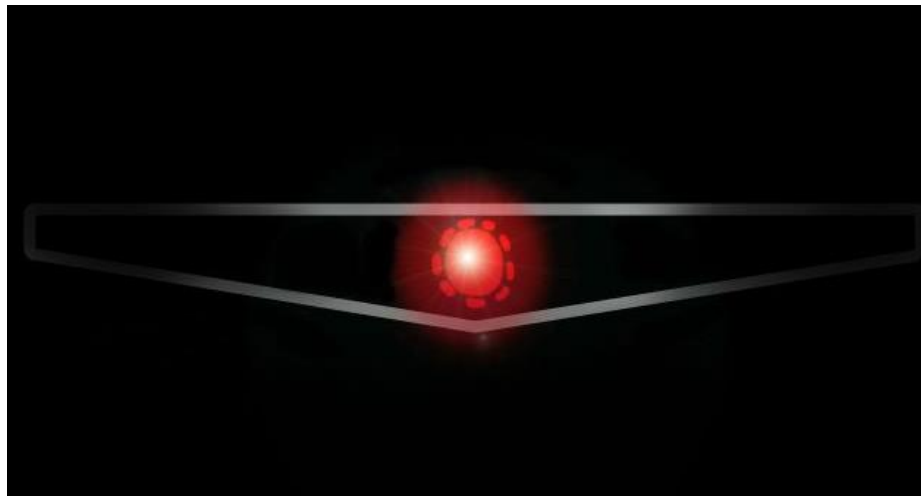

Senior Design I

CSS: Car Sentry

System



University of Central Florida
Department of Electrical Engineering and Computer
Science

Group 38 Members

Qrizelle Crisostomo - Computer Engineer
Ricardo Nunes Alcobia - Computer Engineer
Ari Pantoja - Electrical Engineer
Robert Zarrella - Electrical Engineer

Customers, Sponsors, Contributors:
Not Applicable

Table of Contents

List of Figures	vi
List of Tables	vii
1. Executive Summary	1
2. Project Description	3
2.1. Motivation and Inspiration	3
2.2. Goals and Objectives	5
2.3. Requirement Specifications	6
2.4. Marketing and Engineering Requirements	7
3. Research and Part Selection	9
3.1. Current Market Comparison	9
3.2. Hardware	10
3.2.1. Single Board Computer	11
3.2.1.1. Arduino Nano 33 BLE Sense	12
3.2.1.2. ASUS Tinker Board S - 2 GB	12
3.2.1.3. Nvidia Jetson Nano Developer Kit - 2 GB/4 GB	13
3.2.1.4. Raspberry Pi 4 Model B - 2 GB	14
3.2.1.5. Single Board Computer Comparison	14
3.2.1.5.1. Processing Power	14
3.2.1.5.2. Memory Size	15
3.2.1.5.3. Energy Consumption	16
3.2.1.5.4. Physical Features & Resources	16
3.2.1.5.5. Cost & Overall Consensus	17
3.2.2. Project Controller	18
3.2.2.1. Sensor Communication Requirements	18
3.2.2.2. Atmel	19
3.2.2.3. TI MSP430	19
3.2.2.4. Comparing Models	19
3.2.3. Accelerometer	21
3.2.4. Camera	21
3.2.5. SD Card	23
3.2.6. Bluetooth LE 5.0	24
3.2.7. GPS	25
3.2.8. Battery Life and Power	26
3.2.8.1. Technology Comparisons	28
3.2.8.1.1. Regulators	28

3.2.8.1.2. Battery: Li-Po, Li-Ion, LiFePo4, Ni-MH, Ni-Cd	29
3.2.8.1.3. Charger	31
3.2.8.1.4. AC/DC Adapter	32
3.2.8.1.5. DC/DC Adapter	32
3.2.8.2. Component Selection	33
3.2.8.2.1. Battery	33
3.2.8.2.2. Voltage Regulator	34
3.2.8.2.3. Charge Controller	36
3.2.9. Enclosure Survivability	37
3.3. Software	38
3.3.1. Computer Vision and Machine Learning	39
3.3.2. Investigation of Computer Vision Solutions	40
3.3.2.1. Microsoft Cognitive Services Computer Vision API	40
3.3.2.2. Amazon Rekognition	41
3.3.2.3. Google Cloud Vision API	41
3.3.2.4. OpenCV and YOLOv5	41
3.3.2.5. SimpleCV	42
3.3.2.6. Scikit-image	42
3.3.3. Comparison of Computer Vision Software and Libraries on the market	43
3.3.4. Mobile Component	44
3.3.5. Investigation of Mobile Application Development Strategies	45
3.3.5.1. Native Applications	45
3.3.5.2. Progressive Web Applications	45
3.3.6. Investigation of Mobile Application Development Technologies	46
3.3.6.1. MERN Stack	46
3.3.6.2. FERN Stack	46
3.4. Overall Structure	47
4. Related Standards and Design Constraints	48
4.1. Project Standards	49
4.1.1. Power Standards	49
4.1.1.1. Power Standards Impact	51
4.1.2. PCB Standards	52
4.1.2.1. Impact of PCB guidelines	53
4.2. Constraints and Operation	54
4.2.1. Constraints Overview	54
4.2.1.1. Ethical Constraints	56
4.2.1.2. Social and Political Constraints	58
4.2.1.3. Economic and Time Constraints	59

