Senior Design 1: Final Report

Group 24
M.A.P.S. (Multi Adjustable Port Station)

Report Authors:

Hunter Volonino	*Computer Engineering*
Sebastian Sotomayor	*Computer Engineering*
Siya Sharma	*Electrical Engineering*
Jonathan Luna	*Electrical Engineering*

Review Committee:

Dr. Avra Kundu	*ECE Department*
Dr. Sonali Das	*ECE Department*
Dr. Nazanin Rahnavard	*ECE Department*
Dr. Azadeh Vosoughi	*ECE Department*

Advisor:

Dr. Saleem Sahawneh

Table of Contents

List of Figures

List of Tables

Chapter 1: Executive Summary

In the age of technology, the amount of portable electronic devices has grown significantly and changed over time. Portable devices will generally have some type of lithium-ion battery, which may have specific charging characteristics. When using any portable device continuously for a long period of time, any battery it has will most likely wear, resulting in decreased battery capacity and increased risk of failure. If the portable device is not used for a long period of time, the battery will lose charge, and will eventually also cause additional wear or failure to the battery. Any possible way to decrease the rate of wear on a battery would be very helpful.

This issue, with aging devices, is also compounded by decreased parts availability with time. Older devices may not receive the same parts availability and support as newer devices. For example, various Apple devices do not receive parts availability after anywhere from 5 to 10 years after the release date [1]. Third-party and used parts do also exist, but as time passes, the supply and availability of these replacement parts continues to decrease. This is particularly important for batteries, which have a finite lifespan. Thus, it is important to ensure the best chance of a longer useful lifespan for portable devices that may work fine as-is or have no direct replacement.

Our project is intended to alleviate this problem as much as realistically possible, for a wide variety of devices. We will cover the background, research, design, development, and implementation of the project in this report, as well as additional information such as work distribution, expenses, and project timing. The content of this report starts with our plan, then will detail our extensive research into various different technologies to assist us with development. Standards and constraints will also be discussed, as well as the final parts and technologies we will select. Finally, our design and implementation will be detailed for both the hardware and software components of the project, as well as our description of testing.

Chapter 2: Project Description

2.1 Motivation and Background

Device hobbyists that collect various types of electronics exist to use and enjoy different portable electronics from different points in time. In particular, any device hobbyist with a large amount of mobile or portable devices must keep them charged if they desire to use them. Most portable electronic devices contain rechargeable lithium-ion batteries, which carry a set of risks in use. However, keeping a device plugged in all of the time while it is unused can degrade the battery and even cause it to expand. Charging a battery to 100 percent of its capacity also can eventually cause accelerated wear over time, and keeping a battery in a low state of charge is perhaps even more detrimental [2].

Our senior design project is designed for such hobbyists with various different devices that they use on a semi-regular basis. A device may be desired every few days, or every few weeks. To use the devices, they must be charged to a usable capacity, but not kept charging to the point where excessive battery wear happens.

Another important consideration is that, as this project is designed for various different types of devices, the battery's capacity cannot be read in a direct and standardized manner over USB or any other charging medium. Thus, the only things that can be directly read are voltage, current, and power draw while charging. It is up to the user to configure the charger based on that data, so the charger can be optimized to the user's preferences.

2.2 Goals and Objectives

2.2.1 Goals

Our basic goals consist of:

- Create a device charger that takes in a DC input
- Distribute charging current to multiple portable devices over USB.
- Be able to turn current to each device on and off individually and with automation.
- Inform the user of the current charging statistics.
- Charging portable devices safely based on their profile (current/voltage/power)

Our advance goals consist of:

- Ability for the user to optimize their charging for the least possible impact on battery health.
- Ability for the charger to immediately start when a device is connected.
- Show charging statistics using a Web-accessible interface

Our stretch goals consist of:

- Be able to set a current limit for each individual device
- Be able to estimate the device's battery capacity based on charging current trends, and adjust charging current based on that
- Build a fully custom enclosure for the device charger

2.2.2 Objectives

Our basic objectives consist of:

- Integrating the device with a DC power source as input.
- Display whether a device's USB charging port is on or off.
- Read the voltage and current at each USB charging port
- Display the voltage, current, and power going to each device in real time.
- Once the charging current for an individual device drops below a user-set threshold, shut off current to the device.

Our advance objectives consist of:

- Create safety thresholds that will automatically shut off current to the device if overvoltage or excessive current is detected.
- Create a user interface where the user can set a minimum allowable current, and the time to wait until charging starts again.
- Be able to detect if a device is plugged in or not.
- If a device is plugged in, immediately start charging it until it hits the user-set threshold.
- Web-based interface monitoring the charging status of the multiple devices alerting notifications in reference to if fully charged, charging conditions, etcetera.

Our stretch objectives consist of:

- Design and 3d print enclosures for both the charger and its control unit.
- Analyze the voltage, current, and power data collected to determine automatically how to regulate charging.
- Create circuitry to limit the current to a user-specified value.

2.3 Description of Features/Functionalities

The primary function of our project is to improve and automate how users manage the devices they are charging. The charger will take a 12V DC input and distribute it across four different outputs. Each output will supply 5V through a USB-A port, allowing multiple devices to charge at the same time.

One of the key features of our project will be a user interface. The interface will display information about each of the devices being charged, such as the voltage, current, and power draw. The user can set the minimum allowable current draw per device and set each device's time in between charges. When a device drops below the minimum allowable current draw (indicating a battery with a high state of charge), the charger stops charging the device. If the device is then unplugged and plugged back in, it will start charging again until the current once again drops below the user-set threshold.

Another feature of our project will be internet connectivity. Through the interface, the user will be able to connect to Wi-Fi. Once connected, the charger will transmit statistics for all devices to a web interface. This web interface allows for direct interaction between the user and the device. The user will then be able to monitor the devices that are being charged, track their charging status, and ultimately optimize battery health and performance by setting the minimum current.

2.4 Existing Products

2.4.1 Chargie

On the current market, few companies are producing an external method for battery management. One of these companies: Chargie by Lightly Electronics are offering a charging regulator that allows users to set max battery to charge, provides warnings of excess temperature, and allows you to set custom deadlines (i.e. Tell the app to charge your phone up to 70% by 3:30 P.M.). While this device is great, there are a few shortcomings. The device being an external piece of equipment and not integrated into the charging brick, it is more likely to be lost as well. Furthermore, since the device has one input, one output this results in having to purchase three Chargie's to ensure that three devices get charged. Lastly the Chargie doesn't have the option to switch back to fast/regular charging. This can lead to a situation where you need your phone to charge quickly but are not near it so you are unable to adjust the rate of charge [3][4].

To bypass needing the app, The Chargie uses a proprietary program that measures power and use that measurement to cut off charging [5], While this feature does bypass the need to use the app which is great to devices such as wireless headphones that cannot simply install an app, the feature is barebones and can be confusing for people who are not tech savvy.

2.4.2 Smart Outlets

Another potentially comparable example is smart plugs and relays. Various different brands of smart outlets exist, but they are all typically controlled by a computer or mobile device. The relays can be turned on and off based on set parameters in a mobile app or configuration server. The outlets can measure power draw. This could be integrated and automated with a solution such as Home Assistant [6]. However, this is more of a general-purpose solution and is not designed for device charging. Additionally, for our purpose, this limits the charging ability of one AC outlet to only one device. For example, to charge four devices, four AC outlets would be needed.

For individuals who need to charge multiple devices. Some products on the market are the Anker 735 Charger which allows for fast charging of up to 3 devices [7] while this does allow for less overall time required to charge the device, it lacks the features on previously discussed products where it doesn't allow the user to set limits on the charging, This can introduce more stress on the battery which can reduce overall battery capacity on the device.

2.4.3 eMpower

As for past senior design projects, there has been one previously done with some similarities to ours. During Fall 2012 and Spring 2013, one group created a device charger called eMpower. It contained many USB ports, and was intended to be powered by a solar panel. The USB ports were intended to be turned on and off by the amount of energy consumed over time by a device connected. Energy credits were intended to be monetized, and the USB ports' status depended on the amount of energy remaining from the user's quota. The master unit was intended to interface with multiple sub-units that also included USB ports [8]. By contrast, our project does not intend to monetize the charging ports, and instead turns on and off USB ports based on the voltage, current, and power provided to each port. It is also only intended to work as a single unit; users who wish to charge more devices could simply use multiple chargers for that purpose. Overall, our project is designed more for the device hobbyist rather than an institution.

2.5 Engineering Specifications Table

The below table (Table 2.5) illustrates all of the requirements we have constrained our project to. Our project's success depends highly on these requirements, and we must pay close attention to them. Any extremely imperative parameters we will be tested on are highlighted in the table.

Table 2.5: Requirements Specification

Parameter	Description	Value
USB charging port voltage regulation	The charger must output an acceptable voltage to charge devices over USB.	5V ± 0.25V

Parameter	Description	Value
Amount of USB charging ports	The multi-device charger must be able to charge a reasonable amount of devices at a time.	At least 4
Power Supply Input Voltage Regulation	The input voltage to the charger must be sufficiently regulated.	12V ± 0.5V
Maximum Output Current per Port	The charger must output an acceptable amount of current to charge devices over USB.	750mA ± 250mA
Minimum Charging Current Range (User-Set)	Range over which the user can set the minimum allowable charging current.	0mA - 1000mA
Charging Delay (Initial Plug-in)	When a device is plugged in, the charger must respond to the input quickly to start charging the device.	<5 seconds
Charge Cut-off delay	When a device drops below its set minimum charging power, the charger must respond reasonably quickly.	< 1 minute
Size (Charger)	The charger must be small enough that it could sit on a floor, table, shelf, or under a table.	< 40cm x 40cm x 40cm
Size (Control Module)	The control module must be reasonably small such that it can sit on a desk or table without using excessive space.	< 30cm x 20cm x 10cm
Weight (Combined)	The charger must not be excessively heavy.	< 5 lbs

2.6 Hardware Block Diagram

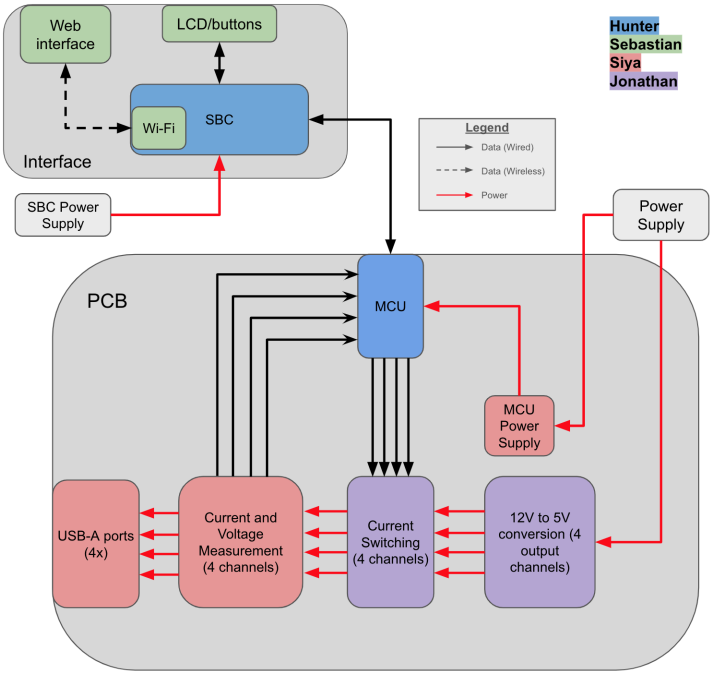


Figure 2.6: Hardware Block Diagram

2.7 Software Block Diagram

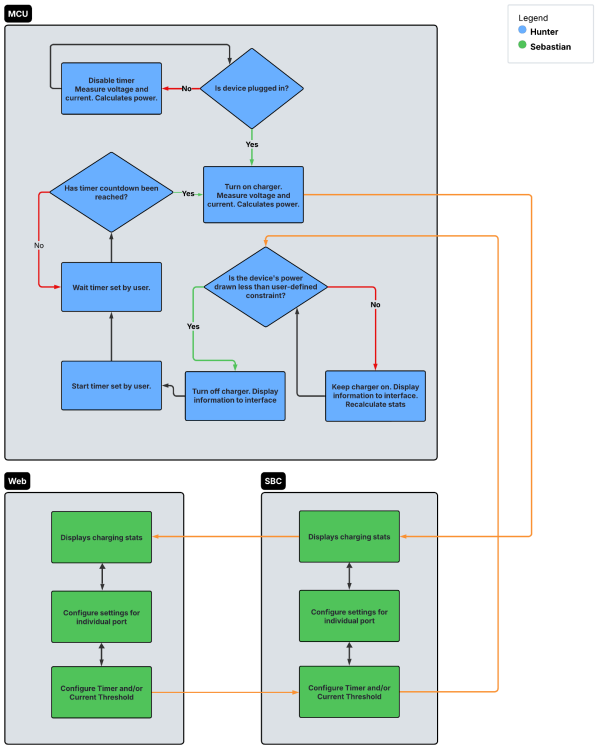


Figure 2.7: Software Block Diagram

2.8 Prototype Illustration

The Smart Device Charger can be seen below as the following illustrations.

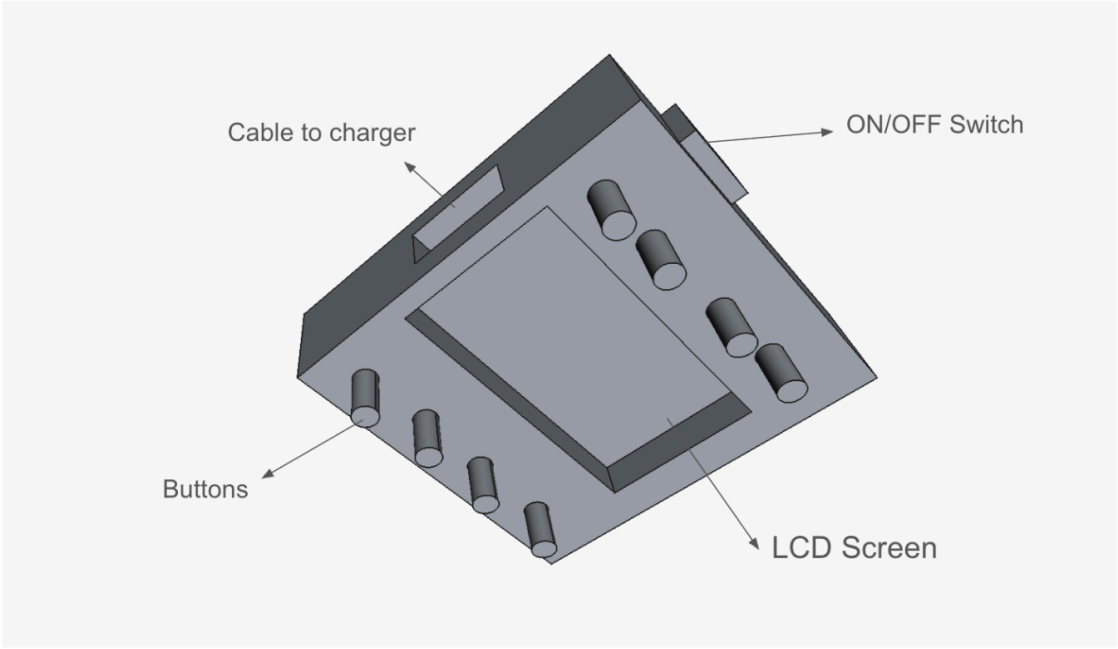


Figure 2.8a: Control Module

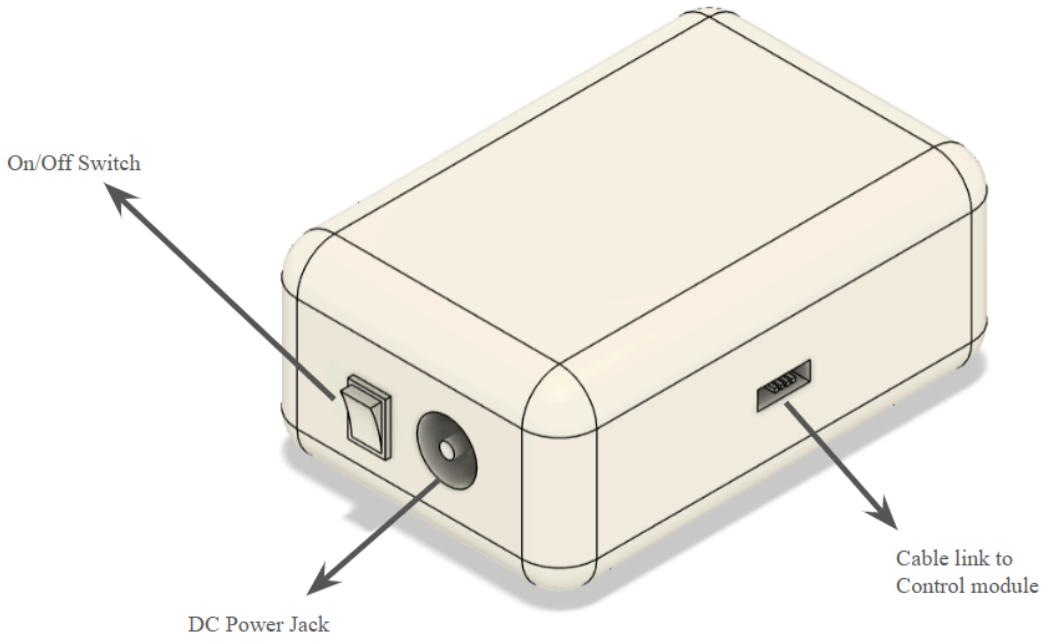


Figure 2.8b: Isometric View of Charging Module (Front)

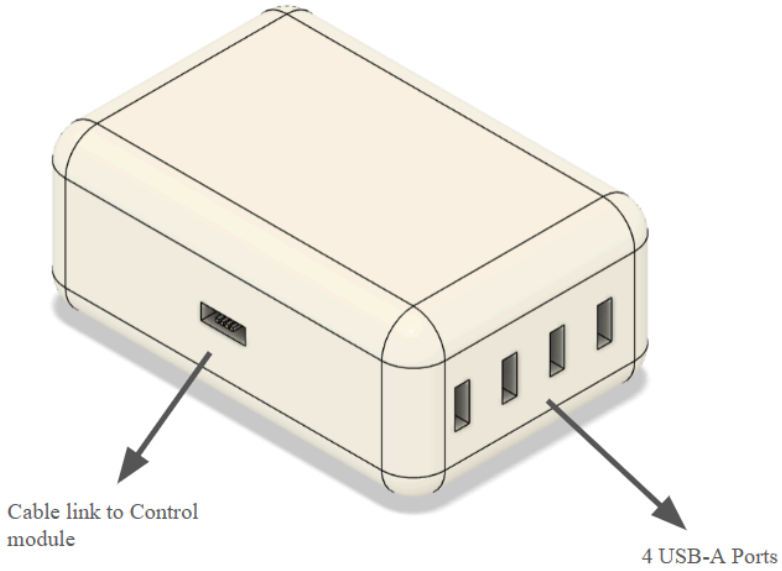
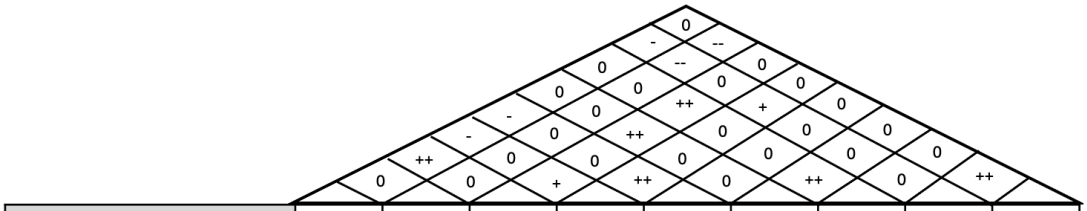


Figure 2.8c: Isometric View of Charging Module (Back)

2.9 House of Quality



Legend Strong Positive Correlation: ++ Positive Correlation: + Negative Correlation: - Strong Negative Correlation: -- No Correlation: 0		USB Charging Port Voltage regulation	Amount of USB charging ports	Power Supply Input Voltage regulation	Maximum Output Current per Port	Minimum Charging Power Range (User-Set)	Charging Delay (Initial Plug-in)	Charge Cut-off delay	Size	Weight
		+	+	+	+	+	-	-	-	-
Cost	-	-	--	-	-	-	-	-	++	-
Durability	+	+	0	+	0	0	0	0	0	0
Device Compatibility	+	+	0	0	-	+	0	0	0	0
Simultaneous Charging	+	0	++	0	-	0	0	+	0	+
Safe Charging	+	+	0	+	-	++	0	+	-	-
Battery Health	+	+	0	0	-	0	0	+	0	0
Portability	+	-	--	-	0	0	0	0	++	0
Target Specifications		5V ± 0.25V	At least 4	12V ± 0.5V	750mA ± 250mA	0.05W - 5.5W	<1 second	< 1 minute	< 40x40x40 cm (Charger) < 30x10x20 cm (Control Module)	< 5 lbs

Figure 2.9: House of Quality

Chapter 3: Research

3.1 Relevant Technology

In order to build a device charger, there are various different technologies required. Hardware and software must each be properly chosen or designed for our use case. As such, we must document our research process into each facet of the project. As a group, we will each choose our respective sub-tasks in the project, research them, describe the information found, and make our design decisions based on it.

3.1.1 Microcontrollers and Field Programmable Gate Arrays

For the main processing of our charger, we are faced with two types of integrated circuits: field-programmable gate arrays (FPGAs), or application-specific integrated circuits (ASICs). The main overarching difference is whether the hardware's configuration can be altered after the part has been manufactured.

A field-programmable gate array (FPGA) is one way of providing main processing in a system. One of the main pieces of an FPGA is a set of configurable logic blocks (CLBs), which can be interconnected and configured with logic gates and other hardware logic devices. This allows the entire logic and processing of the integrated circuit to be changed at any point after it has been manufactured, making it very helpful for prototyping a design. If an issue in the hardware logic has been found, it can be fixed simply by re-programming the FPGA, rather than replacing the device altogether [9].

By contrast, an application-specific integrated circuit (ASIC) does not have the ability to configure the hardware logic after manufacturing. Microcontroller units (MCUs) use ASIC rather than FPGA technology, and thus have another set of benefits. The power consumption and size of ASICs are typically much lower than an FPGA. Additionally, ASICs cost significantly less on average than an FPGA. While the hardware of an MCU cannot be re-programmed, the software is typically designed to be programmable such that it can be used for many purposes. The method of software re-programming is typically less involved than reconfiguring an FPGA, as FPGAs require extensive hardware description language (HDL) experience in order to re-program. Instead, MCU software programming ranges from assembly to C, C++, and other programming languages that can be significantly more user-friendly [10].

Based on the table shown below, we have decided to select a microcontroller unit (MCU) for the main processing of the charger system. Given the time limit of this project, our budget constraints, and our experience with each type of technology, we felt that an FPGA was less feasible to implement for a charger. For a device plugged in all the time, an MCU would consume less power and allow the charging efficiency to be improved while still providing us with enough design flexibility.

Table 3.1.1: FPGA vs MCU

IC Type	Field-Programmable Gate Array (FPGA)	Microcontroller Unit (MCU)
Cost	$100-1000	<$50
Power Consumption	Higher power consumption	Lower power consumption
Flexibility	Most flexible, entire hardware layout can be modified	Less flexible, but has programmable software
Ease of Use/Programming	Difficult to set up and modify. Requires experience in Verilog and/or VHDL.	More user-friendly to configure and program due to using standard programming languages

3.1.2 Microcontroller Architecture

When using a microcontroller, there are multiple different types of microprocessor architectures we can choose. All of these architectures fall into two categories: Complex Instruction Set Computer (CISC), or Reduced Instruction Set Computer (RISC). Our choice of architecture depends on our goals of our charger's implementation.

Complex Instruction Set Computer (CISC) architectures, for a specific instruction, require more clock cycles. For example, an add instruction requires the system to load and store values in order to perform one instruction. Thus, the hardware performs the auxiliary tasks for that instruction, making CISC more hardware intensive. The amount of clock cycles needed per instruction is variable per instruction, which makes instruction pipelining difficult. Architectures such as AMD/Intel x86 and Motorola 68000 are CISC. [11]

By contrast, Reduced Instruction Set Computer architectures have individual instructions for each piece of what would be a single CISC instruction. An add instruction would also require other load and store instructions to be executed in order to function. This means that a compiler for a RISC architecture must also perform the auxiliary instructions for the main instruction. In general, a RISC architecture is more software intensive than CISC. Additionally, since RISC instructions are typically consistently around one clock cycle, instruction pipelining is much more feasible than with CISC. Architectures such as ARM, MIPS, and PowerPC are RISC. [12]

Perhaps the most important difference between RISC and CISC is the efficiency of a set of instructions. Performing multiple operations of 8-bit numbers on a CISC architecture takes around two times the amount of clock cycles as the same operations on a RISC architecture. [12].

When choosing which architecture benefits us in our project, it is important to understand the use case in which CISC is beneficial. CISC, in modern usage, is solely confined to x86 architecture processors. Software support is generally the only barrier between using RISC and CISC in practice [12]. Since there are many well-developed RISC microcontrollers with good software

support in modern times, we will choose a RISC microcontroller. The below table shows a comparison of CISC and RISC architectures.

Table 3.1.2: CISC vs RISC

Architecture	Complex Instruction Set Computer (CISC)	Reduced Instruction Set Computer (RISC)
CPI	2-5	1.5
Extra transistor usage [Citation: Stanford RISC vs CISC]	Storing complex instructions	Memory registers
Location of extra complexity	Hardware	Software
Clock frequency required for same performance	Higher	Lower
Pipelining	Less pipelining possible due to varying instruction length	More pipelining possible due to instructions taking around one clock cycle

3.1.3 SBC Architecture

The control unit of the charger will require a single-board computer (SBC) to run both the user interface and web interface. All of the benefits and drawbacks of CISC and RISC technology mentioned in the previous section for the microcontroller also apply to the SBC. Both CISC and RISC architecture SBCs are available, but RISC dominates many low-power and low-cost SBCs available, just like microcontrollers. Software support for RISC SBCs for our use case is more than adequate, so we will be choosing a RISC SBC for our charger.

3.1.4 Voltage Regulation Methods

In order to maintain a constant output of 5V using a 12V input, a voltage regulator must be utilized to ensure this output is maintained despite any fluctuations that the input can experience during the charging process. The three main Voltage Regulation components that can be utilized are Linear Regulators, Switching Regulators, and Voltage Division using Resistors

A Linear Regulator achieves voltage regulation by utilizing a transistor in a negative feedback circuit to ensure a constant output voltage despite changes in load current and input voltage. A few of the major benefits of this form of regulator is a low output noise making it ideal for Radio Frequency Circuits and for Audio Amplifiers. One major flaw is the design of the Linear Regulator, by having variable series resistance which causes a high input-to-output voltage differential. This differential alongside the high load current results in a large amount of power dissipation in the form of heat. While in isolation this is not problematic on its own, when using

4 linear regulators simultaneously in a small confined space like the inside of a charger can lead to thermal failure which will cause damage to the electrical components housed inside. [13]

A Switching Regulator achieves the same goals as the linear regulator but through using energy storing elements like capacitors and inductors, we are able to reduce the power dissipation levels present in a linear regulator. Furthermore, by using the inductor and capacitor as an LC filter, this results in a smoothed-out voltage which allows for a stable output despite the switch turning on and off. Finally, by altering the duty cycle, the magnitude of the output can vary with a higher duty cycle resulting in an output closer to the input voltage. [14]

Switching Regulators can be further divided into 3 categories: Buck Converters, Boost Converters, and Buck-Boost Converters. A Buck converter is used to lower the output voltage from a high input voltage. This is useful if you know that the output voltage will always be lower than the input voltage. For the Boost Convertor, this is utilized when one wants to increase the output voltage relative to the input voltage where the application of this is for EV vehicles where a lot of the internal components require a higher voltage than is supplied normally by the battery. Finally, the Buck-Boost Converter combines both the buck and boost converter capabilities to make one converter that is able to lower or increase voltage; this option tends to be more expensive. The working mechanics behind this lie in conservation of energy. If the voltage increases, then the current will decrease and vice-verse

The final option would be to create a simple resistive circuit as a voltage divider to achieve the output voltage. While this method is the most simple and easy way to implement this final outcome. It is also the most dangerous way to achieve this outcome. Due to only relying on resistors, any change in input voltage can result in dramatic change for the output. If an electrical surge would arise, any device currently charging can be damaged severely or a potential electrical fire could be caused. Both of these outcomes are undesirable for our end goal.

Based on the table below, we have decided that the Switching Regulator as a Buck Converter would work best in our case.

Table 3.1.4 Regulator Comparison

Voltage Regulation Type	Linear Regulator	Switching Regulator	Voltage Divider
Pros	Maintains Constant Voltage, low noise	Maintains Constant Voltage, High Efficiency	Simple to construct, low cost
Cons	Highly inefficient, High heat dissipation	High noise, Prone to Electromagnetic Interference	No real control on voltage output
Ideal Use Case	RF circuits, low difference between desired output and	Chargers, Sensors, Drones, Radio Controlled vehicles	Low power signal applications

Voltage Regulation Type	Linear Regulator	Switching Regulator	Voltage Divider
	input Voltage		

3.1.5 Current Switching Methods

To manage charging, the system must enable specific charging channels when a device is connected, while also adjusting the current flow to ensure the maximum acceptable current for the load is not exceeded. Two main considerations must be made in order to ensure the switching occurs at the correct moment: 1) Open vs Closed Loop: Determining whether to make the output only reliant on the user input or to include the ability to monitor the current output to determine what changes must be made to ensure the desire output, 2) What type of switch will be used from BJT to MOSFET is also up to consideration as each component has their corresponding pros and cons.

3.1.5.1 Control System

For the open loop system by using a MOSFET for each port as a switch, The Microcontroller (MCU) can turn the switching element using one of the GPIO pins or with Pulse Width Modulation (PWM) to alter the duty cycle. PWM is a technique that uses a digital signal to rapidly turn on and off an input signal. The duty cycle is the parameter that determines the ratio of an input signal being turned on and off with a higher duty cycle meaning the input signal being turned on for longer than being turned off [15]. This process would then lead to an average voltage/current that the load will see. For this use case, this would allow for the charging current to be a lower value than the maximum allowed value on the cellular device. To mimic a charge duration, a timer could be set up which could This method would be most simplest to implement and quickly prototype but suffers from a few key disadvantages:

1) System is not adaptive to the load meaning that while certain parameters may be suitable for an iPod, these same parameters may not be suitable for any other device
2) Due to open nature, users must constantly check and modify parameters to ensure charging is done as the user desires.

Alternatively, a closed loop system can be implemented which is similar to the method mentioned above as it utilizes PWM. A closed loop system is any system that measures the actual output and compares it to the desired output, this leads to the controller to make a choice on which parameters to change to achieve the desired output. This cycle repeats until the desired output is maintained which leads to no further adjustments being made by the controller [16]. In order to implement this system, we would need a sensor to detect the measured current and voltage values. These measurements would then be sent to the MCU, where the MCU alters the duty cycle which affects the average gate voltage which would either increase or decrease the current travelling through the port.

The main difference between these two ideas is whether an open loop or closed loop system will satisfy the demands of our product. In order to make this assessment, one must analyze the

strengths and weaknesses of each loop type. For the open loop system, these systems are generally cheaper to implement, relatively simpler to build, and suited for low power consumption. All of these benefits come at the cost of not being able to adapt to any disturbances in the system. A disturbance is when an event occurs that changes the current output that is outside of the control of the loop system (i.e. a cold draft that results in a change of the temperature) this can cause problems if the system is prone to outside disturbances. This weakness of the open loop system is where the closed loop system can be utilized, the benefits of a closed loop system is being able to adapt to the disturbances that can arise in the system. For the weaknesses, these largely are dependent on factors such as (how many disturbances, how quickly should the system meet the desired goal, and the number of sensors required to detect the changes). Generally speaking, a closed loop system is more complex to implement due to needing to incorporate sensors and communication the the controller, the cost of the system as due to increased demand of parts, and power consumption tends to be higher due to the controller changing how the plant's behaviour.

Table 3.1.5.1 Open and Closed Loop Comparison

	Open Loop	Closed Loop
Cost	Low	Higher
Complexity	Low	Middle-High
Does response change if system changes	No	Yes
Power Consumption	Low	Higher

For the Smart Charger, a Closed Loop system will be considered due to the greater degree of control that can arise with this system. This system would also allow us to set up conditional situations where the smart charger can take a set course of actions to ensure charging is occurring as well as protect the lord device.

3.1.5.2 Switching Component

Now that the control method is selected. One must consider the switching element that will be utilized to ensure the current can flow or open circuit depending on if the desired current can no longer be achieved. The five options that will be talked about are: Metal Oxide Semiconductor Field Effect Transistor (MOSFET), Load switch ICs, Solid State Relays (SSRs), and a Bipolar Junction Transistor (BJT).
A MOSFET is a three-terminal device with a gate (G), drain (D), and source (S) terminals. Current flows between the S and D terminals meanwhile a voltage is applied to the G terminal. This application on the G field allows for an electric field to be generated which controls the amount of current that is allowed to be flowed. MOSFETS come into two different channel types: P-type and N-type depending on if the channel is doped with Accepter atoms or Donor atoms. For a P-type semiconductor, the Gate is considered ON when a negative voltage is applied while for the N-type semiconductor, a positive voltage will allow for the gate to be

turned ON. One further classification is enhancement type and depletion type where in the former case, a voltage needs to be applied in order for current to flow whereas the latter use voltage to close the channel. The considerations on determining which variation of the MOSFET to select is whether we want positive or negative voltage to allow the MOSFET to turn ON and if we want 0V to be considered ON or OFF. [17]

A load switch IC is a semiconductor which is situated between a power supply and a load. These IC's do have a MOSFET present, there exists another circuit setup internally in the system which allows for various protection features to be present in the IC such as a charge pump, slew rate control, reverse-current blocking, overcurrent protection, and a flag circuit. The charge pump allows for an increase in voltage across the load by utilizing two capacitors. By charging the first capacitor fully, another set of switches form a closed circuit leading to another capacitor becoming charged up. This configuration allows for a doubling of the input voltage along the 2nd capacitor. A Slew rate control circuit allows for a slow charging on the capacitive load allowing for no instantaneous voltage drops of the input which prevents the system from being unstable. The reverse-current blocking circuit prevents current from flowing from the output terminal to the input terminal when the IC is turned off. Overcurrent protection limits the output current to protect the load switch and any circuits connected after it. This occurs when there is a short to ground or when load variations create a high demand for current. The FLAG circuit allows a warning to be sent if the circuit identities an external system fault. [18]

A SSR is a switching relay that unlike an electromechanical relay does not have any mechanical parts. This results in a longer lifespan, silent operation, faster speeds compared to their mechanical counterparts. Similar to the load switch IC, an SSR utilizes MOSFETS in order to allow current to flow. Since the design will be utilizing DC current, SSRs with only one MOSFET will satisfy as a switch. An SSR works by using an opto-isolator or opto-coupler which is just a low power LED that will shine a light on a photodiode thus allowing current to flow when the LED is turned on or preventing current flow when the LED is turned off. All of these benefits do come with higher heat generation, a higher upfront cost, and voltage drops across the output terminal leading to lower overall efficiency. [19] [20]

Finally, a BJT is a transistor that utilizes electrons and holes as charge carriers to facilitate current flow. A NJT is a three terminal device (base,emitter, and collector) that facilitates current flow through the emitter or collector depending on the doping of the layers (NPN and PNP). A BJT has 3 useful modes of operation : forward active, saturation mode, and cutoff mode. Forward active mode is the region where voltage and current share a linear relationship. Saturation mode is when the current reaches its maximum value resulting in any increase in voltage resulting in a constant current value. Cut-off mode is when there is no base current which results in no collector current. [21]

Table 3.1.5.2: Switching Component Comparison

Characteristic	MOSFET	IC	SSR	BJT
Type	Voltage - controlled transistor	Integrated high-side MOSFET +	Relay using semiconductor switching	Current-controlled transistor

Characteristic	MOSFET	IC	SSR	BJT
		protections		
Input Drive Requirement	Very low current	Very low control current	Very low control current	Need continuous current
Speed	Very fast	Fast	Slow-medium	Faster than IC slower the MOSFET
Isolation	No	No	Optical Isolation	No
On Resistance/ Voltage Drop	Very low	Very low	Relatively high	Medium loss, mainly in heat
Current Capability	High	Low Medium	Medium	High
Cost	Very cheap	cheap	High	Very cheap
Bidirectional Switching	Limited	Rare	Yes	No
AC Switching	Limited	No	Yes	No
Features	Raw transistor	Overcurrent, soft-start, reverse protect	Built-in protection & isolation	Raw transistor

Based on the table above, we decided to select the Load Switch IC due to being a voltage controlled input that can be turned on or off by the MCU. Furthermore, it comes with in-built protections allowing for redundancy in the circuit. Ensuring a decreased likelihood of critical failure of the whole system. While the SSR is similar in terms of the potential offerings. The optical isolation feature is insignificant as we are not dealing with extremes in voltages in terms of the input voltage compared to output voltage. This design feature is what makes the SSR more expensive as the LED needs to be highly reliable and the photodetector must be able to handle high voltage.

3.1.6 Current and Voltage Measurement

While going about the research process for our Smart Charger, measuring current and voltage is a key component when it comes to this device. More specifically, there are several options that can be used and implemented within this device. In this section, we will be able to determine which device is best to measure the current and voltage enabling best accuracy, precision, and efficiency. Our goal is to also incorporate simplicity in the 12V to 5 V system. Each of the methods presented will offer various advantages and disadvantages when compared against one

another. This could be in reference to cost, power, how precise it can potentially be, and the priorities and overall design it may have to offer. Below presents multiple sections comparing each of the hardware technologies in reference to both voltage and current measurements, then followed by a part comparison as well as the final selections that are to be chosen.

3.1.6.1 Method 1: Shunt Resistor

Shunt resistors are most often used as an approach to measure current in electronics. The way it works is quite interesting, a low-resistance precision resistor is placed in series with a load. The voltage drop is then measured across it, followed by an Ohm's Law calculation where V = I*R. Since both the voltage and resistance are known, we can therefore calculate the current passing through the circuit. Often, the voltage drop is typically small; however, this can be amplified using a current-sense amplifier prior to the microcontroller's reading.

Using this method allows it to introduce very little cost, significant accuracy in gathering the measurements, and also incorporates simplicity. When designing, we also want to make sure that there is low thermal drift, so when selecting a resistor, this is an important criteria to take account of. Furthermore, selecting a device with minimizing power loss to have an adequate power rating is also a vital component towards our choosing. When establishing the trace design and layout, it is important to take note of the two sides that can occur. This includes the high side or the low side. The high side usually is between the supply and load while the low side is between the load and the ground. This is essential to avoid noise or having errors in offset as best as possible.

Table 3.1.6.1 Comparison of Current Measurement Technologies

Technology	Functionality	Disadvantages	Advantages	Accuracy	Cost	Isolation
Amplifier and Shunt Resistor	Voltage drop is amplified along precision resistor and is read by MCU	Generates heat, Significant power loss	Cost friendly, accuracy is high, simple	±0.1–1%	Low	Not Present
Isolated Amplifier (Isolation and Shunt)	Galvanic isolation barrier within shunt	Complex and high costs	Safety, accuracy is high, takes high-voltage	±0.2 –1%	Average	Present
Hall-Effect Sensor	Around current path is able to measure magnetic field	Offset drift, not as precise	Isolated, no insertion loss	±1–3%	Average	Present

Based on Table 3.1.6.1, it is best to consider the shunt resistor with an amplifier to be used in our project. This combination provides high accuracy and can easily be integrated into the overall design. Here, isolation is not a requirement for our device as the system will be operating on a 12V to 5 V, low-voltage domain. Therefore, although this results in no isolation, this ends up being sufficient towards our overall design. [22]

.

3.1.6.2 Method 2: Hall-Effect Sensor

The second method for measuring current and voltage discusses the hall-effect sensor. Current is measured by recognizing a magnetic field that is being generated near a conductor. A hall sensor can be placed nearby when current is being transmitted through a conductor. This is due to the hall sensor being able to produce a voltage in which is proportional to the strength of the magnetic field. The way the hall-effect sensor operates is a method because when the voltage is produced, it can then be converted into a specific signal in which the MCU is able to read and process.

When galvanic isolation is desired and reducing insertion loss is needed, this method would be highly encouraged for those criterias especially since the conductor would not need to be interrupted, physically. Hall Sensors have the capability to measure both the DC and AC current which are often used in high-current or high safety environments and systems. However, although this may be the case, they do incorporate a higher offset drift as well as a higher cost in comparison to the shunt resistors. This then results in this component being less ideal or not as recommended when it comes to systems with low-voltage.

Table 3.1.6.2 Comparison of Hall-Effect Current Sensing

Technology	Functionality	Disadvantages	Advantages	Accuracy	Cost	Isolation
Hall Sensor (Open-Loop)	Directly recognizes the magnetic field	Sensitive towards magnetic noise and temperature	Low loss and isolated	±1–3%	Average	Present
Hall Sensor (Closed-Loop)	Magnetic flux is cancelled from feedback coil	Complex and costly	Significantly precise and linear	±0.5–1%	High	Present

Based on Table 3.1.6.2 we can see that the hall-effect current sensing method can be useful when safety and isolation are highly needed. However, due to average current levels of this specific design and its simplicity, using method 1 as described above with the shunt resistors will still remain more cost effective and practical for this design in specific. [23] [24].

3.1.6.3 Method 3: Isolated Amplifier

Isolated amplifiers are able to incorporate both the galvanic isolation with the high-voltage control circuits as well as how precise the shunt-based measurements are. They are able to send the signals that are being measured along an isolation barrier. This is done by the use of magnetic, capacitive, or even optical coupling. Ultimately, these techniques allow to gather a safe measurement of various current and voltage signals as well as the measurements for high-side.

Isolated amplifiers supply very little to no noise and are very safe; however, they are of high cost and are not as necessary when it comes to low-voltage systems similar to the 12V to 5V conversion that will be used and implemented into the design. Although this is the case, they provide necessary value for designs where there are different ground potentials or where safety compliance is needed when dealing with electrical separation.

Table 3.1.6.3 Comparison of Isolated Amplifier

Technology	How it Isolates	Disadvantages	Advantages	Accuracy	Cost
Capacitive Isolation	Analog signals are transferred by capacitive coupling	High cost, limited range for common mode	Fast speeds, small form factor	±0.2–0.5%	High
Magnetic Isolation	Magnetic field coupling	Complex filtering is needed	Very little to no noise, robust	±0.3–0.6%	High

Based on Table 3.1.6.3 when dealing with applications where high-voltage and safety is critical and needed, isolation amplifiers would be best for these. However, for this design, the non-isolated shunt method still remains as the best choice to be implemented into our project as it incorporates all the necessary criteria and aspects needed for current and voltage measurements. [25].

3.1.6.4 Method 4: Voltage Divider

Voltage dividers are commonly used in DC systems where it is accomplished by using a resistor voltage divider. Essentially, this method is able to reduce a higher voltage, for example 12V, into a range that a microcontroller's ADC, analog-to-digital converter is suitable for. When using the voltage divider method, it consists of two resistors and the output can be seen as the following: Vout = Vin * (R2) / (R1+R2). This equation is right to the point of low cost, and is sufficient when dealing with systems that are non-isolated. Although this seems like a great deal, resistor tolerances as well as drift in temperature can cause several effects when it comes to accuracy in the measurements. A buffer op-amp is most often used to prevent ADC loading and to confirm that the readings are stable throughout the measurement process.

Table 3.1.6.4 Comparison of Voltage Measurement Technologies

Technology	Disadvantages	Advantages	Accuracy	Cost
Voltage Divider with Buffer	Conditional on resistor tolerance	Cost friendly, simple, direct ADC input	±0.5–1%	Low
Differential Amplifier	Costly, Complex design	Common mode noise is rejected	±0.2–.5%	Average
Isolated Amplifier	Costly, not needed for low-voltage systems	Full isolation is offered	±0.3–.6%	High

Based on Table 3.1.6.4 we can see that the voltage divider that incorporates a buffer can be chosen for this design as it is cost friendly, simple, and includes a direct ADC input. Also has sufficient accuracy when it comes to a 12V to 5V conversion circuit which would be present in our project and design overall. [26].

3.1.6.5 Part Comparison and Selection

Based on reviewing the various technologies, the components below are considered to be implemented into our overall design. Each component or each part has been fully evaluated based on how well it is able to perform, how it is able to integrate with the microcontroller, as well as if it is available.

Table 3.1.6.5a Part Comparison for Current Measurement

Part	Manufacturer	Features	Usual Calibration or Example Range	Accuracy (gain error)	~Cost
INA219	Texas Instruments	I2C current or shunt monitor, 0-26V bus, bidirectional	±3.A for external shunt	≤ 0.5% (grade B)	$2-4
INA226	Texas Instruments	I2C low/high side monitor, 0-36V bus, 16-bit ADC	Programmable through shunt calibration	≤ 0.1% gain error	$4-5
INA240	Texas Instruments	-4 to 80V CM, zero drift	Configured through resistor	≤ 0.2% gain error	$3-6

Part	Manufacturer	Features	Usual Calibration or Example Range	Accuracy (gain error)	~Cost
INA219	Texas Instruments	I2C current or shunt monitor, 0-26V bus, bidirectional	±3.A for external shunt	≤ 0.5% (grade B)	$2-4
		amplifier enhanced PWM rejection	or shunt		

The selected part is the INA226 as it has high precision, I2C which incorporates a digital interface, as well as a wide operating voltage range. This makes it most ideal for selecting a component to measure current in comparison to the other suggested in the table. Accurate measurements can be obtained from the INA226 and also provides great microcontroller communication. [107], [108] , [113].

