiFi hotspot.

2.5 Engineering Requirements (ER)
ER-1. 4 USB outlets, 5V @ 900mA max, with local and remote control. **(MR1)**

ER-2. Scheduled on/off control and load prioritization via dashboard, ≤ 60s execution latency. **(MR2)**

ER-3. The dashboard shall display the latest system metrics (outlet voltage/current, outlet status) with a resolution of ±0.1V and ±0.1A. **(MR3)**

ER-4. The local display shall present outlet status, estimated charge/discharge time and allow local outlet control. Displayed information shall refresh at least every 5 seconds. **(MR4)**

ER-5. The system shall operate using a solar panel rated for 12V and provide between 30W to 50W, and sustain functionality for at least 5 hours without solar input. **(MR5)**

ER-6. The internal system (excluding USB outlets) shall consume no more than 5W during typical operation to maximize runtime on battery. **(MR5)**

ER-7. The system shall measure and log voltage, current, and power for solar input, battery, and each output port with an accuracy of ±0.2V and ±0.01A, and synchronize logs with the cloud dashboard every 10 minutes. **(MR6)**

ER-8. All output ports shall include protection via manually resettable fuses rated ≤ 1A.**(MR7)**

ER-9. The entire system, including enclosure, outlets, and display, shall weigh ≤ 10 lbs and fit within a container size of 15 x 15 x 35 cm. **(MR8)**

ER-10. The enclosure and all external components shall meet the IP65 standard for water and dust resistance suitable for outdoor deployment. **(MR9)**

ER-11. The system shall create a local WiFi hotspot supporting at least 4 simultaneous connections with the local control webpage loading within 5 seconds of connection. **(MR10)**

ER-12. The system shall execute user-defined schedules for each output port using an onboard RTC with a timing accuracy of ±1 minute, independent of cloud connectivity. **(MR2)**

ER-13. ER13. The system shall receive 95% of telemetry data and scheduling data and log it to the database. **(MR3)**

Implementation

In order to address all of the marketing and engineering requirements that are set for this project, the plan is to test each section and feature of REACH. The main components of REACH are the main control hub and the remote website dashboard. The main control hub houses the sensor and switches used to take measurements and provide control over the connected devices. The remote website dashboard is where the remote monitoring, control, power management, scheduling, and data analysis takes place. These two main components will be broken down into even smaller sections and individually tested to make sure that once all the pieces are put together, REACH will operate as expected.

Section 3: Design Architecture

3.1 System Architecture

REACH is an external module that connects to the customer's existing remote power system, which in many cases is a solar power system. The field devices are connected to REACH's multiple power outlets so that the user can implement the power management features and have remote control over their devices in remote locations. The main control hub contains the sensors and switches used to take valuable current and voltage readings to check on the health of the system and make power calculations. The website dashboard offers many unique features such as scheduling and power management calculations. It should be user friendly and provide useful insight into what's going on with their devices.

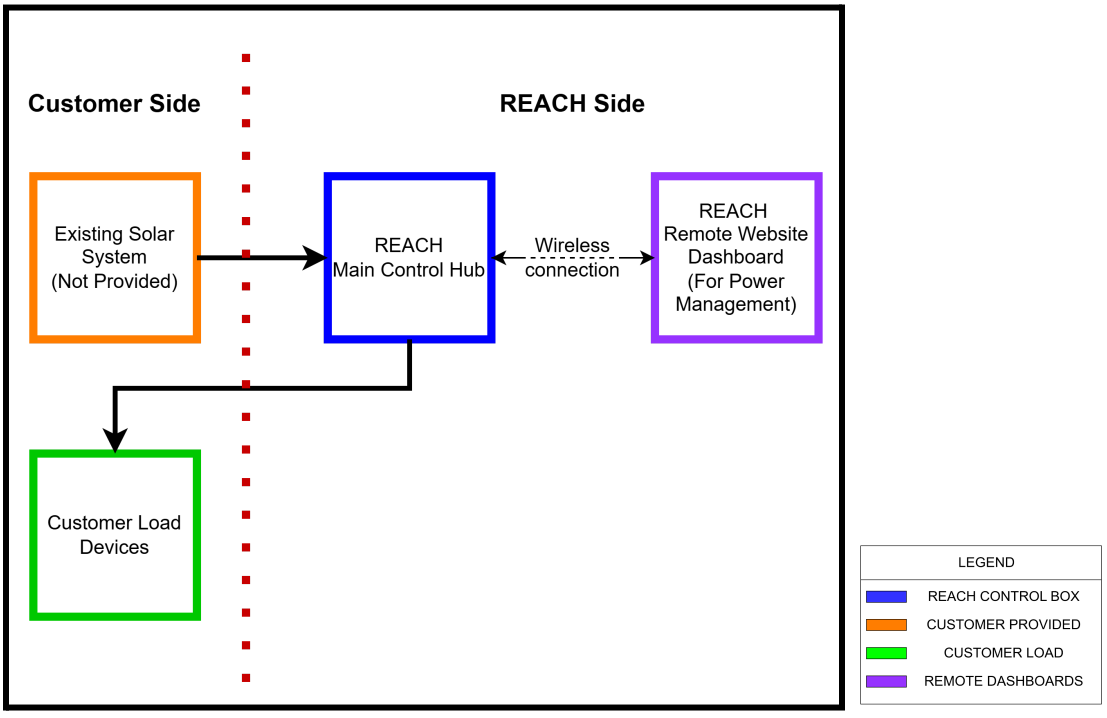

Fig. 1 This is a high level diagram that explains what REACH offers.

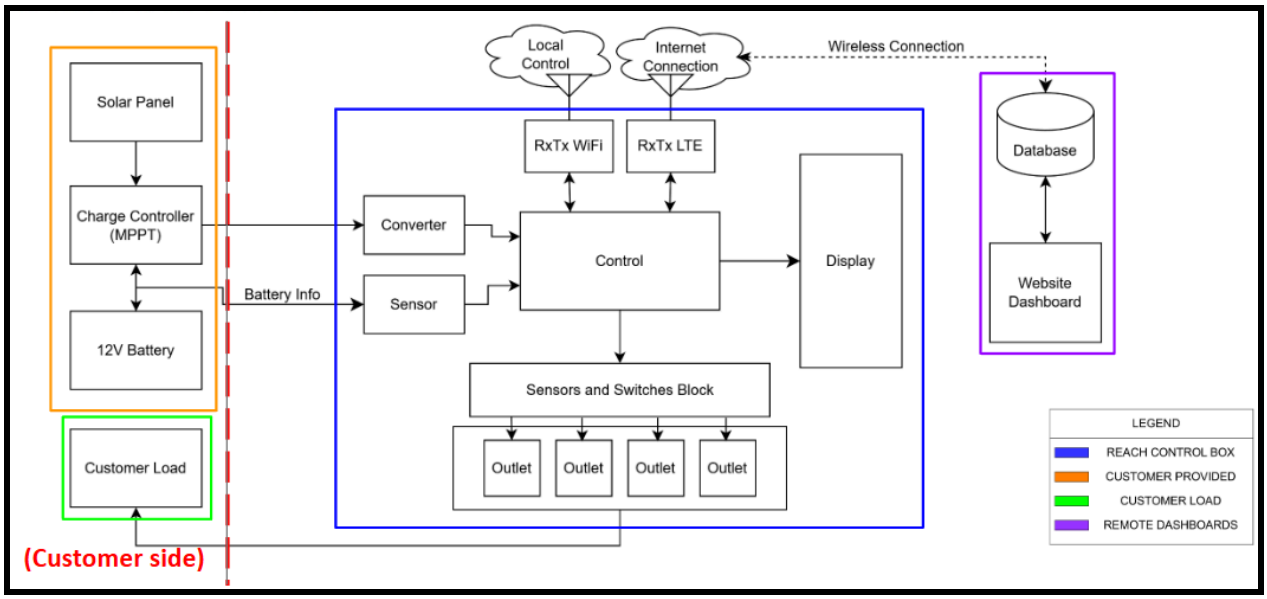

Fig. 2 This is a high level diagram that goes more in depth as to what REACH offers.

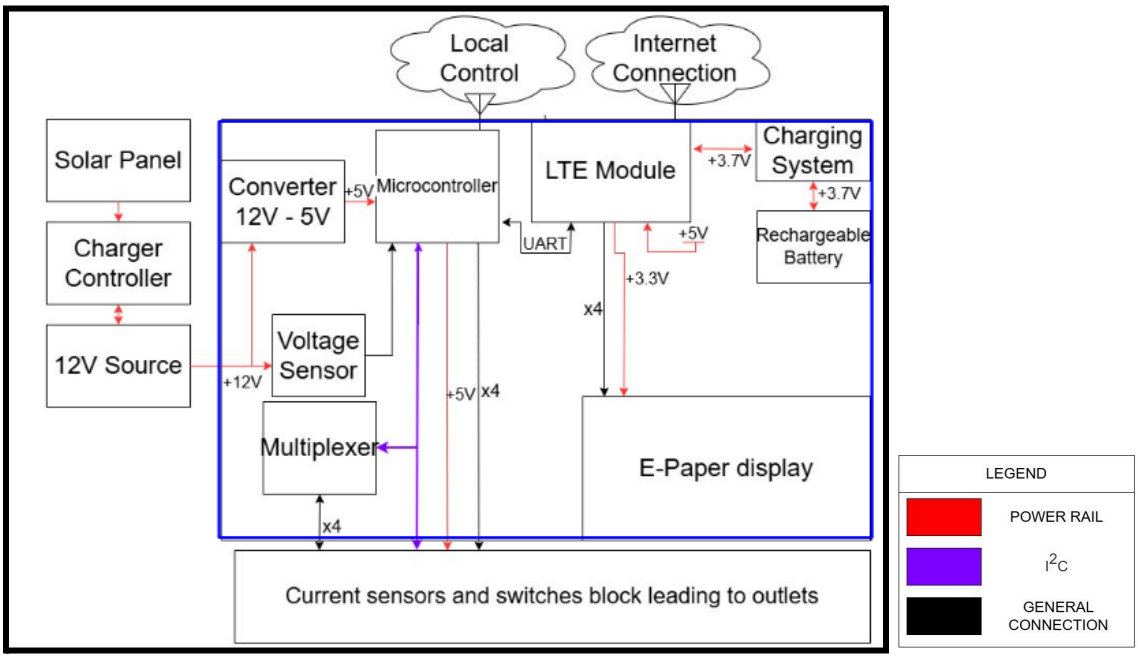

Fig. 3 This is a diagram that shows what is inside the main control hub for our REACH module

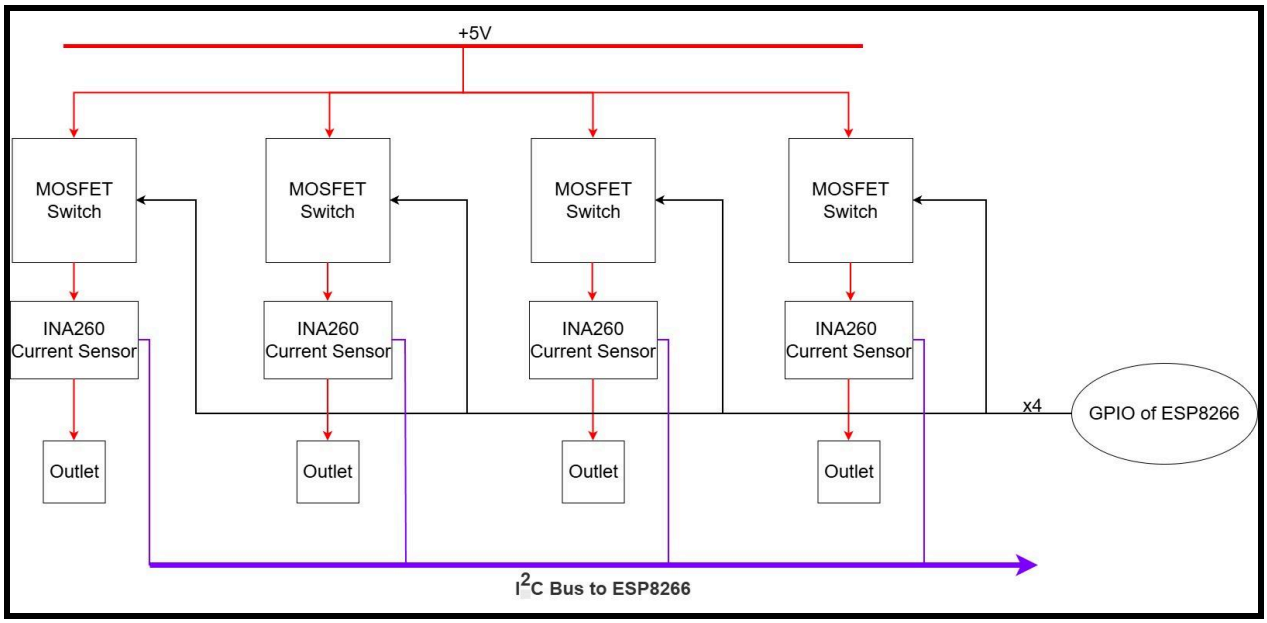

Fig. 4 This is a diagram that shows what is inside the sensor and switches block inside our main control hub for our REACH module

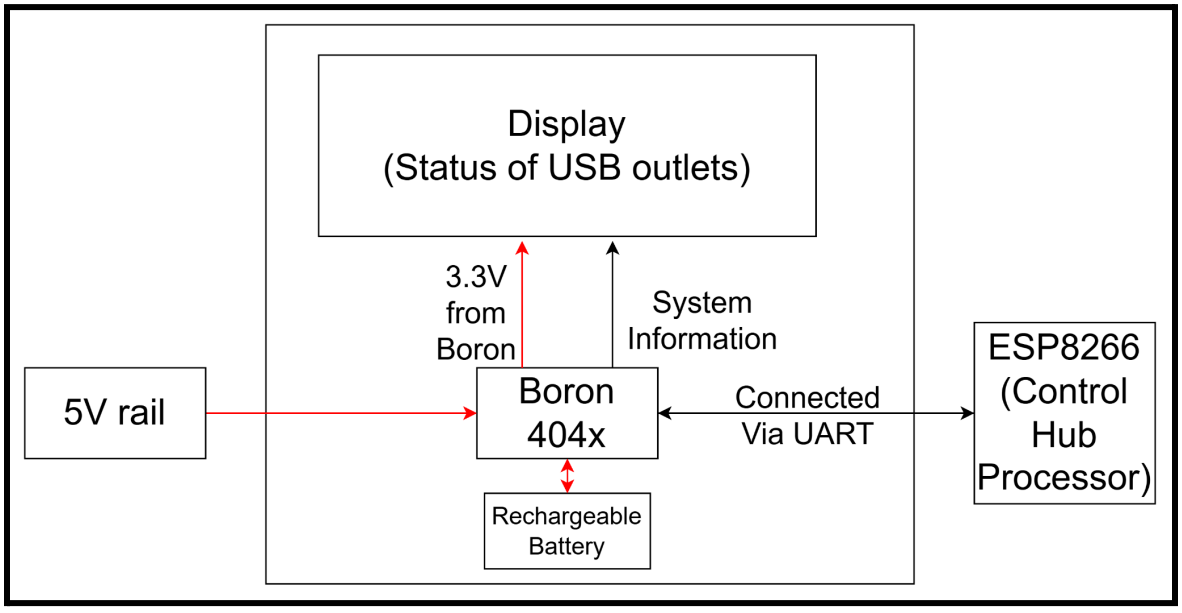

Fig. 5 This is a diagram that shows how we will locally display data on our REACH module