5. System Design	61
5.1. Hardware	61
5.1.1. Schematics	61
5.1.1.1. Power System Architecture and Operation	61
5.1.1.1.1. Charge Controller Primary Block	62
5.1.1.1.2. Charge Controller Output Block	64
5.1.1.1.3. Programming The Charge Controller	66
5.1.2.1.4. 5 Volt and 3.3 Volt Rails.	68
5.1.1.1.5. Power System PCB Considerations	70
5.1.1.2. MSP430 Circuit	70
5.1.1.3. Sensors	72
5.1.1.4. Sensor Connections	73
5.2. Software	74
5.2.1. Flow of Software on Single-Board Computer (SBC)	74
5.2.2. Computer Vision/Single-Board Computer Design	75
5.2.3. Flow of Mobile Application	77
5.2.4. Mobile Application Design	79
5.2.5. Mobile User Interface	80
5.2.6. Communication Protocol Interfacing	85
5.2.6.1. Single Board Computer: Jetson Nano	86
5.2.6.2. Accelerometer	87
5.2.6.3. Bluetooth & SD Card	89
5.2.6.4. GPS	90
5.2.6.5. OLED Display	91
5.2.6.6. Charge Controller Communication with System Controller	91
5.3. Stretch Goals	93
6. Project Integration and Testing	95
6.1. Integration	96
6.1.1. PCB Fabrication	96
6.1.2. Final Coding Plan/Design	97
6.1.3. Prototypes	98
6.1.3.1. Enclosure Prototype	98
6.2. Testing Procedures	99
6.2.1. Breadboard Testing	99
6.2.2. Power Testing	99
6.2.3. Hardware Testing	101
6.2.3.1. MSP430FR6989 Breadboard Testing	101
6.2.3.1.1. External MSP430FR6989 Target	101

6.2.3.1.2. Sensor Hardware Testing	102
6.2.3.1.3. Accelerometer	102
6.2.3.1.4. MicroSD Card	103
6.2.3.1.5. Bluetooth Module Testing	103
6.2.4. Firmware Testing	104
6.2.4.1. MSP430 Code Testing	104
6.2.4.2. Raytac MDBT42Q-P192 nRF52810 Code Testing	104
6.2.4.2. Bluetooth/MicroSD Troubleshooting	106
6.2.5. Software Testing	106
6.2.6. Survivability Testing	108
6.2.6.1. Collision Testing	108
6.2.6.2. Heat Testing	109
6.3. Project Operation	110
6.3.1. Plug and Play Functionality	110
6.3.2. Modes of Operation	110
6.3.3. Troubleshooting	111
6.3.4. Data Access	112
7. Project Administration	113
7.1. IEEE Student Grant Eligibility	113
7.2. Budget Analysis	115
7.2.1. Projected Cost	115
7.2.2. Actual Cost	116
7.3. Work Distribution	117
7.4. Milestones	118
8. Final Thoughts	120
9. Appendix	A
9.1. References	A
9.2. Permissions	E