Table 3.1.6.5b Part Comparison for Voltage Measurement

Part	**Manufacturer**	**Features**	**Input/ Usable Range**	**Typical Accuracy/ Specs**	**~Cost**
OPA333	Texas Instruments	Low offset, zero-drift, rail to rail I/O	1.8 V to 5.5 V supply	Offset ≤10 µV; drift ≤0.05 µV/°C	~$2-3
TLV271	Texas Instruments	Rail to rail output, low power, 3 MHz BW	2.7 V to 16 V supply	Input offset ~0.5 mV typ (spec sheet)	~$1-2
MCP6071	Texas Instruments	Cost friendly precision op-amp	1.8 V to 6 V supply	– (datasheet provides pA bias current, derive full-scale error by user)	~$0.8-1

Based on Table 3.1.6.5b the OPA333 has been chosen as it incorporates high accuracy and is highly stable. It has low input offset and with its rail-to-rail operations, this ultimately makes it highly suited when it comes to voltage sensing in applications that pertain to power DC. [110] [111] [112] [113].

3.1.6.6 Final Selection and Justification

Based on the thorough analyses and comparisons of the different components and technologies above for measuring current and voltage, below portray the decisions for the current and voltage measurements.

For current measurement, the shunt resistor in a combination with the INA226 current-sense amplifier will be best implemented here as a general selection for measuring current and voltage. This design and configuration allows for a cost friendly, simple, and high precision.

For the voltage measurement, the voltage divider with the OPA333 precision buffer amplifier has been chosen. This results in accuracy when taking voltage measurements as well as there being very little component count.

3.1.7 USB Port Output- Measuring Current and Voltage

Gathering accurate data and measurements from current and voltage from the USB port is highly critical when it comes to confirming if there is power delivery or the amount that is taking place and also if it is being adhered with specific USB criteria. Also, making sure protection is implemented correctly and thoroughly when it comes to undervoltage or overcurrent environmental conditions. The key to this is to be able to take the measurement of the voltage supplied to the connected USB device as well as the current being drawn from the port. This is to be done while maintaining very little to no voltage drop so that the USB is able to still perform correctly. Within this section there will be various technologies that will be both introduced and compared for the USB port output where it will monitor and determine which is most suitable to obtain the most reliable and accurate measurements.

3.1.7.1 Technologies of Output Current Measurement

Monitoring the power that is being delivered on the +5V (VBUS) line is highly critical when it comes to USB port current measurement. Essentially, this allows us to confirm that the device that is connected to the USB port is able to properly operate within the specified current limits for the respective USB. Now, the sensing current is to be placed in between the 5V regulator output and the USB connector. It must be placed here because this setup ultimately allows for the microcontroller to detect the current being drawn to each of the ports in real-time, which is exactly our goal. The measurement must be able to maintain a specific voltage drop that would comply with the USB standards. Note that this voltage drop must be below 350 mV.

Method 1: Shunt Resistor and Amplifier

Using a shunt resistor and amplifier allows for a low-value precision resistor to be placed in series with a USB VBUS line. When the voltage drop occurs along the resistor, it is then

amplified through the use of a current-sense amplifier or even an integrated monitoring IC. Thus, it is then producing a signal which therefore can be displayed as the load current. Using this method as the combination of shunt resistor and amplifier, it is able to result in a cost friendly, accurate, and often used USB design to conduct these measurements. However, insertion loss can occur which needs to be minimized through carefully selecting a resistor as well as properly selecting a PCB layout.

Method 2: Integrated USB Power-Monitor ICs

The Texas Instruments INA260 and MaximMAX34407 are integrated ICs that are a combination of both amplifiers and a precise shunt resistor all in one. Current and voltage can both be measured from this device which is needed for our project. Digital outputs are also provided including I2C as well as SPI and are able to have full functioning calibration as well as the alert option. This method has an average cost but can improve overall precision for this circuit to be simplified in its design.

Method 3: Hall-Effect Sensors

When dealing with hall-effect sensors, these operate by detecting the magnetic field with the current that is flowing through the conductor. This results for isolated measurement with there not being a voltage drop where it is also non-intrusive. Although these sensors are great for when it comes to isolation, they are costly, large in size, and not as accurate. Therefore, this results in the hall-effect sensor not being as ideal when it comes to application in relation to compact USBs.

Table 3.1.7.1 Comparison of Technologies for the USB Port Output Current Measurement

Technology	Functionality	Advantages	Disadvantages	Accuracy	Cost
Amplifier and Shunt Resistor	Voltage drop is measured along a low-value resistor that is in series with VBUS.	Cost friendly, very precise, simple	Layout is sensitive and small insertion loss	±0.1–0.5%	Low
Integrated USB Power-Monitor IC	I2C/SPI-digital output; Internal shunt and amplifier	Calibrated, compact/ small, simple MCU interface	Possibly high costs	±0.1–0.3%	Average
Hall-Effect Sensor	Magnetic field is measured along VBUS	Isolation, voltage drop is not present	Larger in size, higher cost, not as accurate	±1–3%	Average to High

Based on table 3.1.7.1 we can see that the integrated USB power-monitor IC is best to be selected for measuring the current from the USB port output. High accuracy is available for this device as well as not much design effort to be incorporated. There is also direct digital communication with the microcontroller. This is especially essential as it minimizes the board space and other outside components that may need to be used. [27].

3.1.7.2 Technologies of Output Voltage Measurement

The VBUS line must remain within the desired 4.75-5.25 V range which is defined by the USB specification requirements. This is in reference to gathering the measurements for the USB output voltage. The voltage sensing circuit is placed along the USB VBUS and the ground lines. Thus, there should also be a high input impedance so there loading at the port does not take place and therefore is not an issue when measuring voltage. With this configuration, the system can then identify the voltage drops that are potentially being caused by cable resistance or even large amounts of current.

Method 1: Buffer and Voltage Divider

Using a voltage divider allows for the USB voltage to briefly scale down so that it can match the ADC input range that the microcontroller can handle. When a buffer-opamp is added this essentially does not allow the ADC loading to occur and essentially helps to control the signal. Using this method promotes a cost friendly, compact, and also allows for high amounts of accuracy for the fixed 5V USB outputs which will be used here.

Method 2: Integrated Power-Monitor ICs

The INA260 and MAX34407 are both USB power-monitor ICs. These devices help integrate both current measurements and voltage measurements onto one singular chip. The voltage is reported digitally along with the readings to be calibrated. For this specific design, there is no need for an amplifier or a separate voltage divider. This is mainly ideal for small and multi-port systems.

Method 3: Isolated Voltage Measurement

Isolated voltage measurement uses isolation amplifiers or also opto-amplifiers where voltage can be measured through an isolation barrier. Note that it is not necessary for there to be a 5V USB port where a common ground is shared by the system; although important for high-voltage and applications where safety is vital.

Table 3.1.7.2 Comparison of Technologies for the USB Port Output Voltage Measurement

Technology	Functionality	Advantages	Disadvantages	Accuracy	Cost
Buffer and Voltage Divider	Scaled to 5V for ADC for divider, no loading	Cost friendly, reliable, simple	Needs resistor tolerance, no isolation	±0.5–1%	Low

Integrated Power-Monitor IC	Current and voltage are both monitored along with calibration	Accurate, simple design, small	Higher costs	±0.1–.3%	Average
Isolated Voltage Sensor	Measurements are isolated from MCU	Safety isolation is provided	High costs, not needed for 5V	±0.3–.5%	High

Based on Table 3.1.7.2 we can see that the integrated power-monitor IC is best selected for measuring the voltage from the USB port. This is due to the fact that it is highly precise, compact, and is able to measure current and voltage simultaneously which is necessary in this application for this specific design we are implementing. The design is further simplified as another analog circuitry is not needed here. [28] [29]

3.1.7.3 USB Power-Monitor IC Comparison and Selection

When identifying which component is best needed for monitoring the USB port output, there are several integrated ICs that are compared based on various specifications or requirements. This includes current accuracy, voltage range, cost, and even its communication interface. Below demonstrate the devices that are selected and most commonly used when it comes to monitoring the USB VBUS lines that adhere to the 5V standard.

Table 3.1.7.3 Part Comparisons for the USB Output Measurements

Part	Manufacturer	Functionality
INA260	Texas Instruments	Internal shunt, I2C output, integrated voltage, power, and current monitor
INA230	Texas Instruments	I2C interface, alerts can be programmed, high-speed 16-bit power monitor
MAX34407	Maximum Integrated	Multi-channel I2C power monitor for the 3.3V or 5V rails.
ACS712	Allegro Microsystems	Hall-effect sensor with also the analog input

Based on Table 3.1.7.3 the INA260 is best to be chosen as it is able to measure both the current and voltage from the USB port output along the USB VBUS line. Note that a single shunt resistor has been integrated as well as an amplifier within this configuration. High accuracy as well as digital communication is also provided here with the microcontroller using I2C

communication. As for the other parts mentioned in the table, the INA230 and MAX34407 can be used for monitoring several channels; however, the INA260 has the ability to offer the combination of cost, preciseness, as well as its simplicity when it comes to the USB port. [114] [115] [116] [117].

3.1.7.4 Implementation

We want to make sure that when we are receiving the current and voltage measurements from the USB port output it must meet or comply with the USB standards. This involves placing the sensing IC in the correct area, using low-resistance traces, confirming that the voltage drop across the sensing remains a certain value, ensuring that the microcontroller is configured in a way that is sampled power data at times for overload current, and more. Note that it is highly needed for this design to maintain the USB VBUS voltage within the 4.75 to 5.25 V under a full load. This system will then be able to provide very accurate readings for the protection of the port and managing power overall.

3.1.7.5 Final Selection

The INA260 integrated power-monitor IC will be selected here to measure the current and voltage from the USB port output. It incorporates an internal precision shunt, the gain is calibrated, and there is also the I2C communication interface. All of these aspects combined make it a complete option to monitor the USB in real-time. The device reduces complexity overall and is able to preserve how well the signal is transmitting. Also, it lies within the USB voltage and current requirements or specifications.

When voltage and current is integrated into a singular device, not only does this make it more efficient, but it reduces PCB area. Although this seems like a small benefit, it improves reliability overall and also is able to enable the microcontroller to recognize if there are any irregularities when it comes to power right away. Using this method allows us to accurately monitor the USB output power delivery where the standards are compliant and this method is overall safely used as well.

3.1.8 MCU Power Supply

There are several microcontrollers available to use when designing a device. However, specifically, there is the ESP32-WROOM-32E. This microcontroller in specific needs a stable low-noise 3.3V DC power source to ensure that there is a reliable operation when it comes to its Bluetooth, Wi-Fi, or even other processing subsystems. Within this specific design, a regulated power supply is needed for the MCU. With the main system having an input of 12V DC this must be brought down to 3.3 V. Within this section, there will be different voltage conversion technologies that will be compared where each of the components are to be evaluated thoroughly and finally a selected component for the final power regulation that is best suited for the ESP32.

3.1.8.1 12 V to 3.3 V Power Conversion Technologies

Specifically for this design, it is needed to have an MCU power supply that must convert 12 V to 3.3V while there is minimal rippled, appropriate energy usage, as well as stable voltage that is being maintained throughout the system. In specific, the ESP32-WROOM-32E usually operates at 3.3V with current averaging about 80 to 160mA. Note that this is during Wi-Fi transmission bursts. THis essentially means that a stable current delivery must be provided by the regulator while heat generation is also being reduced. In reference to the options for the possible conversions, these can be linear regulators, low-dropout (LDO) regulators, or even buck converters.

Linear Regulators

Voltage conversion is provided by linear regulators; however, in specific this is done by dissipating the extra voltage as a means of heat. Linear regulators offer qualities such as low cost, no noise; however, they are not used for large amounts of voltage drops. For example, this can be seen as converting 12V to 3.3V where there is a 150mA load present. THis then ultimately leads into more than 70% of the power that is present turning into heat which does not make this a practical choice for operation on the MCU.

Low-Dropout (LDO) Regulators

In reference to LDOs, these have a greater efficiency and are more standard linear regulators. They are also able to maintain the regulation where there are minimal differences within both the input and the output. However, when reducing voltage from 12V to 3.3V, there is then low-efficiency that is now occurring as well as higher large amounts of thermal loss that is taking place. LDOs would be best for small voltage drop applications, not large amounts.

Buck (Step-Down) Converters

Buck converters use high-frequency filtering and switching to allow it to efficiently lower its voltage from previous higher voltages. They are highly efficient which results in the buck converters being a great and most suited choice for converting 12V to 3.3V for the MCU. Buck converters are able to offer a compact size which is ideal for the PCB layout, low-ripple output, as well as protection for the thermal aspects. Note that these qualities result in reliable operation for microcontrollers, in specific the ESP32.

Table 3.1.8.1 Comparisons for the Power Conversion Technologies

Technology	Functionality	Efficiency	Dropout Voltage	Noise Level	Cost	Suitable
Linear Regulator	Dissipates voltage as heat, output is simple fixed	25–40%	> 1 V	Low	Low	Poor

LDO Regulator	Better performance for small step, low dropout	30–50%	> 0.3 V	Low	Average	Average
Buck Converter	High-efficiency switching with capacitor and inductor filtering	85–95%	Not Applicable	Medium	Average	Great

Based on Table 3.1.8.1 it can be seen that the buck converter would be most suitable for the MCU power conversion. Note that the conversion will be going from 12V to 3.3 V. Buck converters have great amounts of efficiency and are also able to provide stable amounts of current with there being very little to no heat being dissipated. [30] [31] [32]

3.1.8.2 Voltage Regulation Technologies for the ESP32

In specific, the ESP32-WROOM-32E is several sensitive when it comes to stable voltage. This can be mostly seen during Wi-Fi transmission bursts. 3.3V ± 5% must be maintained or stabilized by the power supply and output ripple must also be limited below 50mV. This is crucial because it allows us to confirm that there is proper functionality within the RF subsystem. Common voltage regulation devices or technologies will be compared in this section when dealing with low-voltage digital system applications for the MCU power supply.

Buck Regulators (both Synchronous and Asynchronous)

The buck regulators use MOSFETs for power switching specifically for efficiency. When dealing with synchronous buck converters, they essentially are able to obtain greater amounts of efficiency where diodes are replaced by MOSFETs for rectification. However, asynchronous are not as efficient but are very simple in design. Transient response and ripple response work for the buck regulators for applications in reference to the MCU.

LDO Regulators (After regulation has occurred)

In some cases, LDO regulators are often used after the buck converter so that noise can be further filtered out and voltage is also stabilized. This specific technique lowers efficiency; however, the voltage cleanliness for the sensitive analog circuit is improved significantly. However, for the ESP32 in specific, a low-ESR capacitor along with proper filtering results in the LDO to not be needed within this design.

Table 3.1.8.2 Comparison of the Voltage Regulation Technologies

Technology	Functionality	Efficiency	Output Ripple	Thermal Performance	Cost	Suitable

Asynchronous Buck	Diode rectification along with simple switching design	85-90%	Average	Good	Low	Excellent
Synchronous Buck	Higher efficiency due to dual MOSFET rectification	90-95%	Low	Excellent	Average	Excellent
LDO (Post-Regulation)	Noise filtering (Linear regulator)	70-85%	Very Low	Average	Low	Average

Based on Table 3.1.8.2, a synchronous buck converter is to be selected as it is the most efficient and most reliable option when the ESP32-WROOM-32E needs to be powered from the 12V input. A stable, low-ripple 3.3V output is provided and a peak current for transmitting Wi-Fi is also provided from using the synchronous buck converter. [33] [34].

3.1.8.3 Part Comparison and Selection

Based on the table below, we are able to compare the possible buck converter modules that are to be used for the 12V to 3.3V step-down voltage conversion. Note that this is specific for the ESP32-WROOME-32E MCU. The devices will be selected on output ripple, current rating, and efficiency.

Table 3.1.8.3 Comparisons of the MCU Power Supply Part

Part	Manufacturer	Features	Input Range	Output Voltage	Max Current	Efficiency
MP1584	Monolithic Power Systems	Output is adjustable, compact, 1.5 MHz	4.5–28 V	3.3V	3A	~90%
LM2596	Texas Instruments	150 kHz switching regulator, available	4.5–40 V	3.3V	3A	~88%

AMSR-7812-3.3	Recom Power	Integrated non-isolate Dc-Dc module	6.5–36 V	3.3V	500 mA	~92%

Based on Table 3.1.8.3, the MP1585 buck converter is to be selected. This buck converter incorporates compactability, great efficiency, as well as its ability to control or readjust to the transient current spikes on the ESP32 when wireless transmission takes place. A 3.3V regulation is confirmed when there is a low output ripple and very little to no thermal that is built up. [118], [119], [120].

3.1.8.4 Implementation

As the buck converter has been chosen for the part selection, it is critical to consider how it may be implemented or placed within the design. The buck converter can be placed near the MCU. This is essential to reduce the amount of voltage drops that take place as well as possible coupling of EMI. Ripple can also be reduced here with the use of low-ESR capacitors on the input and output end, and more. Furthermore, the regulator's transient response will furthermore be checked and will meet the ESP32's160 mA peak current that is a requirement that has been stated. Note that the design considerations for when to be implemented as stated above help to ensure that the MCU is able to receive a stable power supply that is also suited for high-speed Wi-Fi operation. Note that this will be implemented into the design overall.

3.1.8.5 Final Selection

Based on the comparisons above, the MP1584 buck converter has been selected for a regulated 3.3V DC supply to the ESP32-WROOM-32E. It incorporates great amounts of efficiency, compactability, as well as a fast transient response. Overall, these qualities make it a great choice to support the MCU's load conditions that would always be changing. The buck converter also has low ripple which helps to confirm that the ESP 32 is able to operate with no interference taking place or any sort of instability while there is wireless communication. Note that this configuration, in specific, provides a medium between performance, how efficient it is, as well as how reliable the device is. Thus, this results in optimal operation of the MCU overall where thermal and power loss is also reduced throughout the entirety of the process.

3.1.9 Energy Source Research

3.1.9.1 12V AC-to-DC Wall Adapter (External Power Brick)

A 12-V AC-to-DC wall adapter is a commonly used power source for applications in embedded systems and consumer electronics. These adapters have an internal isolated power supply that converts 120 AC mains into a stable, low voltage DC output. Due to this AC-to-DC conversion device being commercially sold, all safety considerations and compliance has been addressed pertaining to isolation, EMI filtering, and protection circuitry have all been integrated within the adapter itself. This results in a simplification in the PCB design of the downstream system. 12V

adapters are available in current rating from 1A to 10A. They are commonly used in monitors, laptop chargers, printers, and industrial controllers.[141]

A 12-V DC rail provides sufficient voltage for any buck converters, allowing efficient step-down to 5V and 3.3V for the USB power outputs and for the microcontroller and sensor subsystems respectively. Operation at 12V improves the conversion efficiency as buck converters have lower conduction losses and smaller inductor sizes as input voltage increases. Another benefit of using an external 12-V adapter is the automatic safety compliance- such as insulation, creepage, and surge testing. Since the device is already sold on a commercial scale, this would mean that any safety and compliance laws any electrical device must meet was met by the manufacturer. This reduces the need to ensure the AC to DC conversion meets specifications on the end of the PCB. The PCB can only handle low-voltage DC and does not interact with the AC mains. [142]

One drawback to this configuration is the need for separate buck regulators to ensure each device is handling their required voltage without going over the limits since no USB device can be powered directly from 12V. This increases component count and board area but offers the advantage of per-port isolation and independent regulation. For this design, the 12-V adapters route is great due to balancing safety, modularity, and efficiency while supporting high power activity the four USB charging channels would need to perform.

3.1.9.2 12V High-Current 5-V DC Supply (Direct USB Power Rail)

Another potential energy source is a high-current 5-V power supply, commonly found in USB hubs, SBC power adapters, and industrial 5-V power modules.The supply can output 5 to 20 A from a regulated 5-V rail. Using this implementation, a single 5V high current supply can power all ports directly without requiring external 5V buck converters. [143]

This architecture simplifies the power system as it reduces the need for the 5V DC converter, lowers component count, and reduces potential thermal hotspots. It also eliminates efficiency losses that are associated with 5V buck converters. Implementation of these supplies are present in USB charging stations and SBC clusters like Raspberry Pi farms. [144]

However, this implementation has many drawbacks for its application in a multi port charger. Since all ports share the same 5V rail, it becomes more difficult to identify faults. Another drawback would be in the case of an overload or a short condition occurring, if these conditions happen then the entire 5-V rail could potentially shutdown which would result in other ports also shutting down even if the fault condition did not occur at the specific location. Implementing current limiting and protection becomes more complex as the main regulation happens in 1 rail that could affect 4 potential devices instead of only needing to monitor each regulator that is in control of one USB port. Finally, the 3.3V rail for the ESP32 and INA260 sensors would still require a regulator as these components cannot handle the 5V regulator directly.

3.1.9.3 On-Board Off-Line AC-to-DC conversion (Integrated PSU)

A third option is to integrate an off-line switch-mode power supply directly on the PCB. These configurations would have converters that accept 120 AC line voltage and generate low voltage

DC outputs. Commercial USB chargers and power adapters have optimized SMPS topologies such as flyback converters with rectification. Integrating this into the PCB offers the highest efficiency and most compact solution as it removes the need for an external power brick powering the PCB. [145]

Integrating a SMPS would require the supply in being able to handle rectified line voltages up to 170 VDC. This requires components rated for high voltage and high dv/dt stress. An example of high level equipment that would be required to handle these tasks would be a 600V superjunction MOSFET, high-voltage bulk capacitors, and reinforced-isolation transformers. Constructing a flyback transformer is a highly specialized task in itself as it must balance inductance, leakage, saturation limits, and insulation thickness. [146]

In addition to the electrical complexity, off-line PSU must comply with strict safety standards such as IEC61010-1, IEC-62368-1 and UL-60950. The standards listed above define mandatory isolation levels, protective earth requirements, fault testing conditions, and maximum leakage currents. Designing this system must make sure that clearance distances are met, certified capacitors be utilized for EMI suppression and that thermal fuses or protective MOVs for surge events. Failure to meet these requirements would result in fire hazards or complete failure of the device.

Another concern would be Electromagnetic Interference (EMI). Off line converters generally operate at high switching frequencies, these frequencies would cause strong electromagnetic interference that would require the use of complex filters, a shielded transformer, snubber networks, and optimized PCB strategies to minimize the effect of EMI on the device operation. Normal production on the commercial level typically has an entire team that specializes that EMI is having minimal effect on the circuit which would be far beyond the scope of this current design.

Thermal management would present as the final difficulty in constructing this converter. During the rectifying process, components such as MOSFETS, diodes, and transformers can dissipate a lot of watts of heat. Proper heat spreading, airflow considerations and PCB copper thickness must all be taken into account in order for this design to be operational. Furthermore, commercial chargers use clamp networks or synchronous rectification that would improve efficiency and reduce thermal losses, it would also further increase the design complexity in order to ensure a working device.

Given all of these conditions to consider. The manufacturing and prototyping stage for the project would be the most riskiest as any mistakes during the manufacturing stage could destroy components, improper isolation may allow for lethal voltages to be in contacts with user accessible ports and specialized equipment will need to be utilized to ensure the PCB meet all design safety.

3.1.9.4 Additional Engineering Considerations for Power Source Selection

When examining the strengths and weaknesses between these three power supplies, there are many advanced engineering considerations that further influence the practicality and safety of

each candidate. These factors must be considered carefully to ensure working operation of the device in order to serve as the main power architecture supplying the power. [147]

One major factor is line regulation and transient immunity. AC voltage is not a perfectly stable line. It can experience many disturbances and variances such as surging or harmonic distortion from shared household devices including but not limited to HVAC motors, refrigerators, and car charging stations in the case of electrical cars. An integrated SMPS would need to be carefully designed in order to handle the many variables that exist during power delivery. [148]

Another important consideration would be power factor and harmonic content, All Modern Power supplies in the US that are available on the commercial market must comply with IEC61000-32 and related harmonic limits. Supplies that within a certain wattage threshold like 60W for example must include a stage that deals with power factor corrections. This ensures that the waveform of the input signal is in phase with the voltage from the AC mains. In order to implement this successfully, the advanced control ICs high-voltage inductors and precision sensing resistors. Failure to ensure proper design of the system would result in significant distortion on the main power lines resulting in a device that would be unusable in real world applications.

Thermal management also becomes critical as the power supply must not get hot too quickly to ensure a thermal shutdown event is kept to a minimum. All modern power supplies integrate MOSFETs, Diodes, transformers, and bulk capacitors to combat the rise in temperature. Temperature also directly affects reliability, mean time between failures, and capacitor lifespan. An excellent design would maximize the first and third parameters while trying to reduce the 2nd. Commercial charges generally employ multi-layer copper planes, thermal pads, and as well as optimized airflow paths. Careful material selection is also employed to reduce the heat buildup present in charging applications.

Furthermore, Mechanical and practical factors also play a role in picking a AC to DC converter as going for a commercial purchased one allows for a great size reduction in the PCB as it will not have to manage the thermal effects of converting from AC to DC which would be given to the external applications. If opting for a custom AC to DC converter. This would require not only a greater PCB size but also additional modifications to the PCB so it can accommodate the increased thermal demand, EMI hindrances, and finally be acceptable for standards to be used on an AC mains source.

From a system-integration perspective, the 12-V adapter allows for cleaner power distribution as it simplifies PCB design so the main focus is on DC-DC conversion which is simpler and generally lower in thermal demand compared to AC-DC conversion. The 12V adapter would also draw less current compared to a 5V source in order to reach the power demands of the device. This architecture would also simplify current monitoring which is a main focus for the design,

Finally, from a commercial and time perspective. The 12V adapter proves the superior choice as it will not require a custom design which can exceed costs compared to the other collective components of the project and avoids the requirement of needing approval from standards as the

device will already be pre approved with all the safety features and capabilities that would be required in this project.

3.2 Part Selection

3.2.1 Microcontroller Selection

As discussed in section 3.1.2, we have decided to choose a microcontroller with a RISC architecture. Many such options exist on the market from various different vendor types, and we will describe each feasible option to us below.

3.2.1.1 ESP32

The ESP32 is a family of microcontrollers created by Espressif Systems with a 32-bit architecture. Various different models exist with either Xtensa or RISC-V processors. RISC-V processors are only available in single-core, but Xtensa processors are available with two cores. All ESP32 microcontrollers have wireless capabilities via Wi-Fi and/or Bluetooth. Some ESP32 microcontrollers have the antennas integrated into the PCB, and others have a connector to add an external antenna. The ESP32 is available both as a bare microcontroller, or as a development board. Various product families exist, including WROOM, SOLO, MINI, and WROVER. For this project, we will primarily be considering the ESP32-WROOM-32E. It has a dual-core Xtensa LX6 processor, paired with 520KB of RAM, and the option for 4, 8, or 16MB of flash storage. It is equipped with 802.11n Wi-Fi capability and Bluetooth 4.2, and has 26 GPIO pins. It has PWM support and an ADC, along with UART, I2C, and SPI capability. As for programming with an integrated development environment (IDE), it can be programmed with the Arduino IDE, or Espressif's own integrated development framework (ESP-IDF) attached to a conventional IDE.

3.2.1.2 Texas Instruments MSP430

The MSP430 is a family of microcontrollers created by Texas Instruments. All MSP430 microcontrollers are 16-bit systems with their own RISC microarchitecture. MSP430s have been produced in various different versions for over 30 years, and many options are available with different hardware configurations. However, no MSP430 has built-in wireless connectivity. RAM can vary from less than 16KB to over 256KB, and GPIO pin count can vary from less than 24 to over 80. Bare microcontrollers and development boards are available in different packages over different generations to suit a wide variety of use cases. In order to develop for an MSP430, Texas Instruments provides its own integrated development environment (IDE) tools and libraries for specific parts. For this project, we will be considering the MSP430FR6989. It is available as a development board or bare chip, and features a 16MHz processor with 128+2KB of RAM. The 128KB of FRAM doubles as its storage. It has an ADC and such communication protocols as I2C, SPI, and UART.

3.2.1.3 Raspberry Pi Pico

The Raspberry Pi Pico is a family of 32-bit microcontroller boards created by Raspberry Pi. There are several different models available, from both the original Pico family and the Pico 2 family. The Pico family uses the RP2040 microcontroller, and the Pico 2 family uses the RP2350 microcontroller. The "W" board variants have Wi-Fi (2.4 GHz only) and Bluetooth 5.2. We would be considering using the RP2350 microcontroller as a bare chip for this project. It has a maximum CPU clock of 150MHz, 520 KB of RAM, and 2MB of flash storage. The bare microcontroller chips are available, however they do not have built-in wireless connectivity like the ESP32. Wireless is only available as part of a dev board. This decreases the cost of the microcontroller itself, but makes integrating wireless as part of a custom PCB more difficult. Thus, if we choose to implement wireless communications on the charger's main board, the RP2350 would be a much less convenient option than an ESP32. The RP2350 also offers UART, SPI, I2C, and PWM support built-in. The Raspberry Pi Pico (and by extension, the RP2350 microcontroller) can be programmed in Python using a tool called MicroPython, but a C/C++ software development kit (SDK) is also available.

3.2.1.4 ATmega 2560

The ATmega 2560 is an 8-bit microcontroller created by Atmel, used in such development boards as the Arduino Mega 2560. It features a 16 MHz CPU, 8 KB of RAM, 256 KB of flash storage, and a 4 KB EEPROM. As for communication, the ATmega 2560 supports UART, SPI, and I2C. It also supports PWM, and has 86 GPIO pins, however, it does not have any form of wireless connectivity.. It is programmable via the Arduino IDE. Aside from the development board, the ATmega 2560 is available in both BGA and QFP package types as a bare chip. However, the cost of this microcontroller is the highest of any of the microcontrollers we are considering. Additionally, it has the least RAM and nearly the least storage out of each of the microcontrollers. Thus, it offers the least value and no compelling feature set for us to use.

3.2.1.5 STM32

The STM32 is a family of 32-bit microcontrollers created by STmicroelectronics. All STM32 microcontrollers use an ARM processor, with many different variants available. Some STM32 microcontrollers are available with wireless (Wi-Fi and/or Bluetooth) connectivity. Single or dual-core microcontrollers are available, to target either high performance, general use cases, low-cost use cases, or low-power use cases. We will be considering an STM32F722ZE, which is available as a bare chip, or as part of a development board (part NUCLEO-F722ZE). It features an ARM Cortex-M7 CPU, 256 KB of RAM, and 512 KB of flash storage. It has three 12-bit ADCs, two 12-bit D/A converters, as well as I2C, UART, SPI, SAI, and CAN communication protocol support. However, it does not have wireless communication support. Programming an STM32 is commonly done via the Arduino IDE.

Table 3.2.1: Microcontroller Comparison

Architecture	ESP32-WROOM-32E-N16	TI MSP430FR6989	Raspberry Pi Pico RP2350B	ATmega 2560	STM32 F722ZE
Processor (SoC)	Xtensa LX6	TI MSP430	RP2350	ATmega 2560	ARM Cortex-M7
Cores	2	1	2	1	1
Bits	32-bit	16-bit	32-bit	8-bit	32-bit
Clock Speed (Max)	240MHz	16MHz	150MHz	16MHz	216MHz
RAM	520 KB + 2 MB PSRAM	2KB (SRAM), 128KB FRAM	520 KB	8KB	256 KB
Nonvolatile Storage	16MB Flash, 448 KB ROM	128KB FRAM	2 MB Flash	256 KB Flash, 4KB EEPROM	512 KB Flash
GPIO Count	26	83	48	86	
ADC	Yes, 12-bit	Yes, 12-bit	Yes, 12-bit	Yes, 10-bit	Yes, 12-bit
UART	Yes	Yes	Yes	Yes	Yes
I2C	Yes	Yes	Yes	Yes	Yes
SPI	Yes	Yes	Yes	Yes	Yes
Wireless	Yes, 2.4GHz + Bluetooth 4.2	No	No, but dev board has 2.4GHz Wi-Fi and Bluetooth 5.2	No	No
Cost (Bare chip)	$5.71	$9.47	$1.00	$17.59	$14.06

Based on the table and the different microcontroller options available to us, we have determined that the ESP32-WROOM-32E and STM32F722ZE are the best fit for our project. However, the STM32F722ZE lacks the wireless capabilities of the ESP32-WROOM-32E, and is also more expensive. So, we will be using the ESP32-WROOM-32E as a microcontroller for the charger's mainboard. [121] [122] [123] [124] [125]

3.2.2 SBC Selection

For this project, we will also have a single board computer (SBC) that provides the user interface for the charger and hosts the web-based interface. Thus, the SBC must be able to provide sufficient performance and network capability to host both interfaces, as well as communicate both with the microcontroller and over a network to a client. There are many different types of SBCs available on the market, each with various benefits, drawbacks, and features. As such, we must select one that properly fits our needs.

3.2.2.1 Raspberry Pi 5

The Raspberry Pi 5 is Raspberry Pi's latest single-board computer. It features a 64-bit quad-core 2.4GHz ARM Cortex-A76 processor paired with either 2GB, 4GB, 8GB, or 16GB of LPDDR4X RAM. Storage is handled by a microSD card slot. The Raspberry Pi 5 has 2.4GHz and 5GHz 802.11ac Wi-Fi capability, as well as Bluetooth 5.0. For wired networking, it has Gigabit Ethernet, and it offers both USB 2.0 and 3.0 ports. It also offers a 40-pin GPIO header. It is able to connect to displays with two micro-HDMI ports. It requires 5V and 5A for power, and also adds a power button. This is one of the more expensive options, with a 2GB RAM model starting at $50, while the 16GB model costs as much as $132. A 4GB version costs $66.

3.2.2.2 Raspberry Pi 4

The Raspberry Pi 4 is the previous generation of Raspberry Pi's main single board computer. It features a 64-bit quad-core 1.8GHz ARM Cortex-A72 processor paired with either 1GB, 2GB, 4GB, or 8GB of LPDDR4 RAM. Storage is also handled solely by a microSD card slot. Like the Raspberry Pi 5, the Pi 4 has 2.4GHz and 5GHz 802.11ac Wi-Fi with Bluetooth 5.0, as well as Gigabit Ethernet. It also has two micro-HDMI ports, and a 40-pin GPIO header. It requires 5V 3A for power, and lacks the power button of the Pi 5. Compared to the Pi 5, the Pi 4 still provides good performance and lower power consumption at a lower cost. A 4GB version provides better value than the Pi 5, at $55.

3.2.2.3 Raspberry Pi Zero 2 W

The Raspberry Pi Zero 2 W is the latest version of Raspberry Pi's smaller, lower-power, and lower-cost Zero series. It features a 64-bit quad-core 1GHz ARM Cortex-A53 processor, paired with only 512 MB of RAM. Video output is handled by a micro-HDMI port. Like the main series, a microSD card slot is provided for storage capability. The Pi Zero 2 W does not have an Ethernet port, but it supports 802.11n 2.4GHz Wi-Fi and Bluetooth 4.2, and has the same 40-pin GPIO header as the main series. However, the Pi Zero 2 W is available either with or without a pre-soldered GPIO header. The version with a pre-soldered GPIO header costs $19.80, and the version without costs $19.05. Overall, the Pi Zero 2 W is a well-priced option, but especially lacking in RAM. For hosting the user interface and web interface, this may work but could run into performance issues due to the lack of RAM.

3.2.2.4 Le Potato

The Le Potato AML-S905X-CC is a single board computer made by Libre Computer. It intends to be size-compatible with the Raspberry Pi 3, while offering a different feature set. Unlike the Pi 3, the Le Potato offers 2GB of DDR3 RAM at a $35 price point. It does, however, have a slower quad-core 1.416GHz ARM Cortex-A53 processor. Video output is handled by an HDMI port. It has the same GPIO header as a Raspberry Pi, but adds some extra peripherals in comparison. It offers a USB header, a separate UART header, a separate ADC header, and infrared receiver support. For storage, it offers a microSD card slot just like a Raspberry Pi, as well as an eMMC header to attach eMMC storage devices. However, it does not offer Wi-Fi connectivity, and the Ethernet port is only 100Mbps ("Fast Ethernet"), rather than 1Gbps ("Gigabit Ethernet") on the Pi. The current requirement of the system is not listed by Libre Computer, but full load power consumption is listed at 4W. Thus, it would only need around 1A with a 5V power supply, making it considerably less power hungry than a full sized Raspberry Pi.

3.2.2.5 NanoPi M5

The NanoPi M5 is a single board computer made by FriendlyElec. Unlike the Le Potato, it is not size-compatible with a Raspberry Pi. However, it offers a different feature set than any of the other single-board computers previously discussed. Instead of a quad-core processor, the NanoPi M5 has an octa-core 2.2GHz Rockchip RK3576. It has four ARM Cortex-A72 cores, and four ARM Cortex-A53 cores. It has a 30-pin GPIO header with SPI, UART, I2C, and PWM support. Video output is handled by an HDMI port, and two Gigabit Ethernet ports are available for networking. The NanoPi M5 does not have Wi-Fi or Bluetooth built-in, but it does offer an M.2 slot for an SDIO Wi-Fi module to be installed. There are three storage options available. An M.2 NVMe slot is available for a conventional SSD, a UFS connector is available for add-on UFS modules, and a microSD card slot exists just like the Raspberry Pi. It is available with either 4GB of LPDDR4X, 8GB or 16GB LPDDR5. It is priced very competitively at $55, less than any Raspberry Pi with 4GB of RAM. If we were performing especially performance or multitasking-intense scenarios, then the NanoPi M5 would be a good option

Table 3.2.2: Single Board Computer Comparison

Architecture	Raspberry Pi 5 Model B	Raspberry Pi 4 Model B	Raspberry Pi Zero 2 W	Le Potato AML-S905X-CC	NanoPi M5
Processor	ARM Cortex-A76	ARM Cortex-A72	ARM Cortex-A53	ARM Cortex-A53	Rockchip RK3576 (4x ARM Cortex-A72, 4x ARM Cortex-A53)
Cores	4	4	4	4	8
Bits	64-bit	64-bit	64-bit	64-bit	64-bit

Architecture	Raspberry Pi 5 Model B	Raspberry Pi 4 Model B	Raspberry Pi Zero 2 W	Le Potato AML-S905X-CC	NanoPi M5
Clock Speed (Max)	2.4GHz	1.8GHz	1 GHz	1.416GHz	2.2GHz
RAM	2GB, 4GB, 8GB, or 16GB LPDDR4X	1GB, 2GB, 4GB, or 8GB LPDDR4	512 MB	2GB DDR3	4GB LPDDR4X, 8GB or 16GB LPDDR5
Nonvolatile Storage	microSD card	microSD card	microSD card	microSD card, eMMC connector	microSD card, UFS connector, M.2 NVMe SSD
GPIO Count	40	40	40	40 + ADC, UART, USB headers	30
Communication Protocols	UART, SPI, I2C	UART, SPI, I2C	UART, SPI, I2C	UART, SPI, I2C	UART, SPI, I2C
Wi-Fi	Yes, 802.11ac 2.4GHz + 5GHz	Yes, 802.11ac 2.4GHz + 5GHz	Yes, 802.11n 2.4GHz	No	No, M.2 Wi-Fi module slot
Bluetooth	Yes, Bluetooth 5.0	Yes, Bluetooth 5.0	Yes, Bluetooth 4.2	No	
Cost	$50 (2GB) to $132 (16GB)	$38.50 (1GB) to $82.50 (8GB)	$19.80	$35	$55 (4GB)

Based on the above chart and our analysis of each single-board computer option, we will be choosing the Raspberry Pi 4 with 4GB of RAM. It provides plenty of performance for hosting the user interface with acceptable power draw, and is very well-documented. The Pi 5 with 4GB of RAM (or even a NanoPi M5) would be faster, but also is more expensive and draws considerably more power. [126] [127] [128] [129] [130]

3.2.3 Switching Regulator for USB Output

As discussed in section 3.1.4, A switching regulator in the form of a Buck Converter will be utilized as it will allow for a constant output voltage of 5V while also preventing the user from encountering overvoltage and overcurrent. In order to expedite this process. TI WEBENCH was

used to create regulator designs that would be suitable for our task. The constraints were that the regulator must use a Buck Converter at an input of 12V and regulate a 5V output at a 1A max. While the list is quite extensive, for the sake of brevity, we will be discussing 3 regulators that can be suitable for this project.

3.2.3.1 TPS56339

The TPS56339 includes two integrated switching MOSFETS, internal loop compensation and an internal soft-start. These devices come in the SOT-23 package allowing for a compact but powerful device capable of high efficiency. The regulator has an efficiency from 80% to 95% starting from 40mA to to 3 A which would fit well for the project specifications mentioned in 2.5. This regulator has Advanced Emulated Current Mode Control. With Current Mode Control (CMC), the inductor current is sensed using a resistor or transformer which does allow for fast response at the cost of electrical energy being dissipated by the sense resistor and a noisy signal. Emulated Current Mode Control eliminates the physical component utilizing the input voltage, output voltage, and the inductance. The controller can directly read these values and generate a current reading based on mathematical derivations.

3.2.3.2 : TPS563200 Synchronous Step Down Voltage

The TPS 563200 comes from the TPS56x200 series of regulators offered by Texas Instruments. The efficiency of the regulator is when the input voltage is 12V and output voltage at 5V ranges from 65% efficiency at 1mA up to 92% Efficiency at 3V. The regulator comes in a SOT-23 package offering features such as Adaptive On-Time Control & PWM Operation, Advanced Eco-Mode control, and soft start. Adaptive On-Time Control allows the regulator to quickly adapt to changes in load conditions such as current and more simple loop compensations allowing minimal oscillations occurring at the output terminal. This is in contrast to Conventional PWM where the switching period is only defined by the internal clock oscillator. A downside of this variability of the switching period does lead to more complex output filters being required to accommodate the various switching frequencies present in the circuit. Advanced Eco-Mode Control allows for the regulator to maintain high efficiency despite low current output, the feature allows for the regulator to avoid overheating and thermal shutdown as a result.

3.2.3.3 : LM2670 Simple Switcher Power Converter

The LM2670 is a power converter designed to minimize the required external components in order to maximize available space for other components and required circuits. The regulator is capable of maintaining an efficiency of 95% from 1A to 3A delivered at the output when the input is at 20V. The regulator has a SYNC pin that allows for an external clock to modify the switching frequency from the normal 225 kHz and 280 kHz operating range. When this mode is active, the current limit frequency foldback protective system is not active leading to potential vulnerabilities from output short-circuit conditions. At lower loads, the regulator efficiency will drop as WEBENCH gives an 85% efficiency for input 12V, output 5V, and max current 3A.

3.2.3.4 : MAX16939

This regulator is a 2.2MHz Step-Down Converter from Analog Devices that supports Adjustable Frequency, External Sync, and Power Good/ Reset Output . The regulator has an internal oscillator (FOSC) which allows for the switching frequency to be set. This feature leads to a more customisable design. The external sync feature on this regulator allows an external signal to be applied and the regulator to synchronize itself, this allows for skip mode operation to occur where current consumption will be reduced or into a fixed-frequency Forced Pulse Width Modulation (FPWM) depending on the PIN configuration that is set up. The Power Good signal allows for the regulator to send a signal indicating the device is delivering the desired voltage. This allows for proper sequencing without the intervention of a MCU, preventing damage from unstable voltage outputs, and monitors output stability to reduce the likelihood of unstable operation.

3.2.3.5 : Comparison of Regulators

Table 3.2.3.5: Comparison of Regulators

Regulator	TPS56339	TPS563200	LM2670	MAX16939
Brand	Texas Instruments	Texas Instruments	Texas Instruments	Analog Devices
Input Voltage	4.5 V - 24 V	4.5 V - 17 V	8V - 40 V	3.5 V - 36 V
Switching Frequency (Default)	500-kHz	650-kHz	260kHz	220-kHz - 2.2MHz
Price	$1.10	$1.02	$5.53	$2.38
Efficiency	95.2%	94.6%	85%	91%
Overvoltage Protection		Yes		
Thermal Shutdown	Yes	Yes	Yes	Yes
Overcurrent protection	Yes	Yes		
Sync with external clock			Yes	Yes

Based on the table above, we will be selecting the TPS563200 due to its many protective features and price point. [131] [132] [133] [134]

3.2.4 Switching Regulator for MCU

For the MCU, we will require an input of 3.3 volts as this is the amount required to turn. For the current. We will be supplying a max of 400mA as this is generally the highest current draw when performing startup procedures. Similar to the Switching Regulator for the USB output, TIWEBENCH was utilized to deploy quick prototyping of designs that would normally take weeks at a time. The following constraints are as follows: Input range 11.5 V to 12.5V with the output voltage being 3.3 V with max current at 0.4A.

3.2.4.1 TPS560430X3F

The TPS560430 is a synchronous step-down DC-DC converter that is able to accept an input of 4V to 36V and is able to produce 0.6 Amps. The regulator comes with PFM and forced PWM capabilities , a soft start feature, and compensation circuits internally allowing for a reduced reliance on external electrical components. For this specific part number, we will have access to the FPWM which also gives a fixed 3.3V output. Further features present in the IC are Fixed Frequency Peak Current Mode Control (FFPCMC) and Bootstrap. FFPCMC is a control mode where the switching frequency is held constant and utilizes the inductor's peak current in order to regulate the voltage. This is achieved by utilizing a current-sense circuit that monitors the current of the inductor. Once the peak is achieved, the controller turns off the MOSFET which allows the inductor to completely discharge which prompts the MOSFET to turn back on. Bootstrap voltage is implemented to fix the issue of a source terminal that isn't connected to ground. A common problem that occurs is that when the high-side MOSFET turns on, the switch node's voltage rises close to Vin. However, in order to turn on, there must be a higher voltage relative to ground. Using a diode and capacitor, the capacitor charges up when the low-side MOSFET is turned on. Once the MOSFET switches, the capacitor starts to discharge which allows the potential difference to be large enough where the gate driver is now active. The down side to this technique is that the capacitor will need to charge up.