3.2 Design Architecture

Table 1 Main Microcontroller options

Criteria	Weight	PIC18F47K42	ESP8266	Raspberry Pi	ESP32S3
Cost	0.2	0.29	0.30	0.15	0.26
Processing Power	0.05	0.10	0.30	0.40	0.20
Availability	0.05	0.02	0.01	0.96	0.06
Familiarity	0.3	0.22	0.32	0.17	0.30
Clock Speed	0.1	0.04	0.09	0.83	0.04
Power consumption	0.3	0.31	0.33	0.06	0.30
Score		0.23	0.28	0.25	0.25

Table 2 Transceiver Microcontroller

Criteria	Weight	Particle Boron	ESP32 S3	Raspberry Pi	ESP8266
Cost	0.2	0.2	0.32	0.12	0.36
Type of Transmission	0.3	0.33	0.27	0.17	0.23
Ease of Use	0.2	0.25	0.15	0.3	0.3
Support	0.1	0.26	0.22	0.3	0.22
Transmission Rate	0.1	0.18	0.23	0.27	0.32
Power Consumption	0.3	0.45	0.2	0.1	0.25
Score	1	0.37	0.28	0.22	0.33

Table 3 Transistor for Switching

Criteria	Weight	Dual Mosfet Switch	AO3400A	IRLZ44N
Loss	0.2	0.33	0.34	0.33
Cost	0.25	0.36	0.29	0.35
Availability	0.15	0.33	0.33	0.33
Temperature Resistance	0.1	0.43	0.14	0.43
High-Side Simplicity	0.3	0.63	0.25	0.13

Criteria	Weight	Dual Mosfet Switch	AO3400A	IRLZ44N
Score	1	0.44	0.28	0.28

3.3 Budget/Parts List

The estimated budget and parts list is used to keep track of the project's finances and to make sure that the components used are within a designated budget. Cost is definitely a factor that is considered when completing design matrixes before a specific component or part is decided on. It's important to stick within the project budget and make sure all parts are accounted for and justified.

Table 4 Budget and List of Components

Component	**Quantity**	**Individual Price**	**Costs**	**Description**
ESP8266	1	7.99	7.99	Used to control the main power hub
Particle Boron 404x	1	49.95	49.95	Used for long range transmission
YRDZXG [Buck converter]	1	9.99	9.99	Used to step down the voltage from 12V to 5V
ADS1115 [Multiplexer]	1	3.96	11.89	Allows the ESP to interface with multiple analog sensor
INA260	4	17.99	71.96	Used to measure current
DCT-Electronics [Voltage sensor]	5	1.18	5.89	Used to measure voltage
Female USB	4	2.50	10.00	Outlets used to deliver power
Dual MOSFET PWM Switches	4	1.29	12.99	Used to shut off power to outlet
Fuses	4	0.20	9.00	Used to protect circuits
Waterproof Box	1	Free	Free	Used to house our components
Waveshare E-INK display module	1	21.99	21.99	Used to display outlet states and power metrics
Total Cost		117.04	211.65	

3.4 Project Schedule

The project schedule is a guide used to make sure that sufficient progress is made throughout the year and time is used efficiently to reach all of the goals set in place. It is essential that the engineering requirements are satisfied in a timely manner and enough time is set aside to complete all of the testing requirements. The project schedule is also used to delegate tasks between group members and hold everyone accountable for accomplishing assigned work by the deadlines set by the group. Planning out the available time helps with organization and project management.

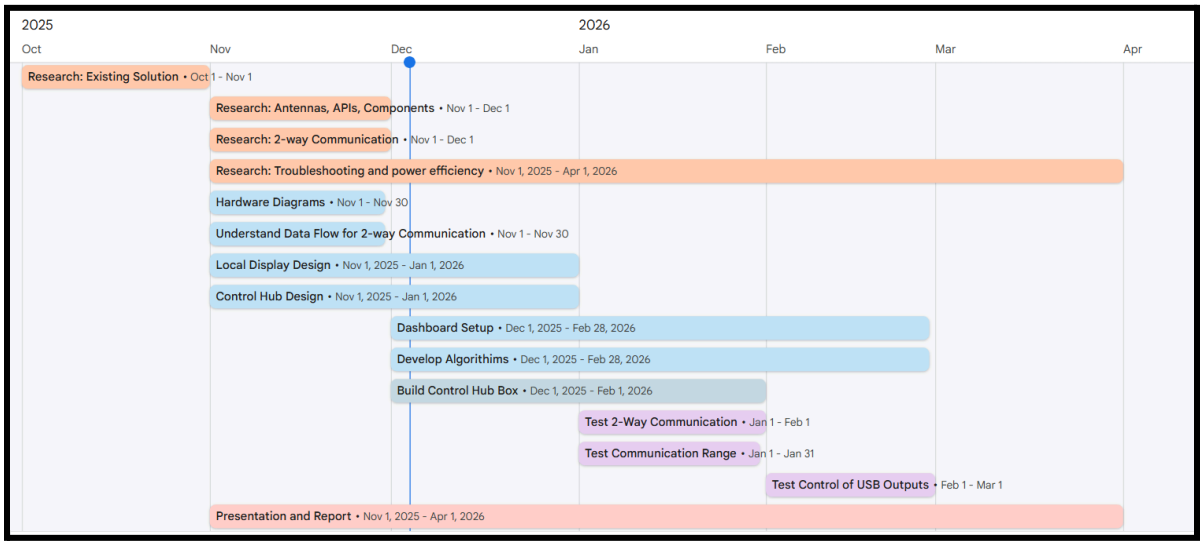

Fig. 6 Proposed schedule that details specific tasks that need to be accomplished in this timeline.

Section 4: Tests

List of Tests

4.1 Fall 2025 Summary of Tests

Below, we present a summary of tests that have been conducted so far.

Table 5 Summary of Proposed and Conducted Tests

Test #	Objective	Related ER	Status	Notes

FT.1	Test basic two-way communication between ESP and Particle Boron	ER1, ER2, ER3	Complete	Continue to test more advancements
FT.2	Test control of website through WiFi connection	ER11	Complete	Successfully changed status of output over WiFi
FT.3	Test current and voltage readings from INA219	ER7	Complete	Looking to optimize IC accuracy or look for another
FT.4	Test range of communication using ESP network	ER11	Incomplete	Looking into antennas to extend range
ST.1	Calculate power consumption of the system	ER6	Complete	Power calculations completed

4.2 Fall 2025 - Description of Tests

FT-1: 2-Way Communication Test

- **Objective:** Test basic two-way communication between the ESP8266 and Particle Boron microcontrollers.

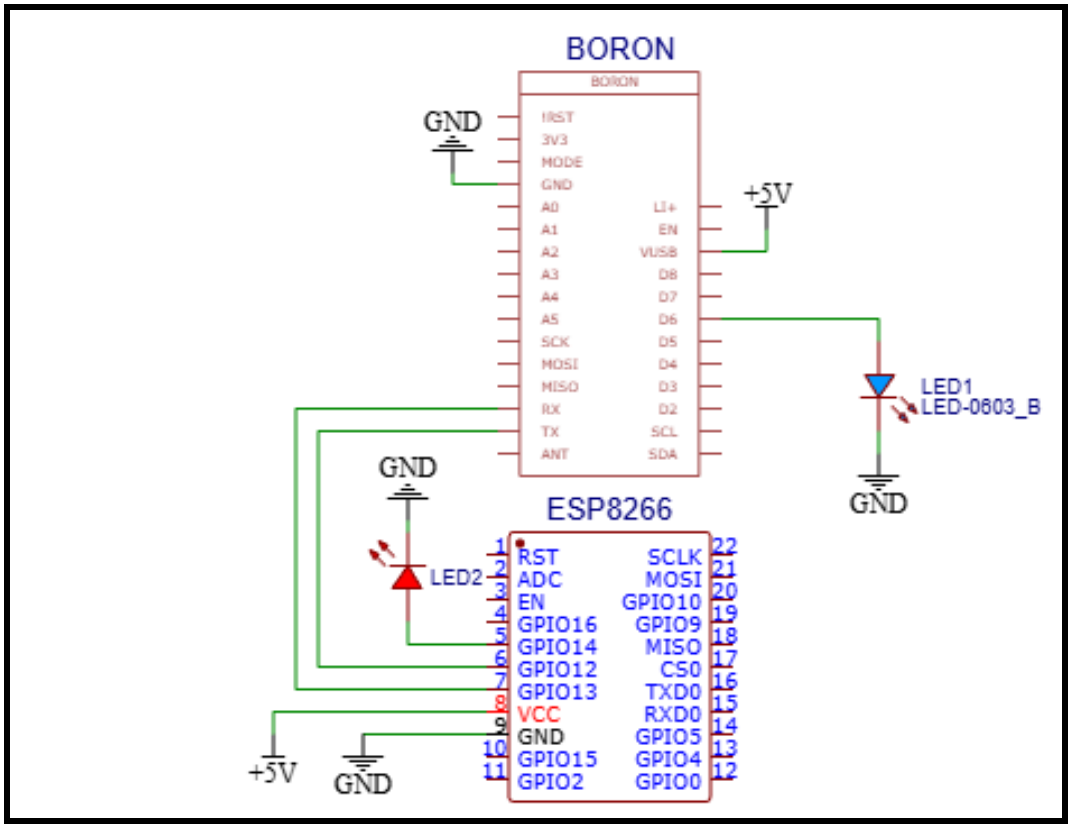

Fig. 7 This is a diagram of the test set up used for the 2-Way Communication Test.

- **Setup:** We will connect the two microcontrollers together serially and utilize UART to send information between them. They will also be connected to power and ground. There will be a type of load connected so that we can test if the two microcontrollers can control it via commands sent. We will also look at the serial terminals on our laptops to see what messages are being sent and received between the microcontrollers.
- **Procedure:** Connect the two microcontrollers serially and make sure they are correctly powered and grounded. Also have the USB connection to our laptops to see the serial terminals. Run code we developed to send messages to other microcontrollers and read what messages are being sent to it. Test commands to turn and LED on and off from each microcontroller.
- **Pass/fail criteria:** Pass if we are able to communicate between the 2 microcontrollers and turn the LED on/off based on commands from ESP website or Boron commands. Fail if you can't send and receive messages between the microcontrollers or control LED.
- **Conclusion:** We were able to successfully achieve 2-way communication between the Boron and ESP8266 because the LED was able to be controlled to turn on and off.

FT-2: WiFi Control Test

- **Objective:** Test website control through WiFi connection.

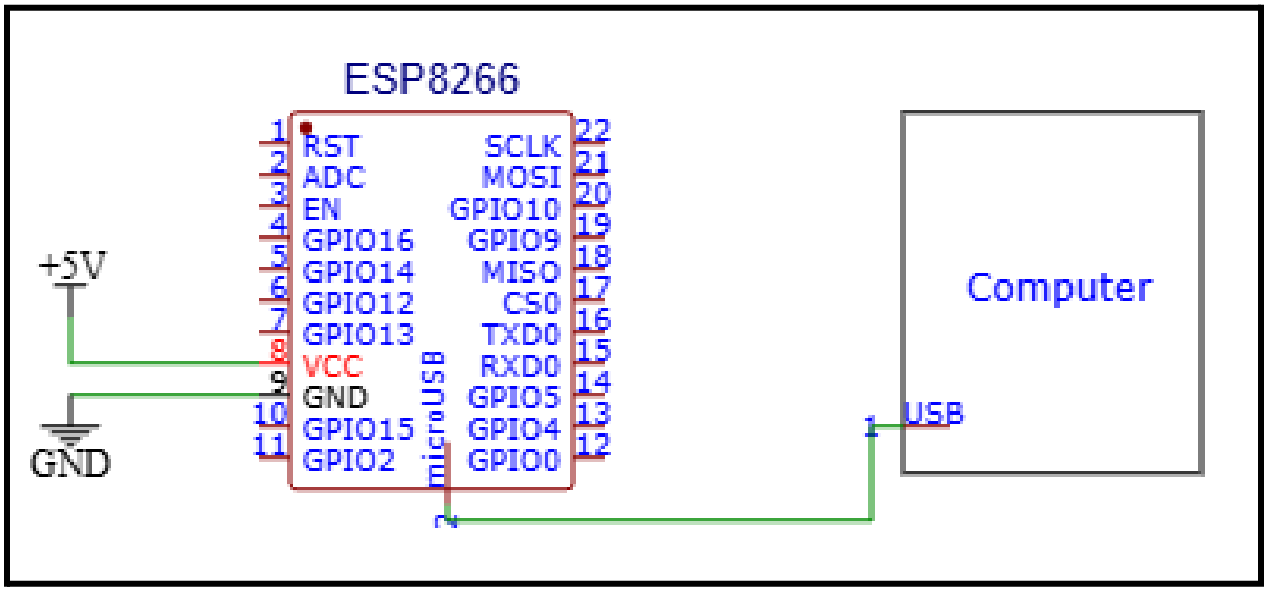

Fig. 8 This is a diagram of the test set up used for the WiFi Control Test.

- **Setup:** The setup just involves connecting the ESP8266 to power, ground, and a laptop so that the serial terminal can be read. Also another device, like a smartphone, to connect to ESP's WiFi network and access local webpage.

- **Procedure:** After writing the code for the program and website, run it on the ESP8266. Then connect the smartphone to the ESP's network and go to the URL of the locally hosted website. Test if the status of the USB outlet can be changed to on/off from the website. Make sure the most recent status of USB is accurately displayed on the webpage.