List of Figures

- Figure 2.1: Example of License Plate Detection Using Computer Vision*
- Figure 2.2: House of Quality Diagram*
- Figure 3.1: Electronics Block Diagram*
- Figure 3.2: Field of View Example*
- Figure 3.3: Power Block Diagram*
- Figure 3.4: Software Block Diagram*
- Figure 3.5: Costs of Computer Vision Solutions at Scale*
- Figure 3.6: High-level Overview of CSS Functionality*
- Figure 5.1: Power System Schematic: Charge Controller Primary Block*
- Figure 5.2: Power System Schematic: Charge Controller Output Block*
- Figure 5.3: Power System Schematic: 5 Volt Rail and 3.3 Volt Rail*
- Figure 5.4: MSP430FR6989 Net Label Connections*
- Figure 5.5: Accelerometer/BT/MicroSD/MSP430 Schematic*
- Figure 5.6: Headers for Programming on the PCB/Jetson Communication*
- Figure 5.7: Single Board Computer Software Flow Diagram*
- Figure 5.8: Mobile Software Application Flow Diagram*
- Figure 5.9: Mobile Application Component Use Case Diagram*
- Figure 5.10: Mobile Application Component Wireframe*
- Figure 5.11: Login Page and error message on iPhone 12*
- Figure 5.12: Registration Page on iPhone 12*
- Figure 5.13: “Forgot Password” Page with error message and notification on iPhone 12*
- Figure 5.14: Main Page on iPhone 12*
- Figure 5.15: SPI Interface between MSP430FR6989 and Jetson Nano*
- Figure 5.16: SPI Timing Diagram. The use of this image is governed by the GFDL.*
- Figure 5.17: I2C Interface between MSP430FR6989 and Accelerometer*
- Figure 5.18: I2C Timing Diagram. This image has been released to the Public Domain.*
- Figure 5.19: SPI Daisy Chain Interface between MSP430, BT, and μ SD*
- Figure 5.20: UART Interface between MSP430FR6989 and GPS*
- Figure 5.21: Charge Controller Input and Output Level Shifters*
- Figure 5.22: Charge Controller Battery Charge Current Measurement Output to MSP430*
- Figure 6.1: Integrated System Schematic*
- Figure 6.2: 3D Rendering of CSS Enclosure First Draft*
- Figure 6.3: SBW Connection for External MSP430 Programming*
- Figure 6.4: Development Boards Reference: Green - Direct Pin Program, Blue - J-Link*
- Figure 7.1: Achieving IEEE 1725-2021 Standards*

List of Tables

- Table 2.1: List of Objectives*
- Table 2.2: List of Hardware Requirements*
- Table 2.3: List of Software Requirements*
- Table 3.1: Single Board Processor Power Comparison*
- Table 3.2: Single Board Computer Memory Size Comparison*
- Table 3.3: Single Board Computer Energy Consumption Comparison*
- Table 3.4: Single Board Computer Physical Features Comparison*
- Table 3.5: Single Board Computer Cost Comparison*
- Table 3.6: Summary of Sensor Communication Method*
- Table 3.7: Microcontroller Specifications*
- Table 3.8: Accelerometer Specifications*
- Table 3.9: Camera Comparisons*
- Table 3.10: PCB Breakout Comparison*
- Table 3.11: Bluetooth Version Comparison*
- Table 3.12: Battery Technology Comparison*
- Table 3.13: Battery Pack Configuration Comparison*
- Table 3.14: 3.3 Volt Regulator IC Comparison*
- Table 3.15: 5 Volt Regulator IC Comparison*
- Table 3.16: Charge Controller IC Comparison*
- Table 3.17: Comparison of Computer Vision Solutions and Desired Characteristics*
- Table 4.1: List of Project Standards*
- Table 4.2: Components, models, certifications, and standards based on IEEE*
- Table 4.3: List of Project PCB Design Standards*
- Table 4.4: List of Constraints*
- Table 4.5: Comparison between default mode and LPM*
- Table 5.1: System Current Consumption Verification*
- Table 5.2: SPI Mode Comparisons*
- Table 5.3: I2C Mode Comparisons*
- Table 5.4: Charge Controller and System Controller Communication Signals*
- Table 5.5: List of Stretch Goals*
- Table 7.1: Minimum Subsystem Requirements for IEEE 1725-2021 Standard*
- Table 7.2: Project Purchasing and Financing*
- Table 7.3: Actual Cost*
- Table 7.4: Work Distribution*
- Table 7.5: Project Timeline of Fall 2021*
- Table 7.6: Project Timeline of Spring 2022*

1. Executive Summary

The inspiration for this project arose from personal driving experiences and experiences with a dash cam. Driving is very dangerous and accidents can occur. Bad drivers can worsen these situations by leaving the scene. A dash cam oftentimes does not provide the quality necessary to make out license plate information and identify them. Higher quality ones that are able to deliver the resolution to do so often costs a fortune. Even then, the further away they are, the harder it is to make out the plate information. The solution to this is an automatic license plate reader (ALPR) that will scan and log the license plate information of all vehicles within its field of view.

The ALPR device for this project will have the objectives of being lightweight, low profile, and portable so that a user can place it in any car. It will also have the ability to run solely on battery power for several days. The device to be developed will be similar to a dash cam in the sense of the constraints imposed upon by the environment it is being placed in, that being, the windshield or the dashboard of the vehicle. This comes with size limitations and high heat considerations. These constraints will be driving factors in determining the outcome of the device, since it must be able to fit either behind the rearview mirror, somewhere on the windshield, or the dashboard. The last two options require that it is low profile so that it will not obstruct or distract the driver from the task at hand.