3.2.4.2 TPS561208

The TPS561208 belongs to a family of synchronous step-down converters. The design for these IC minimizes the external component counts while achieving a low standby current. This TPS561208 also offers a D-CAP2 control scheme with a continuous current mode. In continuous current mode, the inductor current never goes to 0 meaning the inductor is either always storing or releasing energy throughout the mode of operation. This process leads to the output capacitor not needing to handle full current pulses which puts less stress on the electrical components. D-CAP2 is a proprietary control scheme by TI that allows for adaptive on-time control with an internal circuit setup for a pseudo-fixed frequency composed of low-ESR and ceramic output capacitors. This configuration allows for stability of the output signal with little ripple. The adaptive on time control is called this due to a one shot timer duration being determined by the input and output voltage to maintain a pseudo-fixed frequency over the input voltage range. Once the timer is reset, the high-side MOSFET is turned back on. An internal ramp is added to the reference voltage to simulate a ripple at the output which bypasses the need to create the ripple due to ESR.

3.2.4.3 TPS54202

The TPS5402 is a synchronous buck converter that has two integrated switching FETS, internal loop compensation, and an internal soft start. The TPS5402 comes with features like Fixed-Frequency PWM control, Pulse Skip Mode, and Bootstrap Voltage. For the fixed frequency PMW control, the output voltage is compared through an external resistor to an internal voltage reference generated by the error amplifier. This output is then compared to the current of the high-side power. Once these numbers match, the high-side turns off while the low-side turns on. The voltage of the error amplifier is proportional to the output current . In order to ensure current limiting, the voltage on the error amplifier is clamps to a max. There is a secondary clamp present in the minimum level to ensure improved transient- response performance. Pulse Skip Mode is activated when the peak inductor current is lower than .3A. When this mode is activated, the error amplifier output voltage is clamped to prevent the high-side MOSFET from switching, the only way out is for the inductor current to rise beyond 0.3 A. This mode is well suited for light loads and is designed to increase efficiency at those loads. Bootstrap voltage as mentioned in 3.2.4.2 is designed to ensure the MOSFET is fully turned on.

Table 3.2.4.3: Switching Regulator Comparison for MCU

Components	TPS560430X4F	TPS561208	TPS54202
Input-Voltage range	4V to 36V	4.5V to 17V	4.5V to 28V
Switching Frequency	1.1 MHz	580-kHz	500-kHz
Efficiency	85.4%	89.3%	90.4%
BOM Cost	$0.94	$0.61	$0.81
Boot Voltage	Yes	Yes	No
Protection features	OCP UVLO TSD	OCP UVLO TSD	OCP OVP TSD
SYNC Option	No	No	Yes
Soft-Start Time (ms)	1.0	1.3	Adjustable (external cap)

Based on the table above, we will be selecting the TPS561208 due to having a boot voltage capability, comparable efficiency to the TPS54202, and low price point. [135] [136] [137]

3.2.5 Current Switch

An IC load switch is a power-gating device that uses a MOSFET that is integrated with control circuitry to connect or disconnect a load from its supply rail. These devices replace discrete MOSFET solutions by integrating features such as controlled slew rate, current limiting, thermal shutdown, undervoltage lockout, and reverse-current blocking. The integrated control circuitry

improves reliability, reduces the PCB size and increases design consistency when compared to manually implementing the protection circuitry for a discrete MOSFET device. These load switches are selected based on metrics such as the on-resistance, Maximum continuous current, turn on and turn off characteristics, and available protection features. Below is a comparison of three potential IC load-switch options.

3.2.5.1 TPS22918

The TPS22918 is a compact load-switch IC built around a low on-resistance N-channel MOSFET. It is optimized for efficient power gating in low voltage systems. It supports continuous current of up to 2A allowing it to be suitable for moderate power loads. A key characteristic of the device is the configurable slew rate control which allows designers to adjust the turn on behavior ensuring a minimal inrush current or reducing signal noise. The TPS22918 also has a Quick Output Discharge (QOD) feature that rapidly pulls the output node to ground when the switch is disabled. This prevents unintended floating nodes and ensures predictable shutoff behavior. Its low quiescent current and compact footprint makes this component a strong candidate for designs requiring tight control of power distribution with minimal board real estate.

3.2.5.2 LS0501EVT223

The LS0504EVT223 is an integrated load switching device designed with additional protection circuitry in the internals compared to MOSFET-only switches. It includes overcurrent protection which limits the output current during abnormal loading conditions. This reduces the risk of damage to downstream components. The device also features over voltage protection preventing excessive input voltage from reaching the load. An internal soft-start mechanism moderates the output ramp during activation which prevents sudden current draw and ensures smoother transitions. These protections are built into the IC. The LSO504EVT223 is suitable for applications where fault tolerance and long term reliability are important considerations.

3.2.5.3 DML3012LDC

The DML3012 Load switch offered by Diodes incorporated is a performance oriented load switch that integrates a MOSFET with a very low on-resistance. This minimizes conduction losses during operation. The result being a highly efficient device involving higher continuous currents. The device incorporates over current limiting, thermal shutdown, and soft start control which offers protection against short circuits, overheating, and startup surges. The robust design ensures consistent voltage delivery while protecting both the load and the switch from stress conditions. Due to these strengths, the DML3012LDC is well united for demanding power-gating applications.

Table 3.2.5.3: Current Switch Part Comparison

Feature/IC	TPS22918	LS0501EVT223	DML3012LDC
Switch Type	N-channel MOSFET	Integrated MOSFET	Integrated MOSFET

Package	SOT-23-6	SOT-23	SOT-23
Max Continuous Current	2A	2-4A	~3A
Input Voltage Range	0-5.5V	2.7-6V	1.2-6V
QOD	Yes	No	No
Internal Soft-Start	No	Yes	Yes
Overcurrent/Overvoltage	No	Yes	Yes

Based on the options, we chose to go with TPS22918 as the most suitable option, while both LS0504EVT223 and the DML3012 offer overvoltage/overcurrent protections, these already exist on the regulator thus would prove quite redundant in the design of this circuit. Both of these devices lack key features in fast switching

In contrast, the TPS22918 is well suited for fast and controlled power delivery. The critical feature of the IC load is the Quick Output discharge which ensures the output rail rapidly discharges when the switch is turned off. This prevents floating voltages and ensures predictable behavior for all connected devices. [138] [139] [140]

3.2.6 Power Supply

Based on our system requirements, the total power consumption will come from four USB charging ports with additional draw coming from the ESP32, INA260 Current sense IC, TPS22918 Load Switch IC, and status indicators. Each USB port is designed to supply up to 2A at 5V, giving a maximum continuous power of 10W per port. When factoring in all four ports and the low-voltage control electronics, the total output power of the system is estimated to range between 6.25W during light loads and up to 42.5 W when all ports are loaded to their maximum. This range drives the selection of both the main power and individual voltage regulators mentioned previously in this chapter.

3.2.6.1 Power Distribution

Table 3.2.6.1 Power Distribution Table

Component	Supply Voltage (V)	Estimated Current Draw (Max)
ESP32	3.3V	240mA
INA260 (X4)	3.3V	4-5mA
I2C pull-ups, GPIO pull-ups	3.3V	5mA

Status/Indicator LEDs (x4)	3.3V	~35-40 mA
3.3V Regulator	12V	~300mA
TPS22918 Load Switch (x4)	5V	~1uA
USB Port Load (x4)	5V	8A
5V Regulator (x4)	12V	8A
USB-UART/Debug Interface	3.3V	10-20 mA

From this table, the 3.3V rail requires roughly 310 mA while each 5V rail must support up to 2 A independently. In order to facilitate this, using one 3.3V Regulator for the shared control electronics and four 5V regulators for each USB port to allow for current limiting at each port and fault isolation.

3.2.6.2 Power Supply

The charger is intended to be a mains powered device rather than fully battery portable. This means that the device will be supplied by an external DC supply rather than a battery pack. We considered three main options:

1. Single 12V AC-DC wall adapter that will feed on-board buck converters.
2. Single 5V high-current brick feeding the ports directly and local 3.3 regulator
3. Integrated AC-DC PSU on the PCB

Table 3.2.6.2 Power Distribution Comparison

Power Source	Output	Integration Complexity	Safety / Compliance	Cost	Best for	Drawbacks
12 V AC-DC wall adapter	12V @ 4-6 A	Low - simple DC jack + on board DC - DC	High - adapter is pre-certified	Moderate	Modular Systems with multiple DC - DC rails	Requires several buck converters on board
5V High Current Supply	5V @ 10-15 A	Low single 5V rail + 3.3 V regulator	High - pre certified supplies available	Moderate - High	Simple USB hubs, no per port 5V regulation	Harder to isolate ports; no per port regulators
On-Board Off-line SMPS	120 VAC - 5/12 V	High - isolation, creepage,	Designer must meet IEC/UL	Highest (design + safety)	Mass produced commercial	Not suitable for student

		EMI design	requirements		chargers	prototype

For our prototype, we selected a 12V off-the-shelf AC-DC wall adaptor. This method provides a safe isolated DC rail with sufficient voltage headroom for efficient buck conversion to both 5V and 3.3V, while leveraging the adapter's existing safety certifications. This method also allows us to be in compliance with IEC-61010-1 and avoid having a high-voltage main directly interact with the PCB.

3.2.6.3 AC-DC Wall Adaptor

Choosing an AC to DC wall adaptor is crucial for our device as it needs to supply enough Watts to ensure working order. The adaptor would need to be able to convert 120 V AC mains voltage to 12 V DC that will then be sent downstream to the buck regulators. Choosing the appropriate adapter would require that at minimum 40 Watts can be supplied to the USB output plus any overhead that will be for regulator losses and as well as for the control electronics. An under-rated adaptor can lead to voltage drop, system instability, and degraded charging performance. Furthermore, a high quality AC-DC adaptor would provide all the essential safety features to ensure no harm comes to the user and/or device.

An additional benefit of choosing a 12V supply is its efficiency as the main bus voltage. Due to the higher voltage, there will be lower current drawn from the adapter compared to the 5V high -current source which lowers the conductors losses. This also reduces any voltage drop across PCB traces while minimizing heating. This will overall improve the efficiency and reliability of the system. This will prove key during the heavy load phase when all USB ports are being used. By choosing 60 Watts, this will provide more than enough power to run during heavy operation or quick changes in the system such as switching or removal of load devices. By opting for this design choice. The final device will have a more simple power routing while reducing further considerations down the road

Table 3.2.6.3 AC-DC Wall Adaptor Comparison

Adapter Model	Output Voltage	Current Rating	Total Power	Certifications
NEEWER AC 100-240V to DC12	12V	5A	60W	CE,FCC
Mean Well GST60A12	12V	5A	60W	UL,CE,IEC
QualTEK QFWB - 60- 12 - USO1	12V	5A	60W	UL,FCC

After evaluating multiple 12V AC-DC wall adapters in the 60W range, the Mean Well GST60A12 was selected as the primary power source for the charging system. This adapter is

able to provide a regulated 12V at up to 5A which would deliver a max of 60W. This aligned well with the system's calculated power requirement that is approximately 50-60 W for four independently regulated 5V outputs and 3.3V logic rail. This adapter also comes fully certified to UL, CE, and IEC safety standards. This ensures reliable isolation, electrical protection, and compliance with international mains-voltage regulations. Compared to lower-cost generic adapters, the Mean Well Unit offers a superior build quality, longevity, and electrical noise performance. [149] [150] [151]

3.3 Full Stack Development Technologies

A full stack refers to the complete collection of technologies, tools, and software layers required to build and operate a web application from end to end. It encompasses every component involved in delivering functionality to the user, beginning with the frontend, which defines the visual interface and interactive elements that users see on their screens. This includes layouts, buttons, graphs, controls, and all client-side logic responsible for rendering information and capturing user input. Behind the frontend lies the backend, which performs the application's internal operations such as processing commands, executing system logic, managing hardware communication, and exposing APIs that allow client devices to interact with the system. Finally, the database layer provides persistent data storage, ensuring that configuration values, historical information, and other critical system records are maintained reliably across reboots and user sessions. These three layers work together to create a seamless, responsive, and maintainable software system capable of supporting both local and remote interactions with the smart charger.

3.3.1 Local UI Frontend

The Local UI Frontend serves as the on-device interface that runs directly on the Raspberry Pi and provides users with immediate, localized access to the smart charger's controls and real-time charging information. This interface enables full operation of the charger without requiring an external device such as a phone or laptop, and it remains fully functional even when the system is disconnected from a network. By integrating the user interface directly into the SBC, the smart charger behaves like a standalone appliance. A user would be able to walk up to the device, view live voltage, current, and power readings for each USB port, and adjust charging settings instantly through the touchscreen. The Local UI is designed with simplicity, responsiveness, and clarity in mind, ensuring that essential actions can be performed quickly and intuitively. This approach greatly improves the usability and accessibility of the smart charger, allowing it to operate effectively in environments where Wi-Fi may be unreliable, optional, or deliberately disabled for security reasons.

3.3.1.1 Tkinter

Tkinter is a standard Python library that provides tools and widgets to create Graphical User Interfaces (GUIs). Tkinter is usually included by default when installing Python and comes preinstalled on Raspberry Pi OS, which makes it accessible for developers without the need of additional installations or software packages. The name Tkinter is short for "Tk interface" and refers to the GUI toolkit that Tkinter is based on. There are a variety of use cases for Tkinter

such as creating dialog boxes, building a GUI for a web application, implementing a GUI to a command-line tool, creating custom widgets, prototyping a GUI, and lightweight embedded interfaces. [35]

There are many advantages to using Tkinter as the Raspberry Pi's Local UI Frontend, such as its amazing performance. Tkinter is very lightweight and efficient for embedded hardware, and it would not require any additional imports or installs since it's already built into the Raspberry Pi's OS. Another advantage is that Tkinter would have a straightforward implementation since the controls for the smart charger are relatively simple. There are a few disadvantages to Tkinter, with one of them being that there are limitations to customizing the GUI. Tkinter only provides a small number of basic widgets and lacks modern tools. Tkinter also does not have a native look or feel, since it uses the Tk toolkit and does not use native operating system components. Another disadvantage would be that implementing Tkinter would mean that there would be two separate UIs for the local SBC and the web application. Despite its drawbacks, Tkinter would be the best fit for the smart charger due to its simplicity, performance, and its efficiency for embedded hardware. The use of Tkinter will also ensure that users can control the charger independently from any network connectivity.

3.3.1.2 PyQt

PyQt is another GUI module that is built by integrating the Qt C++ cross-platform framework with the Python programming language. Qt encompasses a wide range of functionalities, which offers abstraction for Unicode, SQL, SVG, and XML. Qt also has a built-in web browser and a sophisticated service system, which contains an extensive collection of GUI widgets. Qt also works with Qt Designer, which is a tool that allows developers to create GUIs. Qt Designer helps Qt convert the GUI design into Python code. [36]

The benefits of using PyQt is that it offers more flexibility in its UI design and it supports a native-like feel with more modern customization options. PyQt also includes robust tools for event handling and multi-page layout. The disadvantages to using PyQt is that its install and setup process is more complex, and it would require more development time due to its broad feature set. PyQt also has higher resource usage, which could impact its overall performance. PyQt is a good option, but its toolkit exceeds the needs of the smart charger, which makes it a less efficient choice for this project.

3.3.1.3 Kiosk Mode

Kiosk Mode allows the web application frontend to be displayed locally on the Raspberry Pi in a full-screen browser environment. The user can interact with the smart charger through a browser that automatically starts and hides standard browser elements to create the appearance of a dedicated interface. [37]

The benefits to this approach is that the web application's UI can be reused instead of needing to develop and update two separate UIs. Kiosk mode also supports real-time data through the use of REST API or WebSockets. The disadvantage to this approach is that it introduces additional

resource overhead since the web browser must remain active to operate the interface. There is also a risk of instability if the network or browser services encounter errors.

Table 3.3.1.3: Local UI Frontend Comparison

Frontend	Tkinter	Kiosk Mode	PyQt
Use Case	Ideal for lightweight, embedded graphical interfaces	Best reusing the same web interface locally in a fullscreen browser environment	Suitable for feature-rich desktop-style applications
Real-Time Support	Moderate, supports periodic updates	Strong, can display live data through WebSockets	Strong, supports threaded upbates
Performance	Excellent, minimal CPU and memory usage	Moderate, depends on browser performance	Good, faster than web-based solutions but heavier than Tkinter
Scalability	Limited, suitable for simple GUI	High, it uses the same web technology as my laptop	Moderate. Increases memory footprint
Complexity	Simple	Moderate	High

3.3.2 Web Application Frontend

A full stack refers to the complete collection of technologies, tools, and software layers required to build and operate a web application from end to end. It encompasses every component involved in delivering functionality to the user, beginning with the frontend, which defines the visual interface and interactive elements that users see on their screens. This includes layouts, buttons, graphs, controls, and all client-side logic responsible for rendering information and capturing user input. Behind the frontend lies the backend, which performs the application's internal operations such as processing commands, executing system logic, managing hardware communication, and exposing APIs that allow client devices to interact with the system. Finally, the database layer provides persistent data storage, ensuring that configuration values, historical information, and other critical system records are maintained reliably across reboots and user sessions. These three layers work together to create a seamless, responsive, and maintainable software system capable of supporting both local and remote interactions with the smart charger.

3.3.2.1 LAMP Stack Frontend

In a traditional LAMP stack, the frontend is primarily generated on the server using PHP, which constructs the HTML, CSS, and JavaScript before sending the completed page to the client's browser. Because the browser receives a fully rendered page, the frontend relies heavily on server-driven updates, meaning most interactions require either a full page reload or an AJAX request to fetch new data. This server-rendered model works well for applications that focus on static content, data submission, and basic CRUD operations, where real-time responsiveness is not essential. For simpler dashboards or form-based systems, this approach remains reliable and easy to maintain, as the logic is tightly integrated with the backend and does not require specialized frontend frameworks or complex build pipelines.

The main advantages to using this frontend is that it's simple and lightweight, it does not require complex tools or frontend frameworks. Another advantage is that PHP works closely with the backend to deliver dynamic content without much effort. The disadvantage to using a PHP-based frontend is that PHP frontends are considered to be outdated, especially in comparison to React or Vue. PHP-based frontend also struggles with real-time support, as it relies on polling and is not event-driven. PHP frontends also do not scale well for complex dashboards or highly interactive user interfaces. While these are not issues for basic applications, the lack of real-time data transfer and a limited user interface would make this approach a poor fit for our project. [38]

3.3.2.2 MERN Stack Frontend

In contrast to traditional server-rendered architectures, the MERN stack uses React as its frontend component, enabling highly interactive and dynamic user interfaces. React functions as a modern JavaScript library that builds Single Page Applications (SPAs), where the entire application loads once and all state changes and interface updates occur directly in the browser. This allows the frontend to react instantly to changes in data without requiring page reloads or excessive server communication. Because React efficiently updates only the parts of the interface that change, it is especially well-suited for displaying continuously updating values such as voltage, current, and power readings. This event-driven, real-time-capable structure aligns well with projects that demand live feedback and constant interface responsiveness.

The main advantage to using React is that it offers fast client-side updates without the need to reload pages, which is perfect for our type of project, which involves real-time user interaction. Another advantage to using React is that it offers the capability to build an interactive user interface, which is critical for our project to function as intended. React allows the capability for JavaScript to be leveraged on both the client side and the server side, which enables fast service rendering and an overall better user experience. Some disadvantages to using React would be that it has a steeper learning curve, since it requires knowledge of JavaScript. It also can struggle with search engine optimization (SEO) due to its dynamic nature. There are ways to mitigate these issues, such as implementing frameworks like Next, but other frontends like PHP would be inherently better since it delivers static content. Overall, the advantages to using React significantly outweigh the disadvantages, which would make it the ideal choice for our project. [39]

Table 3.3.2: Web Application Frontend Comparison

Frontend	PHP and HTML/CSS/JavaScript	React
Use Case	Best for content-heavy websites and server-side logic.	Best for dynamic user interfaces and real-time updates.
Real-Time Support	Weak, relies on polling	Strong, updates content dynamically
Performance	Dynamic content rendered on the server	Dynamic content rendered on the client
Scalability	Effective for traditional websites	Strong for complex user interfaces
Complexity	Simple to set up, easy for beginners	Complex build process, requires JavaScript knowledge

3.3.3 Backend

The backend is the portion of the system responsible for handling all server-side logic, data processing, and communication with the database. It manages the functionality of the web application by receiving requests from the frontend, performing the necessary operations, and returning the appropriate responses. Whenever the user interacts with the interface, such as submitting a configuration change, requesting system information, or loading a page, the frontend sends that request to the backend for processing. The backend typically completes this process by interacting with the application's APIs and database, ensuring that data is stored, retrieved, and updated correctly. Because the backend acts as the “brain” of the system, it plays a critical role in coordinating the logic that makes the application work.

3.3.3.1 LAMP Stack Backend

In a traditional LAMP stack, the backend is built using the Apache web server paired with PHP as the main scripting language. Apache handles incoming client requests through HTTP and is responsible for serving static files as well as routing dynamic page requests. PHP runs on the server and generates dynamic responses by interacting with the database, building HTML templates, and executing the application’s logic before sending the final output back to the browser. Because Apache and PHP work closely together, they form a simple and reliable backend architecture that is widely used for content-driven websites and applications that rely heavily on server-side rendering. [40]

An advantage to using Apache and PHP as a backend would be its reliability. Apache is a widely used web server that excels at delivering reliable server-rendered content. Another advantage is

that the setup for this backend is very beginner-friendly and straightforward. A major disadvantage to using this backend structure is that it struggles with real-time content. While this backend is excellent at delivering static content, it needs to reload the page if there has been a change in content, which would cause significant issues with our project. One of the objectives of the website is to be able to display the stats of each charging port in real-time, and needing to refresh the page to update it would be troublesome.

3.3.3.2 MERN Stack Backend

In a MERN stack architecture, the backend is powered by Node.js, which enables JavaScript to run on the server, and Express, a lightweight framework that provides routing and middleware functionality. Node.js uses a non-blocking, event-driven execution model, allowing it to handle many tasks simultaneously without the overhead of traditional multithreading. Express builds on top of Node.js by simplifying the creation of APIs and backend logic, making it easier to define routes, manage requests, and structure the server's functionality. Together, Node.js and Express form a highly flexible and efficient backend environment that is well-suited for applications requiring real-time updates and fast communication with sensors or databases.

An advantage to using Node.js and Express as a backend would be that it is event-driven and lightweight, which would be perfect for our type of project. Node.js can handle multiple requests simultaneously, which would make it effective for our project, which will rely on constantly changing state of charging ports and updates from the sensors. Another advantage is that Express would allow for the project to implement a simple and flexible framework for routing and creating APIs. One of the few downsides of choosing this backend is that Express does have a lengthy setup process. Another disadvantage would be that it shares the same complexity with its frontend counterpart, in that it has a complex build process and requires knowledge of JavaScript. The advantages greatly outweigh the disadvantages, which is why using this backend structure would be best for our project. [41]

Table 3.3.3: Backend Comparison

Backend	PHP and Apache	Node.js and Express
Use Case	Request-Response, blocking model	Event-driven, non-blocking model
Real-Time Support	Limited, requires constant refresh	Strong, works best for real-time monitoring
Performance	Reliable, but slower under high concurrency	Lightweight and efficient
Scalability	Scales well for dynamic and concurrent applications	Scales well for static or CRUD-based applications

Complexity	Complex build process, requires JavaScript knowledge	Simple to set up, easy for beginners

3.3.4 Database

Most full stack applications depend heavily on a database to organize, store, and manage important information. Databases act as digital repositories that allow applications to access, update, and retrieve data whenever needed, which makes them essential for maintaining system state and user-specific information. Our project also requires a database so that the smart charger can store configuration values, such as the user's preferred current thresholds and timer settings. Without a reliable database, the system would not be able to retain these values after a reboot or ensure that both the SBC and microcontroller operate with the same configuration.

3.3.4.1 MySQL

MySQL is one of the most widely used open source relational databases and is a common choice in traditional LAMP stack applications. It stores data in structured tables that can be linked together, and it relies on Structured Query Language, or SQL, to define, manipulate, and query information. MySQL is well suited for both client server environments and embedded systems, and its multithreaded server allows it to efficiently manage multiple database requests at once. This makes MySQL a strong option for applications that require structured data and reliable long term storage. [42]

Some advantages of MySQL are its stability and scalability. It is capable of managing large amounts of data and works perfectly with PHP for dynamic websites. This sounds like it would be perfect, but this is also a major disadvantage. The data our database would need to store would be very resource-heavy, and we need to store the database within the Raspberry Pi. Another disadvantage would be that the configuration for MySQL would be more complex than other options.

3.3.4.2 MongoDB

MongoDB is a core database technology in the MERN stack and is classified as a NoSQL database, meaning it uses flexible BSON based documents rather than fixed relational tables. This document based structure allows developers to define and adjust data models without rigid schemas, which makes MongoDB ideal for handling unstructured or rapidly changing data. The setup process for MongoDB is simple, and developers often use an Object Data Modeling tool to add structure and validation on top of its flexible design. MongoDB works very well with Node.js applications and provides a scalable solution for large or distributed systems. [43]

The advantage of using MongoDB is that its flexibility and scalability are perfect for handling unstructured data. It works excellently with Node.js and makes it the best option for large-scale applications. Unfortunately, MongoDB is also very resource-heavy, so the same issues with

using MySQL would still be present. MongoDB would be perfect if the data were to be stored in the cloud, but our project would need the data to be locally stored in the Raspberry Pi.

3.3.4.3 SQLite

SQLite is a lightweight SQL database engine implemented as a compact C library that operates without a dedicated server process. Instead of running a separate database server, SQLite writes all tables, triggers, and metadata into a single local disk file that the application can read and modify directly. This design allows SQLite to be extremely fast and resource efficient, which makes it a strong choice for embedded systems, portable applications, and devices that need reliable data storage without the overhead of a separate database service. [44]

The biggest advantage to SQLite is that it is resource-efficient and portable. SQLite would work perfectly for logging device data, storing user configurations, and utilizing minimal overhead. The only significant downside to SQLite is that it has limited scalability. Since it stores information locally, it would not have the capacity to support as much as MySQL or MongoDB. It would perform poorly if too much data is being stored, but this just means that our project would have to be optimized to work well with SQLite. This database is the best fit for our demands.

Table 3.3.4: Database Comparison

Database	MySQL	MongoDB	SQLite
Type	Relational	NoSQL	Relational
Use Case	Best for structured, large-scale applications	Best for flexible, large cloud applications	Best for lightweight, embedded systems
Real-Time Support	Poor real-time support	Excellent for real-time, large, unstructured data	Handles small-scale real-time data efficiently
Performance	Reliable, heavy on resources	Reliable, heavy on resources	Very lightweight and efficient
Scalability	Scales well for large workloads	Scales across distributed systems	Limited scalability, not ideal for heavy concurrency
Complexity	Requires installation, configuration, and management	Straightforward setup, no configuration required	Simple setup, but requires running database server

3.3.5 Operating System

An operating system is the core software that manages a system's hardware, processes, and resources. It acts as the bridge between applications and the underlying hardware by allocating CPU time, handling memory usage, managing storage devices, and coordinating input and output operations. Every program that runs on a device depends on the operating system to schedule tasks and ensure that system resources are used efficiently. There are many operating systems that could potentially be used for our project, each offering different levels of performance, flexibility, and hardware compatibility. This section will focus on the two options that are most suitable for our design and evaluate which one provides the best support for the smart charger.

3.3.5.1 Linux

One of the most popular operating systems worldwide, Linux is an open source operating system created by Linus Torvalds in 1991. Linux was designed to be similar to UNIX, but has evolved to run on a variety of hardware. Linux operating systems use the Linux kernel, which manages hardware resources, and a variety of software packages that form the complete operating system. [45]

One advantage of using Linux as our operating system is that it is versatile. Linux powers a wide range of technology from small, data gathering devices to complex cloud applications. Linux also has a worldwide community, so there are plenty of resources available to work on its implementation into our project. The drawback to using Linux is that it is a heavy, server-oriented setup. This would be very excessive for what we would need an operating system to do for our project. Its large-scale application would not be necessary for this type of embedded project.

3.3.5.2 Raspberry Pi OS

Raspberry Pi OS a free, open-source Debian Linux-based operating system developed for use on Pi boards.The first version, originally named Raspbian, released in 2013 and was an independent project by programmers Mike Thompson and Peter Green. Over time, the Raspbian got rebranded to Raspberry Pi OS and evolved into a 64-bit operating system and can handle up to 8GB of RAM. [46]

The biggest advantage to using Raspberry Pi OS is that it was built specifically to be optimized on the Raspberry Pi, which is the SBC that we're using for our project. It provides all the core elements of Linux while also being a streamlined and resource-efficient operating system for the Raspberry Pi. This operating system is ideal for managing the different processes that we'll need for our project, such as the user interface, database storage, the connectivity between the MCU, and the data for the web interface. The only notable disadvantage would be that it does not have the capability to host a large, cloud-based server, but our project wouldn't require that as much since it will be hosted locally. The advantages of utilizing Raspberry Pi OS far outweigh its disadvantages.

Table 3.3.5: Operating System Comparison

Operating System	Raspberry Pi OS	Linux
Purpose	Designed specifically for Raspberry Pi	General purpose OS for desktops, servers, etc.
Use Case	Best fit for local IoT projects	Best fit for large, hosted web environments
Resource Usage	Lightweight and efficient for low-power hardware	Very resource-demanding, best for high-performance servers
Performance	Excellent for SBCs, such as Raspberry Pi	High performance for large-scale systems
Scalability	Limited to local or small-scale embedded systems	Scales up to large enterprise and cloud networking
Complexity	Simple setup with built-in Raspberry Pi tools	Requires moderate configuration
Networking	Ideal for local networks (UART, MQTT)	Ideal for large-scale web hosting, cloud networking

3.4 Communication Protocols

3.4.1 Communication Between MCU and SBC

One of the more important parts of our project is the ability for the microcontroller unit (MCU) to communicate with the single-board computer (SBC). There are various different options we have, using both wired and wireless connections.

3.4.1.1 UART

Universal Asynchronous Receiver/Transmitter (UART) is a basic wired asynchronous communication protocol used for many years in various types of devices. It is still in use for many types of embedded applications. UART allows two devices to communicate using two pins. Each device has a Tx (transmit) and an Rx (receive) pin. One device's Tx pin is connected to the other device's Rx pin, and vice versa. UART has "packets" of data that have three to four sections, depending on the implementation. There is one start bit, 5 to 9 data bits, one (optional) parity bit, and one to two stop bits. UART is normally high (1), so the start bit is set to low (0). Then, the data bits are sent. The parity bit is used optionally to check the validity of the data bits. If the data bits total an even sum, the parity bit should be 0. If they total an odd sum, the parity

bit should be 1. This is not foolproof, as if enough bits are flipped a sum can still stay even or odd. Then, the stop bits are set from low (0) to high (1). Unlike many other communication protocols, UART does not need a clock signal. Instead, it uses a "baud rate" which is based on the frequency of the bits being sent. A higher baud rate means a higher rate of data transmission. There is a tolerance of around 10%, which allows the baud rates of two devices to only slightly differ while still being able to transmit and receive. For most devices, baud rate is configurable. [47] Overall, UART is a very simple and very compatible method of transmitting data, because it only requires two wires and no clock signal.

3.4.1.2 I2C

Inter-Integrated Circuit (I2C) is a commonly used wired synchronous communication protocol. Unlike UART, it does explicitly require a clock. Like UART, I2C requires only two pins to communicate. However, I2C is a bus that works with more than two devices. I2C has "master" devices and "slave" devices. The clock is generated by the master. There are two I2C pins, the SDA (serial data) line, and the SCL (serial clock) line, which carries the clock signal. I2C data is organized into "messages", which are divided into several pieces. The first part is the start condition. SDA is normally high (1), so it is switched to low while the clock line is high. Then, the address frame is sent, which contains a 7-bit or 10-bit address of the desired "slave" device, depending on implementation. A read/write bit follows, where high (1) is read and low (0) is write. Then, an ACK (acknowledge bit) follows, where SDA is set to low (0) if acknowledged. After the acknowledge bit, 8-bit data frames containing the data are sent, with an ACK bit after each set of 8 bits. When the transmission is complete, a stop condition occurs, where SDA is set from low (0) to high (1) while the clock line is high. I2C is a more complicated method of wired communication, but is used in many applications. It is possible to have both multiple "master" devices and multiple "slave" devices, making it versatile. [48]

3.4.1.3 SPI

Serial Peripheral Interface (SPI) is another commonly used wired synchronous communication protocol. Unlike UART and I2C, SPI requires at least four pins to communicate: MOSI (Master Out Slave Input), MISO (Master Input Slave Output). SCLK (clock signal), and SS/CS (Slave Select/Chip Select). Multiple "slave" devices are possible, but there can be only one "master" device. It is also possible for a master to have multiple slave select pins for multiple slaves, but the slave devices can also be daisy-chained in the event that there is only one slave select pin. The clock signal comes from the master. Transmission is started once there is a clock signal and when SS/CS is set to low (0). Data transmission then occurs from master to slave at one bit per clock cycle via the MOSI pin, and then from slave to master via the MISO pin. Unlike UART and I2C, data does not have to be divided into frames or packets. With SPI, there is no limit on the amount of data that can be continuously transmitted. SPI is also significantly faster than both UART and I2C, making it a good choice for large amounts of data. However, there is no parity bit or ACK bit like UART or I2C, so a bit flip or loss of data in transmission is more difficult to detect. [49] Overall, SPI is another good potential option for our use case.

3.4.1.4 MQTT

Moving into wireless communication protocols, MQTT is a network messaging protocol that is commonly used by many Internet of Things (IoT) devices. This means that each device needs a network (i.e. Wi-Fi or Ethernet) connection. It requires a “broker”, which acts as an interface between multiple clients that can send each other messages. Clients can “publish” messages, and other clients can “subscribe” to messages. The broker’s responsibility is to receive the published messages and send them to any client who has “subscribed” to the message. Brokers can be on the same local network, or in the cloud. [50] Either way, to provide simple communication between our microcontroller and single-board computer, we would also need a broker, adding extra complexity to the project.

3.4.1.5 Bluetooth

Bluetooth is another wireless communication protocol, which uses its own network. Each device has built in 2.4GHz radios which can operate at various frequencies around 2.4GHz. Over many years, Bluetooth has evolved and added features. The most recent revision is Bluetooth 6.1, although many current devices do not support it. The speed of data transfers has improved, and Bluetooth Low Energy (LE) was introduced. [51] Bluetooth Low Energy allows for much lower idle power consumption when data is not being transmitted. However, the bandwidth is decreased compared to regular Bluetooth. Thus, there is a trade-off between regular Bluetooth and Bluetooth LE. Bluetooth allows for higher bandwidth, which makes it a better choice for continuous usage. However, it consumes around 100 times more power than Bluetooth LE. [52] This means Bluetooth LE is a better choice for infrequent data transmission. If we implemented Bluetooth for our application, we could test both Bluetooth and Bluetooth LE in order to determine which one to use. The below table showcases each communication method we have considered.

Table 3.4.1: MCU to SBC Communication Comparison

Architecture	UART	I2C	SPI	MQTT	Bluetooth
Wired or Wireless	Wired	Wired	Wired	Wireless	Wireless
Minimum number of devices needed	2	2	2	3	2
Synchronous or Asynchronous	Asynchronous	Synchronous	Synchronous	Asynchronous	Either
Maximum Speed	115200 baud (115.2	5 Mbps	10 Mbps	Network dependent	3 Mbps, or 1 Mbps (LE)

	Kbps)				
Pins required (Wired)	2	2	4	N/A	N/A

Based on the above chart, we will be choosing UART. The scope of data transferred and received on each side is small enough that the speed limitation should not cause a problem. Additionally, since it is arguably the simplest method of communication discussed here, debugging it should be simpler as well in the event of a problem.

3.4.2 Communication Between SBC and Web App

The communication between the SBC and the web application operates at a higher software level. The SBC processes the information received from the MCU while serving as the gateway for the user interface. To display information such as port status, voltage, current, and timers, the SBC must transmit this data using communication protocols that are suitable for web systems. This interaction is handled through APIs, which define how the SBC and the web application request, update, and exchange data.

Application Programming Interfaces, or APIs, are essentially protocols and rules that allow software applications to communicate with one another to exchange data and allow for certain functions and features. APIs are usually structured in terms of client/server architecture. The application that is making a request is the client, while the server is the application responsible for sending the response. There are four different types of APIs, these are Private APIs, Public APIs, Partner APIs, and Composite APIs. Since the project needs to be a working prototype, the API type that the smart charger would use are private APIs. This section will go over three different API styles and determine which ones would be best suited for the smart charger. [53]

3.4.2.1 REST API

Representational State Transfer (REST) APIs are some of the most commonly used APIs across the web. A REST API is a simple, uniform interface that is used to create data, content, algorithms, media, and other digital resources available through web URLs. REST APIs are able to provide a consistent way for developers to consume APIs and improve the performance of applications that rely on them. Rest APIs are also known as RESTful APIs.There are six constraints that must be met to make an API service RESTful. It must be client-server based, use a uniform interface, must be labeled as cacheable or non-cacheable, must be a layered system, code on demand, and use stateless operations. There are four types of requests that can be made with REST APIs. The four types of requests are GET, PUT, POST, and DELETE. GET allows for the retrieval of requested data from a server. PUT will let the server update an entry in the database. POST tells the server to create a new entry in the database. DELETE lets the server delete an entry in the database. [54]

There are several advantages to using REST APIs for the smart charger, such as flexibility, portability, and simplicity. It's simple to perform a migration from one server to another, and

RESTAPIs are also adaptable to the working syntax and platform. Another advantage is that REST APIs are lightweight and fast, which improves its reliability. REST APIs support multiple formats such as XML, HTML, and JSON, which makes it perfectly compatible with the React frontend and the Node.js backend running on the Raspberry Pi. Unfortunately, there are also a few limitations of REST APIs. REST APIs are not optimized for real-time data, since each request needs to be made by the client. For an application that will have constantly changing data in real-time, such as the continuous changing of voltage, current, and power readings, relying on polling will lead to increased bandwidth usage and lower responsiveness. Overall, REST API has great resources, but it alone would not be enough for the smart chargers real-time data delivery. REST API would need to be paired with another real-time protocol to ensure that the data being delivered is instantaneous.

3.4.2.2 WebSocket API

Websocket API is another API format, but with much better support for real-time data transfer. Websocket is a full-duplex protocol, meaning that it can transfer data between a client and a server over a persistent network connection. Websocket uses a TCP/IP connection for creating a communication network between the client and the receiver, and the connection between the two will not be terminated until either the client or the receiver closes the connection. Websockets is a popular API to use in real-time applications, such as chat applications and video calls applications. There are wide uses of WebSockets API, such as online education, gaming, event update applications, real-time data visualization, and collaborative applications. [55]

There are various advantages to using a WebSocket API for the smart charger. The most significant benefit would be WebSockets support of real-time data transfer. The support of real-time communication would ensure that the user interface can reliably display accurate voltage, current, and power readings as they are constantly changing.
Another benefit of using a WebSocket API would be its TCP connectivity. It communicates using a protocol that ensures reliable data transfer and keeps its connection open, which would significantly reduce the overhead that REST API would bring and improve the systems responsiveness. WebSockets would allow immediate feedback when the user would adjust the power configurations and timer settings, overall allowing for an improved user experience. There are a few disadvantages to using WebSockets, with the biggest disadvantage being its added complexity. Due to WebSockets, the server would need to manage active connections and handle disconnections of its TCP networks. Another disadvantage is that the encryption of real-time data would increase the processing demand of the Raspberry Pi. Despite these negatives, the use of a WebSocket API would be essential for the continuous monitoring of the smart charger stats, and it provides the responsiveness and reliability needed to deliver accurate, real-time data.

3.4.2.3 GraphQL API

GraphQL is a query and manipulation language that provides APIs with a flexible syntax that describes data requirements and interactions. This style of API is capable of accessing many sources within one request, which reduces the number of network calls and bandwidth requirements. Updating data is pretty simple when using mutations, as it lets developers describe

how to modify the data. GraphQL is great for real-time applications by enabling subscriptions across multiple devices and letting you access the data offline. [56]

There are several advantages to implementing GraphQL into our smart charger. Some of these advantages would be being able to use one request for getting the information, which would save the energy and CPU cycles consumed by the application. Another advantage is that it's flexible and allows for complex nested relationships to be expressed easily. Unfortunately, this is also a disadvantage to implementing it to our smart charger. Our smart charger will be sending information such as current, voltages, timers, threshold, and status. This information would not be difficult or complex, so using GraphQL API would be unnecessary. Another significant disadvantage is that the API would bring a lot of overhead to the Raspberry Pi, which would be heavier on memory and CPU usage. WebSockets and REST API are more than capable of handling the needs of the smart charger without weighing down the CPU usage. While GraphQL is a powerful tool for large, data-intensive applications, it would be unnecessary and inefficient for the smart charger's straightforward communication requirements.

Table 3.4.2: API Comparison

API	REST API	WebSocket API	GraphQL API
Purpose	Handle on-demand status retrieval	Provides real-time data delivering	Flexible querying of complex data
Use Case	Updating timers/thresholds and retrieving settings	Live feedback of voltage, current, power, and port state	Large-scale applications with dynamic data queries
Resource Usage	Low CPU footprint	Moderate resource usage	High overhead
Performance	Effective for small and infrequent requests	Fast updates with minimal latency	Reduced performance in simple IoT devices
Scalability	Easy to scale with stateless design	Can scale but requires connection management	Built for high scalability with varied data
Complexity	Low complexity, simple to develop and debug	Moderate complexity, connection reliability and authentication required	High, complex schema, backend logic
Networking	HTTP/HTTPS	Persistent TCP	Single endpoint

	request–response model	connection using WS/WSS	using HTTP with schema negotiation

After evaluating REST, WebSocket, and GraphQL APIs for the smart charger, the best option would be to use a hybrid of both REST API and WebSocket API. REST API would offer great connectivity with the React frontend and Node.js backend, which would make it ideal for receiving static commands, such as changing the charger's threshold and timer configurations. The biggest issue with using REST API alone would be that the smart chargers voltage, current, and power stats are constantly changing, which would result in REST needing to poll excessively and consume a lot more bandwidth than is necessary. This is where WebSocket API would come in. WebSocket creates a low-latency communication network that makes the delivery of real-time data efficient and can display the information to the user as soon as it is available. Combining REST and WebSockets allows the smart charger to maintain efficient resource usage and instantaneous data delivery, which ensures that the smart charger is responsive and reliable when communicating.

Chapter 4: Standards and Design Constraints

4.1 Industrial Standards - Current and Voltage

4.1.1 IEC 61010 - Safety Requirements for Measuring Equipment

The IEC 61010-1 standard references the safety measurements and requirements that are applied in laboratory, control, and measurement applications. These global safety norms or multilateral lateral safety regulations that are promoted internationally confirms that the instruments that are used or enabled here to measure the voltage and current are designed in a way that electric shock would not occur, prevent fire hazards, as well as mechanical failure while a certain process or application is taking place. Furthermore, there are four critical category ratings that are essential criteria to this standard. Category 1 protects the electronic equipment that is being used. Category 2 deals with "single-phase receptacle-connected loads such as appliances and portable tools". Category 3 discusses a 3-phase distribution where there is also single-phase equipment and commercial lighting in certain areas or specific locations such as motors or switchgear. Finally, category 4 entails a 3-phase utility connection, conductors used outdoors, service entrances, and electricity meters. These four categories are essentially the measurement categories that state a certain electrical tool's tolerance to withstand voltage. Different categories will apply to the different applications and electrical equipment depending on its criteria and how it is being implemented into the system.

This standard also incorporates other types of criteria to reinforce the safety when in operation. For example, different types of insulation or even clearance distances. Referencing back to the standard specifically, IEC 61010-1 confirms that when using voltage dividers or shunt resistors, these components will function safely when under a momentary condition. This can be specific towards consistent dielectric strength.

When applying this standard while designing a device, this allows to ensure that the measurement systems achieve reliability, safety, and industry and laboratory safety measure credibility. [57] [58] [59]

4.1.2 IEC 61326-1 Electromagnetic Compatibility (EMC) for Equipment Measurement

The IEC 61326-1 aids in promoting individual's safety in the industrial, medical or scientific environment. Proper electromagnetic compatibility must be available and present so the output of the readings are correct and operate properly and effectively. This standard allows us to confirm that the instruments are able to operate correctly when exposed to environments that include electromagnetic interference. This standard covers electrostatic discharge (ESD) and electrical fast transients (EFT).

Immunity testing is performed by manufactures to confirm if compliance is able to be reached by this standard. This includes the IEC 61000-4-series including IEC 61000-4-2, IEC 61000-4-4, and IEC 61000-4-6 series. ESD immunity, EFT/Burst Immunity, and the conducted disturbance immunity respectively. These series help confirm that when these transient events take place, the device remains fully functional in reference to its overall performance.

Now, in reference to the current and voltage measurements, this is a critical component as it is able to ensure that the amplifiers, microcontrollers, or the high-resolution ADCs are able to maintain accuracy and provide correct data when under Electromagnetic Interference or EMI for short.