- **Pass/Fail Criteria:** If we are able to change the status of the USB on the website to on/off then that's a pass. The refresh button needs to show the most recent status of the USB as well in order to pass. Failure would be failure to get the device connected to the website to control the USB and the refresh button doesn't show the correct status of USB.

- **Conclusion:** The website was successfully used to wirelessly control the outlets and refreshed to show the most updated status.

FT-3: INA219 Current and Voltage Sensor Test

- **Objective:** Test current and voltage readings from INA219.

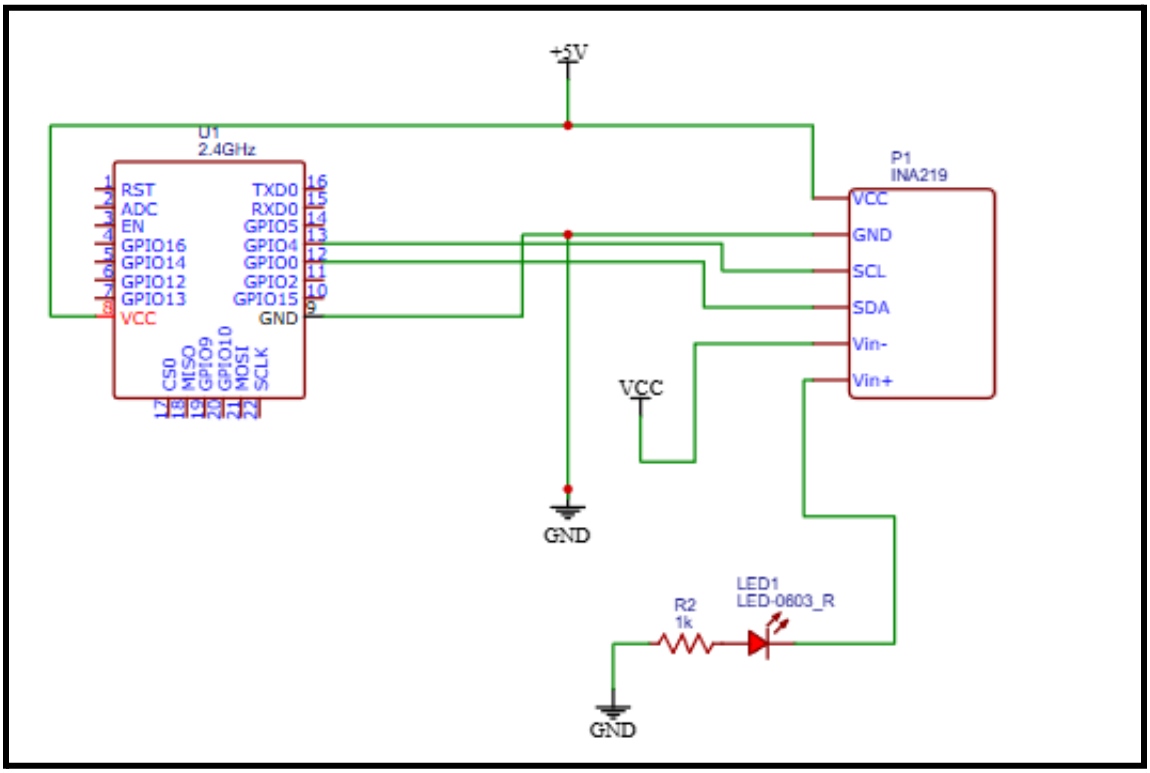

Fig. 9 This is a diagram of the test set up used for theINA219 Current and Voltage Sensor Test.

- **Setup:** Connect INA219 sensor to ESP8266 via I2C. The load connected to the sensor will be an LED.

- **Procedure:** Take voltage and current measurements using the sensor and compare it with the actual voltage and current. Make appropriate comparisons to see how accurate it is and if it's a good fit for the project.

- **Pass/Fail Criteria:** Pass if it's within the accuracy range that we specify in our engineering requirements. Fail if it isn't as accurate as we need it to be.

- **Conclusion:** The current sensor wasn't within the accuracy range we needed it to be, so we will conduct another round of testing with a different sensor to satisfy our accuracy requirements.

FT-4: ESP Network Range Test

- **Objective:** Test range of communication using ESP network.
- **Setup:** Power the ESP8266, upload network code, and connect a device like a smartphone to the network. Measure out distances outside to use to measure range.
- **Procedure:** We would power the ESP8266 with a power supply so that we can take it outside. Then connect a device to the WiFi network. Take measurements at certain intervals and see how far we can go with the connected device before it's out of range and can't connect to the ESP's network.
- **Pass/Fail Criteria:** We will set a certain distance that we hope to reach (like 20m) and if we are still connected to the network at that distance then it's a pass. Otherwise it's a fail.
- **Conclusion:** This was a fail because we didn't have time to get to this test. We dedicated most of our time to actually getting the components to work and decided that this was a lower-priority test compared to the other project demands.

ST.1: System Power Consumption Test

- **Objective:** Measure power consumption of the system.
- **Setup:** Have the whole system put together and take measurements of the power being consumed by each component.
- **Procedure:** Take measurements using the sensors and perform the necessary calculations to figure out how much power is being consumed by each component during different conditions.
- **Pass/Fail Criteria:** If the system consumes 5W or less during typical operation then it's a pass. If it consumes more than 5W it's a failure.
- **Conclusion:** This test was delayed until next semester because we needed a physical system to test.

4.3 Spring 2026 Summary of Tests

Below, we present a summary of tests that have been conducted so far.

Table 6 Summary of Proposed and Conducted Tests (Spring 2026)

Test #	Objective	Related ER	Status	Notes
FT.5	Test scheduling functionality for all 4 outlets for single	ER2, ER12	Complete	Successfully able to run scheduled for all 4 outlets

	instances in one day			multiple times in a day
FT.6	Testing ESP reliability	ER3, ER7	Complete	Successfully able to prove ESP reliability, telemetry data was received by dashboard and database
FT.7	Testing the email alert system for load prioritization	ER2	Complete	Successfully able to send email alerts to user when load prioritization is enabled
ST.2	Testing power consumption of the built system under normal operation	ER4, ER5, ER6	Complete	Successfully operated well below the engineering requirement
FT.8	Testing current readings from the INA260	ER7	Complete	Successfully measured the current accurately and precisely
FT.9	Testing voltage readings from the DCT-Electronic 5:1 voltage sensors	ER7	Complete	Successfully measured the voltage precisely and accurately
FT.10	Load Prioritization Test	ER2	Complete	Successfully shut off low priority lines while leaving high intact

4.4 Spring 2026 - Description of Tests

FT-5: Schedule Reliability Over Time Test

- **Objective:** Test to see if the scheduling feature works for all four outlets for single instances throughout one day.

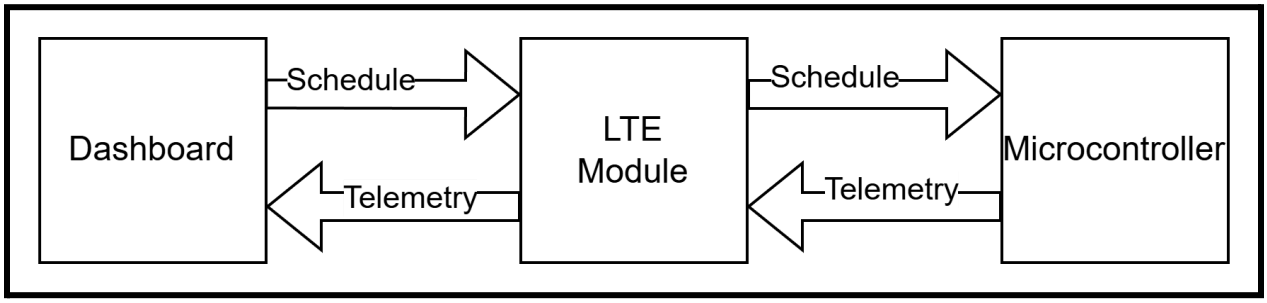

Fig. 10 This is a diagram of the test set up used for the Schedule Reliability Over Time Test.

- **Setup:** The REACH circuit was set up with loads connected to all four outlets. For this test, we used three ESP8266 microcontrollers and one LED as loads. We then used the

single instance scheduler on the Grafana dashboard to schedule periods of time where each outlet was turned ON and OFF. For this specific test each outlet was turned OFF and ON for 15 minutes at a time, and we repeated that three times in one day.

- **Procedure:** Set up the schedules for all four outlets using the single instance scheduler in the Grafana dashboard. Connect loads to each of the USB outlets so that we have evidence that current is being drawn when the load is turned on. Pay attention to the start time for each ON or OFF interval and record the readings from the current sensors at each outlet. Record the current values at each interval and make sure that the outlets are turning on or off at their scheduled times, and lasting for the scheduled duration.

- **Pass/fail criteria:** Pass if all four outlets are able to be turned on and off at the scheduled start times and maintain the target state for the scheduled duration (15 minutes). Fail if any of the outlets aren't able to turn on and off according to the programmed schedule.

- **Conclusion:** In conclusion, we were successfully able to turn all four outlets on and off, 3 times each, in one day. All scheduled segments started at the correct time and most of the data points were sent over to the database, just one point was lost in transmission.

FT-6: Link Reliability Test

- **Objective:** Test link reliability by checking how accurate and consistent the ESP can send telemetry data to the database and dashboard over a period of time.

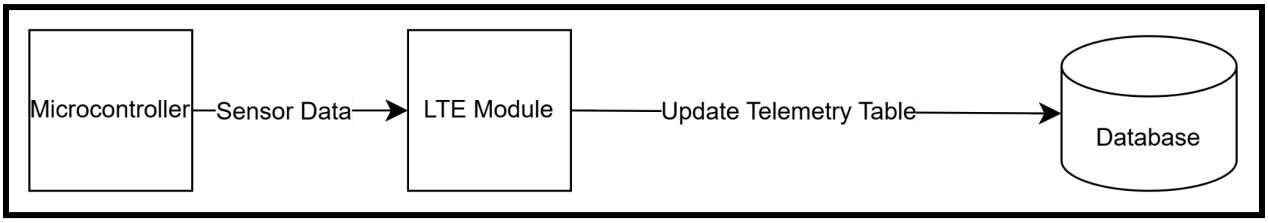

Fig. 11 This is a diagram of the test set up used for the Link Reliability Test.

- **Setup:** We had the REACH system powered on and operating for extended periods of time while doing testing and improvements, so we looked into our database to find a long stretch of time during which everything was operating, loads were connected, and data was being sent over. One specific example we found was on March 7, 2026, when we had the circuit on for about 13 consecutive hours.

- **Procedure:** While analyzing the database information and comparing it with what we were seeing on the dashboard, we expected to see data being sent over 57 times from the ESP. The telemetry data included voltage and current data for each outlet and the battery supply voltage. Once powered on, the ESP sends telemetry data to the database in 10-minute intervals for all four of the outlets. We checked the database to make sure that each time the telemetry database table was updated, it contained data for all four outlets, and none of the values were bogus or unexplainable.

- **Pass/fail criteria:** Pass if all, or most, of the telemetry data was received by the database for all 4 outlets. Fail if we were missing a significant number of data points and/or received bogus values that didn't make sense for the operation at each load.

- **Conclusion:** In conclusion, the test was successful because out of the 57 times data was expected to be sent over only one data point was dropped for outlet 4. This proved that link reliability is reliable and timing was accurate when the system was powered on.

FT-7: Load Prioritization Email Alert Test

- **Objective:** Test to see that Grafana alerts work for the load prioritization function.

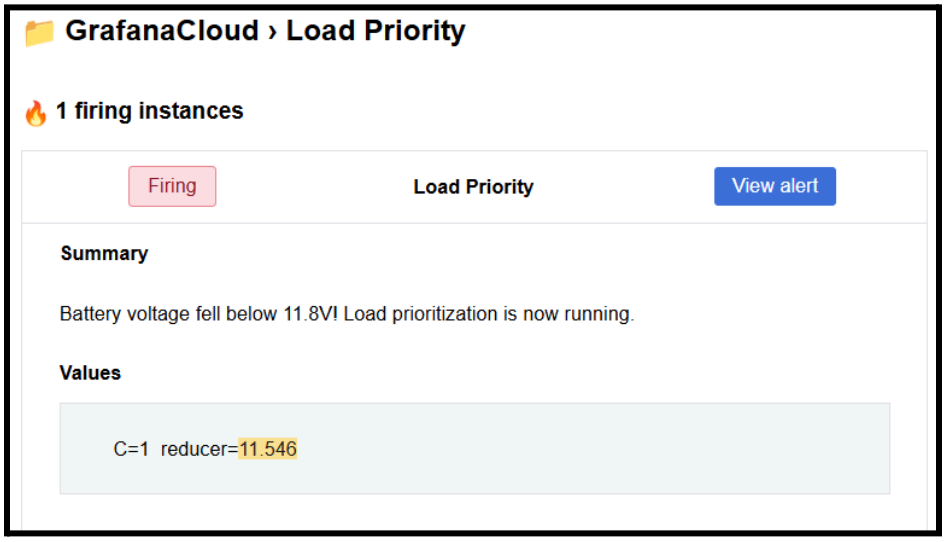

Fig. 12 This is a diagram of the test set up used for the Load Prioritization Email Alert Test.

- **Setup:** We checked email alert functionality three times over a period of two hours. Grafana should send an email alert to the user that load priority is being enabled when the battery supply voltage drops below 11.8V. That voltage is sent over by the ESP in 10-minute intervals as part of the telemetry data.

- **Procedure:** We analyzed the data being received by the ESP on the dashboard and our database in the telemetry table. We looked for instances where the supply voltage fell below 11.8V and looked for an email from Grafana alerting us that load prioritization would be enabled until the voltage rose above the threshold value again. Grafana also has a cool feature that lets us look at the history of the alert over time, and we can see when it was checking the supply voltage value and when the alert was triggered. If the voltage was above 11.8V, we shouldn't be receiving an email alert.

- **Pass/fail criteria:** Pass if we received an email alert when the battery supply voltage fell below 11.8V. Fail if we didn't receive an email when the voltage was below 11.8V or if we received emails when they weren't needed.

- **Conclusion:** In conclusion, the email alerts were sent out successfully to alert the user of load prioritization activation when the voltage level of the battery fell below 11.8V.