Using the IEC 61326-1 to include filtering, grounding, and other various techniques, this further allows us to boost the precision of our measurements as well as incorporate protection against any external sources or even noise that may be present during the process. {[60], [61], [62], [63], [64], [65]}

4.1.3 ISO/IEC 17025 and NIST TN 1297 - Measurement Traceability and Calibration

The ISO/IEC 17025 engages in providing reliable results for any sort of organization where testing, calibration, or sampling is performed. This includes any type of laboratory and that is owned by anyone as well, whether that be by the government, another organization, or a specific industry, This specific standard can also be useful for research centers, universities, or any organization or community that needs to accomplish testing, sampling, or calibration and obtain reliable results and data. This is critical when referencing current or voltage measurements as it states that the use of calibration methods as well as the equipment meet internationally accepted performance and requirements that may include uncertainty.

The NIST Technical Note 1297 discusses measurement uncertainty when evaluating or expressing the measurements. This combines both statistical (Type A) and systematic (Type B) uncertainties so that the overall uncertainty budget can fully be formed and can be referred to

international comparisons of measurement standards, basic research, applied research and engineering, calibration, reference material, and generating standard reference data. As a result of adhering to this standard, engineers are able to report results with quantifiable confidence levels. This allows and enables us to confirm that the measurement values are both traceable and credible to the national standards.

Thus, when both of these standards are being implemented, this can further confirm or ensure that when taking the voltage and current measurements, they are scientifically valid and can also be replicated or repeated in the laboratory and industry. [66] [67] [68].

4.2 Design Constraints - Current and Voltage

4.2.1 Measurement Uncertainty and Accuracy - Design Constraints

When conducting measurements, in this case, measuring current or voltage, this results in a significant constraint in regards to current and voltage measurement systems. The configuration of various components in a specific design are the basis of measurements whether that be resistors, operational amplifiers, analog-to-digital-converter, etcetera. Each individual component can bring uncertainty in measurements. Based on NIST TN 1297, total uncertainty should be calculated statistically. This should solely be done by using the RSS, Root-sum-square method. To confirm that there is reliable performance that is occurring, uncertainty below 1% is recommended here of the full-scale range. This essentially means that there is reliable performance.

Incorporating low-drift voltage references, high-resolution ADCs, and ±0.1% precision resistors can help to accomplish this. Calibration of the ISO/IEC 17025 can occur at times; however its procedure allows to ensure traceability as well as validity amongst its testing intervals. [69] [70].

4.2.2 Input Impedance and Circuit Loading

The loading effect is critical when it comes to voltage measurement. The goal here is to minimize the loading effect, which can ultimately result in a design constraint. The oscilloscope and digital multimeters have been designed with a specific input impedance to prevent disturbances within the circuit so that the voltage value has not been changed or effected when using a tool to measure onto the circuit.

Furthermore, when taking measurements where a shunt resistor is being used, this lessens the burden voltage so that the circuit performance is not affected. To reduce or eliminate thoroughly the resistance error, certain devices are used such as differential amplifiers that include common-mode rejection or even buffer op-amps. To confirm that there are accurate readings that are being drawn or to conclude from, high impedance allows us to confirm and ensure this technique. This can be done even under robust operating conditions. [71][72]

4.2.3 Thermal and Power Constraints

When measuring current and voltage, thermal management is a significant constraint in reference to these systems. For example, both voltage dividers and shunt resistors, especially,dissipate heat proportional to voltage or current. This can be referenced to P=I^2*R. If there is too much heat, this can result in negative effects such as instability for long-term as well as resistance drift that can occur over time.

Using the methods below, this can help minimize the drift within the measurement and also maintain accuracy when dealing with operations incorporating high-current.

Based on Texas Instruments' "Engineer's Guide to Current Sensing", designers should select low temperature coefficients or manganin shunts. Heat-spreading planes on PCBs are essential when thermal energy is needed to be distributed evenly. Using the IPC-2221 guideline in reference to conductor sizing, this ensures that traces are able to withstand certain current loads and no temperature rise will take place during the entirety of this process. [73].

4.3 Additional Standards and Constraints

In addition to the standards and constraints discussed above, there are also other considerations that may play a role in the effectiveness of the current and voltage measurements.

The IEC 60068 essentially allows testing certain equipment when under vibrations, shock, temperature stress, etcetera. This can be applicable to shunt resistors and other components so that consistency and accuracy is maintained regardless of the conditions that may be present during testing or measuring.

Diving deeper into the IEC 60068 it lies on the basis of environmental testing along with its severities. Confirming that it is able to test the electrical, electronic, and other sorts of equipment related to this to confirm how reliable and how well it is able to perform overall in relevance to the environmental testing.

There is also the IPC-2221 standard which expresses the guidelines for a printed circuit board or PCB that incorporates spacing, thermal management, and other specific criterias. This allows us to confirm that the traces onto the board are able to carry and maintain structure without ever overheating. This is an important and critical aspect as measurement drift should not occur as it can result in changes in the values and data produced.

Tolerance drift, calibration, and component accessibility are all practical design constraints that should further be considered and implemented. There is no single standard that incorporates these; however, it is important and allows us to confirm the voltage and current measurement circuits remain safe and accurate while they are operating. [74] [75]

4.4 Industrial Standards- USB Ports

4.4.1 USB 2.0/3.0 Electrical Standards and Power Delivery

There are certain specific electrical standards as well as safety standards for USB ports where they are essentially supplying power that is going into a device. This can be referenced as the USB Specification or Universal Serial Bus Specification. Beginning with the USB 2.0, for each of the standards present here the maximum current must include 5 Volts, where there is +5% or -5%. Furthermore, it should also incorporate 500mA. As for USB 3.0, the current limit can be noted down here to be 900mA. For the USB 3.0 Specification, it mentions that it allows for dynamic voltage for 20V and current can be up to 5A. These criteria are also dependent on the device that is being used and how much it can hold for the current and voltage.

Overloading is critical to avoid here especially when current and voltage are involved. Since current and voltage are being measured from the USB ports, making sure that overloading does not take place is essential. To best avoid this, adhering to the limits present within the standard are ideal. In order to reinforce this concept as well as encourage safety amongst users, there is the USB Implementers forum (USB-IF) which essentially confirms that these safety measures and limits are met to reinforce safety amongst users. The equipment that is being used to also take measurements must also adhere to the specific load resistance that needs to be used or implemented. This is especially critical so that the USB host is able to effectively detect and take note of regulating the current and voltage and also taking in account its measurements as well.

Based on the limits and specifications mentioned within this standard, this helps to further ensure or confirm that the specifications are being met when it comes to the USB ports. This is in reference to delivering power where there is a certain amount of voltage and current that can be handled or tolerated within the USB host. This encourages precise and accurate communication as well to the devices that are connected. [76][77]

4.4.2 IEC 62680- USB Interfaces for Data and Power

There are several interfaces within the USB that are vital towards its overall functions. This includes communication protocols, the physical layer, as well as its characteristics in reference to power delivery. This can be referenced to the IEC 62680 series which deals with the USB-IF specifications. Going more depth into this standard, this allows for us to confirm that no interference would be occurring or taking place with the USB measurements along with normally how the USB would tend to operate. More specifically in relation to the measurements, current and voltage at the USB port.

When referencing the IEC 62680, there are also certain specifications that must meet certain criteria that need to be met here. This includes there being proper shielding, controlled voltage ripple when onto the VBUS line, as well as correctly implemented of the cable impedance. Note that the cable impedance should be about 90 ohms. Adhering to the components of this standard mentioned above are vital when it comes to making sure that loading the USB bus would not occur or any communication issues that were to ever come up or take place.

When designing this device, making sure that the design is met by the IEC 62680 guidelines is significantly critical. Note that engineers are able to integrate integrity while operating as well as electrical safety. This provides much significance to the user and overall as to why these guidelines should adhere and why they should be met as well. [78][79]

4.4.3 IEC 60950-1 and 62368-1-Safety of Information Technology Equipment

The safety standards for information technology equipment apply to the USB ports as they supply power to external devices. These standards are IEC 60950-1 and IEC 62368-1 which is what it is replaced as currently. Several components within these safety standards are present to prevent any sort of hazards occurring. This includes insulation, electric shock protection, as well as limited power source classification also known as LPS.

When wanting to measure a specific unit from the USB port, these measurement devices that would connect to the USB ports must adhere to these standards that were previously mentioned above. This ensures safe equipment design. This can be done through current-limiting circuits as well as maintain a clearance distance when it comes to low-voltage DC circuits. By doing this, if a configuration were to short or fail throughout testing, this would result in absolutely no effect towards the USB host or the user as well.

Note that the IEC 62368-1 can be understood as a general safety standard for insulation, energy source classification, and implementing a safe design. USB current-specs are not provided here. [80][81].

4.5 Design Constraints

4.5.1 Measurement Accuracy and Load Regulation

Load regulation and obtaining accuracy within the measurements that are taken are highly essential when it comes to measuring current and voltage from a USB port. Depending on the device that is connected to the USB port and how much power it can withstand, the USB host will be able to modify the current output based on the device's power requirements. This is so that the data and measurement readings that are being produced remain to be accurate and also that there is very little resistance within the circuit path itself while being able to sense the voltage along the VBUS line.

High accuracy resistors where the tolerance is less than or equal to 1% are able to achieve precise readings without there being any burden voltage that takes place. Also low-offset current-sense amplifiers are able to accomplish this same thing and have a similar concept. Furthermore, Kelvin connections (4-wire) should be placed here so that the trace resistance errors are reduced; this should be within the sensing circuit. Furthermore resulting in the current and voltage readings that are able to reflect the accurate data on the USB ports when there could be various loads taking place. It is important to note that a sense resistor that involves Kelvin within the sense terminals will result in the best results. [82] [83]

4.5.2 Data Integrity and Isolation Constraints

Preventing interference with data transmission is significantly critical here. The USB port incorporates both the data lines and power (VBUS), so in order to prevent this, measurement circuits therefore need to be created. There is inductance as well as parasitic capacitance which can lessen the quality of the signal. This then results as issues such as failures within communication between the USB and the devices that are connected to its ports.

An approach for this to avoid happening is that for the current sensing and voltage, it should be implemented with differential probes, USB-approved breakout boards, or even low-capacitance measurement nodes. To maintain the quality and integrity of the signal,IEC 62368-1 should be used here when dealing with digital isolators or even isolated amplifiers.

Note that another form or method that can be used here are the industrial communication standard such as the RS-485. This indicates different signal transmissions that are being used with no present ground connection. This helps to prevent potential ground loops if there were to be noise in the system or any errors in the data. This can be in reference to the magnetic flux; however, we want to maintain the quality of data throughout. [84] [85]

4.5.3 Thermal Management and Over-Current Protection

Over-current thermal protection, OCP and thermal management are vital constraints when it comes to design through measuring the current and voltage from a USB port output. Based on the USB-IF Power Delivery Design Guidelines, there must be OCP within the USB power delivery circuitry. This is especially essential because it allows for a safety net between the USB host and all the other devices that are connected to it also, or the external devices for short. This basically reinforces the concept where if something wrong were to occur in the circuit or if something were to short this would immediately cut off any current so that overheating is prevented as well as no damage within the components also. [86]

Current can also be restricted to certain levels where the USB Power Delivery and USB4 ™ Thunderbolt 3TM Compatibility Requirements are necessary. It mentions how OCP must be implemented in the circuit so that current can be restricted to safe levels for the device and user. Over-current events will not compromise the integrity of the USB data lines or failure when it comes to the power delivery subsystem. [87].

The concept of thermal management allows for it to work simultaneously with the safety features previously mentioned. If current levels were to approach the delivery limits of the USB power, then the connectors, sensing components, and even dissipation among traces can become essential here. The dimensions must be accurate when it comes to the copper trace widths or even the power planes so that it is able to handle the specified load and the temperature rise would not exceed its limits. Note that this is also specified in the USB and IPC-2221 guidelines. Drift therefore is minimized when dealing with sensing components and thermal damage is also prevented overtime over using it for a long period or course of time.

When OCP and thermal designs are incorporated and used efficiently, this allows for safe and reliable measurements from the USB ports output power while the official USB-IF safety and performance standards are also able to be maintained at the same time.

4.6 Additional Standards and Design Constraints

The IEC 60068 standard references electronic components when it comes to environmental testing to confirm that the USB port measurement circuitry can hold differences when it comes to humidity, vibration, or even temperature without lessening the performance of the device. Note that this can be any form of atmospheric conditions for measurements and tests.These are essentially occurrences that could happen in the real world, so having the IEC 60068 to check this helps with reinforcing situations or circumstances when it comes to real world applications. [88]

Using high-quality USB receptacles, signal shielding integrity maintenance, or even connector durability are all considered practical considerations. Routing, cable length, and electromagnetic testing must adhere to the guidelines that discuss USB-IF. This is vital as it helps to prevent voltage drops taking place or any electromagnetic interference coupling that could take place when measuring the voltage present at the USB port output. [89]

4.7 Industrial Standards - MCU Software

4.7.1 Inter-Integrated Circuit (I2C)

The Inter-Integrated Circuit (I2C) bus specification exists as a standard created by NXP Semiconductors. The specification was first released in 1982, and is now in its seventh revision. I2C enables communication between many types of devices. It uses two wires, SDA (Serial Data) and SCL (Serial Clock). The low (0) and high (1) level of each wire depends on implementation, but is typically determined by a percentage of a voltage VDD, which is attached to SDA and SCL with a pull-up resistor. Low (0) is defined as below 30% of VDD, and high (1) is defined as above 70% of VDD. One device, known as the controller, must generate a clock signal for SCL. The controller must also start and stop each data transfer, regardless of whether it is intended to receive or transmit data to the target device(s). Multiple controllers can exist, but are required to undergo a process known as arbitration. Arbitration allows multiple controllers to work simultaneously until one controller attempts to send a high (1) signal while SDA is low (0). If this happens, the offending controller “loses” and stops its output to SDA. I2C data transfers consist of a start condition, an address, a read/write bit, an ACK/NACK bit, data in 8-bit chunks separated by ACK/NACK bits at the end, and a stop condition. Addresses can have 7 or 10 bits. To use 10-bit addressing, the process differs between whether the controller is primarily the sender or receiver of data, but requires the address to be split up. If the controller is primarily the sender, the address is split into two blocks, where the second part of the address (after the first 7 bits) is after the first R/W and ACK bits, and is followed by another ACK bit. If the controller is primarily the receiver, the R/W bit will be a 0, the second part of the address is sent after the ACK bit, then after its ACK bit is a repeated start condition (Sr). Then, the first 7 bits of the

address are sent again, and the R/W bit will be a 1. 10-bit addressing is less common, but both 7-bit and 10-bit addresses can coexist on the same I2C bus. There are also reserved addresses that can mean specific things, such as the optional Device ID. Device IDs are split into three parts: a 12-bit manufacturer name, a 9-bit part identification, and a 3-bit revision. That way, important device information is possible to read over I2C to get more info about a connected device, which may be helpful for debugging purposes. As technology has advanced, I2C also offers a configurable speed, to support many types of devices with increasing speed requirements. Standard mode is the original implementation of I2C with a 100Kb/s bidirectional speed limitation. Fast-mode allows 400Kb/s bidirectional, while Fast-mode Plus allows 1Mb/s bidirectional. High-speed mode allows an even higher speed of 3.4Mb/s while still being bidirectional. Ultra Fast-mode also exists, which allows 5Mb/s transfer speed, but is only unidirectional. [90]

I2C also has some similar derivatives that have been similarly used over time. An application of I2C used in many cases is called SMBus (System Management Bus). SMBus works very similarly to I2C, however the addressing differs slightly from I2C. They are interoperable, however SMBus devices may not work with SCL (clock) below 10KHz or above 100KHz. Another application is called PMBus (Power Management Bus), which is effectively SMBus intended for use with power converters. Perhaps another useful adaptation of I2C is Intelligent Platform Management Interface (IPMI), which is intended for monitoring and control of computer systems such as servers. It is not directly interoperable with I2C, but is based on it. [90]

4.8 Industrial Standards- MCU Power Supply

4.8.1 IEC 62368-1 Safety of Information and Communication Technology Equipment

The IEC 62368-1 standard discusses critical requirements to encourage safety amongst users. This is in reference to audio and visual as well as any sort of information and technology equipment. Note, that this can be in reference to microcontrollers which is discussed in the report as well as embedded systems. Different sorts of electrical energy sources are grouped and named for the hazard-based safety engineering approach that is introduced here. This essentially describes crucial safety measures that are applicable to electrical energy or equipment to protect fire, shock, or any sort of hazards that were to take place. Essentially, its goal is to reduce any injuries, pain, or fire that were to take place.

In reference to MCU Power supply specifically adhering to one of the standards as previously mentioned, the 62368-1 standard, it is able to confirm that the components that go into the design are designed in a safe manner. This can include insulation boundaries, circuit separations between AC mains, low voltage DC lines, and also voltage rails. Certain requirements must be met in order for there to be the safety net which can include maintaining thermal limits or meeting clearance requirements. The MCU must meet these criteria so that all of its entirety is designed safely. Note that for the thermal limits this needs to be maintained under certain load conditions as well. [91].

4.8.2 Safety Requirements for Measurement, Control, and Laboratory Equipment

The standard IEC 61010-1 discusses safety requirements for electrical equipment in reference to control, measuring data, or even usage of the laboratory. These electronic controls and measurements are being operated on low-voltage power sources. So, when designing the MCU's power supply, it is highly needed for it to meet what the IEC 61010-1 standard requests to ensure that there is safety during electrical testing and development sections. The standard discusses grounding, transient-overvoltage protection, as well as insulation coordination. All of this is to essentially protect the user from being exposed to these hazardous conditions that could potentially take place if this standard is not followed.

When dealing with MCU boards that are powered through the use of a lab power supply or test fixtures, this standard, the IEC 61010-1 allows to ensure that regulators, connectors, or any sort of protective component that were to be used in the design are safe to use and will hold any transient surges that were to take place as well as not affecting the domain for the input and the output. [92].

Failure to address these standards would result in situations leading to electric shock, equipment damage, and fire hazard. For electric shock, this could be a result of inadequate grounding or single fault insulation. This fault may result in external conductive parts being live resulting in an increased likelihood of shock.

4.8.3 ISO/IEC 17025 and NIST TN 1297- Calibration and Traceability in Voltage Regulation

There must be both precision and accuracy when it comes to current and voltage regulation to have an extremely precise MCU power-supply, the ISO/IEC 17025 describes "general requirements for the competence of testing and calibration laboratories" while the NIST Technical Note 1297 provides various methods for uncertainty within measurements, and quantification.

When both of these standards are further applied to the verification of the power-supply this further allows us to confirm that the output voltages, for example the 3.3 V± 2% meet the national standards and are valid under specific conditions. As a result, this helps in reducing the error when taking measurements during how well the device is performing through the usage of voltage regulators to ensure that the operation of the MCU overall remains consistent under transient and nominal loads. [93] [94].

4.9 Design Constraints

4.9.1 Voltage Regulation and Ripple

The MCU must need a noise-free and stable supply voltage in order for it to efficiently operate and work correctly. If there is too much voltage rippled or change occurring, this can result in unpredictability in regards to how it is operating or functioning, its behavior. Furthermore, this can also start affecting how it communicates resulting in communication errors as well as possible damage to the integrated circuits. Incorporating power regulators allows to maintain the output voltage within the MCU specific tolerance amongst the variations of the different types of load conditions it offers.

Low-dropout regulators, also known as LDOs, and switching regulators are chosen based on noise, efficiency, as well as its transient response. Proper filtering should be introduced and incorporated in the design. THis can be done with the use of bypass capacitors which can be placed close to the VDD pins which would be on the MCU. This essentially helps to subdue the high frequency ripples as well as the switching transients. WHen designing a regulator, there are ways that this can be best laid out and decoupled which can furthermore be shown based on the citation mentioned here. This overall helps and encourages reinforcement with this specific design constraint.[95]

4.9.2 Thermal Dissipation and Efficiency

Thermal performance and efficiency associate very closely to the MCU power supply. Junction temperatures can elevate through severe heat when using linear regulators and can also affect other components nearby as well as the regulators. When dealing with switching converters, they require a need for there to be proper selection for the diode and inductor so that there is a decrease in the switching losses. Although the switching converters are highly efficient, this can be a downside for those components. Dissipation must also be calculated where P = (VIN - VOUT * ILOAD when it comes to linear supplies. This allows to reinforce that the thermal rise will remain in between the limits of the device being used or operated on. Adequate airflow or heat sinking can be used when needed. Furthermore, using the IPC-2221 PCB thermal guidelines help to support safe temperatures when operating and also avoid any voltage drift that were to occur by a certain component that would overheat.

When efficiency is maintained as well as the thermal margin, this helps the MCU's supply system to avoid component destruction or voltage drift to promote stable operations across all loads. [96] [97]

4.9.3 Over-Current Protection and Startup Safety

The MCU power supplies must incorporate OCP, also known as over-current protection. THis allows it to incorporate safety towards the circuitry as well as the regulator. Based on USB-IF's"Power Delivery Design Issues for Hi-Speed USB on Motherboards", this references that there must be current-limiting components for all regulated power paths. This can be for example, current-limit Integrated chip or also sense resistors. These ultimately help in preventing the thermal overstress as well as any voltage collapses during faults. [98]

Incorporating adequate startup sequencing and brownout protection are vital and key components here. Microchip's Application Note AN794 describes undervoltage detectors should

make note and reset if any form of instability were to occur as well as the supply rails must be stated in the correct order. [99]

Combining all together, current-limit mechanisms as well as power-up sequencing allow for us to confirm that the outer or peripheral domains as well as the MCU's core are to stay in the limits that are considered safe for electrical systems. This helps to avoid latch-up or any sort of damage that were to take place during power transients and operations.

4.10 Additional Standards and Design Constraints

4.10.1 EMC and PCB Layout Design Standard

An additional standard can be stated as the EMC and PCB layout design standards. Electromagnetic compatibility, also known as EMC, is critical when dealing with the power-supply design for the MCU. THis is because when using or incorporating switching regulators into the system, this can welcome noise into the power and signal domains. With there being incorrect trace routing or grounding this can result in voltage ripple as well as radiated emissions that therefore result in degrading the MCU performance.

Based on the article, "PCB Design Guidelines for Reduced EMI" by Texas Instruments, this provides different layouts where the loop area can be minimized, ground impedance, as well as information related to high-frequency noise. Decoupling capacitors can be placed close to the MCU's VDD pins where solid ground planes are used. THis essentially helps to separate the signal lines from the power traces. Overall, this helps to improve the EMC standards when followed along thoroughly. [100]

4.10.2 Inrush Current and Startup Transient Management

During the initial process when powering-up, the MCU power supply must be able to control the inrush current that is present and is essentially due to the charging input or any of the bypass capacitors. When there is high inrush current, this can result in over-current protection which can also ultimately result in voltage dips, although temporary, this can ultimately reset the MCU.

Based on the article "Managing Inrush Current", by Texas Instruments, it describes how these surges could potentially occur and how one can control them. For example, soft circuits can be implemented to help prevent or control this in taking place. Series resistors can be used as well as current-limiting controllers. When the inrush current is being properly managed, this allows to ensure that there is a stable voltage ramp, the regulators are protected, and MCU brownouts are also avoided when startup is initially taking place. [101]

4.11 Industrial Standards - SBC / Web Application

4.11.1 RESTful API Design Standards

The communication between the SBC and the Web Application follow the principles of REST (Representational State Transfer) defined by the Internet Engineering Task Force (IETF) in RFC 7231. REST is a standardized architectural style of network-based software architectures that operates over Hypertext Transfer Protocol (HTTP). The system uses RESTful design to allow the Web Application to interact with the SBC backend in a consistent and stateless manner. [102]

Each RESTful message is formatted using the conventional HTTP methods such as POST, GET, DELETE, and PUT. These methods are used to retrieve system information and modify charging configurations for specified USB ports. Data exchange is formatted in JavaScript Object Notation (JSON) as defined by RFC 8259 (The JSON Data Interchange Standard). JSON provides a lightweight, language-independant data structure, making it ideal for embedded systems. RESTful API design introduces stateless communication, which means that each request from the web application contains all context. This allows the SBC to process data without maintaining a session state. RESTful API design also introduces a layered system architecture. The web application's frontend, backend, and database all communicate through distinct layers, which allows easier scalability and debugging. [103]

4.11.2 Database Design Standards

The SBC's database stores data such as USB port current thresholds and timers. To maintain data integrity, the database follows ANSI SQL (American National Standards Institute Structured Query Language) standard. This standard defines the syntax, schema management, and query operations for relational databases. Following the ANSI SQL also allows queries to remain portable and adaptable across different database engines without needing major modifications. [104]

The key design considerations of database design standards include primary and foreign keys. Each table has a unique identifier that ensures relationships between datasets remain consistent. Another consideration is data type enforcement, where the current threshold and timers use INTEGER types to maintain precision and compatibility with the SBC's embedded system constraint. By adhering to the ANSI SQL standards, the database can be easily migrated to larger systems if needed, thus increasing the database's scalability and portability.

4.11.3 Web Accessibility and Interface Design

The Web Application frontend must comply with the Web Content Accessibility Guidelines (WCAG 2.1) developed by the World Wide Web Consortium (W3C). These are guidelines that define the best practices for ensuring that web interfaces are usable by all individuals, including those with disabilities or using various types of devices, such as a smartphone or tablet. [105]

The WCAG 2.1 framework is organized around four main components, which are Perceivable, Operable, Understandable, and Robust. To comply with the Perceivable requirement, all voltage, current, power, and configuration options must be displayed with a high color contrast and must be readable. Any icons or graphical indicators on the Web Application should include alternative text so that users using screen readers or non-visual interfaces can access equivalent information. Under the Operable requirement, all functions must be designed to function with both keyboard

and touchscreen input to ensure that the system is fully controllable whether accessed from a mobile device, tablet, or laptop. The Understandable requirement refers to consistent labelling and logical grouping of data fields, which reduces the risk of user error during configuration changes. Lastly, Robust means that the frontend code must remain compatible across modern web browsers and devices, which improves maintainability by allowing developers to update or expand the application without compromising accessibility features. [106]

4.11.4 WebSocket Real-Time Communication Standard (RFC 6455)

The smartcharger's real-time communication between the SBC backend and the web application must follow the WebSocket protocol defined in IETF RFC 6455. WebSocket provides a full-duplex communication channel that runs on a persistent TCP connection. When compared to the request-response model of HTTP, WebSockets reduce overhead by eliminating user-driven polling and repeated handshakes. Under RFC 6455, WebSocket communication is established through an HTTP-based handshake that converts the HTTP connection into the WebSocket protocol. The handshake includes fields such as Sec-WebSocket Key, Sec-WebSocket-Accept, and Connection: Upgrade, which ensures compliance guarantees that both the SBC server and the web application support the protocol. Once the communication upgrades, communication is handled with framed binary or UTF-8 encoded text messages. [152]

The smart charger relies on WebSockets to continuously display the voltage, current, and power readings for each USB port to the user. RFC 6455 states rules for latency handling and message ordering, which ensures that real-time readings are received sequentially and without unnecessary queuing. This ensures that the interface updates smoothly across multiple platforms. The protocol also allows the SBC to detect dropped connections and resume its connection without manual client intervention.

4.11.5 HTML5, CSS3, and ECMAScript Standards (ECMA-262)

The Web Application will follow the W3C standards for HTML5, CSS3, and ECMAScript, which collectively define the structural, visual, and programmatic behavior of all web-based user interfaces. HTML5 establishes the structure of the application, which includes elements such as headers, sections, and articles. HTML5 ensures accessibility tools and modern browsers correctly interpret the layout. HTML5 also defines the standardized APIs that the Web Application uses, such as the WebSocket API and the DOM manipulation interfaces that allow seamless integration with the SBC backend. CSS3 states the styling rules in regards to color contrast, responsive scaling, layout grids, and theming. It allows the project to utilize Flexbox, Grid Layout, and Media Queries, which ensures that the user interface remains constant across desktops, tablets, and mobile devices. CSS3 standards will ensure that the web application remains legible and properly aligned across all display designs, which satisfies both accessibility and usability requirements. Lastly, ECMAScript defines the core scripting language used for dynamic updates, event handling, and real-time rendering of WebSocket data. The Smart Charger will rely on ECMAScript features, such as Promises, typed arrays, and event-driven callbacks. Following the ECMA-262 standard ensures compatibility with all major browsers while enabling parsing, memory management, and garbage collection. These standards guarantee that the Web Application remains stable, maintainable, and accessible. [153]

4.11.6 JSON Data Interchange Standard (RFC 8259)

JavaScript Object Notation (JSON) is standardized under RFC 8259 and it serves as the primary data-interchange format between the SBC backend, the microcontroller, and the Web Application. RFC 8259 defines the syntax rules for JSON structures, such as object notation, arrays, numerical formats, string encoding, and requirements for escaping control characters. These rules guarantee interoperability across programming languages such as Python, JavaScript, and C++, all of which are used across different layers of the smart charger system and are essential for the smart charger to work.

JSON is essential to the smartcharger's communication model. REST endpoints use JSON for configuration commands, as seen in the current threshold and timer commands, while real-time WebSocket messages use JSON-formatted messages containing voltage, current, power, and port number data. JSON has a very lightweight structure that minimizes message size and refuses parsing overhead on the SBC, which is essential for maintaining responsiveness under limited CPU and memory constraints. Following JSON's formatting standard also avoids ambiguities related to encoding errors, type mismatches, and inconsistent message formatting. This standardization is essential for ensuring that the SBC can correctly validate incoming data and that the Web Application can reliably interpret telemetry updates without additional schema translation. [154]

4.11.7 SQLite File Format Standard and ACID Compliance

The SBC will be using SQLite as its database, which means that it will follow its own SQLite File Format Standard and maintain full ACID compliance. The SQLite database resides entirely on a single file, providing simplicity and portability. The file-based approach eliminates the need for a server and makes SQLite ideal for applications with limited resources. The SQLite format defines page structures, B-tree indexing rules, journal and write-ahead log (WAL) modes, record alignment constraints, and the binary structure to persist relational tables within a single file. This approach is essential as it ensures that the database is portable, stable, and resistant to corruption. [155]

SQLite also maintains full ACID compliance, meaning it has atomicity, consistency, isolation, and durability. Atomicity means that all transactions on the database must be completed or rollback the complete transaction in case of any failures. An atomic system must guarantee that atomicity will apply in each and every situation, such as a power failure or crash, where the full transaction will need to be rolled back. Consistency means that the data is always consistent and that no database rules will be violated during any transaction on the database. This ensures that any changes to the values in an instance are consistent to other values of the same instance. Isolation means that one transaction is unable to read the data of another transaction until that is completed. If there are two transactions happening concurrently, only one respective transaction results will appear to the client , not the other client's transaction results. Durability ensures that configuration changes persist even in unexpected events, such as power loss, system resets, or SD card interruptions. SQLite ensures these compliances which allow for predictable performance and a reliable storage method that aligns with the SBC's hardware constraints. [156]

4.12 Design Constraints - SBC / Web Application

4.12.1 SBC Hardware Processing and Memory Constraints

While the Raspberry Pi 4 Model B provides significantly more processing capability than microcontroller-based embedded systems, it still has constrained memory when compared to a full desktop application. The SBC is responsible for running the backend server, WebSocket manager, SQLite engine, operating system services, and UART communication simultaneously, all under limited CPU and memory resources.

The backend needs to perform real-time data parsing, database writes, HTTP routing, and WebSocket routing, which collectively creates continuous computational load on the SBC. Due to Python and Node.js being interpreted languages, their runtime environments cause higher memory and CPU overhead in comparison to compiled languages like C++ and Haskell. The SBC's 4GB RAM also needs to be shared with the operating system kernel, GUI, caching layers, and background daemons, which leaves limited room for memory-intensive processes.

Due to these memory constraints, the smartcharger's design needs to minimize unnecessary data copying, reduce JSON processing overhead, and ensure that the database transactions are lightweight. Operations that are inefficient or block other operations could cause increased latency in charging stat updates, reduce the user interface's responsiveness, or starve the UART communication with the ESP32. These constraints directly impact the architecture of the backend service, requiring careful resource management and asynchronous, non-blocking I/O.

4.12.2 Network Connectivity and Communication Constraints

The SBC relies on a local network for both REST and WebSocket communication. Network conditions such as bandwidth, packet loss, connection stability, and Wi-Fi interference can affect the reliability of real-time telemetry. WebSockets requires a persistent TCP connection to transmit data, so any interruption may temporarily halt data delivery to the Web Application. The frontend needs to be capable of detecting and handling unstable or dropped connections.

Latency is also a critical constraint, as the charging stat updates need a low-latency environment to maintain their real-time responsiveness. Mild to high latency may cause delayed UI updates, stale sensor readings, and inconsistent timing behavior. In environments with congested internet traffic or many wireless devices connected, the consequences of latency can be amplified. The smart charger is intended to operate within a localized network, so it cannot assume the availability of external DNS services, cloud API endpoints, or internet connectivity. This constrains the smartcharger to a self-contained networking model where the SBC serves as the internal API host, network edge device, and data source. These network constraints impact the design of the REST endpoints, WebSocket reconnection strategies, and timeout intervals.

4.12.3 Security and Access Control Constraints

Even though the smartcharger operates under a local, private network, the system must still take security risks with unauthorized configuration changes into account. The REST endpoints that modify current thresholds or recharge timers must validate all input parameters to prevent malformed or malicious requests from causing operational instability.

The SBC does not include multiple users authentication or role-based access control system in the prototype due to complexity and limited CPU resources. While this does help by simplifying the development, it also constrains the system's deployment to trusted environments only. Without TLS encryption, all REST and WebSocket data are transmitted in plaintext, which limits the system's safe use on unsecure or public networks. Additionally, the SBC functions as both the application host and the data controller, which means that any compromise or unsafe exposure of the SBC would leave the entire system vulnerable. These constraints guide the SBC's architecture towards a closed-network model, controlled access points, firewall restrictions, and input sanitization as part of the REST and WebSocket layer.

4.12.4 User Interface, Display, and Responsiveness Constraints

The Web Application must be usable across different devices such as laptops, mobile devices, tablets, and computers, each of which have their own different screen sizes, aspect ratios, pixel densities, and input modes. This imposes constraints on the UI design, in that they'll require responsive layout rules, scalable vector graphics, adaptive font sizing, and component realignment for smaller screens.

Real-time data updates will also create additional constraints, as frequent DOM updates could worsen the performance of the UI, especially on older or lower-power devices. The browser's rendering engine, garbage collection behavior, and JavaScript event loop all limit just how quickly the user interface can display an updated charging stat for a port. Excessive updates could also lead to input lag, frame drops, and overall worse performance. In addition, user interaction must remain predictable even under network delays or WebSocket interruptions. The UI must be able to handle missing telemetry frames, partial updates, or reconnection attempts, which significantly constrains the frontend architecture to be event-driven.

4.12.5 Power and Thermal Constraints

Although not as limited as a microcontroller, the Raspberry Pi still experiences thermal throttling when performing sustained high-load operations. Running the SBC in an enclosed space such as a charger housing introduces heat buildup from both the Pi and voltage regulators. If the Pi exceeds its thermal threshold, the CPU frequency is automatically reduced, which in turn impacts WebSocket throughput, REST response times, and database write speeds.

Power availability is also limited. The Pi must share a single power supply with the rest of the system. Voltage drops or inadequate current delivery may lead to service instability, corrupt SD card writes, or unexpected reboots. These constraints necessitate thermal management, adequate ventilation, and a stable power input to ensure consistent SBC performance.

Chapter 5: Application of ChatGPT and Other LLMs

Due to innovation and emerging trends, Large Language Models (LLMs) have been on the forefront of everyone's mind. From OpenAI releasing an early demo of ChatGPT in 2022 to new competitors such as Google's Gemini and Anthropic's Claude. It is impossible to escape the usefulness and adaptability of AI. Due to this, LLM were utilized in order to explain the general concepts and principles present in the Senior Design idea. The main three LLM being ChatGPT, Gemini, and Claude as these three are the most popular LLM out there on a commercial scale.

LLMs are generally useful in searching for a range of topics that are applicable to the prompt. It is important to note that when asked to cite sources, LLMs due to have a tendency to incorrectly locate key information from the cited source as well as in the case of mathematical formulas, unable to correctly perform mathematical calculation. Due to this fact, any calculation or cited information from any LLM has been reviewed by members of the Senior Design team.

It is important to note that due to the length of the answer a LLM can generate, a summary of the answers will be provided in this chapter. For the full answer provided by the LLM, please refer to Appendix E.

Case Study 1: Question - "What is the criteria that should be used to decide what is the correct electrical component to use as a switch"[1A]

ChatGPT [1B] was able to generate a main criteria list with the following parameters: Voltage & Current Requirements, Signal type, Switching Speed, Control Signal Requirements, Power Dissipation & Efficiency, Isolation Requirements, Complexity and Integration, and Cost & Size Constraints. ChatGPT was also able to provide a mini decision guide, while this guide was elementary in terms of factors to be considered, it was useful in understanding some general applications where one switching element was superior to another. On the criteria list itself, ChatGPT focused a lot more on the device characteristics rather than the system as a whole. In Physical & Design Considerations, the LLM was able to generate some components that do go into the physical design, but it failed to consider design considerations such as IP rating. While the ideal use application for our device would be in an environment where it would not be expected to deal with liquids or large particles, it is crucial in other design applications where one would want to make a device more resilient to outside factors. One benefit of ChatGPT over the other LLM was that it was able to go more in depth on key information such as switching speeds and use applications.

Gemini [1C] also generated a main criteria list but with different criteria. The main differences that Gemini took into consideration of the load that the circuit would encounter. It took into account that both inductive and capacitive loads would either generate large voltage spikes or draw massive current respectively. Gemini also provided extra context when dealing with non-resistive loads as well as general safety rules to keep in mind for switches. Furthermore. Gemini was able to expand on switch terminology which while was not useful for our use case mainly as the switching circuits needed to be independent of each other was a main difference between ChatGPT and Gemini. Finally, Gemini delved more into the actuation method that will

be used by the switch such as a slide or a rotary as well as the mounting style that should be considered. While this information is not a good consideration to make as the switch will need to be controlled using a closed loop system, this is a key difference between ChatGPT and Gemini.

Claude [1D] was the least developed of the LLMs to generate a list for this project. While it was able to generate a list of criteria similar to how Gemini curated its list. It was not as in depth as the other language models present. For example, while it did delve into the same mechanical requirements as Gemini, it was unable to expand on the terminology unprompted. Only when asked did the LLM delve more into the terminology. One area that Claude had over Gemini was the Application-Specific Needs section which discussed briefly about safety and user interface considerations.

When all of these prompts were fully assessed for their content and depth of answer, All of the LLMs were asked to cite their sources. ChatGPT provided a list of their sources as well as included a link to each one. It is important to note there were links from credible sources, nearly 50% of them were pulled from forum pages. While these forum pages did not contradict any of the main articles utilized in the LLM, it did demonstrate the need to verify what ChatGPT responded with was actually factual. When Gemini was prompted for this, it was able to generate a list of sources that were pulled more from professional companies such as Carling Technologies, Acromag and OMRON in comparison to ChatGPT's preference of aiming for websites that speak about technology as a whole. Lastly, Claude was the least forthcoming with how it pulled its information claiming the AI used training knowledge on electrical engineering concepts and standard terminology, when further prompted about where it pulled its information from, it did provide access to links from professional companies that support the claims it was making however, it did provide a link that led to a dead domain.

Case Study 2: Question - What MCUs will be best suited to be used in a closed loop system where the MCU picks up current and/or voltage readings and needs to either open or close a switch depending on the reading? [2A]

ChatGPT [2B] was able to come up with a comprehensive list that had both strong analytical and technical depth . It also provided a list on what components and characteristics the MCU should have or support to allow users to do their own independent research. The top three characteristics that ChatGPT seemed to prioritize was the ADC performance, DMA capabilities, and Real-time control features. Furthermore it also explained why each of those characteristics were important when needing to decide on what . Some of the MCU offerings that GPT mentioned were the TIMSP430, ESP32, and STM32 which were MCU we researched independently on. It went on to further explain why the MCUs offered were selected, allowing for a greater understanding of the requirements in order to succeed in this project.

Gemini [2C] came up with a list more focused on specific parts that were analog and control related such as ADC's, Comparators, Op-Amps, and advanced timers, all of these components are crucial when designing a response control system. This hardware focused criteria allowed for a deeper analysis on why these parts are important. This is in contrast to GPT's analysis which was more focused on what features the MCU had largely granted by the internal part

composition. Gemini was also useful in dividing the MCUs based on their design which further highlights how different architectures lead to different specializations and use cases.

While Gemini's list was not as extensive as the one offered by ChatGPT, it did further explain how component behavior is directly linked with system reliability.While both LLM did mention how the C2000 series was an excellent fit. Gemini focused more on the hardware descriptions to elaborate on how the MCU was a good fit such as bringing up core count, ADC with synchronized sampling, and a more advanced ePWM module on board. This is in contrast to ChatGPT's response which focused more on timing and the control system side in general. Furthermore, Gemini was able to provide a recommendation list that was more detailed in comparison to the tradeoff summary that was provided by ChatGPT. Gemini did fail in explaining the strengths and weaknesses of the DSC in comparison to the traditional MCU and Digital Signal Processor that is commonly used in more conventional systems like the ESP32 and MSP430.

Claude [2D] Generated a much more simplified response in comparison to the other two LLMs that summarized the strengths of each individual MCU. While it did mention the key factors that are important for a closed loop system such as ADC resolution, Comparator functionality, and PWM timers, it failed to elaborate on the importance of these factors on either performance or stability. While the structure was the simplest to follow, it was also less of an analysis in comparison to ChatGPT and Claude. For example, Claude was able to identify that ADC and comparators were required for it failed to elaborate that high sampling rates permitted more accurate readings and comparators were useful in detecting critical fault events that could then be relayed back to the MCU in order to make readjustments as necessary. This lack of detail limits the practical value of understanding how each of these components works and fails to show how the system interacts in order to ensure optimal performance is achieved.

On MCU recommendations, while it was able to make a similar list to the covering the same MCU as Gemini and ChatGPT. However it only provided a sentence as to why the MCU was picked for consideration. An example of this was how STM32 Series was known for fast response times. This is in contrast to other LLMs that brought out their in-built peripheries that enabled the MCU to have these characteristics. Overall, Claude's answers were concise and are suited to the introductory level who want a general overview on the device, but it does not have the in-depth reasoning required for an in-depth engineering evaluation.

When taking into account all LLM responses. ChatGPT provided the most balanced response out of the two LLMS by providing the theoretical explanation as well as an application portion. Gemini focused more into the peripherals of the MCU and hardware configuration. Lastly, Claude delivered a clear but surface level overview of the MCUs.

Case study 3: Question - What considerations would need to be made to communicate with an MCU to SBC to allow for potential web interface to be sent to the SBC to be sent to the MCU?[3A]

ChatGPT [3B] focused more specifically on the communication protocols and how to send data to and from the MCU to SBC. ChatGPT excellently summarized the communication protocols' pros & cons, GPT was also able to provide some side commentary on the general applications these protocols are used providing a nice context to the real-world applications these protocols will be used for. For the data formatting, while it was a brief explanation, it did provide the strengths and weaknesses of binary vs text-based protocols and the tradeoff between efficiency and readability/troubleshooting. Some of the weaknesses of this LLM was on the SBC Web interface integration as it was unable to go beyond a surface level explanation of the protocol and implementation. Overall, ChatGPT was able to discuss more on communication protocols between the MCU and SBC but was weaker overall on the web interface integration aspect of communication.

Gemini [3C] in contrast discussed evenly on the communication between the Web Interface and SBC to the SBC to MCU. Gemini was able to explain the terminology of the interaction between Web interface and SBC by going through the importance of each component. Furthermore, it provided languages and frameworks that would best be suited for the tasks mentioned. When approaching the SBC to MCU communication, while it didn't go through every protocol like ChatGPT did, it was able to mention the most applicable communication protocols and setups like USB, UART, and I2C. It further explained the specific application for each one. One weakness that Gemini did have was not explaining the cons of each communication protocol in the same manner as ChatGPT. Gemini did however explain what a user must do if they wanted to use SBC and MCU requiring different operation voltages. Overall Gemini was able to to focus more on the communication between web interface and the SBC as well as offer suggestions on frameworks and setup.

Claude[3D] was able to provide a quick reference to the important components of each communication line from interface to SBC and from SBC to MCU, generated a list of design constraints to consider such as Data serialization and flow control, ending on an architecture example. Claude was able to discuss a little bit on all the factors required in order for the communication between the three to be efficient whereas ChatGPT and Claude focused more on specific sections of the process. Despite this, it was unable to expand on the topics mentioned in its response and gave little information on what each part was and how to implement it on a deeper level compared to ChatGPT and Gemini.

5.1 Cross-Model Comparison and Observed Patterns

Amongst all three of the case studies, there are consistent patterns within each of the LLM approached engineering-focused prompts. In reference to ChatGPT, it is able to supply responses that incorporate balance, concepts, and systematic and practical overall reasoning when providing responses to users. In specific, it does an ideal job when describing why a specific decision of a design matters. However at times, ChatGPT would provide simplification over certain assumptions it may have made as well as questionable citations. However, when discussing Gemini this demonstrates hardware-centric orientation where it is able to reference electrical behavior, embedded systems, and architecture constraints. Gemini is especially strong when it comes to gathering an analysis on microcontroller peripherals; however, it remains to be weak when it comes to high level comparisons. Now, as for Claude, it was able to provide both

clear, simple, and concise responses. Claude is ideal for when it comes to brainstorming or overviews. However, when referencing engineering-level evaluation it tends to lack here. The summaries it provides tends to carry high accuracy, just not when it comes to detail and decision making steps.
Observing these trends helped to provide a good overview of when each LLM was used during this project. Here we can see that ChatGPT was used for theoretical explanation as well as in reference to application level reasoning. Here, Claude was used for quick idea validation so that high-level summaries can be accurately performed before diving into deep research. [4A]

5.2 Advantages of LLM Use in the Senior Design Process

Each model consists of various differences; however when we did integrate the various LLMs into the workflow listed below:

5.2.1 Faster Conceptual Alignment Among Team Members

The LLMs allows team members to obtain consistent explanations and information of various engineering concepts. Using this method, allowed us to reduce time when it came to searching through various texts that we were to reference and use in our report as well as help all members of the team to have a solid foundational understanding prior to discussing the design choices.