FT-8: Current Sensor Accuracy and Precision Test

- **Objective:** Test to see that the sensors are suitable for use of our system for measurements and outlet consumption

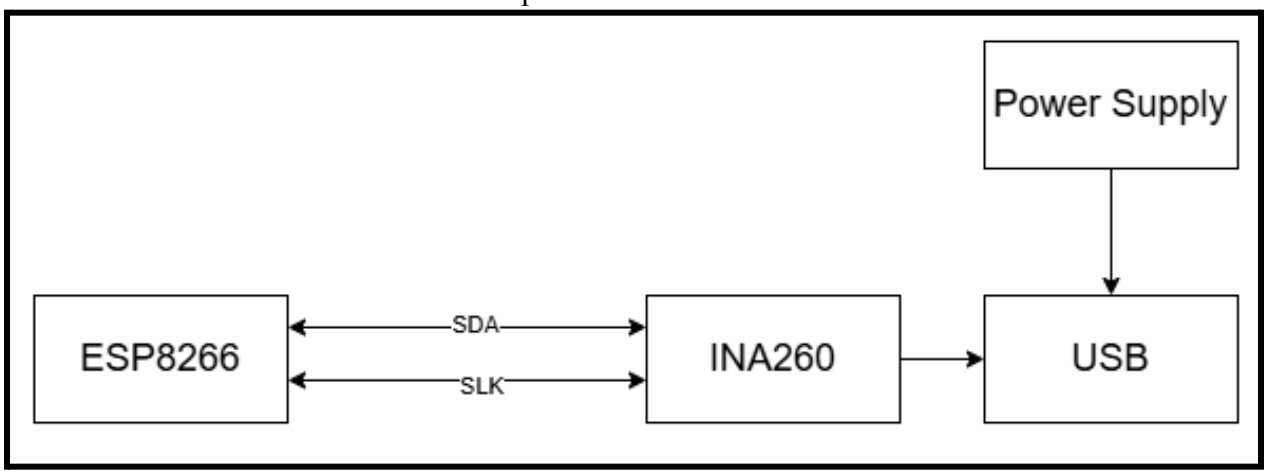

Fig. 13 This is a diagram of the test set up used for the Current Sensor Accuracy and Test.

- **Setup:** We built a circuit consisting of the INA260 current sensor and an arbitrary load, and an ESP8266 microcontroller. The arbitrary load would be powered by a power station and would slowly increase in load current, beginning with a load current of 0 and ending with a current of 1 amp.

- **Pass/fail criteria:** Pass if the sensor can read and send the measured load current accurately within 10 mA of the actual load current value. Fail if this value exceeds more than 10 mA of the actual load current value.

- **Conclusion:** The sensors passed with an accuracy of 10 mA.

FT-9: Volt Sensor Accuracy and Precision Test

- **Objective:** Test to see that the sensors are suitable for use of our system for measurements and outlet operation

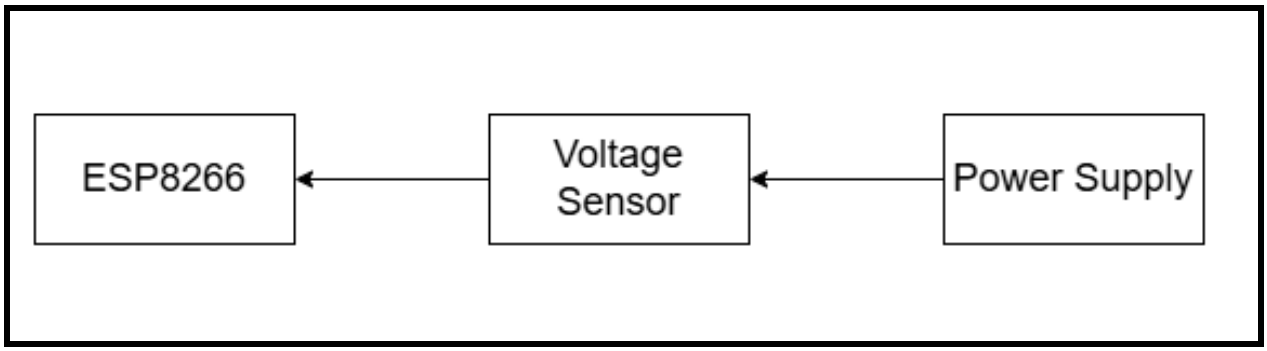

Fig. 14 This is a diagram of the test set up used for the Voltage Sensor Accuracy and Test.

- **Setup:** Using the complete and built circuit, we used the voltage sensors to measure the voltage output of a power bench supply at different voltages ranging from 0 to 15 volts, measuring every .5 volts.

- **Pass/fail criteria:** Pass if after calibration the sensor is able to precisely and accurately measure the voltage at +/- 0.2V and pass that data through the ADS1115 to the ESP data processing. Fail if even after the calibration, the sensor fails to continuously send the correct voltage.

- **Conclusion:** The sensors passed within an accuracy of 0.2 V of the actual measurements.

FT-10: Load Prioritization Test

- **Objective:** Test to see if Load Prioritization Works with varying loads

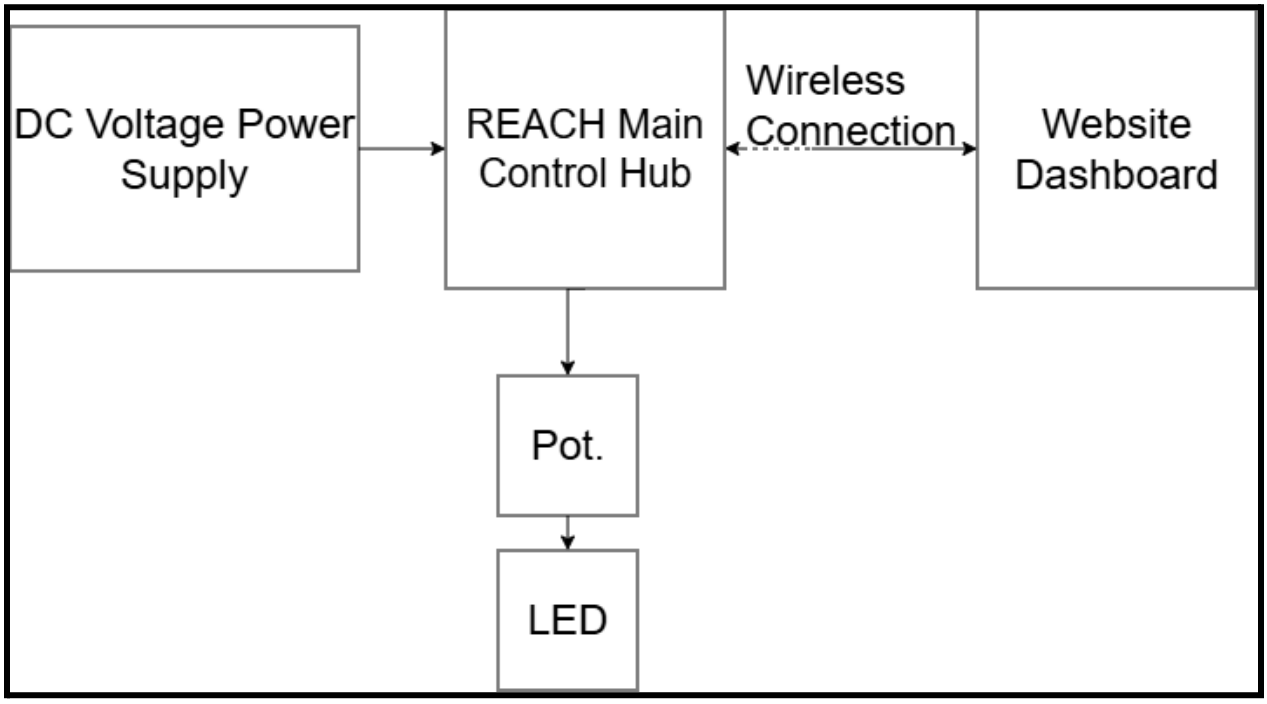

Fig. 15 This is a diagram of the test set up used for the Load Prioritization Test.

- **Setup:** We are using the entire implemented circuit but with the difference being that we don't have outlets for loads. What we have instead are LEDs with potentiometers at the end of the line so that we can adjust current draw and therefore have variable loads of our choosing. Since our load prioritization occurs when the battery starts dropping voltage then instead of a battery connected to our system then we have a DC voltage supply connected in which we can drop below our selected threshold voltages at will.This in conjunction with our variable loads, allows us to test multiple potential cases that our system may experience when the battery stops holding its peak voltage.

- **Procedure:** The procedure was as follows, there were 2 cases of interest that were tested since by proxy they proved all other combinations are good to go. For both cases outlet #1 was considered high priority and every other one was considered low. Case 1 was a scenario in which our high priority outlet (Outlet #1) is drawing a lot of current and other outlets drawing a minimal amount but when the voltage drop occurs, the high priority line is left turned on and despite the other line drawing less, the highest drawing low priority line is shutoff.

    - The case 2 scenario is one in which 2 low priority lines are drawing an equal amount of current (This is defined as 2 lines within 5mA of each other). What this was testing is a scenario in which 2 lines would need to be dropped in order to comply with the “drop the highest drawing load” part of our load prioritization requirement. Once again the voltage is dropped below threshold and the resulting outlets states would show us if it was implemented correctly.

    - It should be noted that the other low priority load should be left on and be untouched unless the voltage drops below our second threshold voltage level [11.5] and this should remain true for all test cases that don't involve it.

- **Pass/fail criteria:** The condition we are checking in order to ensure whether our algorithm is working properly is the outlet state that results after dropping below our first threshold voltage [11.7]. As stated above, our high priority outlet should never turn off under any scenario, provided the system has enough power to run itself. The low priority outlets should be the only things that change during testing and only the highest current drawing load should be shut off. Assuming all conditions are met after the voltage drop then we can conclude that the test was a success, otherwise it has not.

- **Conclusion:** The prioritization algorithm passed as all conditions were met.

FT-11: Power Consumption Test

- **Objective:** Ensure that the entire internal system does not consume more than 5W of energy before outputting power to outlets.

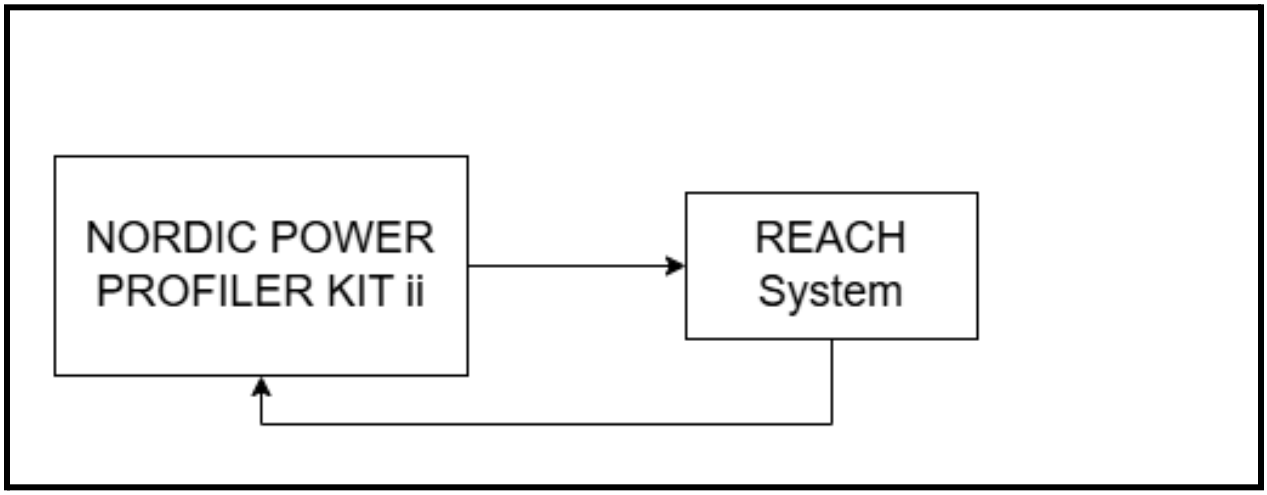

Fig. 16 This is a diagram of the test set up used for the Power Consumption Test.

- **Setup:** Use the NORDIC Power Profiler Kit ii to power and log the power consumption under normal operating conditions of the REACH box. The user interface of the logger module displays the power consumed and current drawn on average.

- **Procedure:** The Power Profiler Kit ii has a downloadable user interface with the capability to log and measure power consumption with an accuracy of 10%. The connection to the REACH box will be operating under normal conditions without activating the mosfet switches. Once the system is powered, the user interface will display and log the power consumption of the system.

- **Pass/Fail criteria:** If the system logs more than 5W of power consumption under normal operating conditions, without attached loads, the system passes the test. If there is more than 5W of consumption, it fails.

- **Conclusion:** The system uses less than 5W of power under normal conditions.

Section 5: Hardware & Software Architecture

5.1 Software Documentation

In this section, we go into more detail about the software architecture we use in our senior design project. Specifically, this section will explain the purpose and functionality of the Grafana dashboard used to control and monitor the system, as well as essential functions used in the coding on the microcontrollers used to operate REACH.

5.1(a) Grafana Dashboard

The purpose of this dashboard is to allow users to remotely monitor and control REACH. The Grafana dashboard is set up to visualize important data for long-term data analysis and real-time observations. It's a user-friendly interface that also provides scheduling and load prioritization capabilities for the four outlets that REACH offers. Since the REACH system will be set up in remote locations, the dashboard gives users the ability to oversee the system and control it in real time.

Justification for Grafana

Our team decided to go with the Grafana platform because we had prior experience using it in an Internet-of-Things class. The familiarity factor was important to us because we didn't want to dedicate too much of our time to learning a new platform, and Grafana had all the capabilities that we required. It also works seamlessly with the database that we have set up with Hostinger, where all of our telemetry, scheduling, and priority data is stored. The visualization panels on Grafana also allow us to easily display important data that the user would want to have easy access to, graph data over a period of time, and take in user input in order to make commands, set schedules, and assign load priority.