5.2.2 Accelerated Design Exploration

LLMs are able to generate quick lists of possible components that can be used, and communication methods. Later verifications are needed as well however these LLMs allow the ideation phase significantly. For example, when selecting the MCU for our project, using ChatGPT and Gemini allowed us to significantly narrow down the choices and compare and analyze the choices before looking at data sheets to further our research and go about our selection.

5.2.3 Immediate Access to Analogies and Clarifications

During our research, several concepts are to come up in our project. When a certain topic were to be unclear, for example ADC sampling, LLMs were able to provide other explanations and analogies to further help and breakdown the concept. This further deepened our knowledge on our project and was essential towards our overall understanding.

5.2.4 Early Error Detection

With the use of LLMs, this also provided a resource to detect any errors or anything that was unclear early on. This essentially acted as a quality-control mechanism. If a response were to disagree, this then signaled that separate research was needed to continue the deep research that was occurring. With this cross-verification, this essentially helped to prevent over-reliance on one model. [5A]

5.3 Limitations, Risks, and Ethical Considerations

Risks and limitations were also key to be aware despite the advantages that were mentioned above.

5.3.1 Citation Hallucinations

When using all three of the LLMs, they would occasionally provide incorrect or citations that were not verified. It was a crucial step to go back and confirm that the information the LLMs were producing and the citations matched. In reference to ChatGPT in specific, it would often list blogs as reliable or reputable sources, when this was not accurate. Gemini provided more professional sources however, double checking was still needed to confirm accuracy within the content as well as their respective sources.

5.3.2 Overconfidence in Answers

When using LLMs, and asking a specific question in which an answer is needed, it often provides an explanation that sounds confident; however, it can technically be incorrect at times. For example, when researching and understanding knowledge in reference to the capabilities of microcontrollers, several of the case studies had overestimated its capabilities or would provide incomplete explanations. As a result, it was needed to verify the technical output with the use of datasheets, documentation, as well as its respective manufacture reference materials depending on which manufacture was being referred to here.

5.3.3 Risk of Plagiarism

The way LLM content is generated, could increase the risk of plagiarism to occur. LLMs tend to cite the model appendix of prompts and outputs as needed as is especially necessary for academic integrity. The AI text has been cited accordingly as per the project guidelines.

5.3.4 Lack of Context Awareness

LLMs need an explicit description to allow them to see the entirety of the system. Most times they will overlook the design constraints, for example, safety margins, environmental conditions, or even real-world noise sources. Therefore, human engineering remains critical when researching. [6A]

5.4 Conclusion

When integrating LLM platforms this enables fast ideation, increase in clarity, as well as helps to improve overall decision making while collaborating throughout the Senior Design process. However, verification on our own was still necessary as LLMs are not always accurate or can provide incorrect information and data. In specific, this is in reference to electrical characteristics, system timing, as well as critical safety components. Through careful citation, cross-model comparison, and independent confirmation using reputable sources, the LLMs overall served as a valuable tool when it came to researching and selecting various components

for our Senior Design project. Independent verification was needed as previously mentioned, however the LLMs served as a vital tool without compromising the integrity in regards to the design process. [7A]

Chapter 6: Hardware Design

The hardware design of our device consists of two main components, the PCB and the physical interface on our device. There are several other elements that go into the Smart Device Charger or the Multi-port system for our project. This includes the web interface, SBC, LCD/buttons, MCU, USB ports, and more. All of these components are essential towards our overall design. A general design for the hardware diagram has been shown above towards the beginning of the report.

6.1 Power Delivery and Electrical Power System

For our multi-port automated device charger, this device essentially takes in a 12V DC input and further distributes a regulated 5V output. This would be amongst the four individual USB-A charging ports that are on the device. Note that each of the USB ports will be managed by the microcontroller and also incorporates specific and critical features such as monitoring voltage and current.

For the 12V input, this will be passed essentially through a protection stage. This includes a fuse, reverse-polarity protection MOSFET, as well as transient-voltage suppression diodes. A buck converter would be used here in this case so that there is enough current that is provided for the devices that are connected. This will be able to contribute towards efficiency as well as thermal stability in reference to this. There will be another 3.3V linear regulator which will help to power the sensing components as well as the microcontroller itself.

Furthermore, there is also the status LED circuit that will be included in both the 5V and 3.3V rails. This will be done with the use of a simple op-amp undervoltage detector. The LED will turn off if the conditions of undervoltage were to occur. This further allows it to apply visual features when verifying the power state of the overall device and those that are connected to it as well.

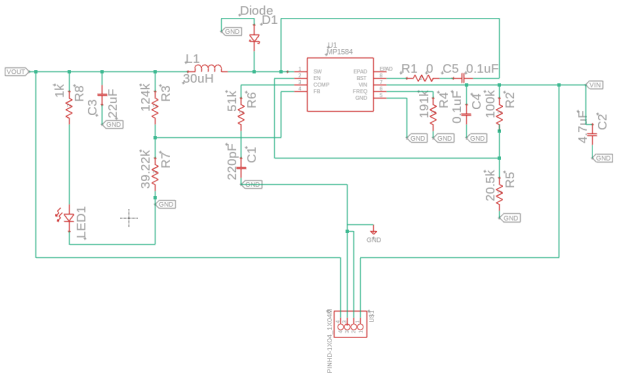


Figure 6.1A: Schematic of 3.3 Volt Regulator

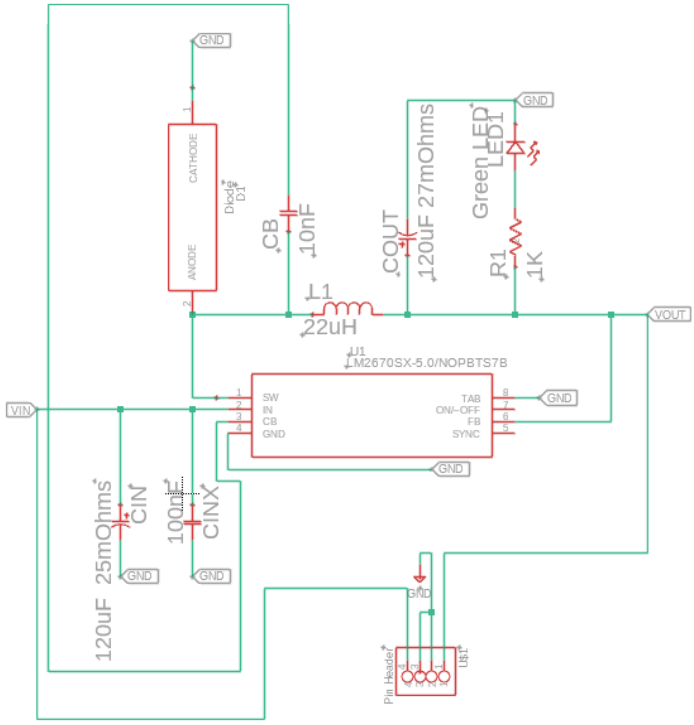


Figure 6.1B: Schematic of 5 Volt Regulator

To implement the required DC - DC regulation, the design incorporates the LM2670 as the primary 5V buck regulator. The LM267 is a high-efficiency switching regulator capable of delivering up to 3A of continuous output current but with our current configuration, the device is limited to 2A to ensure the IC load switch does not handle an excess amount of current flowing through. This regulator satisfies the need of supplying multiple USB ports when combined with the TPS22918 Load switch and the INA260 Current sense IC. The fixed 260 kHz switching frequency helps minimize the external component size while ensuring stable operation with minimal thermal stress.

For the low voltage control rail, the system uses an MP1584 3.3V buck converter to power the microcontroller, INA260 sensors, and other digital logic. The MP1584's ability to operate with a wide input range and excellent conversion efficiency ensures the 3.3V rail remains stable under dynamic conditions. Together, these regulators will be able to provide a robust and efficient power architecture that will enable the device's multi-port charging capabilities while maintaining safe operation and electrical reliability across all subsystems.

6.2 USB Power Switching and Control

A load-switch IC will be used here for each of the USB charging ports and this will be controlled using a GPIO line which will be from the MCU. This is critical as it allows for the user to turn on/off charging on their respective USB ports that will be on the device.

A current-sensing resistor will also be placed in series with each of the outputs. This is so that it can essentially monitor each port in terms of how much current is being drawn or released from each of the port outputs. An INA260 current-sense amplifier will then also be used to amplify the voltage that has been sensed. This then allows for communication purposes with the MCU over the I2C bus and ultimately allows for readings to be produced or sent in real-time.

These readings are used in a way that allows them to automate charging. The MCU will turn off the respective port if a specific current were to drop a certain threshold that has been defined. This is efficient as it allows the port to save or conserve energy while also maintaining and incorporating overall battery health.

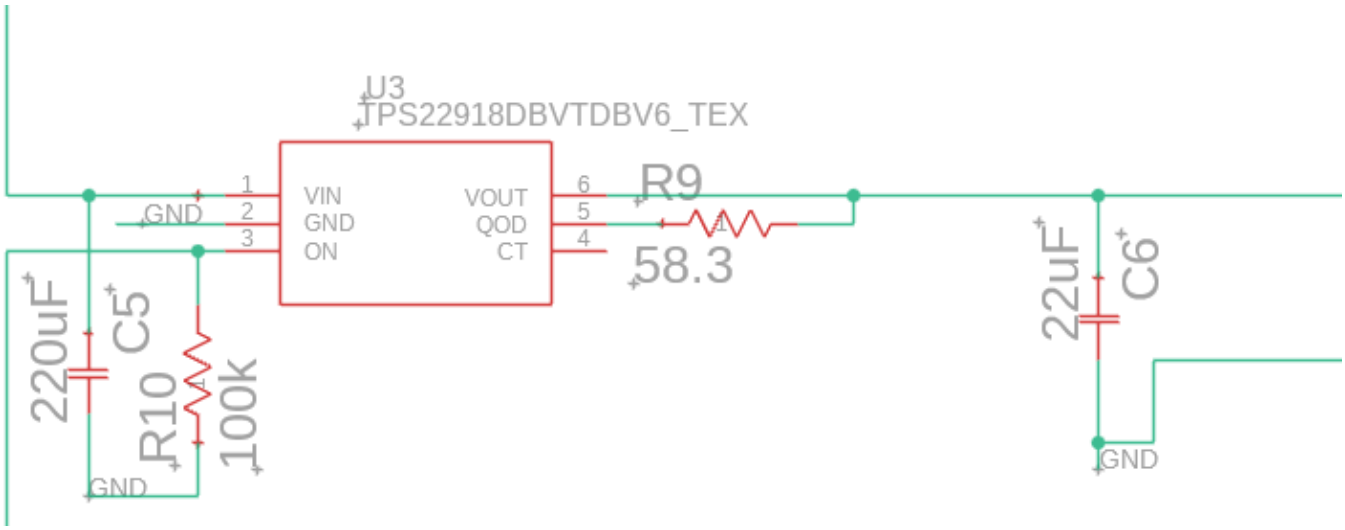


Figure 6.2A: Schematic of IC Load Switch

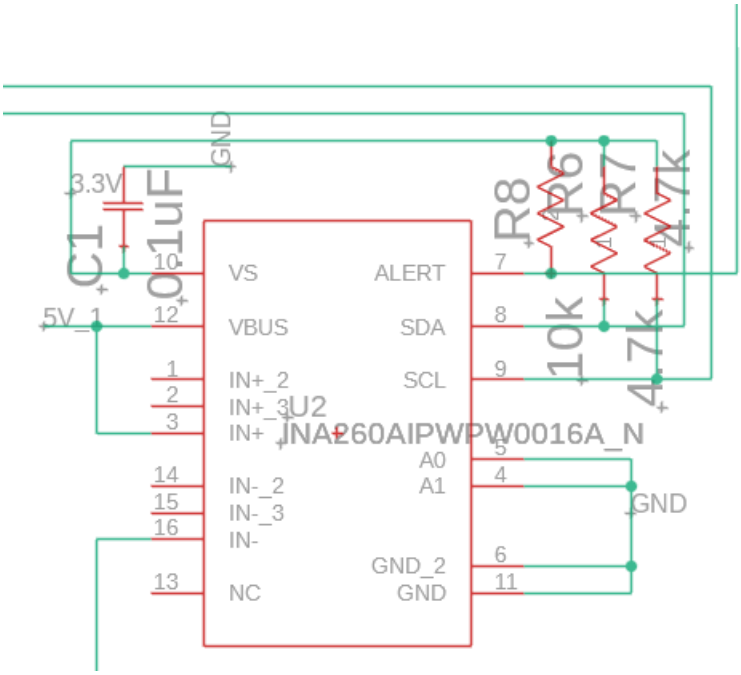


Figure 6.2B: Schematic of INA260 - Current and Voltage Measurement IC

In addition to enabling user-level control, the TPS22918 provides several critical electrical protection and power management features that enhance the safety and longevity of each USB charging port. The TPS22918 has a low-on-resistance MOSFET that minimizes voltage drops while ensuring support for high current draw that can be seen during USB plug in. One of the key advantages of the TPS22918 is the Quick Output Discharge which allows the port voltage to rapidly drop to ground when the load switch becomes open. This ensures that devices disconnect cleanly and eliminate ghost powering or unpredictable behavior when ports switch states.

The TPS22918 also provides reverse current blocking, preventing current from flowing from the output into the INA260 and regulator. This safety feature is important when dealing with USB devices that have their own internal batteries or power sources. By utilizing a TPS22918 at each USB port, the system will maintain a strict electrical isolation between channels. This will allow for continuous system operation in the event of a fault condition occurring at one of the ports without disturbing the remaining ports from operating. The modular approach will ensure greater fault tolerance and will enable for simpler debugging and maintenance.

The INA260 current and voltage monitoring IC is a perfect companion to the load switch by providing accurate and real-time measurements of the electrical characteristics of each port during operation. The INA260 contains a precision 2mΩ, This eliminates the need for a discrete sense resistor to be connected externally to the INA260 which could introduce variability and thermal drift. The device allows for continuous current, voltage, and power reporting over the I2C bus, giving the microcontroller information pertaining to the port to determine if the low switch should be open or closed. Furthermore, these measurements allow the MCU to identify abnormal conditions such as overcurrent events, short circuits, or sudden load changes.

By combining the TPS22918 load switches and INA260 sensors, each USB port effectively becomes a self contained intelligent power channel. The MCU will then use these measurements to enforce automated shutdown rules, allocate power, and protect the system from either unsafe or highly inefficient operating conditions. This system architecture mimics the behavior of modern commercial smart chargers where each port is independently controlled, monitored, and protected. This configuration setup ensures a stable operation under various range of conditions while also supporting energy savings and improved system stability by shutting down unused ports. Overall the integration of both ICs enhances the functionality and safety profile of the multi-port charging system.

6.3 Microcontroller and Communication System

This device is to be built around the ESP32 microcontroller that is being used in our project. The ESP32 allows for multiple analog input channels and also has integrated Wi-Fi features that is a necessary feature towards our overall design. The ESP32 is able to handle USB port control logic, web server hosting for report monitoring, data acquisition, and user interface management which deals with both the display and Wi-Fi.

The MCU will operate from the 3.3V regulator. The I2C lines will then connect the MCU to the current sensing ICs and the display itself. GPIO pins will also be mapped for its respective MOSFET gate driver as well as the LED indicator determining the statuses.

Below will be the full schematic for the Load Switch and Current measurement. It will support a total of four output loads each supported by its own IC Load Switch and INA260 allowing for independent operation and measurements. First, the regulated output voltage will enter the Vin pin of the IC Load switch. Next, the output voltage will go through both the VBus and IN+ of the INA260. Finally on the IN- Pin is connected to the high side of the USB load. This design also ensures minimal power losses to occur such that minimal heating occurs in order to avoid triggering the thermal shutdown of any IC device utilized in the design.

Due to the size of the design; Below are two schematics with one being the complete design and the other figure being a more compact design enabling for easier viewing of the interconnections.

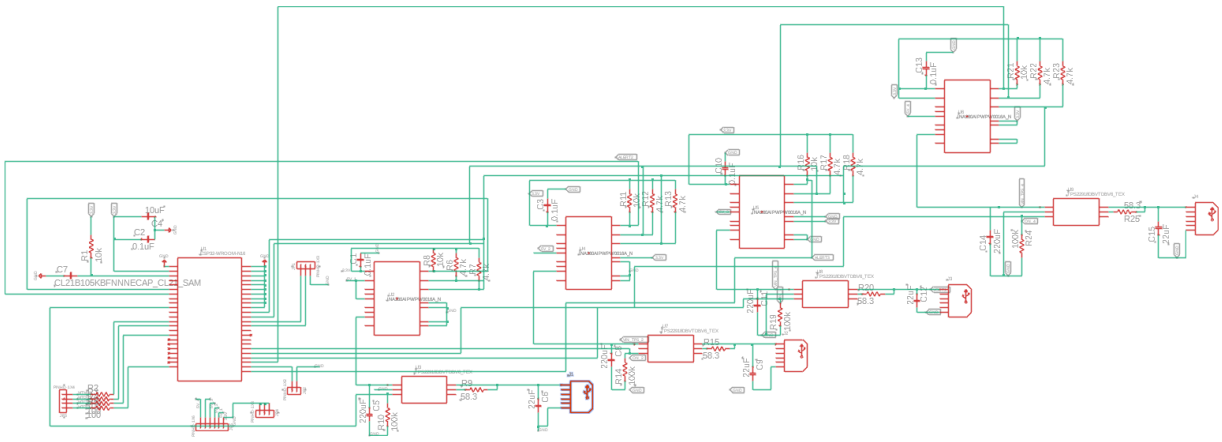


Figure 6.3A: Full Schematic of Charging Circuit - All 4 outputs

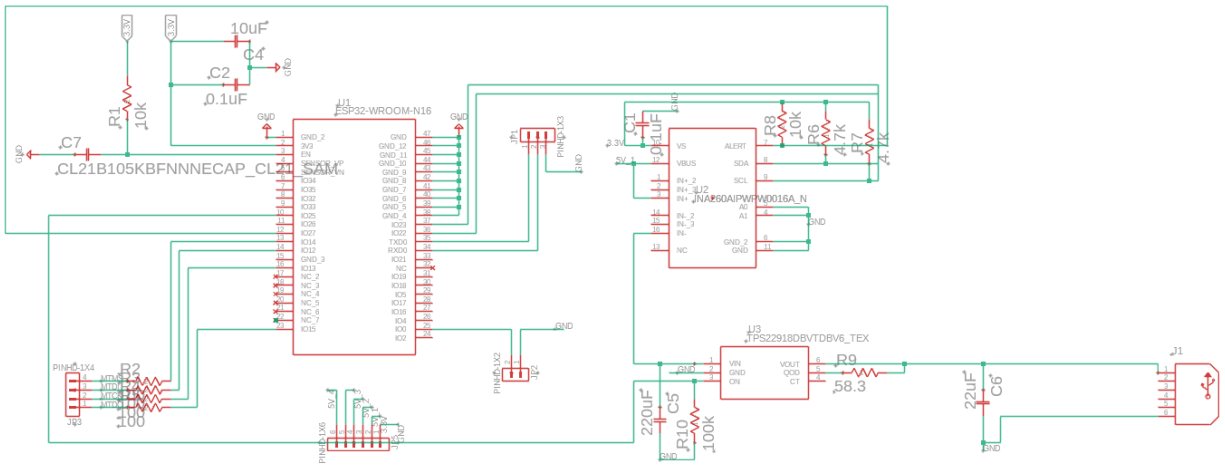


Figure 6.3B: Schematic of Charging Circuit - 1 Output.

The I2C interface also simplifies the system wiring and expandability. By allowing all four INA260 sensors to share the same two-wire bus permits the design to minimize the PCB trace complexity and reduces the number of MCU pins required. Each sensor has its own configurable I2C address which permits the ESP32 to read the measurements and send it to the ESP32. This centralized bus architecture allows for efficient polling routines and precise fault handling, this is

apparent when it takes only a few milliseconds for all the data to be obtained by the MCU. In the event of an overcurrent and irregular charging conditions, the ESP32 can immediately disable the corresponding load switch which protects the USB device and prevents load disturbances from propagating in the system.

The GPIO mapping provides additional structure to the control logic Dedicated pins are assigned to the gate-enable lines of the TPS22918 load-switch ICs, allowing for the ESP32 to actively control the power state of each port with microsecond precision.

Ensuring Proper power routing in the schematic also contributes to system reliability by placing each INA260 in a high-side configuration, this design allows for measurements in the current on the positive rail prior to the current reaching the USB device. This prevents ground disruptions and allows for accurate monitoring despite any variations caused by the downstream circuitry, USB cable, or load behavior. This allows for the micro controller to detect unexpected behavior such as leakage currents, short-circuit conditions, and abnormal voltage levels before power is applied to the USB device which further adds an additional layer of safety and intelligence during the sequence.

Another benefit of this topology is that it enables the firmware to perform pre-enable diagnostics for each port. An example of this would be when a device is plugged in, the INA260 can detect even small changes along the line, this allows for the MCU to verify if the port is electrically safe before turning on the TPS22918. If any port exhibits abnormal idle behavior, the MCU can choose not to energize the load switch and can flag the issue to the user via the display. This proactive behavior is essential in a multi-port system where reliability and controlled charging are priorities to ensure device operations.

Once a port is enabled, the INA260 continues to supply current and voltage measurements in a matter of milliseconds, this allows for the microcontroller to enforce per-port charging rules, implement automatic shutoff thresholds, and detect device removal events. The ESP32 can accurately determine the dynamic behavior of each load without being affected by any switching transients or discharge behavior that can be caused by the TPS22918 as the INA260 is positioned at the output of the 5V regulator directly. This clean measurement contributes to a more stable closed-loop control and reduces false triggers in the automated port management algorithms

Overall, the positioning of the INA260 strengthens the safety, diagnostic capability, and operational intelligence of the system. It ensures that each port is monitored continuously even when disabled and provides the MCU with the most recent data from which to evaluate charging conditions and perform controlled reliable power distribution.

6.4 System Hardware Block Diagram

The figure displays the design or architecture of the system’s hardware. Here it can be shown the power flow that stems from the DC input that will then go into the USB outputs as well as the control paths from the microcontroller to each of the functional subsystems.

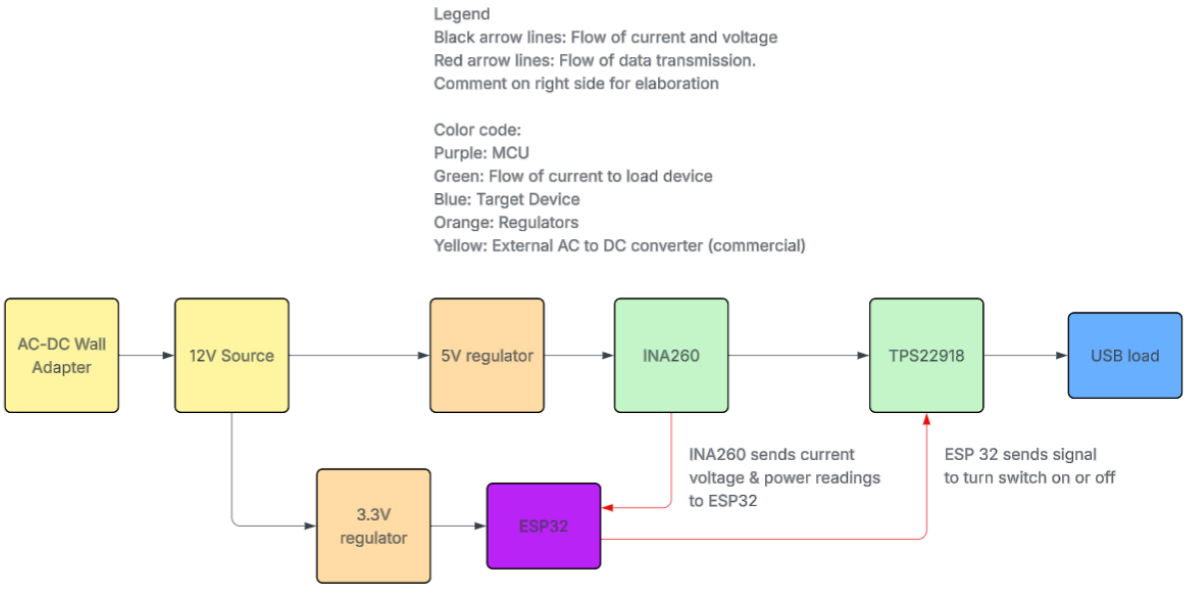


Figure 6.4: System Hardware Block Diagram: Multi-port Automated USB

6.5 Safety and Protection Circuitry

Safety is critical when dealing with electronics and other electrical components when constructing a device. There are several protections that are enforced here to ensure that this involves a safe operation. The following, over-current, over-voltage, thermal, and reverse-polarity are safety mechanisms used here. Over-current protection deal with current-sense feedback. Over-voltage protection is in reference to protection with the TVS diodes. As for thermal protection this is incorporated within the voltage regulators. Finally, with reverse-polarity protection this is a safe-guard on the input line.

If any of the protection conditions are to be set off, the MCU will then turn off the ports that have been affected. This then will inform the user through the display and the web interface that is being used.

The ESP32, TPS22918, and INA260 all have bypass capacitors placed directly at their supply pins to stabilize their local power rails. This protects the components from high-frequency voltage spikes and sudden load transients. Without these protections, this would cause the ESP32 to reset when unprompted or the INA260 to enter undefined states or inaccurate operating states.

Moreover, consideration for the input capacitance and output capacitance was required to minimize the input voltage dipping during high-current switching events. To mitigate this, the design follows the manufacturer-recommended practice of using a 10:1 ration between Cin and CL. Cin allows for the stabilization of the upstream 12 V supply and CL supports the regulator's output. Maintaining this ratio ensures consistent transient response, reduces voltage sagging when multiple USB devices are plugged in and prevents false triggering of undervoltage or brownout conditions in the power management circuitry.

The placement and sizing of the input and output capacitors also contribute significantly to the overall system robustness. For example, the LM2670 requires low-ESR capacitors placed close to the switching node to ensure stable operation and prevent oscillations. Similarly, the MP1584 3.3V regulator benefits from ceramic capacitors positioned at the Vin and Vout pins to ensure a clean ripple-free supply for sensing and control electronics. These capacitors will act as a local energy carrier, supplying instantaneous current during transient loads and preventing rapid voltage changes that could propagate down steam to sensitive components like the INA260 and ESP32

The INA260 is positioned before the TPS22918 load switch in your design, the sensing stage remains isolated from switching disturbances and provides updated current measurements whether the port is enabled or disabled. When the load switch is turned on, inrush current is captured by the INA260, allowing the MCU to detect the abnormal behavior and force the port to shut down. When the port is turned off, the INA260 will report a zero current while remaining fully active on the I2C bus ensuring the measurements do not reach an undefined state during transition.

The TPS22918 supports a controlled slew-rate and Quick Output Discharge (QOD) features to further improve safety. The controlled slew-rate manages the rate at which the output rail rises, this prevents any current spikes that may cause the voltage to droop on the 5V bus. The QOD path ensures that when a port is disabled, the voltage would rapidly discharge to ground. This would prevent ghost powering or latch-up conditions that would confuse the current-measurement logic. The power down behavior will be important for the measurement system the INA260 will be in charge of as this QOD allows for predictable transitions and a decrease likelihood in reaching undefined states.

All of these protective mechanisms would work in conjunction under the control of the ESP32 which would gather the sensor data and system conditions in order to determine if operation is done under normal conditions. If any abnormal event occurs such as over-current, over voltage, excessive temperature or reversed polarity is detected, the MCU will execute a shutdown of the affected port or the whole system depending on the severity of the issue. The web interface and onboard display then will notify the user with a diagnostic message. This interaction with both hardware protection and firmware decision making ensures electrically safely of the load device and user.

6.6 PCB Layout

The PCB will be designed using Fusion360 and will be able to display the wide 5V traces in reference to the high current, the two-layer design with the solid ground plane, analog and digital zones so that noise can be reduced, and decoupling capacitors which will be placed near the ICs.

Overall this hardware design promotes a safe, smart, and a charging solution that is automated and can also handle several devices at the same time. This incorporates a combination of strong power delivery for each of the ports as well as wireless monitoring. This can further satisfy the both of the basic and advanced objectives that have been previously defined or stated above.

Chapter 7: Software Design

The software of our charger design consists of two parts: the microcontroller unit software (which must manage all of the communication and logic in the charger), and the single-board computer software (which must manage. Each part has many sub-parts, which is described in detail in the below sub-sections.

7.1 Microcontroller (MCU) Functionality

The microcontroller unit (MCU) we have chosen for this project is an ESP32, as detailed in Chapter 3. The microcontroller must interface with two main parts of the hardware: the current and voltage measurement, and the port controls. It must also interface with the single-board computer (SBC), in order to provide the necessary details to the user and web interfaces. For this, we will be using FreeRTOS, a real-time operating system which allows for separate tasks and synchronization primitives.

7.2 MCU System Overview/Block Diagram

Since our ESP32 microcontroller has two cores and fairly advanced hardware, we need a software package that can utilize the extra hardware capabilities afforded to us. FreeRTOS allows for a large amount of tasks to run, with different priorities. A task can be pre-empted by another task, and tasks of the same priority can run in "time-slicing" where running time is shared, allowing full use of the ESP32's modern hardware design.

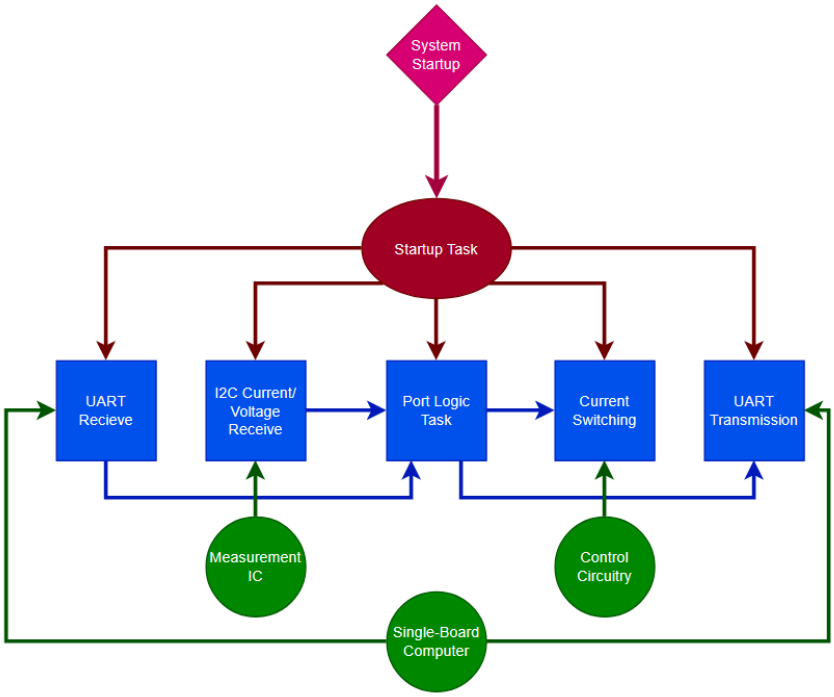


Figure 7.2a: MCU Task Block Diagram

Tasks running in FreeRTOS need to communicate with each other via shared data or some form of task notifications. With shared resources come concurrency issues, so we must keep track of all shared resources in order to avoid bugs such as race conditions. The current, voltage, and power for each port will be written to by the I2C Current/Voltage Receive task, and accessed by the port logic task and UART transmission task. The current threshold for each port, as well as the port timeout, are set by the UART receive task, and used by the port logic task. The on/off status for each port is set by the port logic task, and used by the UART transmission and current switching tasks. Our shared resources are also shown in the below diagram.

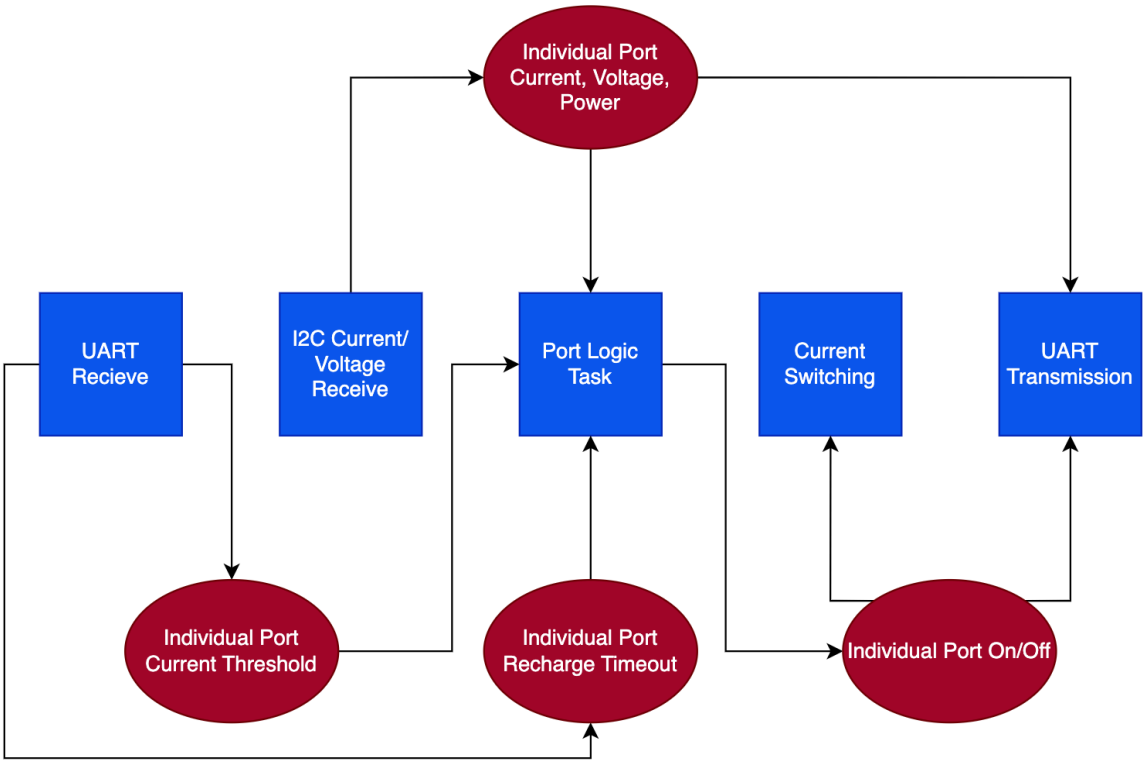


Figure 7.2b: MCU Shared Resource Diagram

7.3 Task Overview

7.3.1 Startup

The startup task, referred to as app_main in FreeRTOS, will contain all of the setup code for FreeRTOS, any synchronization primitives, as well as any communication protocols, timers, or interrupts used. UART and I2C will need to be initialized. It will run once, when the system is booted, before launching all of the other tasks. When launching other tasks, priority can be set, such that a higher priority task can interrupt or pre-empt a lower priority task. The startup task is extremely important, as without it, no other task will be able to be set up or run.

7.3.2 UART Transmission

The UART Transmission task takes in the values of the current, voltage, and power received and calculated by the port logic task, and sends it to the single-board computer over UART. It will be called on a periodic basis in order to ensure that it does not re-transmit out of date data. JSON format will be utilized in order to effectively categorize data into port number, voltage, current, and power for the SBC's user interface.

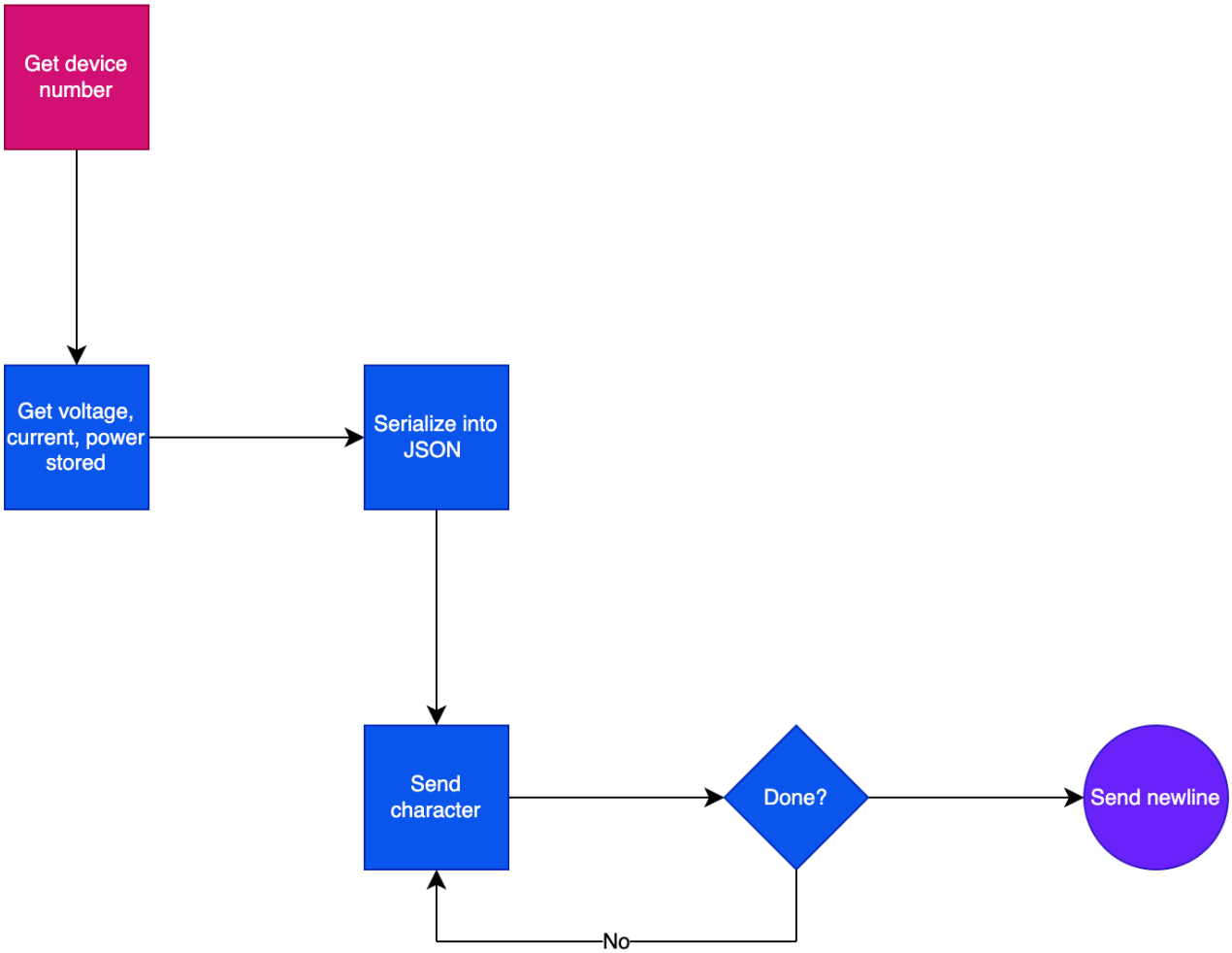


Figure 7.3.2 UART Transmission Task Flowchart

7.3.3 UART Receive

The UART Receive task will retrieve and store any data packets sent via UART, which will set the current thresholds and charge timeouts for each port. The current threshold allows the charger to turn off when the current drops below the received value. The charge timeout allows the

charger to stop charging a device for a set period of time. This task runs periodically in order to check for new UART transmissions from the single-board computer.

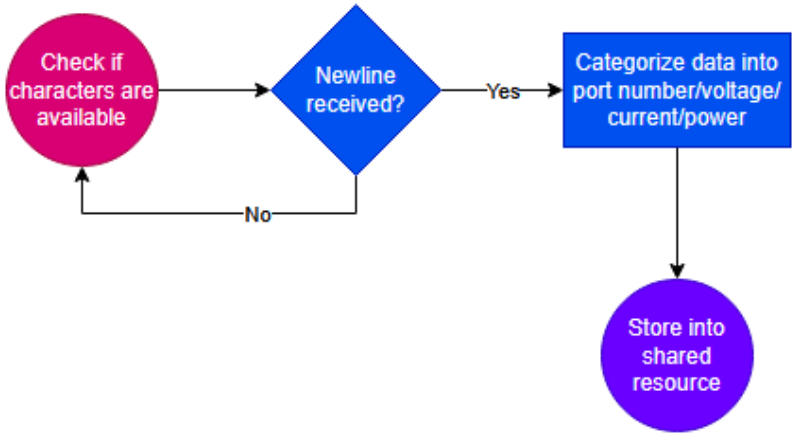


Figure 7.3.3 UART Receive Task Flowchart

7.3.4 I2C Current/Voltage Receive

The I2C Current/Voltage Receive task's job is to get the value of each port's current and voltage through the INA260 measurement IC. Then, the value is updated and stored as a shared resource. It is imperative that this task get data quickly and run often in order for the charger to respond quickly to a drop in current or a new device plugged in. Therefore, this task will run periodically to check for new data.

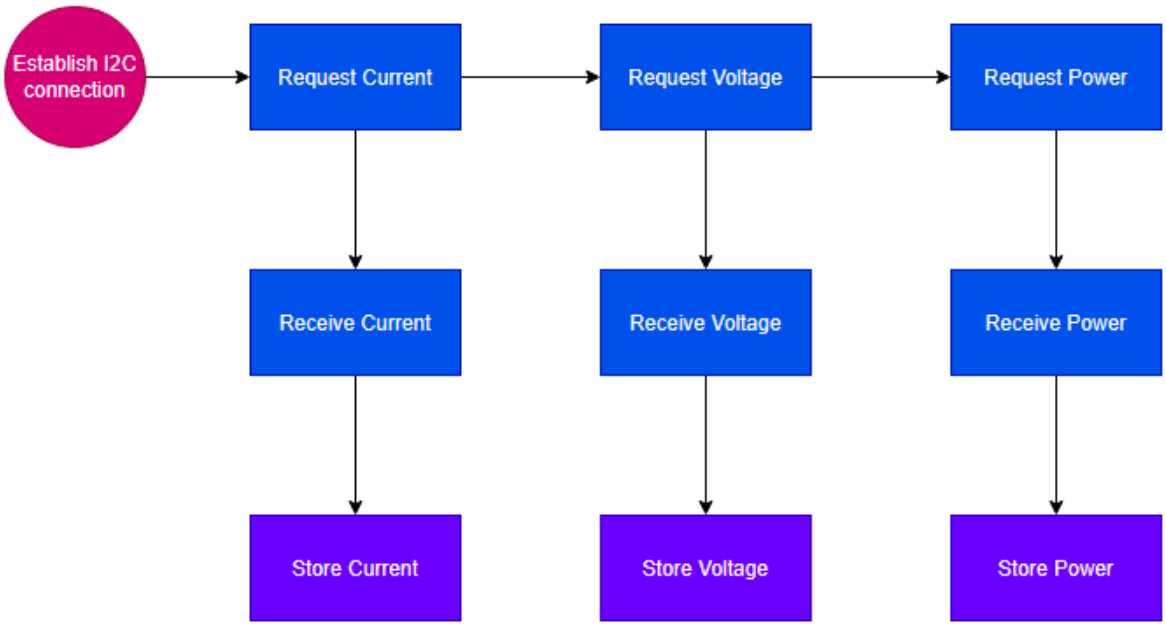


Figure 7.3.4 I2C Current/Voltage Receive Task Flowchart

7.3.5 Port Logic Task

The port logic task is the main task in the system under normal operation, and must interface with every other charging task. The port logic task's job is to take in the results of the UART receive task (if any change), then check the values of each port via the I2C Current/Voltage Receive task. Each value is put into a moving average, and the average value is computed. If the average for a port drops below the set minimum, the current switching task is signaled to turn off the port. If the timer has expired for a given port, it will then turn on a port. If it is detected that a new device has just been plugged in, it will also decide to turn on the port to allow the new device to charge. This task will run periodically and with a higher priority.