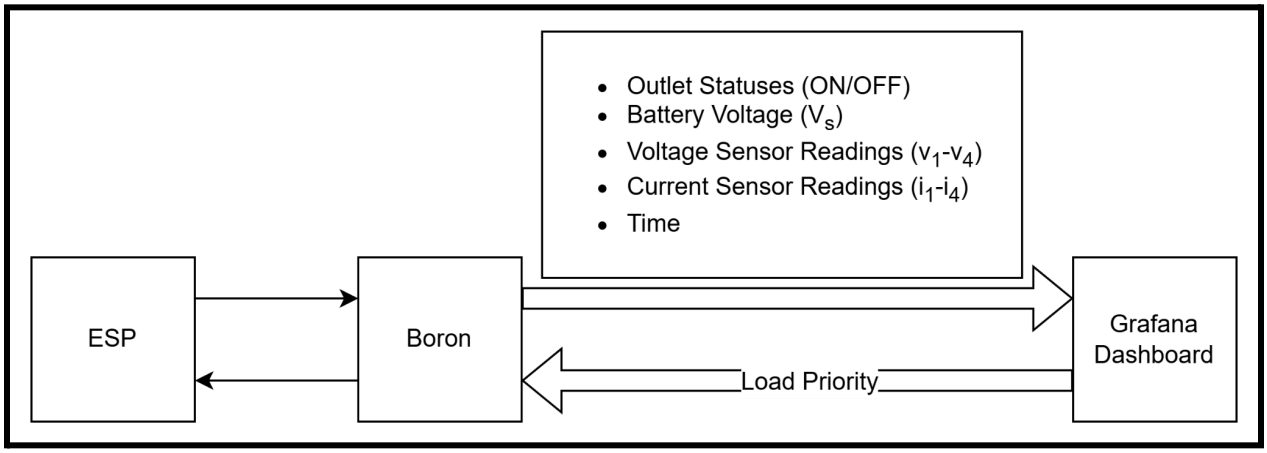

Fig. 17 Data Flow to the Grafana Dashboard

Data Flow

All of the important information for our project is stored in a MySQL database that is run by phpMyAdmin on Hostinger. We have database tables that store telemetry data, schedules, commands, outlet status, and load prioritization. These database tables can be updated from the ESP side of our system or the dashboard side of our system. The ESP sends telemetry data involving current and voltage sensor readings, as well as the outlet status when manual commands and schedules are running. The data from the ESP is sent over to the database tables via the Particle Boron microcontroller, which is able to update the tables through PHP files. It's through the PHP files that the Boron is also able to access data from the database tables and send it over to the ESP in order to run commands, send schedules, and send the load prioritization set by the users. The sensor data that is received from the ESP can be visualized in different ways, like tables and graphs, so that the user can see immediate responses and analyze data collected over long periods of time.

Grafana has a type of visualization panel called Business Form panels that take in user input and can update the database tables when PHP files are set up to use HTTP post methods. These Business Form panels that we have on the dashboard allow the user to set schedules and outlet priority status remotely, and control REACH in real time. When changes are made to the schedules and outlet priority status, the respective database tables are updated with new information that the Boron polls periodically and sends over to the ESP.

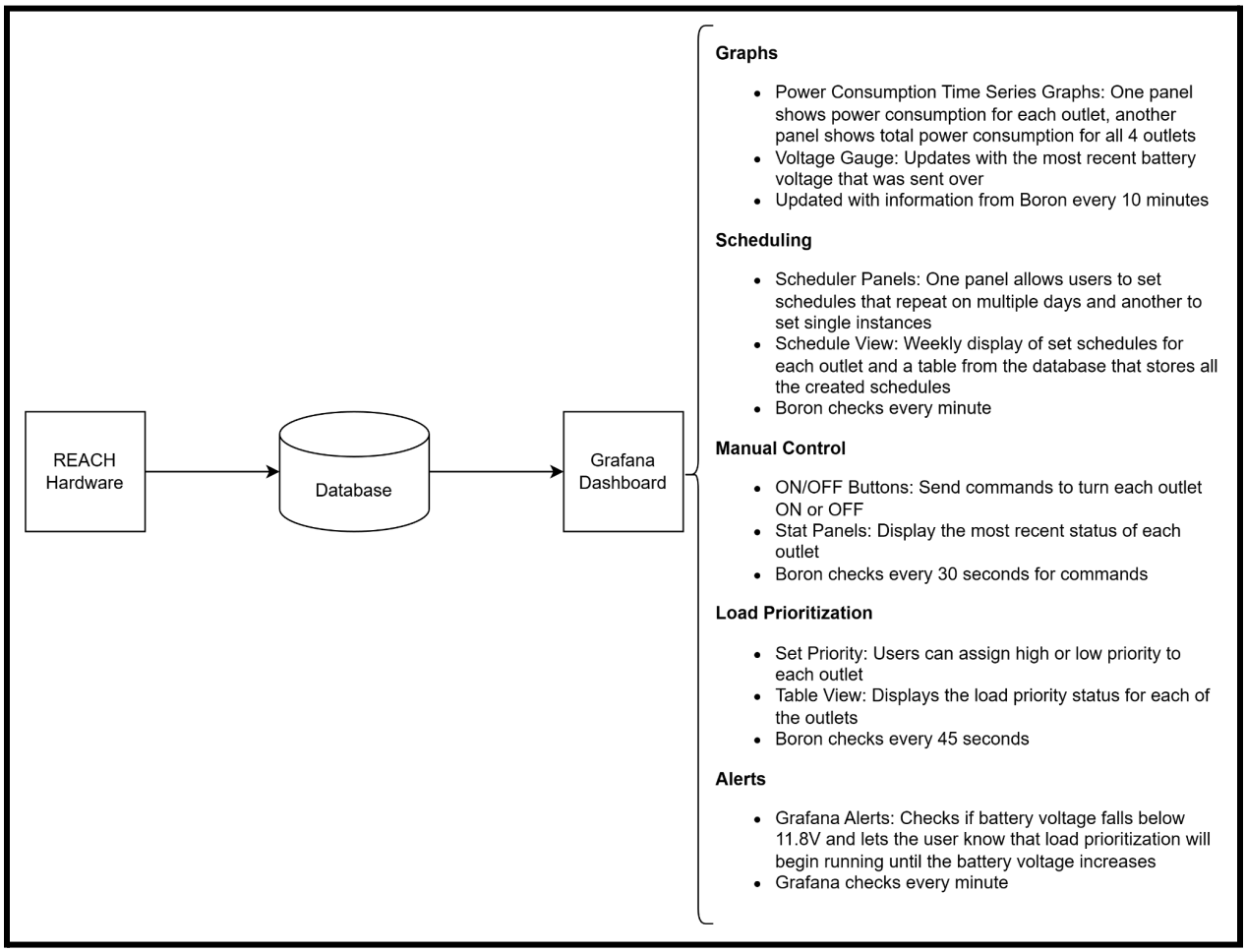

Fig. 18 Visual Explanation of Dashboard Features

Dashboard Features - Real-Time Monitoring

The user interface on Grafana contains several different visualization panels that provide different functions, ranging from real-time monitoring to scheduling. For this section, we'll go into more detail about the panels for real-time monitoring of the system. At the top of the dashboard is a gauge to visualize the voltage reading for the user's battery that's being used to power REACH. It's important to know what this value is because it determines how much power is available to the system, if it needs to be replaced, and it plays a role in determining if load prioritization features should be activated. The battery voltage is an important value, and it is periodically updated when the Boron sends the updated telemetry data every ten minutes.

Grafana has a dashboard variable feature that we took advantage of in order to clean up the dashboard interface. We set the outlet number to be a dashboard variable, and a little

dropdown at the top left corner of the dashboard allows the user to choose one outlet at a time. This affects which power consumption graph and weekly schedule is displayed on the respective panels. The dashboard variable is convenient when the user wants to specifically focus on a certain outlet, for example, if they are only operating on outlet 1. It simplifies the dashboard visuals and gives more control to the user.

Next up is the Time Series visualization panels that are being used to display the power consumption of the outlets over time. There is one panel for specific outlet power consumption and another panel for total power consumption of all the outlets over time. The one for specific outlets changes what data is being displayed based on the outlet that is chosen using the dashboard variable. The calculation for the power consumption is done based on the current and voltage readings in the telemetry table in our database. The total power consumption calculation is also done using the data in our telemetry table. These readings are valuable because it shows how much power is being consumed over time and can be helpful for long-term data analysis.

To continue to go into detail about the visualization of the telemetry data, there is a Table panel used to display the most recent data sent over from the REACH system. It shows the time the data was received, respective outlet IDs, battery voltage, sensor voltage for the outlet, current sensor readings for the outlet, and power consumption calculation. The table is an easy way to see what's going on with the circuit in real time and to make comparisons with previous readings.

There is another Table panel for the history of manual commands. This table shows users the time the command was created, outlet ID, outlet state, and the status of the command. This is useful for the user to see what commands are pending and which ones were executed at each outlet.

Another table is set up to display the outlet priority status for each outlet. There are also columns to show what time the priority status was set and the reasons why each outlet might have been turned off if they were low priority. This is helpful for storing the history of the load prioritization changes.

The final type of visualization panels that are used for real-time monitoring are the Stat panels. The dashboard currently has four stat panels for each outlet's current status, on or off. The stat panels get their status from the database table that contains data on the outlet state and are updated when the Boron passes on the new information. The outlet state can be changed with manual commands, set schedules, and load prioritization logic.

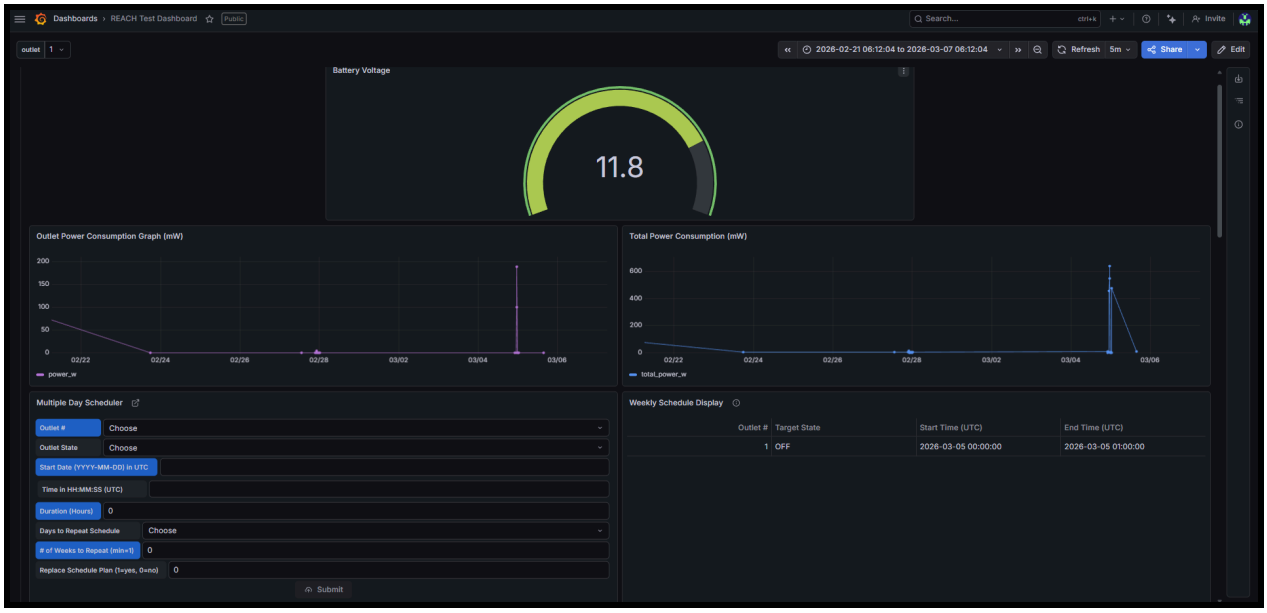

Fig. 19 Battery Voltage Gauge and Power Consumption Graphs

Dashboard Features - Manual Control Interface

In order to implement manual control of the outlets, four panels were created to host on and off buttons for each outlet. It was difficult to find a Grafana panel with buttons that could run the HTTP protocol to update the database tables from PHP files. So instead of using the Grafana button plug-in, we decided to use Text panels and HTML code to create on and off buttons that would update the commands table in our database. The Boron would then check that table, send the pending commands to the ESP, and the ESP would turn the respective outlet on or off based on the command. The Boron also sends an acknowledgement to the database so that the outlet status table updates with the correct status. The stat panels with the outlet statuses updates based on the values in the outlet state table.

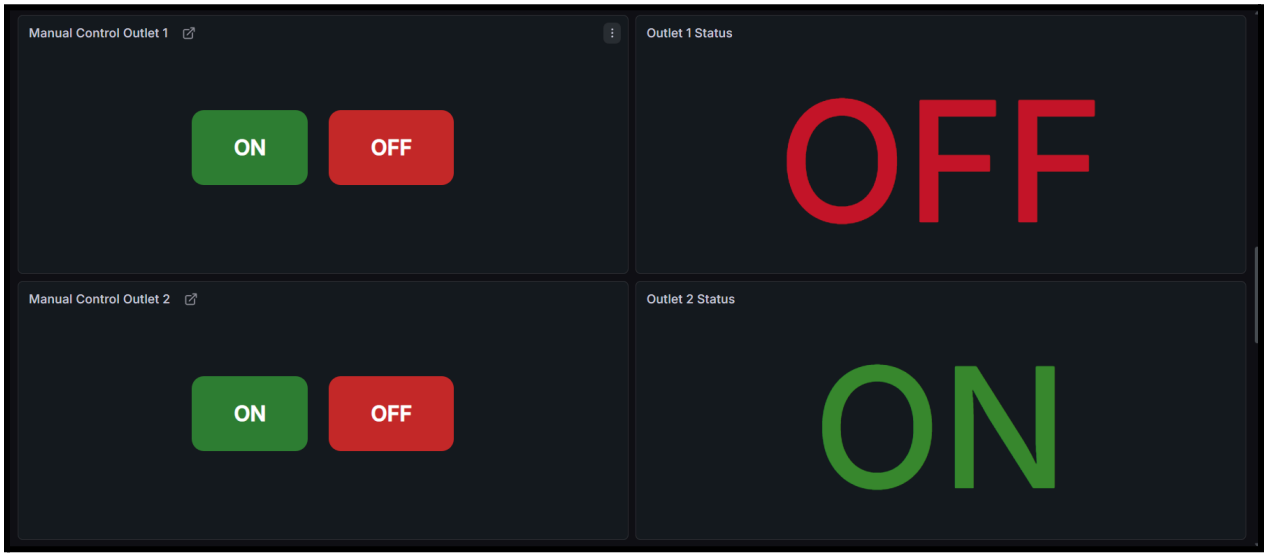

Fig. 20 Manual Control Buttons and Stat Panels with Outlet Status

Dashboard Features - Scheduling

The scheduling feature was a difficult task to figure out the logic for, and it was important that we made the user interface for setting schedules easy to understand. There are two separate Business Forms panels for the user to use to set schedules for the outlets. One is to set schedules that repeat on multiple days in a week, and the other is for schedules that occur as single instances. It's important to note that the scheduling features need time inputs in UTC because that's the timezone the database operates in, and it's simpler to have everything operate in the same timezone. The users need to input outlet ID, outlet state, start date, start time, duration in hours, days to repeat (if applicable), number of weeks to repeat, and whether they want to have the new schedule replace the previously set schedules for that particular outlet. The new schedules are sent to the schedule table in the database.

There is a Table visualization panel that displays the weekly schedule for all four outlets. It has a simple setup with the start and end times, outlet ID, and outlet state. This panel is connected to the dashboard variable, so it displays the weekly schedule for the outlet that's selected. There is another table panel that shows all the scheduled on and off times for all the outlets. It's a comprehensive list with all the outlets so that the user can see all the upcoming and past schedules in one place.

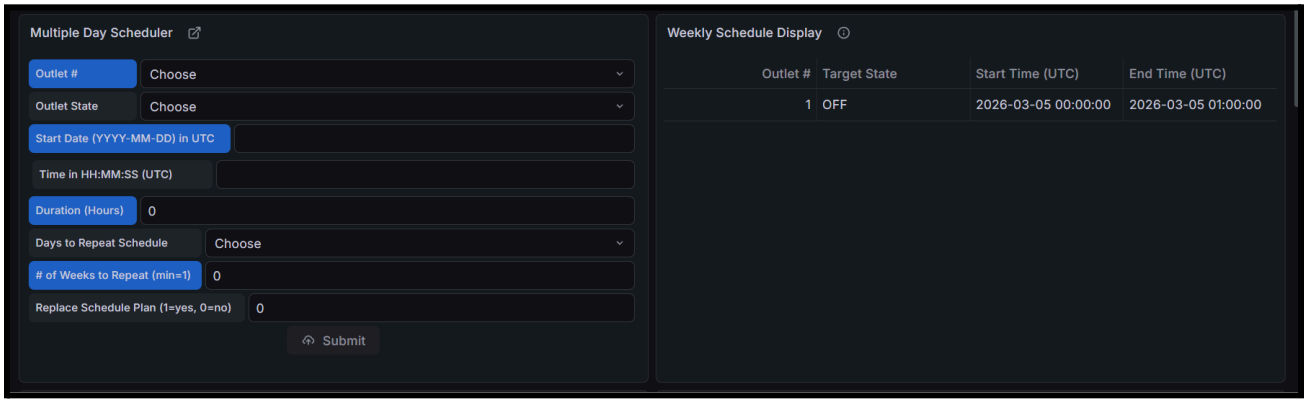

Fig. 21 Scheduling Panel and Weekly Schedule Display

Dashboard Features - Load Prioritization

Another feature that we worked to implement on the dashboard is load prioritization. This will kick in when the battery voltage of our system hits a certain threshold voltage. In order to maintain battery health and overall system operation, when the battery hits a certain threshold voltage, we will have load prioritization turned on. Outlets that are set to high priority will remain powered on, and the low priority outlets will be turned off according to the logic set by the ESP. If one of the low-priority outlets is drawing more current than the others, it will be turned off first. In the event that all the low-priority outlets are drawing similar amounts of current, they will all be turned off. Once the battery voltage surpasses the threshold voltage again, the low-priority outlets that were shut off will be turned on again.

There is a simple Business Forms panel on the dashboard where the user can select an outlet and set the priority, either high or low. The data is then reflected in a Table panel next to it so that they can see the priority status of all four outlets. There are also columns for the time that the priority was set, the time an outlet was shut off, and the reason why the outlet was turned off.

The priorities set by the user will update the outlet state table with the corresponding changes. The goal for this feature is to maximize system efficiency and optimize battery health.

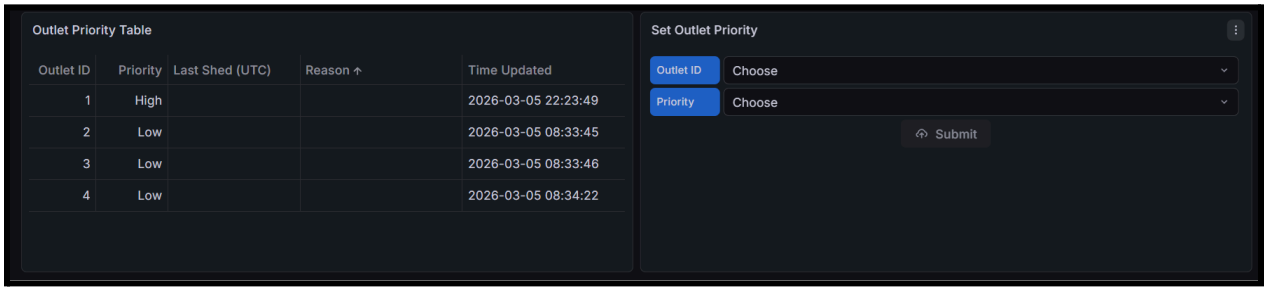

Fig. 22 Load Prioritization Panels

Dashboard Features - Alerts

In Grafana, we set up an alert that will check the battery voltage every minute. If it notices that the battery voltage falls below 11.8 V, an alert will be sent to the user letting them know about the voltage level and that load prioritization will be enabled. This is helpful to let the users know that the system is being optimized, and the low-priority outlets might get shut off based on the load prioritization algorithm.

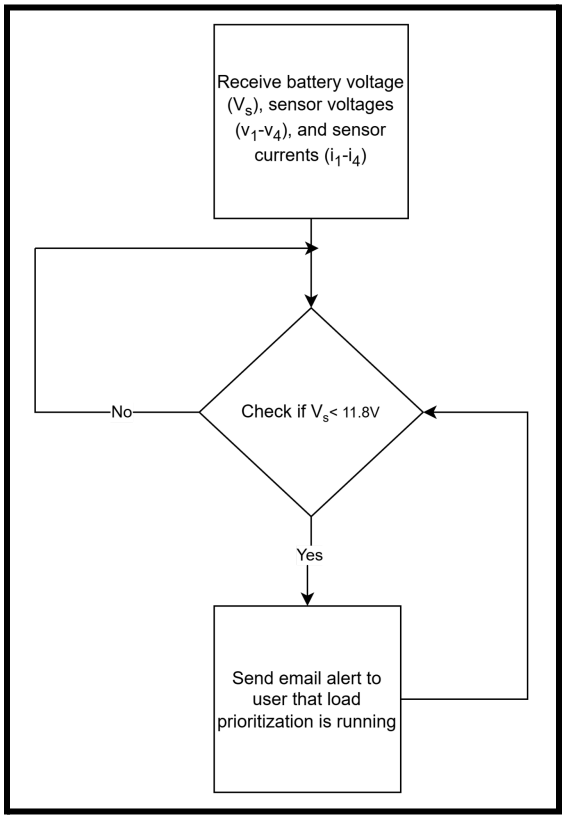

Fig. 23 Load Prioritization Alert Flow Diagram

Challenges and Future Improvements

One of the big challenges that we faced while making the dashboard was keeping the time zones consistent across all the platforms and components that we're using in the project. It's easiest to keep everything in UTC, but it can be confusing for the user to set schedules using a timezone that they aren't currently in. Sometimes we would mess up with scheduling because we'd have them start at the wrong times, and we'd wonder why the outlets weren't responding.

Another challenge was figuring out how to get the data flow working correctly. We would be able to get the connection from the database to Grafana working, but then there would be a disconnect from the Boron to the dashboard, or something else would go wrong. We learned that it was easier for the Boron to parse CSV format instead of JSON, so we had to modify the format for the PHP files.

5.1(b) Particle Boron

Particle Communication Layer

The boron has a lot of jobs to perform since it's our only way of communicating between the ESP and the dashboard. For example, one of the things the Boron does is send out the sensor readings and current outlet status. So every few minutes, the dashboard can see what the outlets are doing and how much is being drawn.

The function used in order to publish this data is:

"publishUSBTelemetry()"

This information goes to the database and is then displayed nicely on the dashboard. That's about as much interaction as the Boron has putting data into the tables.

The webhooks are mostly used to read data from these database tables, and there are a few tables of interest, such as:

"Particle.publish("serverSchedule", String(outlet), PRIVATE);"

This function and topic are used in order to retrieve the schedule from the user, the user-inputted schedule being found in our database. We have other functions that serve the same functions but for our features:

"*Particle.publish("pollCommand", payload, PRIVATE);"*
"Particle.publish("reachPrio", String(outlet), PRIVATE);"

These are used to access the Particle webhook and get the manual commands which are executed quickly after being received or the user-inputted priority value for the outlet.

UART Communication Layer

The UART communication does a lot of heavy lifting in terms of the local communication between the two microcontrollers. Setting up the communication method is easy, but what they send is invaluable,
for example:

"SCHED,<outlet>,<start>,<end>,<target>,<enabled>"

Where every bracket outline details parameters of the user-inputted schedule that was retrieved and must be executed by the ESP. The same could be said for these other strings:

"USB1=ON"
"USB3=OFF"

"PRIORITY,2,LOW"
"PRIORITY,4,HIGH"

"TIME=<epoch>"

In these examples, "*USB1=ON"* is sent by manual commands, Priority sets priority for the outlet, and time is sent to the ESP so that it knows when to execute a schedule.

What the ESP sends to the Boron is:

"USB1,STATE=ON,CURRENT=0.123,VOLTAGE=5.01,SUPPLY=12.3,EPOCH=1736238123"

"OVERRIDE=1"
"OVERRIDE_TARGET=0"
"OVERRIDE_UNTIL=1736240000"

"schedAck = true"

This single line (USB1, STATE….) gives the Boron everything it needs to update the e-ink display and upload telemetry to the cloud: outlet state, current draw, voltage, battery level, and the ESP's current time. The override lines tell the Boron whether an override is active, what the forced state is, and when it expires, while schedAck confirms that it received and stored the latest schedule.

E-Ink Display Rendering

Fig. 24 E-Ink Display with system and switch information

Each line follows this format:

"swX: <voltage>V <current>mA <state>"

where:
"Voltage (V)": the measured output voltage of that USB port
"Current (mA)": how much current the connected device is drawing
"State (0 or 1)": whether the outlet is OFF (0) or ON (1)

On the right side of the display, there are two additional fields:

"VS": the system's supply voltage (battery voltage),
"ACK": whether the ESP successfully received the latest schedule

Altogether, we get the local display and a simple way to read important values about our system.

System Timing

The REACH system relies heavily on precise timing intervals to keep everything synchronized between the Boron, the ESP8266, and the cloud. Instead of using delays or blocking code, the Boron uses non blocking timers based on millis(), which lets it handle multiple tasks in parallel without freezing the system. Each task in the firmware runs on its own schedule, and the timing intervals were chosen to balance responsiveness, power consumption, and network usage.

The timing intervals for each of the user-inputted values are as follows:

Time Sync to ESP (every 1 second):
The Boron sends the current epoch time to the ESP so it can maintain an accurate internal clock.

Schedule Polling (every 60 seconds):
The Boron checks the cloud for updated schedules once per minute.

Command Polling (every 30 seconds):
Manual ON/OFF commands are checked more frequently, so the system feels responsive when the user presses a button on the dashboard.

Priority Polling (every 45 seconds):
Priority updates are polled on a staggered interval so they don't line up with the other tasks. This reduces the chance of network congestion and keeps the system responsive.

Telemetry Upload (every 10 minutes):
The Boron uploads sensor data to the cloud at a slower interval to save bandwidth and power. Telemetry doesn't need to be real-time, so 10 minutes is a good balance.

E-Ink Display Refresh (minimum 5 second spacing):
The display only refreshes when new data arrives, and even then, it enforces a minimum interval to avoid unnecessary power usage and extend the life of the e-paper panel.

All of these timers run independently, which means the Boron can handle cloud communication, UART messages, display updates, and button presses without blocking or slowing down.

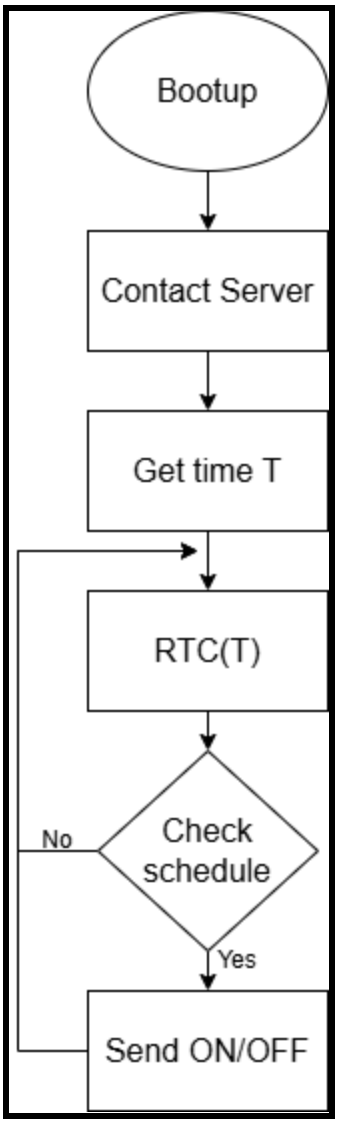

Fig. 25 Software Flowchart for Scheduling

5.1(c) ESP8266 12-E NodeMCU

Overview of the ESP8266

The ESP8266 12-E NodeMCU acts as a local execution engine for all of the USB outlets, with the capability to execute schedules, priority rules, and manual commands received by the Boron, as well as executing manual commands from a local page. The module is able to measure voltage, current, and supply level for each outlet and package that data in the correct order to send to the Boron via UART.

UART Communication Layer

The ESP receives structured commands that drive all of the outlets' behavior.

Incoming Commands from the Boron:

The ESP8266 parses multiple command formats, each mapped to a specific subsystem.

Time Synchronization:
TIME=<epoch> Updates usbRef.nowEpoch, which drives schedule execution and override expiration.

Priority Commands:
Supports three formats, all handled in parseLine():
PRIORITY,<outlet>,<priority>
Example: PRIORITY,2,1

- Sets priority for a single outlet.

PRIO=1010

- Compact bitmap for all outlets.

PRIO,1,0,1,0

- CSV list of priorities.

All formats call usbRef.setPriority().

Schedule Commands:
SCHED,<outlet>,<start>,<end>,<target>,<enabled>

The ESP:

- Parses all five fields
- Clears any active override for that outlet
- Stores the schedule via usbRef.setSchedule()

Manual Commands:
USB1=ON or USB3=OFF

The ESP:

- Activates an override for 24 hours (nowEpoch + 86400)
- Immediately applies the state via usbRef.applyManual()

Outgoing Messages to the Boron

Compact Snapshot Line:

Before sending per-outlet telemetry, the ESP sends:

PRIO=1010,EOFF=0000,EPOCH=1736238123

This gives the Boron:

- Priority bitmap
- Emergency-off bitmap
- Current epoch time

Useful for quick display updates.