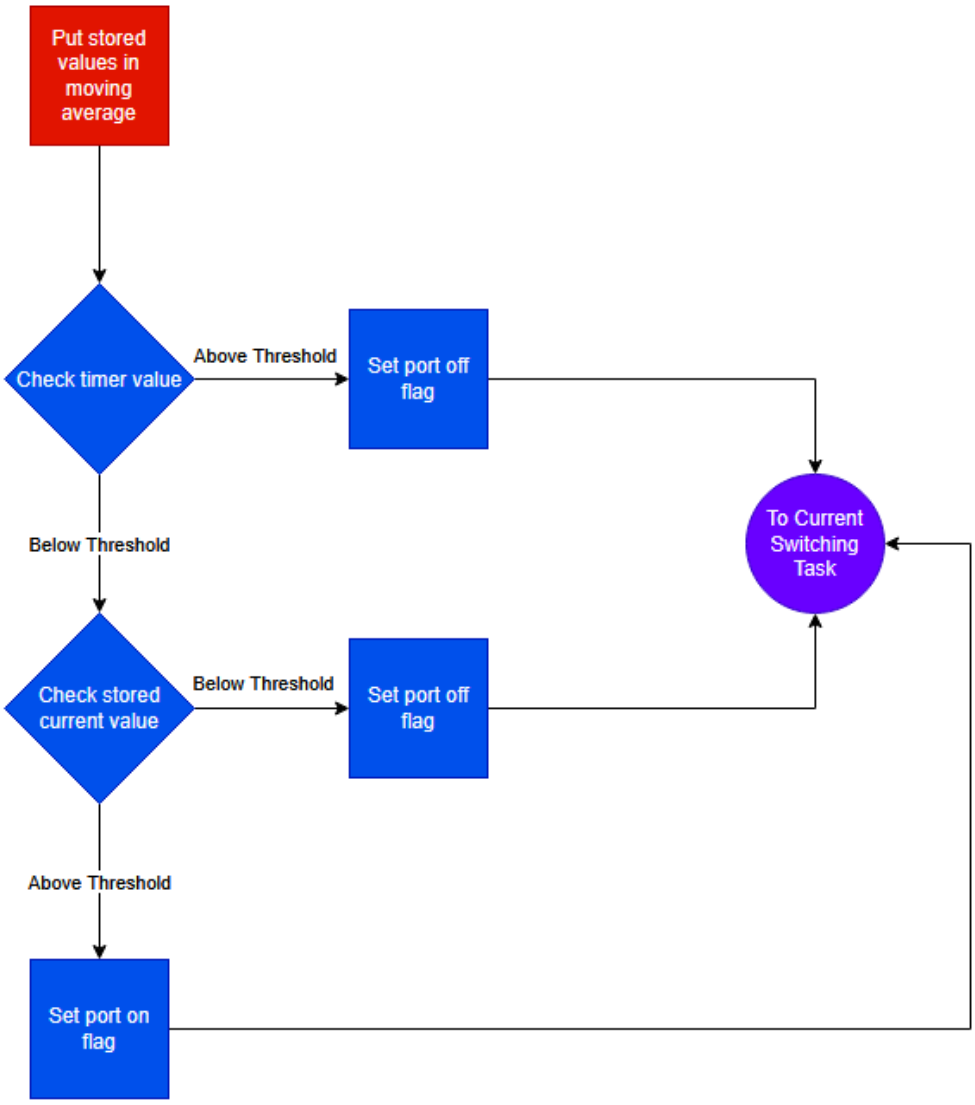


Figure 7.3.5 Port Logic Task Flowchart

7.3.6 Current Switching Task

The current switching task's job is to change the output to the current control circuitry in order to turn a specific port on or off. This is another important component in the charger's response time, as it actually allows the device's charging to be directly altered. It runs periodically in order to respond quickly to events decided by the port logic task.

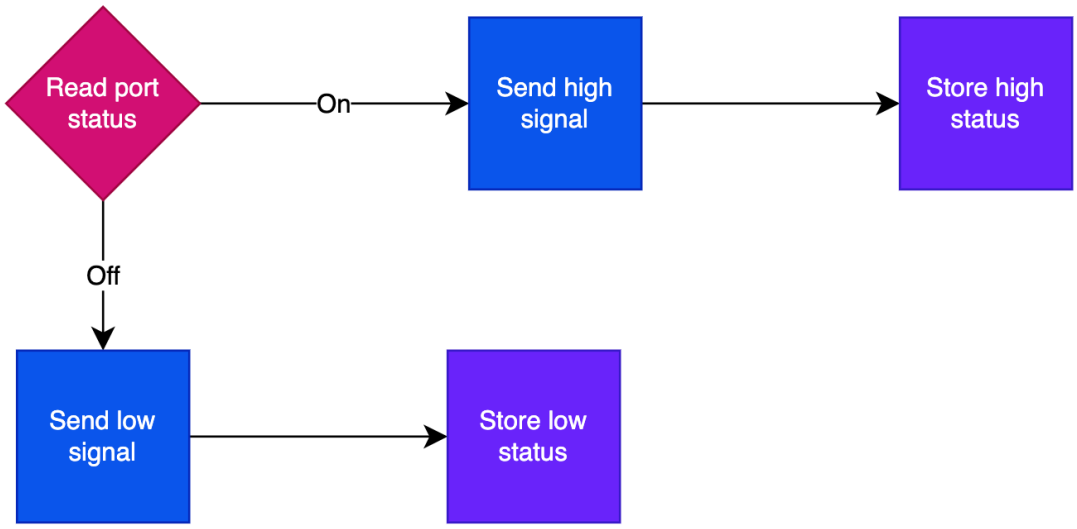


Figure 7.3.6: Current Switching Task Flowchart

7.4 SBC Software Overview

The Single-Board Computer (SBC) serves as the processing layer that connects the firmware of the microcontroller to the interfaces seen by the user. The SBC chosen for the smart charger is the Raspberry Pi 4 Model B, and its tasks are to receive real-time data through UART, store the information locally, and provide simple controls to allow users to configure the timer and threshold of each charging port. The SBC will continuously receive messages from the MCU that contain the voltage, current, and timer of a specified port, while also receiving user commands that configure or adjust the threshold and timer of a specified port.

The SBC runs on Raspberry Pi OS, which is a lightweight Linux-based operating system optimized for the Raspberry Pi 4. This operating system allows multitasking and network connectivity, enables UART communication, database management, and can host a user-interface. The SBC software consists of a backend server, a local database, a local frontend interface, and a web-based frontend interface. These components will work together to set up seamless communication between the user and the smart charger, whether it's physically at the device or remotely over a network.

7.4.1 SBC Startup Behavior

When the SBC is plugged in, it will initiate its startup sequence. The Raspberry Pi will boot into Raspberry Pi OS and start its backend initialization. This is where the Linux kernel initializes core system services necessary for the smart charger, such as the filesystem, UART drivers, and networking stack. Once the operating system reaches its target state, several custom systemd services begin to ensure that the smartcharger software launches reliably and without the need for user intervention. One of these systemd services is responsible for starting the backend server, which is implemented by Node.js, Express, and a WebSocket manager. The backend is responsible for establishing UART communication with the ESP32, hosting the REST API, managing real-time data streaming, and interfacing with the SQLite database. A second systemd service is responsible for launching the Tkinter-based local user interface in fullscreen mode, bypassing the desktop environment so that the SBC behaves more like a dedicated device rather than a traditional computer.

Once the backend is running, the SBC will perform its initialization routine. The backend begins by opening the UART communication and configures the port with the specified baud rate and communication parameters. The SBC will clear corrupted or partial data left in its UART buffer and wait for its first valid JSON-styled format. While in this stage, there is a possibility that the ESP32 has not finished booting, in which the SBC will perform periodic retries until communication is established. Once UART communication is synchronized, the backend will load configuration data from its SQLite database. If the database has not been created yet, the backend will automatically generate it from the required schema. After loading these settings, the backend will transmit them to the ESP32 to ensure that the backend, microcontroller, and database all share the same configuration settings immediately after startup.

When the backend is fully initialized, the Tkinter local interface loads and is displayed on the Raspberry Pi touchscreen in fullscreen kiosk mode. The interface connects to the backend to retrieve the most recent port information and displays real-time voltage, current, and power readings as they become available. The Tkinter UI does not communicate directly with the ESP32, but instead all interactions happen in the backend server to ensure that UART communication is handled by a single, centralized manager. This will prevent resource conflicts and ensure that configuration updates from either the local UI or the Web Application follow the same processing path.

The smart charger is designed without a dedicated power switch or a reset button. Its behavior will resemble a consumer appliance. Once the device is being supplied electrical power, the system boots on automatically, launches all necessary software components, and restores the charger to its previous configuration settings. If power is interrupted, the SBC will restart and reconnect with the database and the microcontroller. The use of systemd will also ensure that the backend and the user interface will also restart automatically if there is an unexpected failure or power shortage. Figure 7.4.1 illustrates the startup procedure that the SBC will follow every time it is booted on.

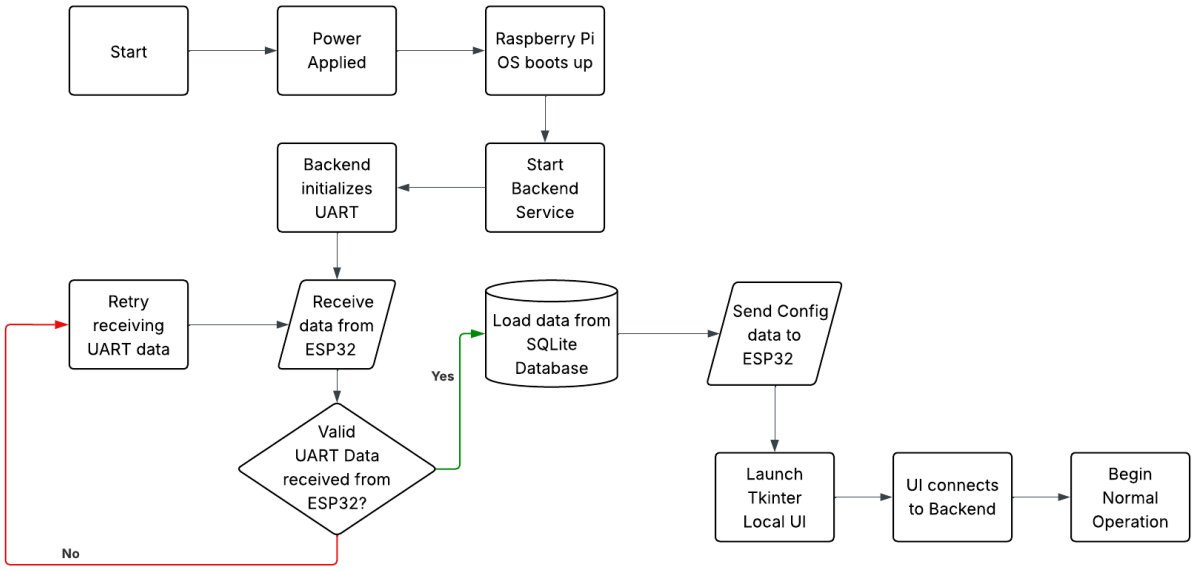


Figure 7.4.1: SBC Startup Behavior Flowchart

7.4.2 SBC System User Configuration Flowchart

The smart charger's configuration setup is designed so that both the local Tkinter interface and the Web Application interact with the same backend logic. This ensures that all configuration changes are processed consistently and stored centrally. The backend server acts as the coordination layer that validates user inputs, updates the database, and communicates the changes to the ESP32. This approach is intended to prevent conflicts between interfaces, ensures data integrity, and maintains synchronized system behavior across all user-access points.

Before a configuration update is applied, the backend performs several checks. It will begin by validating the incoming data to ensure it follows proper JSON formatting and falls within the acceptable operational limits. This check includes that the current threshold values are valid decimal values, recharge timer values are non-negative numbers, and that no fields are left empty or malformed. Once validated, the backend will write the updated values to the SQLite database, which will guarantee that the smart charger can persist user settings across power cycles.Storing those values in the database first ensures that the system state remains consistent even if the ESP32 is temporarily unavailable.

After the database is updated, the backend will prepare a UART command packet in JSON format and transmit the updated configuration to the ESP32 microcontroller. The ESP32 processes the update and applies the new setting to the specified port. At this point, the backend is monitoring the UART channel to receive an acknowledgement that the ESP32 received the configure settings command. If no confirmation is received, or if the ESP32 is currently disconnected, the backend retains the updated configuration in the database and attempts retransmission once communication is restored. This ensures that configuration values remain consistent, correct, and eventually synchronized with the hardware, even during brief communication loss.

Figure 7.4.2 illustrates how the SBC will handle configuration requests initiated by the user, such as modifying a port's timer or current threshold. The user can choose to modify a specified port's configuration either on the local UI or the web application. When the user modifies either a port's timer value or a port's current threshold, the SBC will record that value and save it in the SQLite database, overriding its previous value. Once the configuration is stored, the SBC will then transmit the updated value to the ESP32 microcontroller through the UART interface. This will ensure that both the database and the microcontroller are synchronized with the user's preferences.

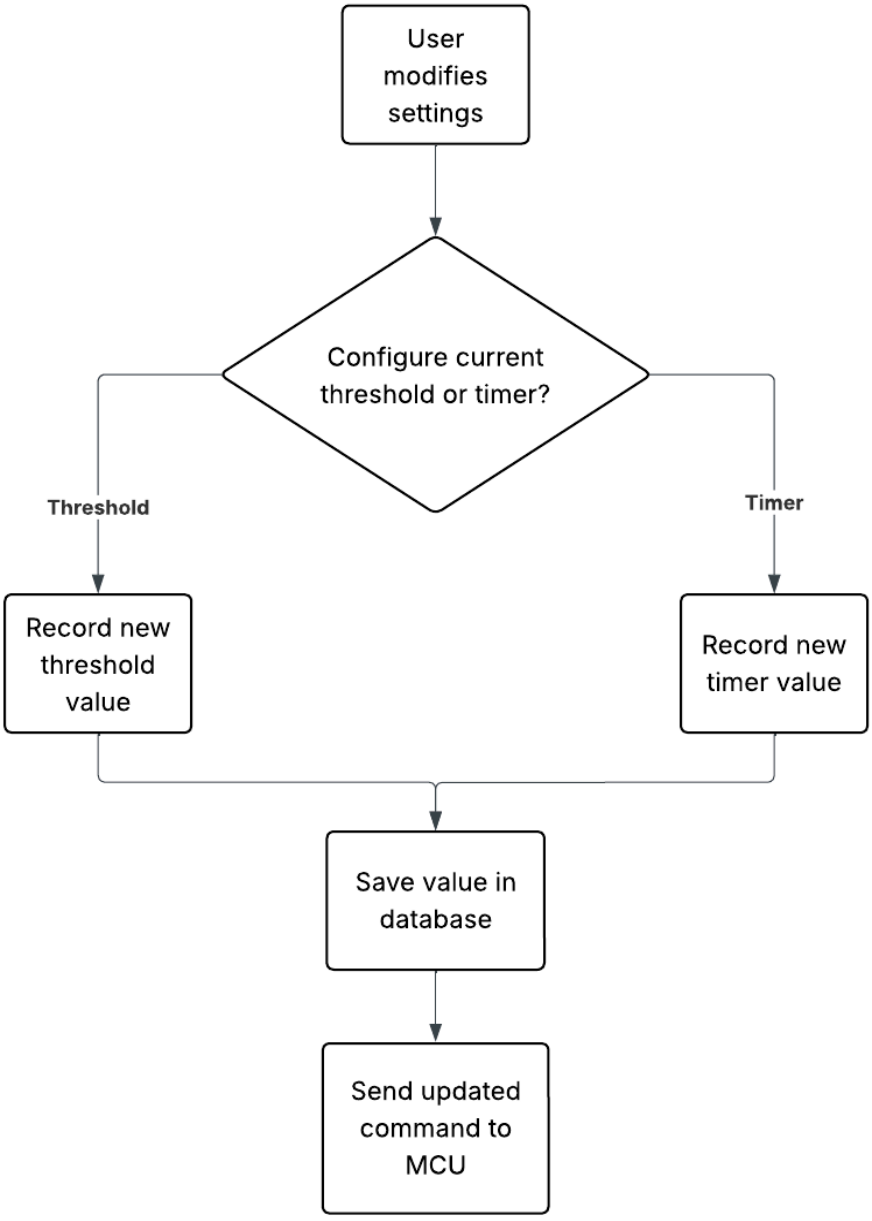


Figure 7.4.2: SBC User Configuration Flowchart

In addition to storing and transmitting the updated values, the backend will also update the local UI and the Web Application's interface to reflect the new changes. This includes refreshing the displayed current threshold and recharge timer values and sending real-time charging status through the WebSocket connection to ensure consistent and accurate information across all devices. Because the SBC treats the backend as the source for data configuration, the user interfaces do not store or compute any values locally; instead, they request and display the data managed by the backend.

7.4.3 SBC Data Processing and Display Loop

Figure 7.4.3 illustrates the data processing and display loop, which represents the continuous operation of the SBC as it receives and processes charging data in real-time from the ESP32 microcontroller. The backend constantly monitors the UART connection for new messages containing voltage, current, and power data for each port. When a new UART message is received, the SBC will parse it to and retrieve the updated values and update the values displayed on the SBC's local UI and web application's UI. If no new messages are received, then the SBC will continue waiting while displaying the most recent information. This loop is what enables live monitoring of the smart charger's ports while ensuring that the user interface accurately displays this data.

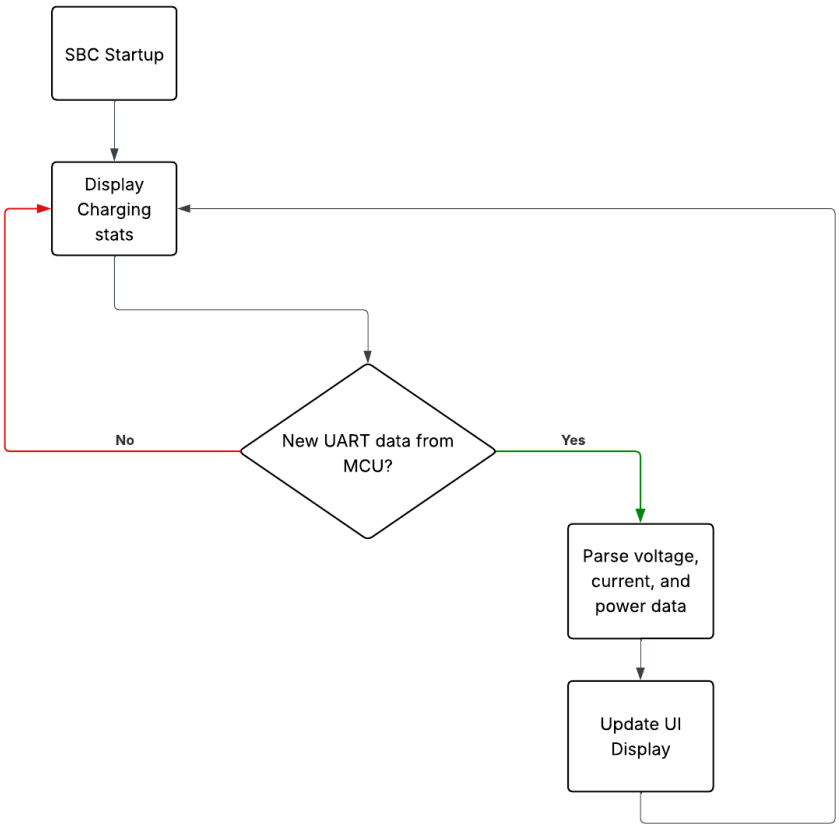


Figure 7.4.3: SBC Data Processing and Display Loop

7.4.4 SBC System Use Case Diagram

Figure 7.4.4 showcases the Use Case Diagram, which shows how the various components of the smart charger interact with each other. The diagram defines the SBC System as the boundary, which contains the actions that are done within the SBC. Actors outside the boundary would include the User, the SQLite database, and the ESP32 microcontroller. The user is primarily responsible for initiating all configuration and monitoring actions, while the ESP32 is responsible for supplying real-time electrical measurements and executes updated configuration commands received from the SBC.

The Use Case Diagram has two primary user-driven interactions: viewing real-time charging statistics and configuring individual port settings. Although the user may interact through either the local UI or the Web Application, both interfaces communicate exclusively with the SBC backend. This design ensures a unified control flow, consistent data handling, and synchronized system behavior regardless of which interface is used. The backend plays a critical role by performing input validation, updating the SQLite database, relaying configuration commands to the ESP32, and broadcasting live data updates to all active clients. This prevents conflicting actions between interfaces and ensures that all system components operate on the same authoritative state.

Figure 7.4.4 illustrates the Use Case Diagram for the SBC System of the Smart Charger. The diagram shows how the SQLite Database, ESP32 microcontroller, and the user all interact with each other through the SBC software. The user has two primary actions when interacting with the SBC: viewing charging statistics and configuring individual port settings. One of these interactions is view charging stats. The ESP32 provides the charging stats of each port, in which the SBC then takes each of the port's information and displays it for the user to view. The "View Charging Stats" use case continuously receives voltage, current, and power data from the ESP32 and presents it to the user interface. The user can also configure individual port settings. When a user configures a port's settings, they can choose to either change a port's current threshold value or adjust a port's timer to restart charging. These configuration updates are sent to both the SQLite database for storage and the ESP32 for execution. Together, these interactions demonstrate how the SBC enables real-time monitoring of charging data and configuration of each charging port.

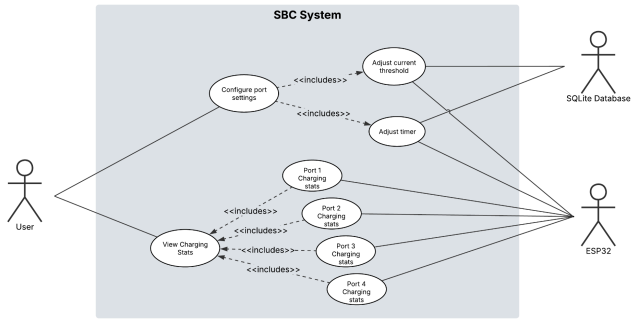


Figure 7.4.4: SBC Use Case Diagram

Each use case also carries a set of preconditions and postconditions that help define the system's expected operating behavior. Before either use case can execute, the SBC must be fully booted, the backend server must be running, the UART connection must be active, and the database must be accessible. The postconditions of the View Charging Stats use case guarantee that the user interfaces reflect the latest real-time values provided by the ESP32, while the postconditions of the Configure Port Settings use case ensure that the updated values are saved in the database, transmitted to the microcontroller, and visually reflected across all frontends. Together, these conditions emphasize system reliability and ensure that both monitoring and configuration remain accurate and synchronized.

7.4.5 SBC Resource Usage Summary

The SBC must allocate its limited CPU and memory resources among several competing tasks. The backend server typically consumes between 5–15% CPU depending on how frequently UART messages arrive and how many clients are connected. WebSocket broadcasting increases CPU usage slightly but remains lightweight due to the small size of telemetry messages. SQLite database reads and writes require minimal memory, as the database file is compact and the operations are short-lived.

Tkinter uses additional CPU resources for rendering the graphical interface, but its load remains low because it only updates numeric values rather than complex graphics. Overall, the SBC maintains a small memory footprint, typically under 300 MB for the full system, leaving enough capacity for background operating system tasks and caching. This ensures that the smart charger runs smoothly without exceeding the SBC’s thermal or performance limitations.

7.4.6 SBC Error Handling and Recovery

The SBC includes several layers of error handling designed to maintain system stability even if unexpected issues occur. One of the most common types of errors arises from malformed or incomplete UART messages produced when the ESP32 resets or when environmental noise interferes with the serial line. The backend checks each message for valid JSON formatting, the correct fields, and appropriate numerical ranges. Any message that fails validation is discarded without interrupting system operation. The SBC continues listening for the next valid message, which prevents corrupted data from reaching the user interface.

The system also contains recovery logic for disconnections or restarts of the ESP32. If the backend detects that the UART connection has gone silent for a certain duration, it will display a “Disconnected” status on the user interface and begin retrying communication. Once the ESP32 resumes sending valid messages, the SBC automatically re-syncs the configuration by resending the stored values from the database. Additional safeguards protect the system from internal errors, such as backend crashes or UI failures. Systemd automatically restarts the backend server and Tkinter interface if they unexpectedly terminate, which ensures that the smart charger recovers without requiring user intervention. These error handling mechanisms allow the SBC to maintain reliable performance under real operating conditions.

7.5 Frontend Architecture

The Smart Charger will have two different frontends that will allow the user to interact with the SBC. There will be a Local UI Frontend that lets a user interact directly with the smart charger physically, and a web application frontend that lets a user interact with the smart charger remotely. Both interfaces will share the same data and backend server, so the information will remain consistent across both frontends.

7.5.1 Local UI Frontend

The Local UI Frontend will run directly on the Raspberry Pi using Tkinter, a Python toolkit for developing GUIs. Tkinter provides a resource-efficient interface for users that are near the physical charger. The display would be a touchscreen that runs the Tkinter GUI, which will allow users to tap on buttons on the screen. The interface will display live port statistics, such as the voltage, current, and power to the user. The GUI will also allow users to configure timers and set a cutoff threshold for each port. Tkinter is lightweight and preinstalled on Raspberry Pi OS, so there will be minimal latency and no need for additional software packages or dependencies.

7.5.1.1 Local UI Dashboard Layout

The first menu of the local user interface will serve as the dashboard for the smart charger. When the application is launched on the Raspberry Pi, the user will be presented with a fullscreen graphical layout designed for a capacitive touchscreen. This will turn the Raspberry Pi into a dedicated appliance that serves as the smart charger's interface rather than a traditional desktop environment. The fullscreen is implemented to remove window borders and operating system elements to keep the user focused solely on the charger's controls and port status.

The main menu of the interface will feature a 2x2 grid of cards, with each card corresponding with one of the four charging ports in the system. Each card will display the current, voltage, and power in real-time going through that port when a device is plugged in. There is also a configure button within each card that will allow the users to configure the settings for each individual port. The port cards are color-coded to create a strong visual separation and help the user easily distinguish each port. On the bottom of the screen, there is a fixed status bar that will display the Raspberry Pi's connection status to the ESP32. This bar will automatically update and display either a green "ESP32 Connected" message or a red "ESP32 Disconnected" message depending on the current communication status.

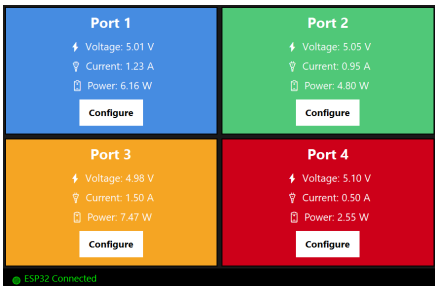


Figure 7.5.1.1 Local User Interface Main Menu

7.5.1.2 Port Configuration Interface

The Port Configuration Interface is the secondary screen within the Local User Interface and is shown when the user selects a specific charging port from the main dashboard. Each specific port's menu color matches the color that it was seThis menu will allow players to configure two parameters of the selected port, which are the current threshold and the charging timer. The interface itself has two input fields for the two different configuration settings. When the user taps either one of the input fields, an embedded on-screen number pad will appear on the right side of the display. The number pad is built into the same window rather than opening in a small pop-up, which ensures that the interaction remains stable and consistent. The keypad includes the numbers 0-9, a backspace key, a period key for decimal values, a backspace key, and an OK button to dismiss the keypad.

Once the user enters new values, they can either save the updated settings or cancel the operation. Pressing save will record the updated threshold and timer values and will cause the SBC to send the new configurations to the SQLite database and the ESP32 microcontroller. Pressing cancel will exit the menu without making any updates to the previous configuration settings. If the user were to leave an input field empty and save the configuration settings, a pop-up message will appear that will warn the user saying "Cannot save empty value." This will prevent the user from saving the empty value. If the user leaves an input field empty, the previous configurations would be reverted, and the empty value will be discarded.

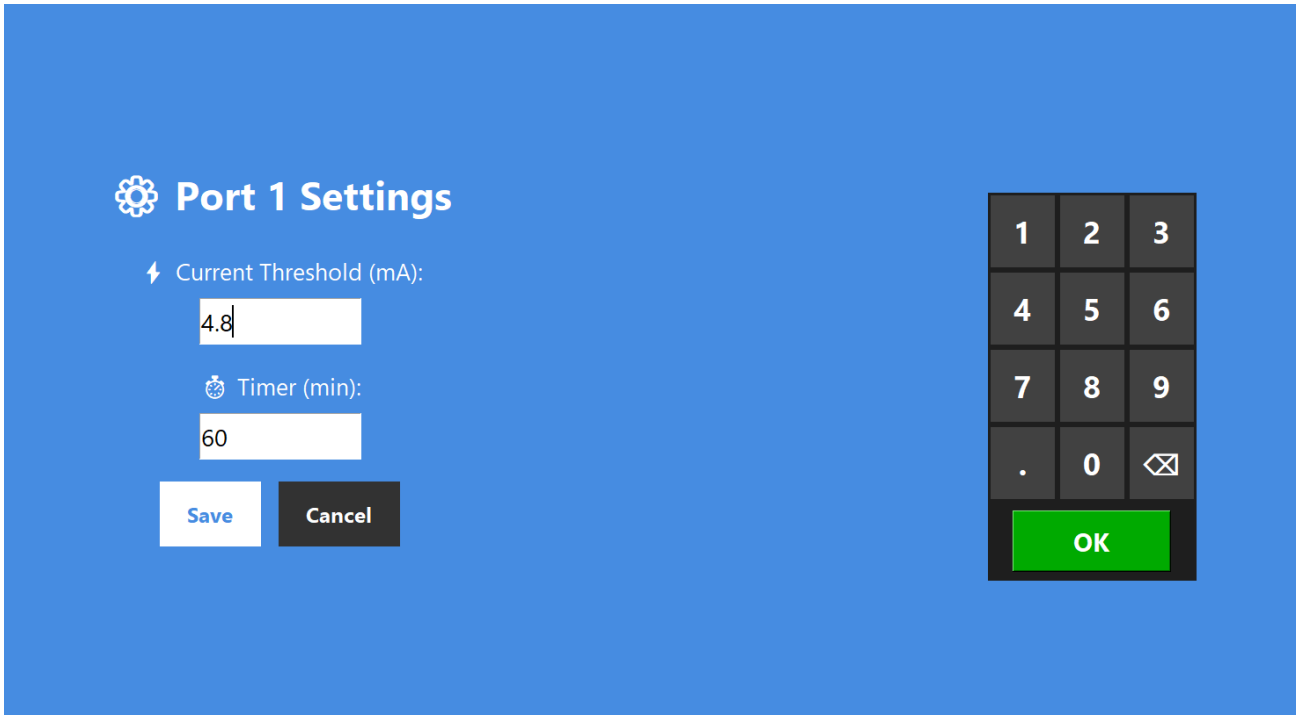


Figure 7.5.1.2 Port Configuration Interface

7.5.2 Web Application Frontend

The Web Application Frontend is implemented using React, a modern JavaScript framework designed for building responsive and interactive user interfaces. This frontend is served directly from the SBC's Node.js and Express backend and can be accessed from any device connected to the same local network as the Raspberry Pi. The dashboard provides a visual overview of all four charging ports, displaying each port's voltage, current, and power in real time. Each port card also contains a "Configure" option that allows the user to adjust the timer and current threshold values associated with that port.

The web application communicates with the SBC backend using both REST and WebSocket protocols. REST API calls are used for configuration-based operations, such as setting a new timer value or updating a current threshold, ensuring that all user actions follow a predictable request and response structure. For real-time monitoring, the application establishes a persistent WebSocket connection with the backend. This allows the frontend to receive live voltage, current, and power updates without requiring manual page refreshes or repeated polling. Whenever the ESP32 microcontroller transmits new data to the SBC, the backend immediately sends the updated values to all connected web clients, keeping their displays synchronized.

To enhance usability, the web dashboard will be designed to be fully responsive, allowing it to scale effectively across laptops, tablets, and mobile devices. This enables users to monitor and configure the smart charger from anywhere within the network. The web interface also includes reconnection logic; if the WebSocket connection is lost due to network interruptions or backend restarts, the application automatically attempts to reconnect and displays a temporary status banner to alert the user.

Additional visualization features may be incorporated into the web interface, such as a real-time voltage and current graph for selected ports. This graph would plot values over time using libraries such as Recharts or Chart.js and would allow users to visually see their individual port charging stats over time. These extended features enhance the capabilities of the smart charger and provide a more comprehensive user experience compared to the local UI's simple numeric display values.

7.6 Data Logging and Storage

The SBC uses a centralized logging system to record all critical events, communication activity, and user actions. Each backend component writes event information to a log file stored within the SBC's filesystem. These logs include timestamps, received UART messages, REST API requests, configuration changes, error messages, and connection status updates. Logging is essential for debugging, validating correct system behavior, and identifying issues such as malformed packets or communication delays.

Logs are stored in a dedicated directory within the smart charger's working folder and are automatically rotated to prevent the SD card from being overwhelmed by large log files. The logging system provides traceability for every major event, which is especially important during

testing and verification. By maintaining a complete diagnostic record, the SBC can be tested more efficiently, and any unexpected behavior can be traced back to its source.

All data that is stored on the ESP32 will also simultaneously be stored on the SBC using a SQLite database. While the voltage, current, and timer will not be logged, there will be records for when a timer was set, what the timer value is, the state of the port, and the threshold values of each port. These settings will ensure that the smart charger can return to its previous configurations after a power reboot. Since SQLite is fully self-contained, which means that no external server is needed and it would have minimal processing overhead.

7.6.1 Entity Relationship Diagram

Table 7.6.1 illustrates the Entity Relationship Diagram (ERD) for the smart charger's SQLite database, which will store the configuration values associated with each charging port on the system. The ERD will contain a single table named SmartCharger, which will represent the four USB-A charging ports on the device. Each row will correspond to one port and include three key attributes, which are the port number, the current threshold value, and the timer value. The Port_Numbers value will store the port's number and will be the table's primary value, which ensures that each port is unique within the database. The Current_Threshold value will store the current level that determines when charging should automatically stop. The Timer value will store the cooldown duration between charging cycles, which allows the system to delay re-enabling charge once a device has reached its threshold.

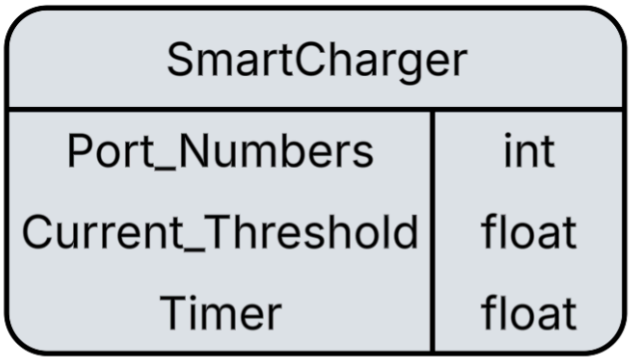


Figure 7.6 Entity Relationship Diagram

Chapter 8: System Fabrication/Prototype Construction

8.1 MCU Board Layout

The MCU board will house the ESP32 microcontroller, INA260 Current and Voltage measurement modules, and the TPS22918 Load Switch modules. This board will house the most critical logic and sensing components of the Multi-Adjustable Port Stations (M.A.P.S), the design must satisfy strict requirements related to power integrity, signal routing, noise reduction, and protection.

- Programmability: The MCU board must be able to be capable of allowing the ESP32 to be programmed using a USB. To ensure on-board USB programming capability, the MCU board integrates CH340C USB-to-UART bridge, which interfaces with a USB-C connector with the ESP32's UART bootloader pins. This permits firmware to be flashed directly over USB while ensuring proper auto-rest and boot mode functionality without the need for external adapters or manual jumpers. Proper placement of the CH340C, its decoupling capacitors and its USB differential-pair routing ensures stable programming and reliable communication

- Dynamic Load Control: The MCU board must be able to enable or disable each charging output based on measured conditions. To achieve this, the INA260 is placed upstream of the TPS22918 module which allows for monitoring of current to determine if the switch is open or closed while allowing for the MCU to determine if the port is active or inactive. To support up to 4 devices at once. I2C communication will be used to allow for the monitoring. When a condition is met based on the current measurements, the information would be sent to the ESP32 which will send an off signal to the TPS22918 to open the switch.

- Power management: The MCU board must have features on board that will allow for an efficient use in power to lower wasted energy and lower heat dissipation. The TPS22918 module supports a Quick Output Discharge (QOD) which allows for the output voltage to drop to ground in micro-seconds. This rapid discharge eliminates floating nodes, enhances port isolation, and contributes to more efficient and reliable multi-port switching.

- Load Monitoring: The MCU Board must be able to measure electrical quantities at each output to provide real-time feedback to the controller. This requires accurate measurements of both current and voltages so the controller can determine load conditions such as charging activity, idle behavior, or fault behavior. This is achieved through the INA260 which allows for accurate current and voltage measurements that are accessible through the ESP32 over the I2C bus. This allows for the ESP32 to track power consumption, detect when a device begins or ends charging, and identify abnormal load conditions.

- Over Current Protection: The INA260 has the alert pin tied to the ESP32 controller. When the INA260 detects a higher current than set in the limit register, the INA260 will send an alert to the ESP32 which would force the load switch to open to cut off the current.

Bypass capacitors are placed near the ESP32 TPS22918, and INA260 supply pins to stabilize local power delivery and suppress high frequency noise. These capacitors help prevent ESP32 brownouts, inaccurate measurements from the INA260, and undefined states from the TPS22918 caused by transient loads or switching noise. Proper capacitor placement and grounding techniques are essential for maintaining low impedance power rails across the MCU board.

To support stable operation, the MCU board employs common PCB layout strategies such as short I2C trace lengths, dedicated ground reference planes, and separated analog and digital-return paths. The INA260 measurement traces are routed away from high current switching paths to minimize noise coupling. Lastly, each USB port is placed closely to the corresponding load switch to reduce voltage loss and decrease the risk of ground-induced measurement errors.

Lastly, due to multiple USB devices being switched on or off at any moment, the MCU board must handle transient EMI and Thermal Gradients. Copper pour regions are used to increase heat dissipation around regulators, the ESP32, and high current traces. Additionally, the filtering on power lines helps reduce EMI that could impede on the measurement accuracy on the INA260.

Depicted below will be the main PCB for the project.

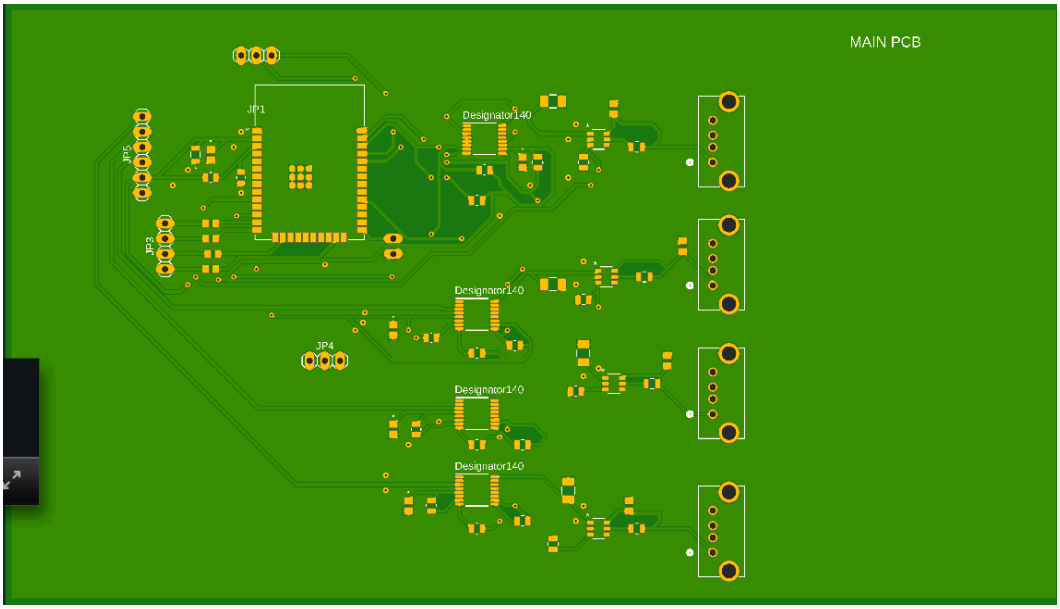


Figure 8.1: MCU Board Layout

8.2 5V and 3.3V Regulator Boards

The Regulator boards for 5V and 3.3V house their respective buck converters. The Regulator boards must be able to achieve the following goals to ensure project success.

- Current Output: The power board must be able to deliver sufficient current to power both the load device and the MCU components. To achieve this, the 5V regulator is designed to deliver 2A to prevent damage to the IC Load Switch and to ensure sufficient current is delivered for slow charging. The 3.3V regulator is designed to deliver 1A as all parts requiring this board require minimal current to turn on with the ESP32 requiring the majority of this current.

- Reverse Current Protection: The Regulator boards must have reverse-current protection to prevent damage when voltage flows back backward into the load. Without this

protection, any stored energy in the capacitors or load device can force current into the output stage which could cause permanent damage to the Internal MOSFETs. This protection. By blocking reverse current, the regulator remains stable during power-down or load switch transitions.

Both regulator boards rely on input and output capacitors to ensure power integrity. Input capacitors reduce voltage sag in the event of a sudden load transient. The output capacitors ensure that a steady output voltage is present during the switching cycles. The designs follow the recommended ratio between the input and output capacitance and the ESR requirements of the LM2670 and MP1584 to ensure minimal ripple, improve load regulation, and controlled thermal performance.

Efficiency plays a huge role in the regulator design. The copper pours and thermal vias are placed around the regulator ICs to dissipate heat and prevent localized hotspots. High efficiency switching minimizes heat generation, improves the reliability of the system, and reduces the probability of triggering a thermal shutdown protection during extended charging sessions.

The regulator level protections from overcurrent, overvoltage, thermal, and reverse polarity work together with the TPS22918 load switches and INA260 sensors to create a coordinated protection strategy, If a regulator detects an overcurrent or thermal fault. It removes power from the rail, the INA260 picks up on this change which is sent to the MCU in order to disable the USB channel.

Throughout the development process, special attention was given to the PCB layout optimization to enhance electrical performance and reduce noise propagation. High current traces for the 5V regulator board were widened and reinforced with copper pours to maintain low impedance paths. Sensitive analog lines for the INA260 measurement stages were routed away from switching nodes to minimize interference. Ground planes were placed and stitched using vias to reduce return-path noise and to maintain a stable reference for both regulator and digital circuity. Additionally test points were added across key nodes to enable streamlined debugging and performance verification during system checks. These layout considerations not only enhance the overall system robustness, but also ensured the regulators operated with the optimal electrical environments.

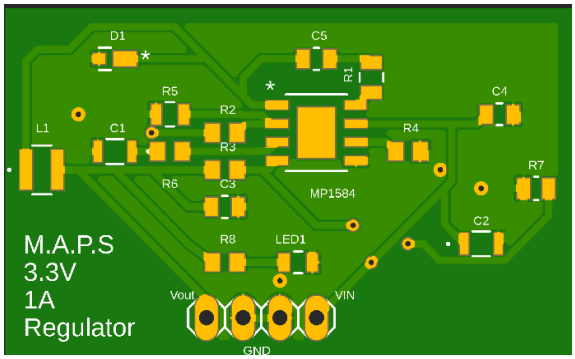


Figure 8.2a: 3.3V Regulator Board

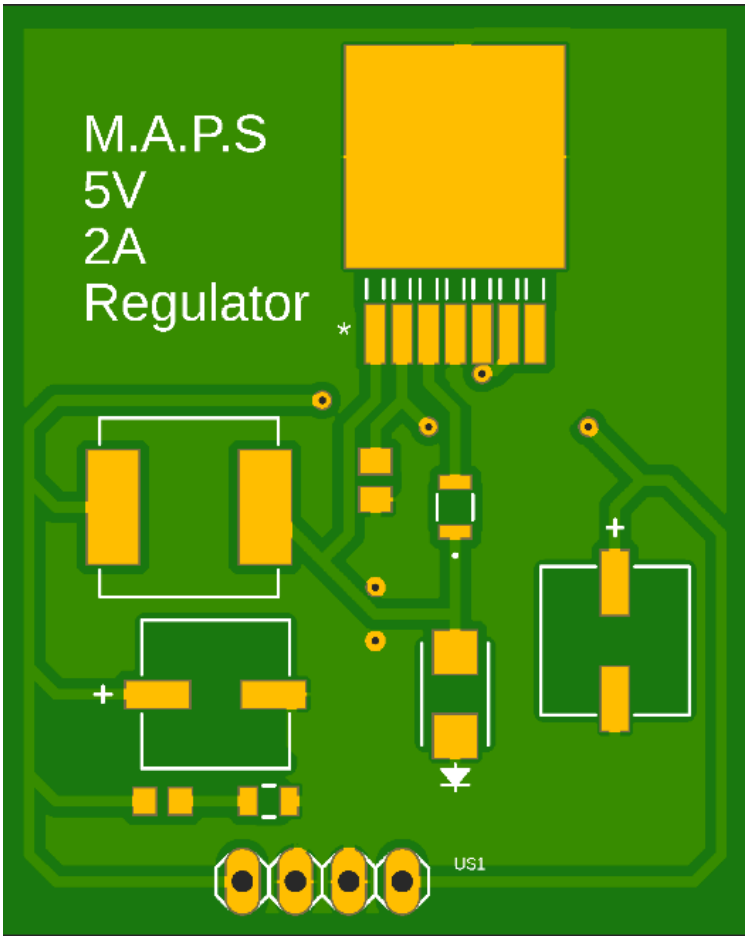


Figure 8.2b: 5V Regulator Board

Chapter 9: System Testing and Evaluation

9.1 Hardware Testing

In order to create fully working software and ensure reliable operation of the charger, all relevant hardware must be tested extensively for full functionality. Here we will outline each type of testing for all individual hardware components. All regulators, custom circuits, and measurement devices have different methods of testing and validation, which we will address in detail.

9.1.1 USB Port Regulators

All four USB ports have their own regulator, and as such it is imperative that each individual regulator functions correctly. The regulator (LM2670) has a datasheet with specifications that must be met according to the regulator's own design. We will be using it to step a 12V input of each down to around 5V.

The first method of testing will be measuring the input and output voltage of the regulator. As the final regulator is implemented on a PCB, the final product will not have an easy method of

current measurement (aside from the INA260 located downstream). However, when implemented on a breadboard, current measurement is very possible, so we will measure both it and the voltage then. The input and output power can then be calculated from the input and output voltage. We can then calculate the efficiency as Vin/Vout * 100 %. The efficiency can then be compared to any reasonable value, which should be at least 50-60% at bare minimum.

We can also measure this with different loads attached. During our initial testing phase, before we have a PCB implementation, we can accomplish this with a variable electronic load resistor. By varying the load resistance, we can verify that the regulator's input and output voltage, current, and power make sense for all values, and calculate the efficiency as well.

Once the prototype is fully assembled with USB port output, the full output voltage, current and power can be externally measured via a standard USB port monitor. However, this also includes both the current switching and measurement aspects of the charging system, and is not purely a test of the regulator.

9.1.2 ESP32 Regulator

Since the ESP32 microcontroller also needs to be supplied with power as part of the main charger, there is another regulator that will power it. The procedure for testing the regulator is effectively the same as the USB port regulators, except the output specification differs. The ESP32 requires a 3.3V input, so the output voltage of the regulator must be stable at roughly 3.3V. The ESP32-WROOM-32E datasheet recommends a 500mA supply to the chip, so we will test the current delivered by the regulator using a digital multimeter before implementing it on the PCB. The datasheet also lists maximum current consumption under load as around 379mA peak [121]. So, we will ensure the supply is at least 475mA to be safe.

Thermal characteristics are another important part of a regulator, and this is no exception. We will measure the regulator using a thermal camera once implemented, and double-check that the operating temperature is well under the maximum junction temperature specified on the datasheet.

9.1.3 INA260 Current Measurement IC

Before implementing the INA260 current measurement IC on our charging system, we must ensure that it is functional as its own unit, and can interface with our ESP32 microcontroller. Conveniently, pre-made testing boards exist that utilize an INA260 and allow it to be wired up to a development board.

Our testing plan is to initially set up the testing board with the INA260 and attach its GND/SCL/SDA pins to the ESP32 development board. Then, we can attach a test source through the INA260 board's Vin+ and Vin- pins. The testing board comes with an Arduino IDE-compatible library that contains functions to read current, voltage, and power. We will then compare the measured values from the INA260 with the measured values from a digital multimeter to ensure accuracy. Additionally, we will measure the response time of a change in

the source to the change in measurement, to ensure acceptable latency. The below figure shows our testing setup.

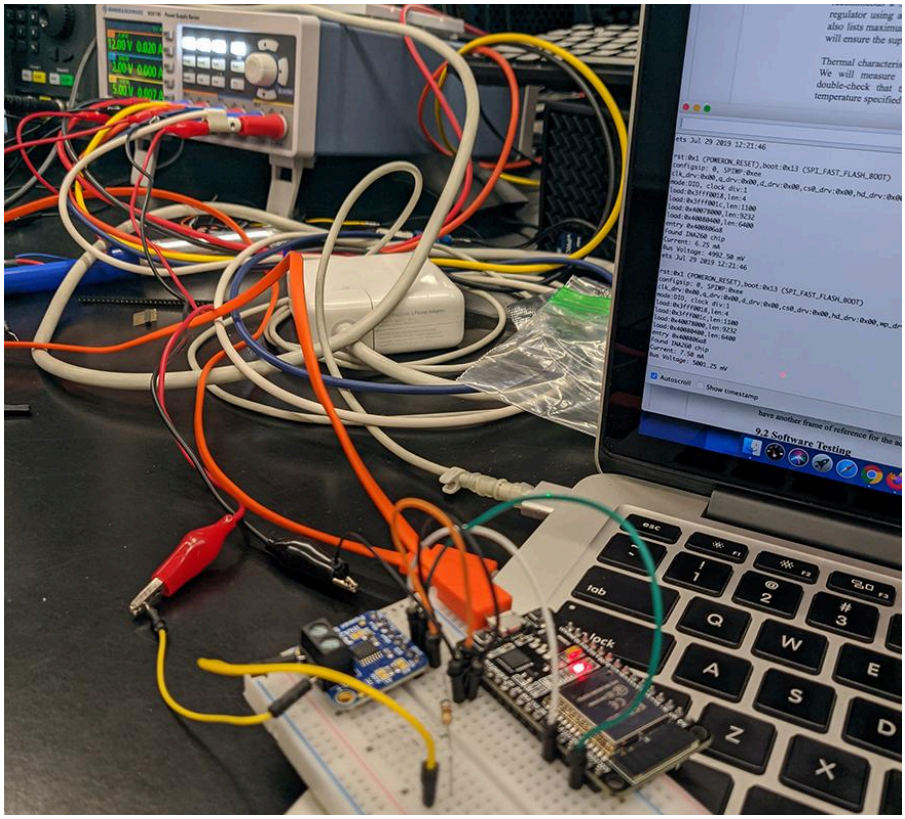


Figure 9.1.3: INA260 Testing Setup

Once implemented on the PCB, we will only be able to measure the current and voltage with an external USB measuring tool or a multimeter after the port. Using those, we can have another frame of reference for the accuracy of the INA260.

9.2 Software Testing

The software subsystem contains both the ESP32 microcontroller and the Raspberry Pi 4 single-board computer. They must both interface with each other, and the ESP32 must also interface with the INA260 measurement IC and the current switching circuit. All pieces of this subsystem must be 100 percent operational, in order to meet our specified requirements.

9.2.1 ESP32 Microcontroller Software Testing

Since the ESP32 microcontroller is an integral part of the charger system's main PCB, we must ensure its full functionality. A failed software system on the ESP32 could potentially make diagnosing small hardware issues even more difficult, and as such, it is imperative that we rigorously validate all tasks as well as the software environment.

9.2.1.1 Development Environment and Setup

We have chosen to use our ESP32 with Arduino IDE, using FreeRTOS libraries. As such, we needed to install our chosen version of the IDE (v1.8.19), and install all relevant configurations to allow the ESP32 to be programmed over UART, as well as the serial console. The development board contains a USB to UART bridge, whose driver must be installed and set up in order to interface over UART in both directions. UART on the development board is also shared with the first UART channel, which is helpful for debugging.

On startup of the ESP32, the setup task runs, which performs a number of functions. The UART baud rate is set to 115200 for both communication and the serial console, and all FreeRTOS tasks are then initialized with parameters and (optionally) cores. I2C is also set up at this time. To debug, we can send messages over UART and I2C only once on startup, to verify that all communication is working properly.

9.2.1.2 UART Receive Task

The UART receive task must parse inputs from the SBC and store them. That depends on both the SBC sending port data over UART. However, this section covers only the ESP32's portion. When receiving inputs, each input can be printed to the console to debug the UART transmission itself. Then, the inputs must be categorized into port number, current threshold, and timeout for each. These parsed inputs can then be printed to ensure functionality of any input. Error handling can be implemented to ensure potentially harmful values (i.e. out of range) cannot be stored.

9.2.1.3 UART Transmit Task

The UART transmit task must take the measured values from each port (as well as their status), and transmit it to the SBC over UART. Successful reception is the job of the SBC, but only if the ESP32 correctly sends the information. Thus, console debug messages will be implemented for each port's data that is to be transmitted, and its status. That way, we can debug and avoid potentially unknown or unintended outputs.

9.2.1.4 I2C Current/Voltage Receive Task

The I2C Current/Voltage Receive task takes in the measured values from each INA260 IC, parses the input, and stores the result for each port. Logic will be implemented to ensure the measured values make sense and are stored in the correct place. A current value below zero or above 3A is likely out of range, and should flag as such. A voltage value above 5.5V is also too high. In this case, we can use a power supply to send out of range voltages and currents through the INA260 to check whether these protection mechanisms work.

9.2.1.5 Port Logic Task

The port logic task takes in the stored values for port status, port measurements, timer value, and current threshold, and makes decisions for each port depending on the case. For each run on each port, the existing values, new values, and changes made can be printed to the console for

debugging. We can compare the expected inputs and outputs of the task to the actual inputs and outputs, in order to check if they match. This is also helpful for debugging inter-task communication and the outputs of other tasks, as it is another way to catch errors in them.

9.2.1.6 Current Switching Task

The current switching task takes in the status of each port and outputs to the current control circuitry a signal corresponding to “on”, or “off”. In order to test the current switching task, we must measure the output voltage. An output voltage near 0V is considered “off”, and a voltage near or above 1V is considered “on”. A digital multimeter attached to the ESP32 can take care of measuring this, so we can check whether the “on” or “off” status matches the output voltage at the correct time with the correct settings.

9.2.2 Raspberry Pi SBC Software Testing

The Raspberry Pi serves as the processing layer that connects the ESP32 microcontroller to the user-facing interface. All real-time data from the ESP32 are received, validated, and stored on the SBC before being transmitted to the frontend through REST and WebSocket APIs. Since the SBC is responsible for coordinating data flow between the embedded firmware and the UI, its software must be tested to ensure accuracy, responsiveness, and reliability.

9.2.2.1 Serial Communication Testing

Testing the serial communication began by verifying the UART configuration on the SBC. The communication parameters of the SBC were set to 115200 baud rate, 8 data bits, no parity, and one stop bit. These configuration settings were confirmed using Pythons’s pyserial settings and ESP32 debug output. Several test messages were exchanged between the ESP32 and the SBC to confirm correct wiring and proper UART initialization. The receipt of JSON messages indicated that the UART hardware and configuration were functioning correctly.

The next step involved the Raspberry Pi’s ability to continuously receive and parse telemetry JSON messages from the ESP32. The Python code was configured to read lines from the SBC’s serial port and decode them into UTF-8 strings. Multiple tests confirmed that the SBC could reliably capture messages at the expected rate without any errors, such as truncation or merging. Command transmission from the SBC to the ESP32 was also tested by sending JSON commands, specifically for setting the current threshold and setting the timer value. The Python script alternated between sending the two operation commands every few seconds to verify that the ESP32 consistently received and applied the configuration updates. The ESP32’s debug UART output was able to confirm that the commands were successfully parsed and stored internally.

9.2.2.2 Local Data Storage Testing

There will need to be database write tests to ensure that telemetry can be reliably stored on the SBC. The software will be tested under various data rates to verify that inserts remained stable even with rapid updates. The storage will also need to maintain integrity, and all records should

remain accessible and correctly formatted. The SBC will also need to support the user interface and backend API, which means the SBC will need to retrieve data for each charging port from the database quickly. The SBC's storage and data retrieval will be tested through repeated SQLite queries on the database.

9.2.2.3 Frontend Integration Testing

To evaluate the frontend, future testing will measure the data latency from the moment the ESP32 generates a message to the moment the updated value is displayed on both the local interface and the web interface. This will be done by timestamping messages at each stage of the following stages: ESP32 transmission, Raspberry Pi receipt , backend processing, and frontend rendering. This will determine the total propagation delay and confirm that the smartcharger can support real-time monitoring. The real-time consistency of the UI will also be tested by verifying that the updates shown on the browser accurately reflect what's in the SBC's internal state. This will involve modifying the current threshold and timer values through the user interface and verifying those commands were sent to both the backend and the ESP32.

9.2.2.4 Data Validation and Error Handling Testing

In addition to receiving telemetry from the ESP32, the SBC needs to make sure that the data being received is valid, complete, and safe to forward to the user interface. Future testing will verify the SBC's ability to correctly parse and validate JSON messages, which includes checking that all JSON messages contain the required fields, which are port number, voltage, current, and power. Tests will also handle how the SBC manages incomplete or corrupted data from the ESP32. These corruptions could be incomplete strings, missing fields, or incorrect data types. The SBC will need to be able to discard these types of messages without causing the code or the serial communication to crash.

Boundary testing will also be conducted on the ESP32. Boundary-value testing will be checking that the SBC is correctly able to filter out unrealistic values to the current threshold and timers. These values could be that the user selects a very high, unrealistic current value threshold or that the SBC correctly filters them out before sending the information to the ESP32 and backend. Intentionally sending out-of-range data will be able to test that the SBC is filtering them out correctly. Duplicate message filtering and timestamp accuracy will also be checked to make sure that the data displayed to the web frontend remains constant and meaningful.

9.2.2.5 Fault Tolerance and Recovery Testing

Since the Raspberry Pi acts as a central processing node for the smart charger, it needs to remain functional even under unexpected events. Planned fault tolerance will evaluate how the SBC behaves under rough conditions. Planned fault tolerance will evaluate how the SBC will behave during temporary ESP32 disconnections, such as if the UART wires become unplugged or if the microcontroller resets. The SBC should notice the interruption, continue running, and automatically resume its communication once the ESP32 gets reconnected. Additional tests will simulate a crash in the SBC's Python script and verify that the system restarts itself without the

need for user intervention. Power-loss recovery will also be tested by rebooting the Raspberry Pi mid-operation and ensuring that it re-initializes the backend and frontend as expected.

9.3 Performance Evaluation

We have previously outlined the specifications by which our project is measured. In order for our project to be successful, we must ensure those specifications are met. However, sufficient margin of error should exist, such that the charger can stay reliably within its constraints. We have specifically highlighted three parameters to be tested on, which we will demonstrate during Spring 2026.

The first parameter needed is USB port voltage regulation. Our specification is 5V, with a maximum tolerance of 0.25V up/down. The output voltage of each port is dependent on the regulator, but also each other component between the regulator and the USB port. Any resistor, inductor, or capacitor that current flows through on its way to the port should be accounted for. Additionally, both the load switch IC and INA260 measurement IC also have current passing through them, and must also be accounted for. All of these components may have a voltage drop that must be designed for with the setup of each. If the output voltage drop is too large, a slightly higher voltage (i.e. 5.1V) may be needed to offset it and bring the voltage well within spec.

The second parameter needed is charging delay for initial plug-in. Our specification is less than 5 seconds. This is fairly simple to test, as we can simply start a timer when a device is plugged in, and stop the timer once reasonable current starts being delivered to the device. This specification is largely software-dependent, as the microcontroller must respond quickly to determine the device has been plugged in. If this does not happen, an issue may be present in either the logic or timing of one or multiple of the microcontroller's running tasks. Another possibility is a starved task that cannot run in time. Hardware faults could also in theory exist in either the load switch IC (failure to switch on a port), or current measurement IC (failure to continue delivering current to the port).

The third parameter needed is the charge cut-off delay. Our specification is less than 1 minute. Testing this is also fairly straightforward. We can keep track of the current measured through the port as well as console output from the microcontroller. The values will be put into a moving average, but the raw measured values will be available in the microcontroller. Once the moving average drops below the threshold, the microcontroller will send a signal to the load switch IC to shut down the port. This timing depends on the response time of tasks in the microcontroller, the size of the moving average, and the response time of the load switch IC. Unless a major issue occurs with the software or load switching IC, the response should be well under one minute.

There are several other parameters which we have specified our project to, which are not necessarily required to be tested. However, we must still test them independently regardless. Power supply input voltage regulation, or the variance in input voltage to the charger, is specified as 12V with a maximum tolerance of 0.5V. We will be able to verify this with a digital multimeter attached to the input terminal and ground of the charger. Given our selected power supply, this falls well within its spec. The port output current is specified as 750mA with a maximum tolerance of 250mA. Using an external tool plugged into a USB port, we can measure

the current. Maximum charging current should occur when the battery is in a lower state of charge, and it is ideal to test with multiple different types of devices with different power and battery requirements. The current threshold set specification we have made is from 0mA to 1000mA. This means that the port is set to be able to turn off at any current dropping below the set point. A 0mA setting means the charger will never shut off as long as any device is plugged in and taking current, and a 1000mA setting will likely shut off a port in a short amount of time. Based on these, we can set the threshold to both of these values and ensure that the expected behavior happens, in order to also cover edge cases.

9.4 Plan for Senior Design 2

As Senior Design 1 ends and Senior Design 2 approaches, the planning and design phase must come to an end. We must end Senior Design 2 with a working charger system, and many steps lie in between our current progress and a functional, demonstrable project.

While we do have PCB designs and schematics currently, they are not final and could use some improvements before ordering. Those improvements will be completed between now and the early parts of Senior Design 2, which should ideally net us a working PCB once it arrives and we assemble it with the correct components. However, the reality of this is that unforeseen issues may arise, potentially costing us valuable time. Thus, we will have the first orderable PCB done before the first couple weeks of Senior Design 2. Any revisions will be done quickly after.

There are a couple other possibilities for things to go wrong that we must plan for. First, the hardware being ready before the software is a potential possibility. As such, software implementation should be finalized before the hardware is ready for testing. Ideally, it should be done significantly before, in case unknown bugs or issues exist. This also highlights the importance of testing software with existing development boards or PCB implementations before ordering custom PCBs. Extensive testing with this method is required, especially with the microcontroller. As for the hardware that the ESP32 directly interfaces with, we have acquired a pre-made INA260 board for port measurement, and are working on a load switching circuit, so the microcontroller software can be prototyped with these. This allows us to test the software without a fully working hardware implementation, and simply replicate the steps four times for each port.

Another potential possibility is that there is an existing uncaught issue with the hardware design, causing the original ordered PCBs to be faulty. Sufficient time must be allotted for this, because in this case, we must wait for another fixed PCB to arrive, setting us back days to weeks. Additionally, all PCB designs must be reviewed with fine detail to lessen the likelihood of this occurring. There is also the possibility in which a faulty PCB can be fixed in order to work, if its design or assembly issue is small enough.

Coordination between the team will be extremely important during Senior Design 2, so we can all be on the same page between tasks. We have a group chat with all members actively contributing, and plan to continue coordinating meetings multiple times per week. Effective communication is another imperative part of the project's success. Any changes or important developments should be stated both in-person and via group chat for all members to stay

up-to-date. During these biweekly meetings, we will discuss timelines for each part of the project so we can complete each task in a timely manner.

As the project continues to progress, the documentation provided in this report should also be updated to reflect any changes that have been made. After working on the full implementation and testing individual pieces, it is possible that some pieces of the project may change. If that is the case, the relevant sections in the document will be updated to reflect any changes.

Despite the difficulties afforded to us in this design process, we will persist, in no small part due to the aforementioned planning, collaboration, and design methods discussed. With the correct methods and resources in place, Senior Design 2 will net us a working charger system well before the deadline.

Chapter 10: Administrative Content

10.1 Budget and Financing

This charger will be created with an initial budget of $500, which will be paid privately by member(s) of this group. We will be keeping costs relatively low through borrowing various tools needed for the Smart Device Charger and reusing lab equipment where needed. It is, however, very possible that the budget for the device will exceed $500 in the end if multiple prototypes are needed or if the parts cost changes between the time of writing and the time we finalize ordering.

10.2 Bill of Materials

Table 10.2: Bill of Materials

Item	Unit cost	Quantity	Total cost	Part Number
ESP32-WROOM	$8.99	1	$8.99	BO7WCG1PLV
RaspberryPi Zero 2W	$21.39	1	$21.39	B09LH5SBPS
USB Type-A single port straight	$0.43	4	$1.72	CU01SAV1S00
PCB			~$150	
PCB Components (est.)			~$50	
LCD display	$10.63	1	$10.63	TS-TFT3.5Z-ND
USB tester	$7.95	4	$31.80	MIS-TLUT658R

Item	Unit cost	Quantity	Total cost	Part Number
5V Regulator	$0.22	4	$0.88	TLV76050DBZR
3D Printing Materials			~$25	
Miscellaneous Costs			~$50	
Total Amount	–	–	~$350.41	–

10.3 Project Milestones

Our group formed towards the end of Spring. We have been collaborating amongst each other throughout the summer onwards in preparation for Senior Design 1 this fall. Below represents the table(s) indicating the timeline and milestones that we intend on accomplishing this fall for Senior Design 1.

Table 10.3: Senior Design I Project Report Milestones

Milestone	**Initial Date**	**Estimated Final Date**	**Duration**
Project Idea	Summer	Summer	0 Weeks
Finalized Project Idea	08/07/2025	08/14/2025	1 Week
Independent Background Research	08/07/2025	08/21/2025	2 Weeks
Divide and Conquer Report	08/25/2025	09/03/2025	1.5 Weeks
30-Page Milestone	08/25/2025	09/15/2025	3 Weeks
60-Page Milestone	09/16/2025	10/07/2025	3 Weeks
90-Page Milestone	10/08/2025	10/29/2025	3 Weeks
120-Page Milestone	10/30/2025	11/20/2025	3 Weeks
Final Report Draft	11/21/2025	11/28/2025	1 Week

Table 10.3b: Senior Design I Project Design Milestones

Milestone	**Initial Date**	**Estimated Final Date**	**Duration**
Component-Level Design	09/03/2025	09/24/2025	3 Weeks

Component-System Design	09/25/2025	10/16/2025	3 Weeks
Breadboard Prototype	10/17/2025	11/07/2025	3 Weeks
PCB Design and Ordering	11/08/2025	11/29/2025	3 Weeks

Table 10.3c: Senior Design II Project Design Milestones

Milestone	**Initial Date**	**Estimated Final Date**	**Duration**
PCB Testing	01/12/2025	01/26/2025	2 Weeks
System Integration and Evaluation	01/27/2025	02/17/2025	3 Weeks
Project Demo Practice	02/18/2025	03/04/2025	2 Weeks
Finalize Project Documentation	03/05/2025	03/19/2025	2 Weeks
Final Presentation Practice	03/23/2025	03/30/2025	1 Week
Final Project Presentation	TBA	TBA	TBA

10.4 Work Distribution

Table 10.4: Work Distributions

Electrical Engineering	Responsibilities
Jonathan Luna	12V to 5V Conversion
	Current Switching and Control Hardware
	MCU Power Supply
Siya Sharma	Current and Voltage Measurement
	USB Port Output
Computer Engineering	Responsibilities
Hunter Volonino	Microcontroller Setup and Communication
	SBC Setup and Communication

	Current Control Programming
	Charging Data Retrieval
Sebastian Sotomayor	SBC User Interface
	Web User Interface
	Database Storage

Chapter 11: Conclusion

Our project attempts to address a gap in charging technology, where device battery health may not be prioritized under long-term charging situations. Existing products already allow for certain devices to be turned off before a full charge, but only work with a rather narrow variety of devices, and typically do not support charging multiple devices at a time. Technology is available to implement it, but there is no commercial or well-known solution. This is where the Multi-Adjustable Port Station fills the gap. It is primarily designed for hobbyists that may collect various devices and use them on a semi-regular basis. The purpose of this is to maintain them in a usable state of charge, while maximizing the long-term battery health of devices that may or may not have replacement batteries available in the future. Since these devices may greatly vary in age or capability, communication in a standardized format to read battery information inside the device over USB is not possible. Therefore, we have based charging control on the measured output current from each port, which is easily measurable before it reaches the device.

While our charger system is mainly intended for hobbyists, there are various other scenarios in which the Multi-Adjustable Port Station may prove useful. In general, any use case of multiple USB-charged battery-powered devices that must remain plugged in for long periods of time are of particular interest. Store environments, where display devices need to be operational but remain plugged in, are a likely candidate. Another option would be education. Classroom educational devices may be battery-powered to allow students to use them wirelessly, but may also be plugged in for long periods of time when they are not used for a lesson.

The Multi-Adjustable Port Station system physically consists of several parts. First, a main board contains several integral pieces. An ESP32 microcontroller was selected to interface with the rest of the project. A current measurement subsystem featuring four INA260 measurement ICs was chosen, which interfaces with the ESP32 over I2C, and measures each of four USB ports, also located on the main board. The final subsystem on the main board is for load switching, which allows each individual port to be turned “on” or “off” based on the microcontroller’s output. There are headers on the main board for auxiliary PCBs, which contain the various regulators we will be using. Four 5V regulators take a 12V input and allow a 5V output to be sent to the load switching subsystem. An additional 3.3V regulator takes a 12V input and steps it down to the 3.3V required to run the ESP32 microcontroller. For communication, the main board has a header to allow a bidirectional UART interface with the other major hardware component, a Raspberry Pi 4 single-board computer. The Raspberry Pi 4 allows two major interfaces: a

physical user interface when attached to a display, and wireless communication to support a web interface.

The software of our Multi-Adjustable Port Station is split into two categories. The first category is software run on the ESP32. We will be choosing FreeRTOS for its multitasking capability to harness both cores. Tasks will be created for both directions of UART communication, I2C communication, load switching output, and port on/off logic. That way, the ESP32 can make intelligent decisions to control charging. Our second software category is software run on the Raspberry Pi 4, whose purpose is to allow charging parameters to be entered and charging data to be shown to the user. The Raspberry Pi must also take both directions of UART communication with the ESP32, which must also be attached to the user interfaces. There will be two user interfaces: one directly on the screen attached to the device, and one shared on the network. The local user interface allows port data to be displayed on a small screen attached to the charger for easy adjustment in close proximity. The network interface, powered by the MERN stack, enables charging data to be viewed and adjusted over a website, which will be accessible over any standard web browser attached to the same network. This will require additional front-end and back-end design.

As we progress towards the implementation and completion phase of our charger system, it is important to address that this project is intended to educate us about the full engineering design process. This will more adequately prepare us for industry and the future of engineering as a whole, providing a net benefit both for us and the future world.

Appendices

Appendix A - Reference

[1]“Obtaining service for your Apple product after an expired warranty,” Apple Support. https://support.apple.com/en-us/102772

[2] Battery University, “BU-808: How to prolong lithium-based batteries,” Battery University, https://batteryuniversity.com/article/bu-808-how-to-prolong-lithium-based-batteries (accessed Aug. 29, 2025).

[3] “Chargie a gold edition single,” Chargie, https://chargie.org/product/chargie-single/ (accessed Aug. 29, 2025).

[4] Ovidiu, “Why chargie beats Apple’s built‑in charge limiting,” Chargie, https://chargie.org/why-chargie-beats-apples-built%e2%80%91in-charge-limiting/ (accessed Aug. 30, 2025).

[5] Ovidiu, “How to use Chargie to future-proof any device’s battery (not just smartphones),”Chargie,https://chargie.org/how-to-use-chargie-to-future-proof-any-devices-battery-not-just-smartphones/ (accessed Aug. 30, 2025).

[6] Home Assistant, “Integrating individual device energy usage,” Home Assistant, https://www.home-assistant.io/docs/energy/individual-devices/ (accessed Aug. 31, 2025).

[7] “Anker 735 Charger (nano II 65W),” Anker, https://www.anker.com/products/a2667?variant=41581366575254&ref=a2697-anker-charger-140w-4-port (accessed Aug. 31, 2025).

[8] “Empower,” eMpower - Solar Powered Mobile Device Charger With Remote Control, https://www.ece.ucf.edu/seniordesign/fa2012sp2013/g32/ (accessed Sep. 1, 2025).

[9] ARM Ltd, “What is FPGA?,” *Arm | The Architecture for the Digital World*. https://www.arm.com/glossary/fpga

[10] J. Schneider, “FPGA vs microcontroller,” *Ibm.com*, Jun. 03, 2024. https://www.ibm.com/think/topics/fpga-vs-microcontroller

[11] V. Kanade, “RISC vs. CISC: 20 Key Comparisons,” *Spiceworks*, 2023. https://www.spiceworks.com/tech/tech-general/articles/risc-vs-cisc/

[12] E. Roberts, “RISC vs. CISC,” *Stanford.edu*, 2019. https://cs.stanford.edu/people/eroberts/courses/soco/projects/risc/risccisc/

[13] R. Keim, “What Is a Linear Voltage Regulator? - Technical Articles,” *www.allaboutcircuits.com*, Feb. 13, 2020. https://www.allaboutcircuits.com/technical-articles/what-is-a-linear-voltage-regulator/

[14] R. Keim, “What Is a Switching Voltage Regulator?,” *Allaboutcircuits.com*, Jul. 14, 2023. https://www.allaboutcircuits.com/technical-articles/what-is-a-switching-voltage-regulator

[15] “Pulse Width Modulation | DC Motor Drives | Electronics Textbook,” *www.allaboutcircuits.com*. https://www.allaboutcircuits.com/textbook/semiconductors/chpt-11/pulse-width-modulation/

[16] Electronics Tutorials, “Closed-loop System and Closed-loop Control Systems,” *Basic Electronics Tutorials*, Mar. 04, 2018. https://www.electronics-tutorials.ws/systems/closed-loop-system.html

[17] Electronics Tutorials, “MOSFET and Metal Oxide Semiconductor Tutorial,” *Basic Electronics Tutorials*, Sep. 03, 2013. https://www.electronics-tutorials.ws/transistor/tran_6.html

[18] “Basics of Load Switch ICs Basics of Load Switch ICs Basics of Load Switch ICs.” Accessed: Oct. 31, 2025. [Online]. Available: https://toshiba.semicon-storage.com/info/application_note_en_20210326_AKX00144.pdf?did=13765

[19] Arrow Electronics, “What is a solid state relay?,” *Arrow.com*, Jan. 02, 2024. https://www.arrow.com/en/research-and-events/articles/whats-inside-a-solid-state-relay

[20] WIN SOURCE, “What are the advantages and disadvantages of solid state relays?,” *WIN SOURCE BLOG*, May 25, 2020. https://blog.win-source.net/q-a/what-are-the-advantages-and-disadvantages-of-solid-state-relays/ (accessed Oct. 31, 2025).

[21] “Bipolar Junction Transistors: What they are & do,” *www.emcourse.com*. https://www.emcourse.com/news-blog/bipolar-junction-transistors-what-are-they-what-do-they-do

[22] “An Engineer’s Guide to Current Sensing e-book A collection of technical content on current sensing designs.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/eb/slyy154b/slyy154b.pdf?ts=1761062279272

[23] “Hall Effect Current Sensing: Open-Loop and Closed-Loop Configurations - Technical Articles,” *www.allaboutcircuits.com*. https://www.allaboutcircuits.com/technical-articles/hall-effect-current-sensing-open-loop-and-closed-loop-configurations/

[24] “Understanding and Applying the Hall Effect - Technical Articles,” *www.allaboutcircuits.com*. https://www.allaboutcircuits.com/technical-articles/understanding-and-applying-the-hall-effect/

[25] “Isolated Amplifiers | Analog Devices,” *Analog.com*, 2016. https://www.analog.com/en/product-category/isolated-amplifiers.html (accessed Oct. 31, 2025).

[26] J. Blom, “Voltage Dividers - learn.sparkfun.com,” *Sparkfun.com*, 2019. https://learn.sparkfun.com/tutorials/voltage-dividers/all

[27] “An Engineer’s Guide to Current Sensing e-book A collection of technical content on current sensing designs.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/eb/slyy154b/slyy154b.pdf?ts=1761140995150

[28] “Voltage Dividers - SparkFun Learn,” *Sparkfun.com*, 2025. https://learn.sparkfun.com/tutorials/voltage-dividers/all

[29] “Power Delivery Design Issues for Hi-Speed USB on Motherboards.” Available: https://www.usb.org/sites/default/files/power_delivery_motherboards.pdf

[30] “Understanding Buck Power Stages in Switchmode Power Supplies Application Report SLVA057,” 1999. Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/an/slva057/slva057.pdf?ts=1761152515038

[31] “Analysis of Four DC-DC Converters in Equilibrium - Technical Articles,” *www.allaboutcircuits.com*. https://www.allaboutcircuits.com/technical-articles/analysis-of-four-dc-dc-converters-in-equilibrium/

[32] A. Da Silva and Y. Ducommun, “Quick Reference Guide To TI Buck Switching DC/DC Application Notes.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/an/slva958b/slva958b.pdf?ts=1761214064392

[33] A. Applications Journal, “Basic operation,” 2009. Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/an/slyt358/slyt358.pdf?ts=1761196998469

[34] “Linear & low-dropout (LDO) regulators | TI.com,” *Ti.com*, 2025. https://www.ti.com/product-category/power-management/linear-ldo-regulators/overview.html

[35] GeeksforGeeks, “What is Tkinter for Python?,” GeeksforGeeks, May 15, 2020. https://www.geeksforgeeks.org/python/introduction-to-tkinter/

[36] “What is PyQt? | Learn Python PyQt,” pythonpyqt.com. https://pythonpyqt.com/what-is-pyqt/

[37] A. Whittaker, “How to use a Raspberry Pi in kiosk mode - Raspberry Pi,” Raspberry Pi, Nov. 18, 2022. https://www.raspberrypi.com/news/how-to-use-a-raspberry-pi-in-kiosk-mode/ (accessed Oct. 31, 2025).

[38] Mayur Koshti, “Is It Right to Use PHP Instead of React?,” Medium, Sep. 28, 2024. https://medium.com/%40mayurkoshti12/is-it-right-to-use-php-instead-of-react-33698a6c6377 (accessed Oct. 31, 2025).

[39] “A Complete Guide on Why Use React for frontend development,” IT Blog | Mobile App Development India | Offshore Web Development - Bacancytechnology.com, Mar. 11, 2022. https://www.bacancytechnology.com/blog/why-use-react

[40] IBM, “LAMP stack,” Ibm.com, Oct. 12, 2021. https://www.ibm.com/think/topics/lamp-stack

[41] D. Fateh, “What is the MERN stack? A tutorial for modern web development,” Contentful.com, May 26, 2025. https://www.contentful.com/blog/mern-stack/

[42] MySQL, “MySQL :: MySQL 8.4 Reference Manual :: 1.2.1 What is MySQL?,” dev.mysql.com, 2024. https://dev.mysql.com/doc/refman/8.4/en/what-is-mysql.html

[43] B. Krause, “What is MongoDB? - OPC-Router.de explains it to you exactly,” System integration with the OPC Router, Apr. 01, 2021. https://www.opc-router.com/what-is-mongodb/

[44] SQLite, “About SQLite,” Sqlite.org, 2019. https://www.sqlite.org/about.html

[45] RedHat, “What is Linux?,” Redhat.com, Jan. 03, 2023. https://www.redhat.com/en/topics/linux/what-is-linux

[46] M. Toback, “Raspberry Pi OS: Features and Benefits for the Raspberry Pi,” mini pc technology, Dec. 16, 2024. https://minipctech.com/raspberry-pi-os/

[47] S. Campbell, “Basics of UART Communication,” Circuit Basics, Feb. 13, 2016. https://www.circuitbasics.com/basics-uart-communication/

[48] S. Campbell, “Basics of the I2C Communication Protocol,” Circuit Basics, Feb. 13, 2016. https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/

[49] S. Campbell, “Basics of the SPI Communication Protocol,” Circuit Basics, Feb. 13, 2016. https://www.circuitbasics.com/basics-of-the-spi-communication-protocol/

[50] R. Santos, “What is MQTT and How It Works | Random Nerd Tutorials,” Random Nerd Tutorials, Apr. 02, 2019. https://randomnerdtutorials.com/what-is-mqtt-and-how-it-works/

[51] “What is Bluetooth? How to get Bluetooth on a PC,” Digital Citizen, Nov. 24, 2020. https://www.digitalcitizen.life/what-is-bluetooth/

[52] “ESP32 Wireless Communication Protocols | Random Nerd Tutorials,” Oct. 28, 2022. https://randomnerdtutorials.com/esp32-wireless-communication-protocols/

[53] Amazon Web Services, “What is an API? - API Beginner’s Guide - AWS,” Amazon Web Services, Inc., 2025. https://aws.amazon.com/what-is/api/

[54] Postman, “What Is a REST API? Examples, Uses, and Challenges,” Postman Blog, Jul. 09, 2020. https://blog.postman.com/rest-api-examples/

[55] GeeksforGeeks, “What is the use of WebSocket API ?,” GeeksforGeeks, Feb. 10, 2022. https://www.geeksforgeeks.org/html/what-is-the-use-of-websocket-api/

[56] “graphql,” Amazon Web Services, Inc. https://aws.amazon.com/graphql/

[57] “Measuring categories per IEC 61010-1 | GOSSEN METRAWATT,” *Gossenmetrawatt.de*, 2015.

https://www.gossenmetrawatt.de/en/knowledge/measuring-and-test-technology/measuring-categories-per-iec-61010-1/ (accessed Oct. 31, 2025).

[58] Fluke, “IEC Category Ratings: Use the Right Tools for the Job | Fluke,” *Fluke.com*,

[59] “IEC 61010-1:2010,” *Webstore.iec.ch*, 2016. https://webstore.iec.ch/en/publication/4279?utm_source=chatgpt.com (accessed Oct. 31, 2025).

[60] “IEC 61326 EMC Testing for Electrical Equipment in Control, Laboratory, and Measurement,” *Applus+ Keystone*, Nov. 18, 2024. https://keystonecompliance.com/iec-61326/?utm_source=chatgpt.com (accessed Oct. 31, 2025).

[61] “Understanding IEC/EN 61326-1: A Guide for Manufacturers - Compliance Testing,” *Compliance Testing*, Oct. 24, 2023. https://compliancetesting.com/iec-en-61326-1/?utm_source=chatgpt.com (accessed Oct. 31, 2025).

[62] “IEC 61326-1:2020,” *Webstore.iec.ch*, 2020. https://webstore.iec.ch/en/publication/62793 (accessed Oct. 31, 2025).

[63] “IEC 61000-4-2:2025,” *Webstore.iec.ch*, 2025. https://webstore.iec.ch/en/publication/68954 (accessed Oct. 31,2025).

[64] “IEC 61000-4-4:2012,” *Webstore.iec.ch*, 2017. https://webstore.iec.ch/en/publication/4222 (accessed Oct. 31, 2025).

[65] “IEC 61000-4-6:2023,” *Webstore.iec.ch*, 2023. https://webstore.iec.ch/en/publication/65586 (accessed Oct. 31, 2025).

[66] “Basic definitions of uncertainty,” *Nist.gov*, 2025. https://physics.nist.gov/cuu/Uncertainty/basic.html?utm_source=chatgpt.com (accessed Oct. 31, 2025).

[67] “NIST Technical Note 1297 | NIST,” *NIST*, Jul. 06, 2009. https://www.nist.gov/pml/nist-technical-note-1297

[68] “ISO - ISO/IEC 17025 — Testing and calibration laboratories,” *ISO*, Jan. 26, 2021. https://www.iso.org/ISO-IEC-17025-testing-and-calibration-laboratories.html?utm_source=chatgpt.com

[69] “Basic definitions of uncertainty,” *Nist.gov*, 2025. https://physics.nist.gov/cuu/Uncertainty/basic.html

[70] “NIST Technical Note 1297 | NIST,” *NIST*, Jul. 06, 2009. https://www.nist.gov/pml/nist-technical-note-1297?utm_source=chatgpt.com

[71] “Keysight Technologies The Truth About the Fidelity of High-Bandwidth Voltage Probes Who Should Read This Application Note?” Accessed: Oct. 31, 2025. [Online]. Available: https://prc.keysight.com/Content/PDF_Files/5988-6515EN.pdf

[72] “Probing Tips for High Performance Design and Measurement,” *Tek.com*, 2025. https://www.tek.com/en/documents/application-note/probing-tips-high-performance-design-and-measurement?utm_source=chatgpt.com (accessed Oct. 31, 2025).

[73] “An Engineer’s Guide to Current Sensing e-book A collection of technical content on current sensing designs.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/eb/slyy154b/slyy154b.pdf?ts=1760802933685

[74] “Generic Standard on Printed Board Design.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.electronics.org/TOC/IPC-2221A.pdf

[75] “IEC 60068-1:2013,” *Webstore.iec.ch*, 2024. https://webstore.iec.ch/en/publication/501

[76] “USB Power Delivery | USB-IF,” *Usb.org*, 2025. https://www.usb.org/taxonomy/term/60 (accessed Oct. 31, 2025).

[77] “USB 2.0 | USB-IF,” *Usb.org*, 2025. https://www.usb.org/taxonomy/term/40

[78] G. Team, “Introducing the USB-IF Conformity to IEC 62680 (USB) Specifications Program ,” *Graniteriverlabs.com*, Nov. 22, 2024. https://www.graniteriverlabs.com/en-us/technical-blog/usb-if-iec-62680 (accessed Oct. 31, 2025).

[79] “IEC 62680-1-2:2024,” *Webstore.iec.ch*, 2024. https://webstore.iec.ch/en/publication/91099 (accessed Oct. 31, 2025).

[80] “IEC 62368-1:2023,” *Webstore.iec.ch*, 2023. https://webstore.iec.ch/en/publication/69308 (accessed Oct. 31, 2025).

[81] “IEC 60950-1:2001,” *Webstore.iec.ch*, 2023. https://webstore.iec.ch/en/publication/18568 (accessed Oct. 31, 2025).

[82] T. Regan, J. Munson, and G. Zimmer, “AN-105: Current Sense Circuit Collection Making Sense of Current,” *Analog.com*, 2025. https://www.analog.com/en/resources/app-notes/an-105fa.html (accessed Oct. 31, 2025).

[83] “An Engineer’s Guide to Current Sensing e-book A collection of technical content on current sensing designs.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/eb/slyy154b/slyy154b.pdf?ts=1760897094827

[84] M. Cantrell, “Digital Isolator Simplifies USB Isolation in Medical and Industrial Applications,” *Analog.com*, 2025. https://www.analog.com/en/resources/analog-dialogue/articles/digital-isolators-in-medical-and-industrial-apps.html (accessed Oct. 31, 2025).

[85] Breaking ground loops to protect USB data transmission, https://www.analog.com/media/en/technical-documentation/technical-articles/NAppkinNote_Breaking_Ground_Loops.pdf (accessed Oct. 31, 2025).

[86] “Power Delivery Design Issues for Hi-Speed USB on Motherboards.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.usb.org/sites/default/files/power_delivery_motherboards.pdf

[87] “USB4TM Thunderbolt3TM Compatibility Requirements Specification,” 2021. Accessed: Oct. 31, 2025. [Online]. Available: https://www.usb.org/sites/default/files/USB4%E2%84%A2%20Thunderbolt3%E2%84%A2%20Compatibility%20Requirements%20Specification%20Rev%201.0%20-%2020210129_0.pdf

[88] “IEC 60068-1:2013,” *Webstore.iec.ch*, 2024. https://webstore.iec.ch/en/publication/501

[89] “Compliance | USB-IF,” *www.usb.org*. https://www.usb.org/compliance

[90] “UM10204 I 2 C-bus specification and user manual Rev. 6 -4 April 2014 User manual Document information Info Content.” Available: https://www.nxp.com/docs/en/user-guide/UM10204.pdf

[91] “IEC 62368-1:2023,” *Webstore.iec.ch*, 2023. https://webstore.iec.ch/en/publication/69308 (accessed Oct. 31, 2025).

[92] “IEC 61010-1:2010,” *Webstore.iec.ch*, 2016. https://webstore.iec.ch/en/publication/4279

[93] “NIST Technical Note 1297 | NIST,” *NIST*, Jul. 06, 2009. https://www.nist.gov/pml/nist-technical-note-1297

[94] “ISO - ISO/IEC 17025 — Testing and calibration laboratories,” *ISO*, Jan. 26, 2021. https://www.iso.org/ISO-IEC-17025-testing-and-calibration-laboratories.html

[95] “Optimizing the TPS62130/40/50/60/70 Output Filter.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/an/slva463a/slva463a.pdf?ts=1760996103394

[96] “Linear versus switching regulators in industrial applications with a 24-V bus.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/an/slyt527/slyt527.pdf?utm_source=chatgpt.com&ts=1760997709591

[97] Analog, https://www.analog.com/media/en/technical-documentation/application-notes/an46.pdf (accessed Oct. 31, 2025).

[98] “Power Delivery Design Issues for Hi-Speed USB on Motherboards.” Available: https://www.usb.org/sites/default/files/power_delivery_motherboards.pdf

[99] “AN4794 MCP2515/MCP25625 EMC Hints Introduction.” Accessed: Oct. 31, 2025. [Online]. Available: https://ww1.microchip.com/downloads/aemDocuments/documents/APID/ApplicationNotes/ApplicationNotes/AN4794-MCP2515-MCP25625-DS00004794.pdf

[100] “PCB Design Guidelines For Reduced EMI SZZA009,” 1999. Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/an/szza009/szza009.pdf?utm_source=chatgpt.com

[101] “Managing Inrush Current.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/an/slva670a/slva670a.pdf

[102] “RFC 7231 - Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content,” *datatracker.ietf.org*. https://datatracker.ietf.org/doc/rfc7231/

[103] “rfc8259,” *datatracker.ietf.org*. https://datatracker.ietf.org/doc/html/rfc8259

[104] B. Kelechava, “The SQL Standard - ISO/IEC 9075:2023 (ANSI X3.135) - ANSI Blog,” *The ANSI Blog*, Oct. 05, 2018. https://blog.ansi.org/ansi/sql-standard-iso-iec-9075-2023-ansi-x3-135/

[105] Level Access, “WCAG 2.1 | Web Accessibility Standards and Checklist,” *Level Access*, Jan. 20, 2025. https://www.levelaccess.com/blog/wcag-2-1-exploring-new-success-criteria/

[106] CJ, “What Are the Four Major Categories of Accessibility?,” *Boia.org*, 2019. https://www.boia.org/blog/what-are-the-four-major-categories-of-accessibility

[141]T. Hudson, “Understanding AC/DC Power Supplies.” Available: https://media.monolithicpower.com/mps_cms_document/2/0/2020-seo-understanding-acdc-power-supplies_r1.0.pdf

[142]Anker, “Understanding the AC to DC Converter: A Comprehensive Guide,” *Anker*, Jul. 14, 2023. https://www.anker.com/blogs/ac-power/the-world-of-ac-to-dc-converter-an-in-depth-analysis

[143]“USB Power Delivery: The Technology 1 - Convenience and Safety,” *Renesas*, 2025. https://www.renesas.com/en/support/engineer-school/usb-power-delivery-02?srsltid=AfmBOooHmZOG3F2PG0LRHVnr4Ob1ar-LdeHMuuLcKfsJID7NSdFPKNV_ (accessed Nov. 25, 2025).

[144]Michele, “Build Better Devices: USB Power Delivery Explained | Simplexity,” *Simplexity Product Development*, Jun. 03, 2025. https://www.simplexitypd.com/usb-power-delivery-explained/ (accessed Nov. 25, 2025).

[145]brmarcum, “AC to DC Conversion,” *Instructables*. https://www.instructables.com/AC-to-DC-Conversion/

[146]“Beginners approach to DIY AC/DC bench power supply - Page 1,” *Eevblog.com*, 2023. https://www.eevblog.com/forum/projects/beginners-approach-to-diy-acdc-bench-power-supply/ (accessed Nov. 25, 2025).