Full Telemetry Packet (Per Outlet):
Each outlet sends a detailed report as follows:
USB1,
STATE=1,
PRIO=1,
EOFF=0,
SCHED_EN=1,
SCHED_START=1700000000,
SCHED_END=1700003600,
SCHED_TARGET=1,
OVERRIDE=0,
OVERRIDE_TARGET=0,
OVERRIDE_UNTIL=0,
CURRENT=0.123,
VOLTAGE=5.01,
SUPPLY=12.30,
EPOCH=1736238123,
RUNTIME=50.4

Schedule Execution Engine

The ESP8266 maintains a schedule table for each outlet, storing the start time, end time, target state, and enabled flag. It uses the internal epoch clock to determine when a schedule should activate or deactivate an outlet. If an override is active, the ESP temporarily suspends scheduled execution for that outlet until the override expires. Once the override ends, the ESP automatically resumes normal schedule behavior.

Priority and Emergency - Off System

The ESP8266 manages outlet priority using values received from the Boron, allowing the system to resolve conflicts when power is limited or when multiple outlets request activation at the same time. It also tracks emergency-off states, which immediately disable an outlet regardless of schedules or overrides. Both priority and emergency-off values are included in telemetry so the Boron can display them and make higher-level decisions.

Sensor Measurement Layer

The ESP8266 measures the current draw and output voltage for each USB outlet, as well as the system's supply voltage. These measurements are formatted to three decimal places and included in the telemetry packets sent to the Boron. The ESP may apply filtering or smoothing to ensure stable readings and detect abnormal conditions such as overcurrent or undervoltage.

Outlet Control Layer

The ESP8266 controls each outlet through functions that apply manual commands, execute schedules, and enforce priority rules. Manual overrides take precedence over schedules, ensuring that user commands are applied immediately. Emergency-off states override all other logic and force the outlet off. The ESP updates internal state arrays to reflect the current behavior of each outlet and includes this information in telemetry.

5.2 Hardware Documentation

5.2(a) Particle Boron Microcontroller

Particle Boron LTE Module

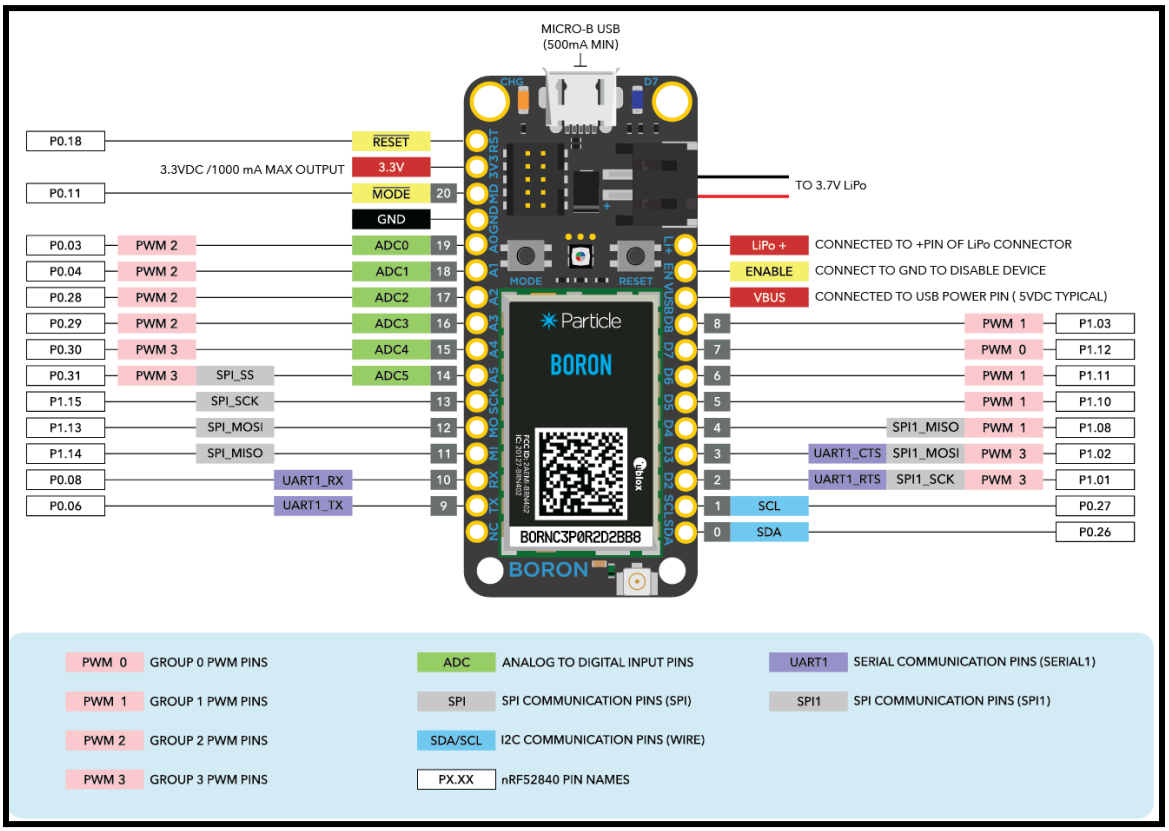

Fig. 26 Particle Boron 404x Pinout Diagram

The Boron is the brain of the system that processes the data imputed by the user and gives feedback to the user. The Boron was selected for this project for a few different reasons and the main one is the LTE connectivity the Particle provides to the system. It does this using a wideband LTE-CAT M1 cell antenna. This allows us to communicate with the device even when they are in places that don't have wifi connections, like a vineyard. It also has an onboard RTC that automatically syncs up on boot up, allowing our system to have an accurate time being kept on the system. This is useful because the system being able to keep track of accurate time means it can execute schedules based on our standard of time which in this case is UTC. This can also ensure that software flow can continue without too many issues, relying on a standard

microcontroller timer. The final feature I would like to mention is the ability to connect the device to a single-cell LiPo 3.7 V Battery. This microcontroller was made specifically for IoT applications, which is why it perfectly fits into our system as our gateway.

E-Ink Display

Fig. 27 Dimensions of the Waveshare E-INK display module

The E-Ink display is how we are implementing the display outlined in Engineering Requirement 4. It was chosen because the display consumes very little power, and since we didn't plan on having a fast display on the system, we could use it to show all the specified information stated in our requirements. This information includes outlet status, voltage on the line, current on the line, and a voltage reading of the main power supply, in our case, a 12V battery.

UART Link to ESP8266

For REACH, the Boron and ESP8266 talk to each other over a simple UART serial link. This is the internal communication backbone of the whole system. We chose UART because it’s simple, doesn’t require any fancy timing, and both microcontrollers support it natively. It’s also low power, which matters since the whole system is running off a battery, and it’s extremely easy to debug. You can open a serial monitor and watch the messages stream by, and it's not hard to wire since the Boron and ESP sit right next to each other.

5.2(b) ESP Microcontroller

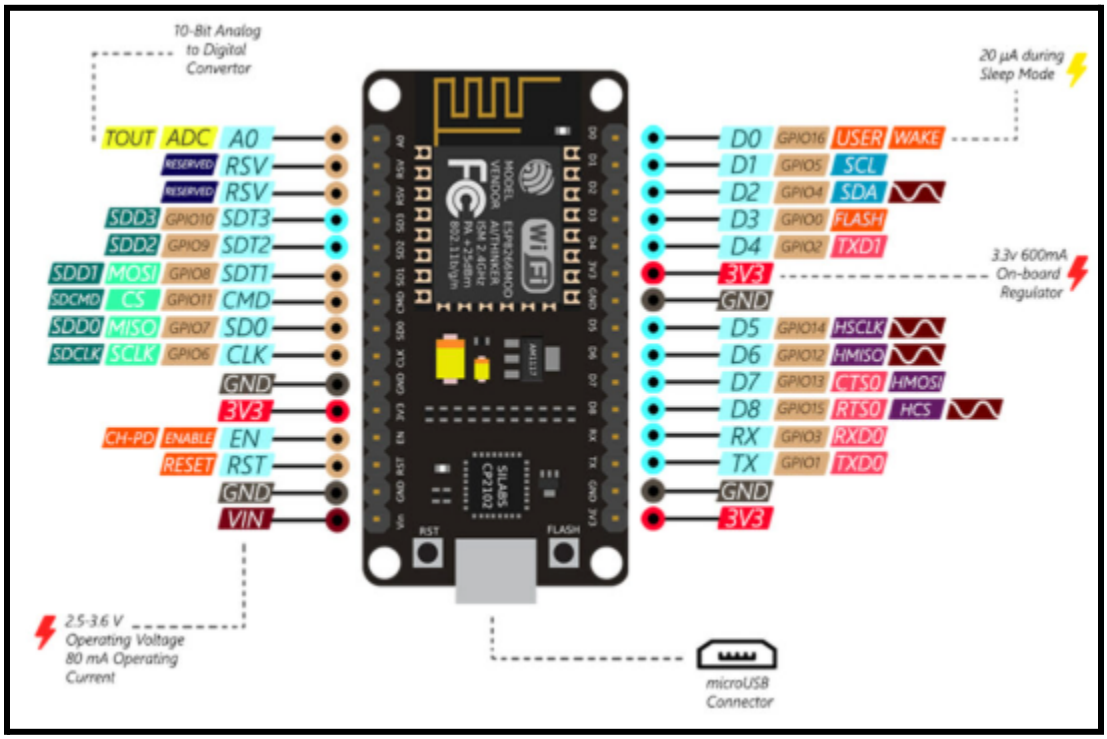

Fig. 28 ESP8266 Module

ESP8266 12-E NodeMCU Kit

The ESP8266 is the microcontroller responsible for the control of the outlets and the reading of sensor data, acting as the body of the system, reacting to the command line of the Boron module, which is connected via UART. The module consists of 30 pins, 17 of which are general-purpose input/output ports (GPIO), SPI protocol, I2C capabilities, UART, I2S interfaces with DMA, and a 10-bit analog-to-digital converter (ADC). The module uses the I2C protocol, which is a master-slave protocol, to read data from the INA260 current sensors, which pulls measurements off of the outlet lines in the form of a bus. On this bus is also the ADS1115, which is a multiplexing module in order to receive various analog signals, due to the voltage sensors being analog sensors. Apart from this, the ESP module controls each Dual MOSFET Switch module by its designated GPIO. The ADC runs on a 10-bit resolution, which is rated for 3.3V. This means the resolution is 3.3mV, which allows the ESP to accurately read analog measurements provided to that pin. This did pose a challenge, as the ESP module only has one ADC, which was not sufficient for this system. The solution ended up being the ADS module, which allowed the use of the I2C bus.

The I2C bus operates as follows: The protocol calls for a master-slave protocol, where there is a system data line and a system clock line. The system data line is for the collection of measured or calculated data. In this case, the ESP8266 polls data from the sensors and modules that operate on the bus. The system clock line is used to time the different modules on the bus itself to not collect bit errors from the I2C modules and confuse data by bit collision. The modules are addressed in order to keep an order of the I2C bus data, which is read on the ESP8266 module to poll and process.

Local Webpage

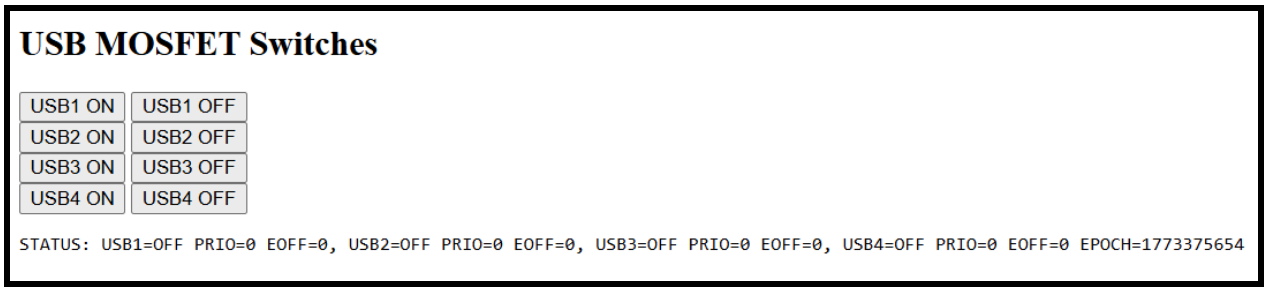

Fig. 29 Local Webpage for Outlet Control

The local webpage provided by the ESP module is for local control of the REACH box. Enveloped in the ESP onboard WiFi is the local PHP page, which allows the user to control the outlets with a touch of a button. While the connection protocol is considered WiFi, the protocol does not allow for access to the general internet.

5.2(c) Input and Output

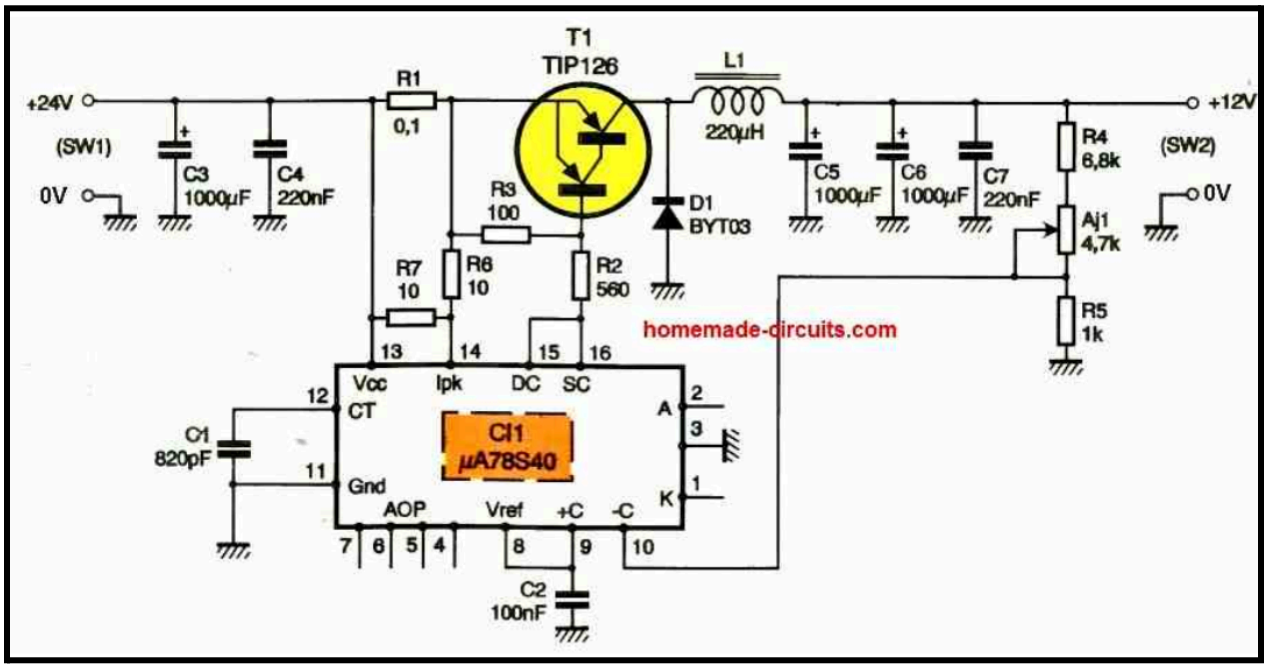

Fig. 30 Klunoxj DC 12V/24V Buck Converter

Klunoxj DC 12V/24V Buck Converter

The buck converter steps down the battery voltage from 12V to 5V, which was the original intended battery voltage; however, it has the capability to step down to 24V, a larger range than needed. The converter offers circuit protection on the output, so there are no surges. The maximum amount of current the module is able to provide is 5A, and it is rated for 25W.

Because there is no documentation provided, below is the following efficiency. This data is to show how effective and efficient the conversion of power is to be applied to our system.

Vin	Current in	Vout	Current out	Power in	Power out	Efficiency
11.800	0.030	5.200	0.053	0.354	0.274	0.774

11.900	0.030	5.200	0.053	0.357	0.274	0.768
12.000	0.030	5.200	0.053	0.360	0.275	0.763
12.100	0.030	5.200	0.053	0.363	0.275	0.756
12.200	0.030	5.200	0.053	0.366	0.275	0.750
12.300	0.030	5.200	0.053	0.369	0.275	0.744
12.400	0.030	5.200	0.053	0.372	0.275	0.738
12.500	0.030	5.200	0.053	0.375	0.275	0.734
12.600	0.030	5.200	0.053	0.378	0.275	0.728
12.700	0.030	5.200	0.053	0.381	0.275	0.722
12.800	0.030	5.200	0.053	0.384	0.275	0.716
12.900	0.030	5.200	0.053	0.387	0.275	0.711
13.000	0.030	5.200	0.053	0.390	0.275	0.705
13.100	0.030	5.200	0.053	0.393	0.275	0.700
13.200	0.030	5.200	0.053	0.396	0.275	0.695
13.300	0.030	5.200	0.053	0.399	0.275	0.688
13.400	0.030	5.200	0.053	0.402	0.275	0.683
13.500	0.030	5.200	0.053	0.405	0.275	0.678

The anticipated operating voltage of REACH is from 11.8V - 13.5V, assuming it is connected to a battery setup. The output voltage of the buck converter is stable, as is the output current of the system, however, much of the provided power is dissipated from heat, which is something that will need to be considered. The efficiency of the system was measured using a bench power supply (VOLTEQ HY3005D) that has the capability to measure the input current as well as provide power, an Agilent multimeter (34405A), the buck converter, a 100 ohm 50W resistor, and a multimeter to measure the output voltage. Above are the measurements calculating the efficiency using the formula:

$$\eta = \frac{V_{out} I_{out}}{V_{in} I_{in}}$$

USB Outlets

The outlets that provide a connection to each load are in the form of USB-A female connectors. This format was chosen because of the simplicity of integration to provide a load, as well as how common devices are able to be powered by the connection.

5.2(d) Sensors

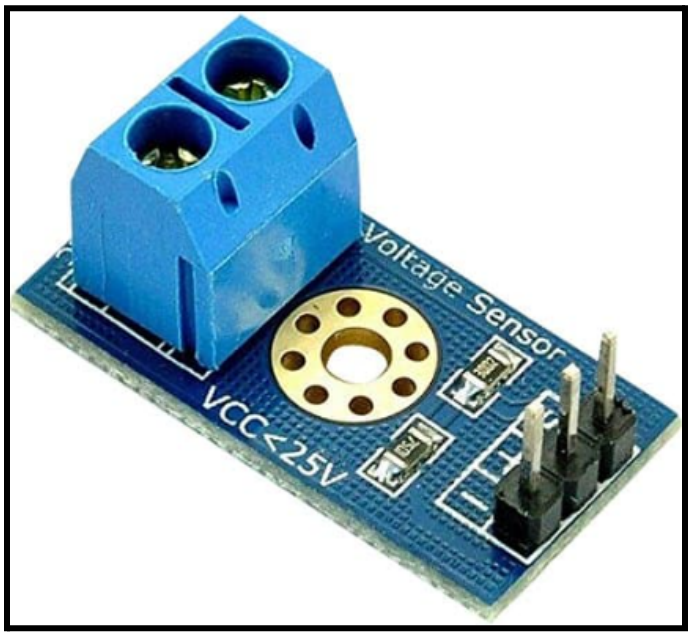

Fig. 31 DCT- Electronic Voltage Sensor Module

DCT- Electronic Voltage Sensor

The DCT- Electronic Voltage sensor uses a voltage divider to scale down higher voltages in order to receive measurements at a safer value. These measurements are received by the ESP module as digital values through most of the application by the ADS1115; however, they are emitted as analog values, as this module is an analog sensor.

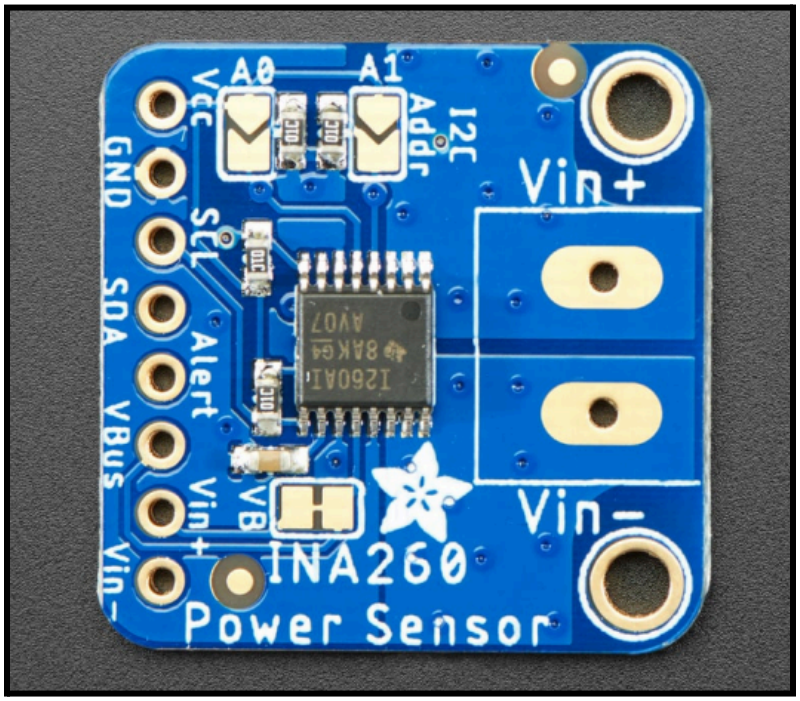

Fig. 32 Adafruit INA260 Module

Adafruit INA260 Current Sensor

The Adafruit INA260 Current Sensor uses an internal shunt resistor that is measured with a 16-bit resolution, allowing for measurements as small as 1.5 mA. These modules measure current in series with the USB outlets in order to accurately measure their current and power output. The module has a voltage measuring capability; however, it is not implemented in this system because of the difficulty and inaccuracy of each measurement.

5.2(e) Additional Modules

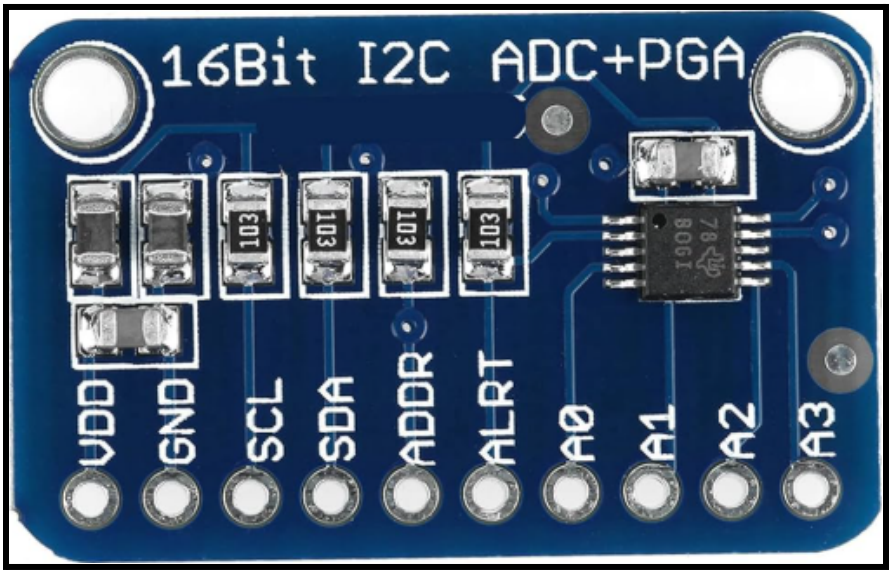

Fig. 33 Adafruit 16-Bit Analog-to-Digital Converter Module

Adafruit 16-Bit Analog-to-Digital Converter

The low-cost module was used to give the ESP more ADC pins for the purpose of reading voltage across each line. Where the voltage supply is being read by the onboard ADC of the ESP module at 10 bits, the ADS1115 has the ability to receive analog signals at a resolution of 16 bits, which allows for accurate and precise measurement.

Section 6: Engineering Ethics

6.1 Ethics of the Engineering Profession and Our Project

The REACH system was designed with ethical engineering practices in mind, mainly focusing on reliability, sustainability, and usefulness in remote environments. Ethical design means creating a system that is safe, dependable, and beneficial to the people using it. In our case, REACH was designed to help users remotely manage and monitor important field devices in places like vineyards and farms where connectivity and access to power can be difficult. By giving users remote scheduling, telemetry monitoring, and load prioritization, the system helps reduce downtime and unnecessary maintenance trips while improving the reliability of remote equipment. Sustainability was also something we considered while designing REACH. The system was intentionally designed to work with existing solar power systems instead of requiring users to completely replace their current setup. The use of low-power hardware like the E-Ink display and efficient microcontrollers also helps reduce overall system power consumption. Another important part of the project is the modular design approach. Components like the Particle Boron, ESP8266, sensors, and display can all be individually replaced if they fail, which helps reduce electronic waste because the whole system does not need to be thrown away. However, we are not currently using biodegradable materials in the enclosure or components. In future revisions of the project, we could improve sustainability further by using recyclable materials and making the enclosure easier to reuse or repair. The REACH system is useful because it helps improve the management of remote systems that people rely on every day. Devices like irrigation controllers, frost alarms, and environmental sensors are important for agricultural operations, and losing access to them can cause serious problems. REACH helps reduce those risks by allowing users to remotely monitor system conditions and manage power usage more efficiently. It can also reduce the number of maintenance trips required to visit remote sites, which helps lower fuel usage and emissions.We also thought about accessibility and ease of use during the design process. The dashboard and E-Ink display were designed to present information in a simple and readable way using large text, clear outlet indicators, and straightforward controls. This makes the system easier to understand for users who may not have a strong technical background. Even so, there are still improvements we could make in the future, such as adding larger scalable text options, voice-assisted controls, or support for users with visual impairments. Finally, we considered whether there could be any harmful aspects to the project. The system uses LTE and WiFi communication, but both operate at low power levels that are commonly used in consumer electronics and are not intended to create harmful RF exposure. Electrical safety was also considered through the use of fuses, current sensing, and voltage monitoring throughout the system. While the current system operates safely during testing, additional long-term environmental testing would help further improve confidence in the durability and safety of the system in harsh outdoor conditions.

Section 7: Conclusion

7.1 Conclusion

Overall, REACH was designed to be an external module with a user-friendly dashboard interface to address the challenges faced in remote locations involving power management, remote device monitoring, and reliable connectivity. Throughout this project, we successfully developed the system to be capable of remotely controlling and monitoring the connected low-power field devices for which they were intended. This was achieved through the modular PCB circuit, LTE communication, and key features on our dashboard, including scheduling, load prioritization, and manual commands. We conducted several iterations of testing to make sure that our system was reliable and operating within the parameters we established with the marketing and engineering requirements. This was to make sure that our system was power-efficient, reliable, and compatible with existing solar-powered systems that customers might have. We encountered our fair share of challenges and difficulties that involved multiple rounds of troubleshooting, attention to detail, and perseverance to solve problems with the hardware and software sides of the project. In conclusion, the project was able to successfully meet the major engineering and marketing requirements that were established and we were able to demonstrate functionality. There is always room for improvement and advancements with further developments, but at the end of the day, we gained a lot of technical, presentation, and professional skills throughout this project process.

References

[1] OutBack Power, "OpticsRE: Monitoring and System Control." https://old.outbackpower.com/downloads/documents/system_management/optics_re/opticsre_manual.pdf

[2] Victron Energy, "Victron GX product range." https://www.victronenergy.com/live/venus-os%3Astart

[3] Victron Energy, "VRM Portal – Dashboard." https://www.victronenergy.com/live/vrm_portal%3Adashboard

[4] Morningstar Corporation, "Solar Charge Controllers & Inverters | Morningstar Off-grid Solar." https://www.morningstarcorp.com/

[5] K. Pitchaimuthu and B. Sridhar, "An IoT Based Smart Solar Photovoltaic Remote Monitoring and Control Unit." https://www.researchgate.net/publication/295611696_An_IoT_Based_Smart_Solar_Photovoltaic_Remote_Monitoring_and_Control_unit

[6] A. Hamied et al., "IoT-Based Low-Cost Photovoltaic Monitoring for a Greenhouse Farm," *Energies*, 2023. https://www.mdpi.com/1996-1073/16/9/3860

[7] "IoT Based Solar System Monitoring and Load Management for Small Farm," 2022. https://www.researchgate.net/publication/363426028_IoT_Based_Solar_System_Monitoring_and_Load_Management_for_Small_Farm

[8] A. Burgio et al., "An IoT-Based Solution for Monitoring and Controlling Battery Storage Systems," *Energies*, 2023. https://www.mdpi.com/1996-1073/16/7/3140

[9] Particle, "Particle Boron: 2G/3G or LTE Cat M1 + Bluetooth." https://docs.particle.io/boron/