[147]R. Santopietro, “The Necessity of AC and DC Surge Protection In Modern Electrical Systems,” *Raycap Industrial Surge Protection Devices and Lightning Strike Protection*, Mar. 27, 2025. https://www.raycap.com/the-necessity-of-ac-and-dc-surge-protection-in-modern-electrical-systems/ (accessed Nov. 25, 2025).

[148]GNS Components Limited, “Thermal Resistance of Power Semiconductors - Knowledge,” *Ictransistors.com*, May 16, 2025. https://www.ictransistors.com/info/thermal-resistance-of-power-semiconductors-102921511.html (accessed Nov. 25, 2025).

[149]File name:GST60A-spec 2025-03-28 60W AC-DC Reliable Green Industrial Adaptor. (n.d.). https://www.meanwell.com/Upload/PDF/GST60A/GST60A-SPEC.PDF

[150]Q. Ualtek, “Energy Efficiency Level VI Output Protections: SCP/OCP QFWB-65 Series P P C.” Accessed: Nov. 25, 2025. [Online]. Available: https://www.qualtekusa.com/images/Power%20Supplies/pdf_files/QFWB-65%20Series.pdf

[151]“NEEWER AC 100-240V to DC 12V Power Supply Adapter - NEEWER,” *NEEWER*, 2025. https://neewer.com/products/neewer-ac-12v-power-supply-adapter-5a-60w-3a-36w-66600755 (accessed Nov. 25, 2025).

[152]“rfc6455,” *datatracker.ietf.org*. https://datatracker.ietf.org/doc/html/rfc6455 (accessed Nov. 25, 2025).

[153]“ECMAScript® 2021 Language Specification,” *tc39.es*. https://tc39.es/ecma262/ (accessed Nov. 25, 2025).

[154]“rfc8259,” *datatracker.ietf.org*. https://datatracker.ietf.org/doc/html/rfc8259 (accessed Nov. 25, 2025).

[155]“Database File Format,” *SQLite*, Apr. 30, 2025. https://sqlite.org/fileformat.html (accessed Nov. 25, 2025).

[156]K. Pykes, “What are ACID Transactions? A Complete Guide for Beginners,” *datacamp*, Feb. 18, 2025. https://www.datacamp.com/blog/acid-transactions

(accessed Nov. 25, 2025).

Appendix C - Data sheet

[107]

"INA226 36V, 16-Bit, Ultra-Precise I 2 C Output Current, Voltage, and Power Monitor With Alert." Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/ina226.pdf?ts=1761062875647

[108] "INA219," INA219 data sheet, product information and support | TI.com, https://www.ti.com/product/INA219 (accessed Oct. 31, 2025).

[109] "INA240," INA240 data sheet, product information and support | TI.com, https://www.ti.com/product/INA240 (accessed Oct. 31, 2025).

[110] "Access to this page has been denied.," *Mouser.com*, 2025. https://www.mouser.com/datasheet/2/405/1/opa333_q1-3379084.pdf (accessed Oct. 31, 2025).

[111] alldatasheet.com, "TLV271 Datasheet(PDF)," *Alldatasheet.com*, 2020. https://www.alldatasheet.com/datasheet-pdf/pdf/84626/TI/TLV271.html (accessed Oct. 31, 2025).

[112] "OPA333," OPA333 data sheet, product information and support | TI.com, https://www.ti.com/product/OPA333 (accessed Oct. 31, 2025).

[113] "MCP6071/2/4 Features." Accessed: Oct. 31, 2025. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/22142_B_MCP6071.pdf

[113] Ti, https://www.ti.com/lit/ds/symlink/ina226.pdf (accessed Oct. 31, 2025).

[114] "INA260," INA260 data sheet, product information and support | TI.com, https://www.ti.com/product/INA260 (accessed Oct. 31, 2025).

[115] "INA230," INA230 data sheet, product information and support | TI.com, https://www.ti.com/product/INA230 (accessed Oct. 31, 2025).

[116] Analog, https://www.analog.com/media/en/technical-documentation/data-sheets/MAX34407.pdf (accessed Oct. 31, 2025).

[117] ACS712 by SparkFun Electronics Datasheet | DigiKey, https://www.digikey.com/en/htmldatasheets/production/1732744/0/0/1/acs712 (accessed Oct. 31, 2025).

[118] AIMTEC, http://www.aimtec.com/site/Aimtec/files/Datasheet/HighResolution/AMSR-78-NZ.pdf (accessed Oct. 31, 2025).

[119] Ti, https://www.ti.com/lit/ds/symlink/lm2596.pdf?ts=1667716842835 (accessed Oct. 31, 2025).

[120] Client challenge, https://www.monolithicpower.com/en/documentview/productdocument/index/version/2/document_type/Datasheet/lang/en/sku/MP1584EN-LF-Z/document_id/204/ (accessed Oct. 31, 2025).

[121] "ESP32WROOM32E & ESP32WROOM32UE Datasheet." Available: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf

[122] "MSP430FR6989 MSP430FR698x(1), MSP430FR598x(1) Mixed-Signal Microcontrollers 1 Device Overview," 2014. Available: https://www.ti.com/lit/ds/symlink/msp430fr6989.pdf

[123] "Raspberry Pi Pico Datasheet: An RP2040-based microcontroller board" Available: https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf

[124] "Atmega640/1280/1281/2560/2561 datasheet" Available: https://ww1.microchip.com/downloads/en/devicedoc/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf

[125] "This is information on a product in full production. STM32F722xx STM32F723xx," 2020. Accessed: Oct. 31, 2025. [Online]. Available: https://www.skytech.ir/DownLoad/File/1616_stm32f722ic.pdf

[126] Raspberry Pi Ltd, "Raspberry Pi 5," 2023. Available: https://datasheets.raspberrypi.com/rpi5/raspberry-pi-5-product-brief.pdf

[127] Raspberry Pi Ltd, "DATASHEET Raspberry Pi 4 Model B Release 1," 2019. Available: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf

[128] Raspberry Pi Ltd, "Raspberry Pi Zero 2 W," Apr. 2024. Available: https://datasheets.raspberrypi.com/rpizero2/raspberry-pi-zero-2-w-product-brief.pdf

[129] "NanoPi M5 - FriendlyELEC WiKi," *Friendlyelec.com*, 2017. https://wiki.friendlyelec.com/wiki/index.php/NanoPi_M5 (accessed Oct. 31, 2025).

[130] T. Co., "AML-S905X-CC Le Potato Overview Resources and Guides," *Libre Computer Hub*, Nov. 07, 2022. https://hub.libre.computer/t/aml-s905x-cc-le-potato-overview-resources-and-guides/288 (accessed Oct. 31, 2025).

[131] "Load Current (A)." Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps56339.pdf?ts=1761920597410

[132] "TPS56x200 4.5-V to 17-V Input, 2-A, 3-A Synchronous Step-down Voltage Regulator in 6-Pin SOT-23." Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps563200.pdf?ts=1761079849500

[133] “LM2670 SIMPLE SWITCHER ® Power Converter, High-Efficiency, 3A, Step-Down Voltage Regulator With Sync.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/lm2670.pdf?ts=1761237650167

[134] MAX16935/MAX16939 - 36V, 3.5A, 2.2MHz step-down ..., https://www.analog.com/media/en/technical-documentation/data-sheets/max16935-max16939.pdf (accessed Oct. 31, 2025).

[135] “I OUT (A) TPS560430 SIMPLE SWITCHER ® 4-V to 36-V, 600-mA Synchronous Step-Down Converter” 2017. Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps560430.pdf?ts=1761780249843

[136] “TPS56120x 4.5-V to 17-V Input, 1-A Synchronous Step-Down Voltage Regulator in 6-Pin SOT-23.” Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps561208.pdf?ts=1761793907397

[137] “TPS54202 4.5-V to 28-V Input, 2-A Output, EMI Friendly Synchronous Step Down Converter Simplified Schematic Output Current (A) Efficiency vs Output Current TPS54202,” 2016. Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps54202.pdf

[138] “Input Voltage (V).” Accessed: Oct. 31, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps22918.pdf

[139] “LS0504EVT233 Series + eFuse | Littelfuse,” *Littelfuse, Inc.*, 2025. https://www.littelfuse.com/products/fuses-overcurrent-protection/efuse/ls0504evt233 (accessed Oct. 31, 2025).

[140] “DML3012LDC SINGLE-CHANNEL SMART LOAD SWITCH Description,” 2024. Accessed: Oct. 31, 2025. [Online]. Available: https://www.diodes.com/datasheet/download/DML3012LDC.pdf

Appendix E - ChatGPT/LLM prompts and outcomes

Case Study Question & Response[1B] What is the criteria that should be used to decide what is the correct electrical component to use as a switch?

That's a great question — choosing the **right switching component** (BJT, MOSFET, relay, SSR, IC load switch, etc.) depends heavily on the **electrical, control, and environmental requirements** of your circuit.

Here's a breakdown of the **main criteria** engineers use when selecting a switch:

⚙️ 1. Voltage and Current Ratings

- **Max Voltage (Vds, Vce, or Vswitch):** Must exceed the highest voltage the switch will see — usually with a **20–30% safety margin**.
- **Max Current (Id, Ic, or Iswitch):** Must handle the maximum load current plus a safety margin.
- For **inductive loads**, the current spike at turn-off should also be considered.

Example:
If switching a 12 V, 2 A motor → choose a device rated for **≥20 V and ≥3 A**.

⚡ 2. Type of Signal Being Switched

- **DC** vs **AC:**
    - BJTs and MOSFETs are ideal for **DC**.
    - Relays or SSRs can handle **AC or DC** but have slower switching.
- **Analog** vs **Digital:**
    - Analog signals (audio, sensors) require low leakage and low ON resistance (e.g., analog switches or transmission gates).
    - Digital signals tolerate higher ON resistance.

⏱️ 3. Switching Speed

- **Mechanical Relays:** milliseconds — slow but can handle high current.
- **BJTs:** microseconds.
- **MOSFETs:** nanoseconds to microseconds (great for high-speed switching).
- **SSR:** milliseconds but with no mechanical wear.
- **Load Switch ICs:** typically microseconds; good for controlled power sequencing.

🔌 4. Control Signal Requirements

- **MOSFETs:** Voltage-controlled → low gate current but need proper gate voltage (logic-level vs standard).
- **BJTs:** Current-controlled → need base resistor and more drive current.
- **Relays:** Coil requires higher current (often needs a transistor driver).
- **IC Load Switches:** Usually logic-controlled, designed for direct MCU interface.

🌡️ 5. Power Dissipation & Efficiency

- Compute **$P = I^2 \times R_{DS(on)}$** (MOSFET) or **$V_{CE(sat)} \times I$** (BJT).
- Choose a switch with **low conduction losses** to reduce heating and improve efficiency.
- Check for **thermal resistance** and consider heat sinking if necessary.

🔄 6. Isolation Requirements

- **Electrical isolation** between control and load?
    - **Relay/SSR:** Provide galvanic isolation.

- **MOSFET/BJT:** No isolation — need optocoupler if required.

🧠 7. Complexity and Integration

- If you want simplicity and protection features (soft-start, current limit, reverse current blocking) → **Load Switch IC** is ideal.
- If you need a **custom discrete design** → use MOSFET/BJT and external components.

💰 8. Cost and Size Constraints

- Relays: inexpensive but bulky.
- BJTs: cheap, small, but less efficient.
- MOSFETs: slightly pricier but efficient and compact.
- IC Load Switches: small and clean, but higher cost per channel.

🧩 Summary Table

Switch Type	Speed	Isolation	Control	Current Range	Typical Use
Relay	Slow (ms)	Yes	Coil current	Up to 10s of A	AC/DC power switching
SSR	Medium	Yes	Logic/low current	A few A	Silent switching, industrial
BJT	Fast	No	Base current	<10 A	Low-power DC, amplifiers

MOSFET	Very fast	No	Gate voltage	<100 A	DC switching, converters
IC Load Switch	Fast	Partial	Logic-level	<5 A	Power sequencing, MCU control

---

If you tell me **your application** (e.g., voltage, current, control source, and load type), I can recommend the **most suitable type** of switch for that scenario.
Would you like to do that?

If you like, I can track down additional application-specific guides (e.g., for inductive loads, high-frequency switching, or automotive use) and we can build a more comprehensive reference list.

Case Study Question & Response [1C] What is the criteria that should be used to decide what is the correct electrical component to use as a switch?

Here are the key criteria you should use to select the correct electrical switch for your application.

Choosing a switch involves matching its specifications to the demands of your circuit and its physical environment.[1] The most critical factors are the electrical ratings, as a mismatch here can lead to switch failure or a fire hazard.[2]

---

1. Electrical Criteria (The Most Important)

These factors determine if the switch can safely handle the electricity flowing through it.[3]

⚡ Voltage and Current Rating

Every switch has a maximum voltage and current it can safely handle.[4]

- **Voltage Rating:** This is the maximum voltage the switch is designed to insulate against when it's in the 'off' position. You'll typically see this in volts (V), such as `125V` or `250V`.

- **Current Rating:** This is the maximum amount of current (measured in amps, A) the switch can safely carry and interrupt when it's in the 'on' position.[5] A common rating is 15A.

Rule: Always choose a switch with voltage and current ratings **equal to or higher** than your circuit's requirements.

🔌 AC vs. DC Rating (A Critical Distinction)

A switch's rating for Alternating Current (AC) is almost always higher than its rating for Direct Current (DC). **They are not interchangeable.**[6]

- **AC (Alternating Current):** In an AC circuit (like your home's wiring), the voltage crosses zero 100 or 120 times per second.[7] This "zero-crossing" naturally helps to extinguish any spark (arc) that forms when the switch's contacts pull apart.[8]
- **DC (Direct Current):** In a DC circuit (like from a battery or solar panel), the current is constant.[9] When the switch opens, the arc does not naturally extinguish. It is sustained, generating intense heat that can melt the switch's internal contacts, weld them together, or destroy the switch.

Rule: If you are switching a DC circuit, you **must** use a switch specifically rated for your DC voltage and current. An AC-rated switch used on a DC circuit will fail, often dangerously.[10]

💡 Load Type (Resistive vs. Inductive)

The *type* of device you are switching has a massive impact on the switch's lifespan. The current rating on a switch is almost always for a simple **resistive load**.

- **Resistive Load (Easy):** These are simple devices that create heat or light, like an incandescent bulb, a toaster, or a heating element. The current is stable.
- **Inductive Load (Hard):** These devices use a magnetic coil, like **motors**, **fans**, **solenoids**, **relays**, and **transformers**.[11] When you turn off an inductive load, its collapsing magnetic field creates a large voltage spike (or "kick") that causes a powerful arc across the switch contacts.[12] This arc is far more damaging than a resistive arc.

- **Capacitive Load (Hard):** These devices (like some power supplies) draw a massive burst of current (an "inrush current") the instant you turn them on, which can also damage contacts.

Rule (Derating): If you are switching a non-resistive load, you must use a heavy-duty switch or "derate" its capacity significantly. A common rule of thumb is to choose a switch rated for **at least double** the load's steady-state current. For motors, you may need a switch rated 5x-10x higher to handle the startup current.

---

2. Functional Criteria

These factors define how the switch behaves in the circuit.

🎛️ Switch Configuration (Poles and Throws)

This describes the internal connections the switch makes.

- **Pole:** The number of separate circuits the switch controls.[13]
    - **Single-Pole (SP):** Controls one circuit (e.g., a standard light switch).[14]
    - **Double-Pole (DP):** Controls two separate circuits with a single action (e.g., a 240V appliance switch that needs to disconnect two "hot" wires at once).[15]
- **Throw:** The number of "on" positions the switch has.
    - **Single-Throw (ST):** A simple On-Off switch. It connects the pole to one other contact.
    - **Double-Throw (DT):** The switch toggles the pole between two different contacts.[16] This is used for "three-way" switching (like controlling a light from two ends of a hallway) or to select between two different functions.

Common Types:

- **SPST (Single-Pole, Single-Throw):** Basic On-Off.[17]
- **SPDT (Single-Pole, Double-Throw):** One input, two outputs (e.g., A or B).[18]
- **DPST (Double-Pole, Single-Throw):** Two circuits, On-Off.[19]
- **DPDT (Double-Pole, Double-Throw):** Two circuits, two outputs (e.g., A or B for both).

🔘 Action (Momentary vs. Maintained)

- **Maintained:** The switch stays in the position you put it in (e.g., a light switch).
- **Momentary:** The switch is spring-loaded and returns to its default position when you let go (e.g., a doorbell, a car horn, or a computer's power button). This is often specified as (ON)-OFF (push-for-on) or ON-(OFF) (push-for-off).

---

3. Physical and Environmental Criteria

These factors relate to the switch's physical form and its ability to survive its environment.[20]

- **Actuation Method:** How do you use the switch?
    - **Toggle:** A lever you flip (classic light switch).[21]
    - **Rocker:** A switch that pivots (common on power strips).
    - **Push-Button:** A button you press.[22]
    - **Slide:** A slider you move back and forth.[23]
    - **Rotary:** A knob you turn (like a fan speed selector).[24]
    - **Key:** A key-operated switch for security.
- **Mounting Style:** How is it installed?
    - **Panel Mount:** Mounted through a hole in a panel or wall.
    - **PC Mount:** Soldered directly onto a printed circuit board (PCB).
    - **In-Line:** Sits in the middle of a power cord.
- **Environmental Sealing (IP Rating):** This is crucial if the switch will be exposed to dust or water.[25] An **IP Rating** (e.g., IP65) tells you its level of protection. The first digit is for solids (dust), and the second is for liquids (water). A higher number means better protection.
- **Lifespan (Cycles):** How many times can the switch be turned on and off before it's expected to fail? This is specified in "cycles" and can range from 10,000 for a cheap switch to over 1,000,000 for a high-reliability one.
- **Size and Aesthetics:** Does it need to be a specific size to fit? Does its color or style matter for the final product?

By considering all these factors, starting with the critical electrical ratings, you can confidently choose a switch that is safe, reliable, and well-suited for your project.

Case Study Question & Response [1D] What is the criteria that should be used to decide what is the correct electrical component to use as a switch?

I should clarify - I didn't use external sources for this information. I answered your questions based on my training knowledge, which includes general electrical engineering concepts and standard terminology that's been established in the field for decades.

This type of information about electrical switches represents fundamental, stable technical knowledge that doesn't change frequently. Switch terminology like SPST, DPDT, poles, and throws are standardized conventions used across:

- Electrical engineering textbooks
- Industry standards (like those from NEMA, IEC)
- Manufacturer datasheets and catalogs
- Technical education materials

Since you asked about sources, if you'd like to verify this information or learn more, I can search for current reference materials, manufacturer documentation, or educational resources that cover switch specifications and terminology. Would that be helpful?

Alternatively, if you're looking for information about a specific application or more detailed technical standards, I'd be happy to search for that as well.

Case Study Question & Response [2B] What MCUs will be best suited to be used in a closed loop system where the MCU picks up current and/or voltage readings and needs to either open or close a switch depending on the reading?

Nice — good question. I'll split this into (A) the design criteria you should use to pick an MCU for closed-loop current/voltage control and (B) specific MCU families (with example reasons and tradeoffs). I searched datasheets and vendor pages to pick well-matched examples. ([STMicroelectronics][1])

A — What matters (design criteria)

1. **ADC performance**

* Resolution (12/16-bit), sample rate (kS/s → MS/s), input range, input multiplexer and differential inputs. For current sensing you often want a differential input or front-end amplifier. ([STMicroelectronics][1])

2. **Analog front-end/analog peripherals**

* Built-in PGA, comparators, op-amps, multiple ADCs (for simultaneous sampling), and injected channels / analog watchdogs for fast clamp/fault detection. ([STMicroelectronics][1])

3. **DMA + low-latency interrupts**

* DMA lets you stream ADC samples without blocking the CPU; low interrupt latency and deterministic real-time performance matter for tight control loops. ([DeepBlue][2])

4. **Math performance / real-time control features**

* If you need fast control (current loop for power converters, motor FOC), choose MCUs with DSP instructions, fast FPU or specialized control cores. ([DigiKey][3])

5. **PWM and gating peripherals**

* PWM outputs with dead-time generation, complementary outputs and capture timers are essential when the MCU must directly drive switching.

6. **Safety & fault handling**

* Hardware comparators, trip inputs, fault pins, on-chip protection and watchdogs help shut switches immediately on overcurrent.

7. **Connectivity & tooling**

* Debugging, libraries (e.g., motor control libraries), and ecosystem (IDE, examples, dev kits).

8. **Analog accuracy caveats**

* Some high-level MCUs (e.g., ESP32) have noisy / non-linear ADCs; they can be used if you add external ADCs or calibration but are not ideal for precision current measurement out of the box. ([Reddit][4])

B — MCU families I recommend (by application)

1) **High-speed power / motor / tight current loops**

* **Texas Instruments C2000 (Piccolo / F28x family)** — purpose-built for real-time control: math-optimized cores, high-performance ADCs, many PWM channels, control peripherals and motor libraries. Great for sub-100 µs loops and FOC. ([Texas Instruments][5])

* Use when you drive MOSFETs/IGBTs directly and need deterministic control and many synchronous ADCs.

* **STMicroelectronics STM32G4 / STM32F3 (analog-heavy ST parts)** — very capable ADCs (multi ADCs, injected channels), strong DSP/FPU performance in higher G4 parts, good timers/PWM for power control. Strong ecosystem (CubeMX, examples). ([STMicroelectronics][1])

2) **General closed-loop control (good analog, mid-range compute)**

* **Microchip SAM E51 / SAM D21 families** (Cortex-M4/M0+) — good analog blocks, dual 12-bit ADCs (some 1Msps), low power options, broad toolchain. Good for battery systems or power monitoring where you want balance of analog + software ease. ([Microchip][6])

* **TI MSP430 series** — ultra low power, good integrated analog on many parts, great when power budget is tight (you already use MSP430). Good for low-sampling closed-loop tasks (slow to moderate bandwidth). (You're already working with MSP430 so it's a natural option.)

3) **Wireless or IoT devices (with caveats)**

* **Espressif ESP32 / ESP32-S3** — offers Wi-Fi/Bluetooth and 12-bit ADCs, but ADC non-linearity/noise is well-documented; acceptable with calibration or if you front-end with an external high-precision ADC or INA amplifier. Avoid relying on raw ESP32 ADC for precision current sensing. ([Reddit][4])

4) **If you need absolute measurement accuracy / isolation / higher resolution**

* **Use an external ADC + current-sense amplifier**: e.g., TI ADS1115 (16-bit I²C) or ADS1015 for faster but lower-res; INA series (INA219/INA226) or dedicated differential amplifiers for shunt measurement. Many designers pair a mid-range MCU (SAM/STM32) with an external 16-bit ADC and/or INA for the analog front end. This gives much better accuracy and isolation options than relying solely on MCU ADCs.

Tradeoffs summary (short)

* Want fastest, most deterministic closed-loop control → **TI C2000** or high-end **STM32G4**. ([Texas Instruments][5])

* Want low power / simple loops → **MSP430** or **SAM D21**. ([Microchip][7])

* Need wireless + can accept ADC quirks → **ESP32** (but add external ADC/PGA for accuracy). ([Reddit][4])

Quick practical checklist for selecting a part

1. Required control loop bandwidth (Hz → defines sample rate).

2. ADC resolution and simultaneous sampling requirement (do you need synchronous sampling of multiple channels?).

3. Do you need integrated gate drivers or will you use external MOSFET drivers?

4. Safety/fault pins and how fast the MCU must react to an overcurrent event.

5. Tools/ecosystem and dev kit availability.

If you want, tell me the **approximate voltages & currents**, how **fast** the MCU must react (e.g., “switch within 100 µs”), whether you need **wireless**, and whether you prefer **TI / ST / Microchip / Espressif** — I’ll give 2–3 exact part numbers + dev-kit links and a recommended front-end (shunt value, amplifier or INA, or external ADC) and a simple block diagram.

[1]: https://www.st.com/resource/en/product_training/STM32G4-Analog-ADC.pdf?utm_source=chatgpt.com "STM32G4 Analog-to-Digital Converter block"

[2]: https://deepbluembedded.com/stm32-adc-tutorial-complete-guide-with-examples/?utm_source=chatgpt.com "STM32 ADC Tutorial + ADC Examples [Ultimate Guide]"

[3]: https://www.digikey.com/en/product-highlight/t/texas-instruments/c2000?utm_source=chatgpt.com "C2000 32 bit Microcontroller – Texas Instruments"

[4]: https://www.reddit.com/r/esp32/comments/z3es25/your_thoughts_on_esp32_adc_accuracy/?utm_source=chatgpt.com "Your thoughts on esp32 ADC accuracy"

[5]: https://www.ti.com/product-category/microcontrollers-processors/c2000-real-time-mcus/overview.html?utm_source=chatgpt.com "C2000 real-time microcontrollers"

[6]: https://www.microchip.com/en-us/products/microcontrollers/32-bit-mcus/pic32-sam/sam-d?utm_source=chatgpt.com "SAM D Arm® Cortex®-M-Based Microcontrollers (MCUs)"

[7]: https://ww1.microchip.com/downloads/en/DeviceDoc/SAM-D21-DA1-Family-Data-Sheet-DS40001882H.pdf?utm_source=chatgpt.com "SAM-D21-DA1-Family-Data-Sheet-DS40001882H. ..."

Case Study Question & Response [2C] What MCUs will be best suited to be used in a closed loop system where the MCU picks up current and/or voltage readings and needs to either open or close a switch depending on the reading?

For your closed-loop system, the best MCU isn't just about the brand or core speed; it's about having the right **integrated analog and control peripherals**. You need an MCU that can sense the analog world (current/voltage) and react (control a switch) very quickly, often without the main CPU even needing to be involved.

The most critical features for your application are:

* **Fast Analog-to-Digital Converters (ADCs):** To accurately measure the current or voltage.

* **High-Speed Comparators (HSCMP):** To *immediately* detect if a reading is above or below a safety threshold (e.g., an overcurrent event).

* **Advanced Timers (PWMs):** To control the switch (likely a MOSFET) and to be "tripped" or shut down by the comparator.

* **Internal Op-Amps (optional but very useful):** To amplify the small signal from a current-sense resistor, reducing external component count.

Based on these needs, here are the MCU families best suited for this task, categorized by their strengths.

---

1. Digital Signal Controllers (DSCs)

These are a hybrid between a traditional MCU and a Digital Signal Processor (DSP). They are **purpose-built for high-speed, real-time, closed-loop control** like power supplies and motor control.

* **Microchip dsPIC33 Series (e.g., CK, EV, EP families)**

    * **Why they're a top choice:** These are arguably the industry standard for this kind of work. They have extremely fast ADCs, high-speed comparators, and "Motor Control PWM" peripherals that are designed to be linked together. You can configure the hardware to have an overcurrent event from a comparator **automatically shut down the PWM output in nanoseconds**, far faster than any software interrupt could.

    * **Key Peripherals:** Fast 12/16-bit ADCs (often > 3 Msps), high-speed comparators with built-in DACs (for setting a precise trip point), and highly flexible PWM generators.

* **Texas Instruments C2000 Series (e.g., Piccolo, Delfino)**

    * **Why they're a top choice:** Like dsPICs, the C2000 family is a powerhouse for real-time control. They are renowned for their high-performance 32-bit core, exceptionally good ADCs (often with synchronized sampling), and sophisticated "ePWM" (enhanced PWM) modules.

* **Key Peripherals:** High-resolution 12-bit ADCs (often 4+ Msps), integrated comparators, and ePWM modules that have a dedicated "trip-zone" input. You can route the comparator's output directly to this trip pin to instantly and safely shut down your switch.

---

2. MCUs with Advanced Analog Peripherals

These are general-purpose MCUs that have special families "tuned" for analog-intensive applications.

* **STMicroelectronics STM32 G4 Series**

* **Why they're a good choice:** The G4 series was specifically designed by ST for digital power and motor control. It integrates a wealth of high-performance analog peripherals.

* **Key Peripherals:** Up to five 12-bit, 4 Msps ADCs; seven high-speed comparators; six operational amplifiers; and "High-Resolution Timers" that can be cross-linked with these analog peripherals. This allows for the same "hardware-only" fast shutdown as the DSCs.

* **Microchip AVR DB / DD Series**

* **Why they're a good choice:** If your application is simpler or more cost-sensitive, these 8-bit MCUs have outstanding analog features. They are one of the few MCUs to integrate **on-chip operational amplifiers** that you can configure in hardware.

* **Key Peripherals:** You can use the op-amps to create a "non-inverting" or "differential" amplifier for your current-sense resistor. The output of that op-amp can then be fed *internally* to the Analog Comparator (AC) or the ADC. The AC's output can, in turn, be used to automatically shut down the timers (TCs) controlling your switch. This creates a complete, fast-acting protection circuit with almost no external parts.

* **NXP LPC55S3x / Kinetis KV Series**

    * **Why they're a good choice:** NXP's offerings are also very strong. The Kinetis KV series is designed for motor control with 16-bit ADCs and "FlexTimers." The LPC55S3x series features a very fast High-Speed Comparator (HSCMP) sub-system that can be directly linked to PWMs for fast protection.

👑 My Top Recommendations

* **For High-Performance / Fast Response:** Start with the **Microchip dsPIC33CK** or the **TI C2000 Piccolo** series. They are made for this exact problem, and the wealth of application notes and example code for "digital power" or "overcurrent protection" will be invaluable.

* **For High Integration / Lower Cost:** If your performance needs are more moderate, look at the **STM32G4** series or the **Microchip AVR DB** series. Their ability to integrate op-amps and link peripherals in hardware can significantly simplify your design and reduce your bill of materials (BOM).

Case Study Question & Response [2D] What MCUs will be best suited to be used in a closed loop system where the MCU picks up current and/or voltage readings and needs to either open or close a switch depending on the reading?

For a closed-loop control system with current/voltage sensing and switch control, here are the best MCU options:

Top Recommendations

1. STM32 Series (STMicroelectronics)

- **STM32G4**: Excellent choice with built-in high-resolution ADCs, analog comparators, and fast response times

- **STM32F3/F4**: Good ADC performance, plenty of GPIO, cost-effective

- Features: 12-bit ADCs (up to 5 MSPS), hardware comparators for ultra-fast response, built-in op-amps on some models

2. Texas Instruments C2000 Series

- Specifically designed for real-time control applications

- Fast ADCs, PWM generation, and control peripherals

- Best for motor control and power electronics applications

3. Microchip PIC/AVR Series

- **PIC16F/18F**: Simple, reliable, good ADCs

- **ATmega/ATtiny**: Cost-effective for simpler applications

- **dsPIC33**: For more demanding control loops requiring DSP capabilities

4. ESP32 (Espressif)

- If you need wireless connectivity alongside control

- Built-in ADCs (though less accurate than dedicated industrial MCUs)

- Cost-effective and easy to program

Key Features to Look For:

- **ADC resolution**: 12-bit minimum (16-bit for precision)

- **ADC sampling rate**: Fast enough for your control loop timing

- **Hardware comparators**: For threshold detection without CPU intervention

- **Interrupt capability**: Fast response to voltage/current changes

- **Timer peripherals**: For precise control loop timing

- **GPIO with sufficient current drive**: Or ability to drive external MOSFETs/relays

Selection Criteria:

For simple on/off control (relay/contactor): Almost any MCU works - Arduino Uno, STM32F0, PIC16F

For fast switching (solid-state, < 1ms response): STM32G4, C2000, dsPIC33

For precision measurements: Higher resolution ADCs (16-bit), consider external ADC chips

For cost-sensitive: ATmega328, PIC16F, STM32F0

What's your specific application? (voltage range, switching speed requirements, current levels?) This would help narrow down the best choice.

[3B] What considerations would need to made to communicate with an MCU to SBC to allow for potential web interface to be sent to the SBC to be sent to the MCU?

Ah, you're thinking about a system where a **Single-Board Computer (SBC)** like a Raspberry Pi or similar acts as a bridge between a **web interface** and a **microcontroller (MCU)**. Let's break this down carefully and cover the main considerations.

---

1. Communication Protocol Between SBC and MCU

You need a reliable link between the MCU and SBC. Common options include:

Protocol	Pros	Cons	Notes
UART / Serial (RS232/TTL)	Simple, widely supported	Slower speeds, requires proper flow control	Common for sensor/control data
I2C	Good for short distances, multiple devices	Limited distance, shared bus can be noisy	Usually for multiple peripherals on same bus
SPI	Fast, simple master-slave	More wires (MOSI/MISO/SCK/CS), short distances	Good for high-speed data like display or ADC updates
CAN	Robust, multi-node	More complex	Often used in automotive/industrial
USB / HID / CDC	Plug-and-play	MCU must support USB device stack	Can appear as a virtual COM port to SBC

| **Ethernet / TCP/IP** | Can directly support web traffic | MCU must have network stack | Good for web-based control if MCU is powerful enough |

Consideration: Pick a protocol your MCU can handle efficiently, considering speed, distance, and robustness.

---

2. Data Serialization and Messaging

You can't just "send raw variables" over most protocols; you need a format:

* **Simple text-based protocols** (JSON, CSV, custom delimiters)

  * Pros: Easy to debug, human-readable
  * Cons: More bandwidth overhead
* **Binary protocols** (fixed-size structs, packed data)

  * Pros: Efficient, lower latency
  * Cons: Harder to debug and extend

You also need to decide **command structure**:

* MCU may expect "commands" (e.g., `SET_PWM:150`)
* MCU may send "status" (e.g., `TEMP:25.4`)

---

3. Synchronization / Timing

* **Polling vs. Interrupt-driven**:

  * SBC can poll MCU periodically (simple but can miss fast events)
  * MCU can send data asynchronously when something changes (requires buffer management)
* Consider **flow control** (especially for UART) to prevent data loss.

---

4. SBC Web Interface Integration

* The SBC usually hosts the **web server** (Node.js, Flask, etc.)
* Web interface sends HTTP requests (REST API or WebSocket) → SBC application → translates to MCU commands
* Consider **latency requirements**: e.g., controlling a motor in real-time may need <50 ms response; temperature readouts can tolerate seconds.

---

5. MCU Resource Considerations

* Limited memory and CPU: can the MCU handle parsing JSON?
* Might need lightweight parsers (e.g., using minimalistic binary protocol instead of JSON).
* Firmware needs a **command parser** and **error handling**.

---

6. Security

* If you want to access this web interface over the internet:

  * SBC should handle authentication & HTTPS, not MCU (MCU typically lacks SSL/TLS)
  * Avoid exposing MCU directly to the internet

---

7. Reliability & Error Handling

* Include checksums or CRC for critical data
* Define behavior on communication failure (retries, default values)
* Log errors on SBC for debugging

---

8. Example Flow

1. User interacts with web interface → sends REST API or WebSocket message to SBC.
2. SBC app translates to MCU command (`SET_PWM:150`) → sends over UART.
3. MCU receives command, executes, and optionally sends back a status.
4. SBC receives status, updates web interface.

If you want, I can **draw a clear diagram showing MCU ↔ SBC ↔ Web Interface** with labeled protocols and data flows—it makes these relationships much easier to visualize.

Do you want me to do that?

[3C] What MCUs will be best suited to be used in a closed loop system where the MCU picks up current and/or voltage readings and needs to either open or close a switch depending on the reading?

For your closed-loop system, the best MCU isn't just about the brand or core speed; it's about having the right **integrated analog and control peripherals**. You need an MCU that can sense the analog world (current/voltage) and react (control a switch) very quickly, often without the main CPU even needing to be involved.

The most critical features for your application are:
* **Fast Analog-to-Digital Converters (ADCs):** To accurately measure the current or voltage.
* **High-Speed Comparators (HSCMP):** To *immediately* detect if a reading is above or below a safety threshold (e.g., an overcurrent event).
* **Advanced Timers (PWMs):** To control the switch (likely a MOSFET) and to be "tripped" or shut down by the comparator.
* **Internal Op-Amps (optional but very useful):** To amplify the small signal from a current-sense resistor, reducing external component count.

Based on these needs, here are the MCU families best suited for this task, categorized by their strengths.

---

1. Digital Signal Controllers (DSCs)
These are a hybrid between a traditional MCU and a Digital Signal Processor (DSP). They are **purpose-built for high-speed, real-time, closed-loop control** like power supplies and motor control.

* **Microchip dsPIC33 Series (e.g., CK, EV, EP families)**
    * **Why they're a top choice:** These are arguably the industry standard for this kind of work. They have extremely fast ADCs, high-speed comparators, and "Motor Control PWM" peripherals that are designed to be linked together. You can configure the hardware to have an overcurrent event from a comparator **automatically shut down the PWM output in nanoseconds**, far faster than any software interrupt could.
    * **Key Peripherals:** Fast 12/16-bit ADCs (often > 3 Msps), high-speed comparators with built-in DACs (for setting a precise trip point), and highly flexible PWM generators.

* **Texas Instruments C2000 Series (e.g., Piccolo, Delfino)**
    * **Why they're a top choice:** Like dsPICs, the C2000 family is a powerhouse for real-time control. They are renowned for their high-performance 32-bit core, exceptionally good ADCs (often with synchronized sampling), and sophisticated "ePWM" (enhanced PWM) modules.
    * **Key Peripherals:** High-resolution 12-bit ADCs (often 4+ Msps), integrated comparators, and ePWM modules that have a dedicated "trip-zone" input. You can route the comparator's output directly to this trip pin to instantly and safely shut down your switch.

---

2. MCUs with Advanced Analog Peripherals
These are general-purpose MCUs that have special families "tuned" for analog-intensive applications.

* **STMicroelectronics STM32 G4 Series**
    * **Why they're a good choice:** The G4 series was specifically designed by ST for digital power and motor control. It integrates a wealth of high-performance analog peripherals.
    * **Key Peripherals:** Up to five 12-bit, 4 Msps ADCs; seven high-speed comparators; six operational amplifiers; and "High-Resolution Timers" that can be cross-linked with these analog peripherals. This allows for the same "hardware-only" fast shutdown as the DSCs.

* **Microchip AVR DB / DD Series**
    * **Why they're a good choice:** If your application is simpler or more cost-sensitive, these 8-bit MCUs have outstanding analog features. They are one of the few MCUs to integrate **on-chip operational amplifiers** that you can configure in hardware.
    * **Key Peripherals:** You can use the op-amps to create a "non-inverting" or "differential" amplifier for your current-sense resistor. The output of that op-amp can then be fed *internally* to the Analog Comparator (AC) or the ADC. The AC's output can, in turn, be used to automatically shut down the timers (TCs) controlling your switch. This creates a complete, fast-acting protection circuit with almost no external parts.

* **NXP LPC55S3x / Kinetis KV Series**
    * **Why they're a good choice:** NXP's offerings are also very strong. The Kinetis KV series is designed for motor control with 16-bit ADCs and "FlexTimers." The LPC55S3x series features a very fast High-Speed Comparator (HSCMP) sub-system that can be directly linked to PWMs for fast protection.

👑 My Top Recommendations

* **For High-Performance / Fast Response:** Start with the **Microchip dsPIC33CK** or the **TI C2000 Piccolo** series. They are made for this exact problem, and the wealth of application notes and example code for "digital power" or "overcurrent protection" will be invaluable.
* **For High Integration / Lower Cost:** If your performance needs are more moderate, look at the **STM32G4** series or the **Microchip AVR DB** series. Their ability to integrate op-amps and link peripherals in hardware can significantly simplify your design and reduce your bill of materials (BOM).

Case Study Question & Response [3D] What MCUs will be best suited to be used in a closed loop system where the MCU picks up current and/or voltage readings and needs to either open or close a switch depending on the reading?

For a closed-loop control system with current/voltage sensing and switch control, here are the best MCU options:

Top Recommendations

1. STM32 Series (STMicroelectronics)

- **STM32G4**: Excellent choice with built-in high-resolution ADCs, analog comparators, and fast response times
- **STM32F3/F4**: Good ADC performance, plenty of GPIO, cost-effective
- Features: 12-bit ADCs (up to 5 MSPS), hardware comparators for ultra-fast response, built-in op-amps on some models

2. Texas Instruments C2000 Series

- Specifically designed for real-time control applications
- Fast ADCs, PWM generation, and control peripherals
- Best for motor control and power electronics applications

3. Microchip PIC/AVR Series

- **PIC16F/18F**: Simple, reliable, good ADCs
- **ATmega/ATtiny**: Cost-effective for simpler applications
- **dsPIC33**: For more demanding control loops requiring DSP capabilities

4. ESP32 (Espressif)

- If you need wireless connectivity alongside control
- Built-in ADCs (though less accurate than dedicated industrial MCUs)
- Cost-effective and easy to program

Key Features to Look For:

- **ADC resolution**: 12-bit minimum (16-bit for precision)
- **ADC sampling rate**: Fast enough for your control loop timing
- **Hardware comparators**: For threshold detection without CPU intervention
- **Interrupt capability**: Fast response to voltage/current changes
- **Timer peripherals**: For precise control loop timing
- **GPIO with sufficient current drive**: Or ability to drive external MOSFETs/relays

Selection Criteria:

For simple on/off control (relay/contactor): Almost any MCU works - Arduino Uno, STM32F0, PIC16F **For fast switching** (solid-state, < 1ms response): STM32G4, C2000, dsPIC33 **For precision measurements**: Higher resolution ADCs (16-bit), consider external ADC chips **For cost-sensitive**: ATmega328, PIC16F, STM32F0

What's your specific application? (voltage range, switching speed requirements, current levels?) This would help narrow down the best choice.

[4A] What are the cross-model comparison and observed patterns based on our content? "Across all three case studies, several consistent patterns emerged in how each LLM approached

engineering-focused prompts. ChatGPT tended to provide the most balanced responses, combining conceptual explanations with system-level reasoning and practical design context (ChatGPT, 2025). It was generally the strongest at articulating why a design decision mattered, although it occasionally introduced questionable citations or oversimplified certain assumptions.

Gemini, in contrast, demonstrated a hardware-centric orientation, consistently referencing electrical behavior, peripheral architecture, and embedded-system constraints with greater specificity than the other LLMs (Gemini, 2025). This made Gemini especially strong for analyzing microcontroller peripherals, but weaker in high-level comparisons or multi-disciplinary tradeoffs.

Claude showed a clear pattern of providing the simplest, most concise responses. While this made Claude useful for rapid brainstorming or introductory overviews, it generally lacked the depth required for engineering-level evaluation unless explicitly prompted to expand (Claude, 2025). While its summaries were accurate, its lack of unsolicited detail limited its usefulness in critical decision-making steps.

These observed trends helped guide when each LLM was used during the project. ChatGPT was used for theoretical explanation and application-level reasoning. Gemini was used when component-level or peripheral-level detail was required. Claude was used for quick idea validation or to obtain fast, high-level summaries before performing deeper research."

[5A]provide me the advantages of using an LLM that were to be usd in the senior design process " While each model had individual differences, several overarching benefits emerged from integrating LLMs into the workflow:

1. Faster Conceptual Alignment Among Team Members

LLMs allowed team members to quickly obtain consistent definitions and explanations of engineering concepts—especially during early design meetings. This reduced time spent searching through reference texts and helped ensure that all team members shared a foundational understanding before discussing design choices (ChatGPT, 2025).

2. Accelerated Design Exploration

LLMs generated rapid lists of potential components, architectures, and communication methods. Even when these lists required later verification, they significantly compressed the ideation phase. For example, the MCU comparison provided by ChatGPT and Gemini allowed the team to narrow down choices before consulting datasheets or application notes (Gemini, 2025).

3. Immediate Access to Analogies and Clarifications

When a concept was unclear—such as load behavior in switching circuits or synchronized ADC sampling—LLMs provided alternate explanations, analogies, and simplified breakdowns that supported deeper understanding (Claude, 2025). This helped especially when preparing documentation or explaining decisions to non-EE teammates.

4. Early Error Detection

Paradoxically, inconsistencies between LLMs acted as a quality-control mechanism. When responses disagreed, it signaled that additional independent research was needed. This cross-verification prevented over-reliance on a single model and reinforced engineering rigor"

[6A] What are the limitations, risks, and ethical considerations for using LLMs"
Despite the advantages, LLM usage posed meaningful limitations and risks:

1. Citation Hallucinations
All three LLMs occasionally produced incorrect or unverifiable citations. ChatGPT, in particular, frequently listed forum posts or generic tech blogs as "sources," which required manual validation (ChatGPT, 2025). Gemini provided more professional sources, but these still needed verification against manufacturer documents. Claude often avoided citations entirely unless prompted.

2. Overconfidence in Ambiguous Answers
LLMs often responded with confident-sounding explanations even when technically incorrect. For example, several responses across case studies overestimated certain microcontroller capabilities or provided incomplete explanations of communication timing. This reinforced the need to verify all technical output using datasheets, TI documentation, or manufacturer reference manuals.

3. Risk of Plagiarism
Due to the nature of LLM-generated content, citing the model and maintaining an appendix of prompts and outputs—as required—is essential for academic integrity (UCF SD Guidelines, 2024). All AI-generated text used verbatim in this report has been cited accordingly, and the full outputs appear in Appendix E per project guidelines.

4. Lack of Context Awareness
Because LLMs cannot see the full system unless explicitly described, they frequently overlooked design constraints such as safety margins, environmental conditions, or real-world noise sources. As a result, human engineering judgment remained critical."

[7A] provide me a conclusion that helps me summarize all that i have talked about for the LLMs and the senior design project: "The integration of LLM platforms supported faster ideation, increased conceptual clarity, and improved collaborative decision-making throughout the Senior Design process. However, the models also underscored the necessity of human verification and

engineering judgment, especially in areas involving safety, electrical characteristics, and system timing.

Through careful citation, cross-model comparison, and independent confirmation using authoritative sources, LLMs served as valuable tools without compromising the rigor or integrity of the design process.