However, unlike a dash cam, which has to be constantly recording footage, the device will make use of computer vision to power the license plate recognition. This allows for frequent images to be taken and scanned to extract only the necessary information from the image before discarding the image. In this manner, the device will not have to consume as much power by constantly recording and storing the data. The objective is to save on battery life so that the device can last several days of a typical daily American commute.

Additionally, the device will also include a low power mode (LPM) that will conserve battery life even more by turning off all unnecessary components and only waking up to begin recording data when the on board accelerometer detects a jolt. This is expected to vastly extend the battery life. The default mode would be one in which the device is constantly scanning and logging data as it encounters them. A user will have the option to decide which mode fits their needs best.

Optimization of battery life is a major consideration due to the fact that the device will have a single board computing module that will be running the computer vision required to detect vehicles and scan license plates. This is an intensive task that requires a lot of power. It is also expected to generate a lot of heat in a small enclosure. This adds to the list of constraints and research into how to best optimize to reduce power draw and heat generation, so that the device may operate reliably as intended.

As with all modern electronic devices today, this ALPR will also include wireless connectivity so that a user can offload the data from the device onto their mobile device and sync it with the cloud. The decision to add a wireless component was made for the convenience of a user. It will give them options as to how to interact with the device. Most

modern electronic devices have also now become internet of things (IoT) devices. It would be out of place and possibly a hindrance to users to not add in wireless capabilities. It will also provide additional backup security for the device. The mobile app will initially be a bare-bones app that will simply be able to sync the data and view data on the database. If time permits, this can later be expanded and fleshed out to provide more functionality for a user such as deleting data or updating device settings remotely.

The scope of the project can get very large very quickly when considering all the different license plates in existence and features for an ALPR. Therefore, the project will primarily focus on scanning Florida license plates and only storing the essential information, such as license plate information and vehicle model and color. The former will reduce the time necessary to train the algorithm and the later will save battery life and storage space in order to achieve the objectives of the project. Expanding the license plates that the device can reliably scan and adding in other features such as the ability to store images or record videos will become stretch goals that will be worked on if time permits.

2. Project Description

2.1. Motivation and Inspiration

Driving is a dangerous activity. People do not realize the risks associated with driving. When it is put into perspective, a human operating three-thousand pounds of moving steel barreling down the road with others seems daunting. To make matters worse, human error increases the chances of vehicle accidents to occur. Many civilian vehicles are not equipped with any form of surveillance camera systems to log these instances for insurance and/or legal purposes.

The motivation behind the project stems from personal experience driving around the city of Orlando and seeing first hand situations in which it would be useful to quickly obtain a vehicle's license plate information. This could be the case of a hit and run situation, reckless driving, simply remembering a vehicle for future reference, or any other myriad of reasons to obtain a vehicle's license plate information. Therefore, this brings up the biggest question, why not simply purchase a dash cam? As it currently stands, a good dash cam with reasonable video quality will cost more than a hundred dollars, if not even more than that. Even then, it might still be hard to read the license plate information of a vehicle more than twenty feet away since the dashcam incorporates a very wide field of view. Additionally, it takes up a lot of storage and fills up local storage rather quickly, thereby limiting the amount of data that can be stored and for how long. This could be remedied with cloud backups or manual backups, however this adds to cost and time. The alternative solution is a license plate scanner.

Automated license plate recognition (ALPR) has existed since the late 1970s. This form of technology has been and is currently being utilized by law enforcement. These computer-controlled camera systems are typically mounted on street poles, streetlights, highway overpasses, and police squad cars. In the field, ALPRs would automatically capture and store data, including but not limited to: license plate numbers, location, date, time, and photographs of the vehicle and the driver/passengers in the vehicle. This information would then be uploaded to a server. Data collected by these devices would be used by police for real-time and historical investigations and be used as evidence in a related crime.

In addition to law enforcement, technological advancements are being made in the automobile industry. Tesla, Inc. is notable for producing smart, autonomous vehicles. Autonomy is achieved by the implementation of cameras, advanced radar sensors, and advanced intelligence. While ALPR technology has not been incorporated in the production of Tesla vehicles, their features can be modified to increase functionality. A security researcher, Truman Kain, took it into his own hands to develop the Surveillance Detection Scout. In this project, Kain developed a computer that plugs into a Tesla's dashboard USB port [1]. The computer then converts the vehicle's built-in cameras into a system that detects, reads, and stores license plate information and recognizes human faces. The security feature comes into play as the computer sends out an alert on the Tesla's display notifying the driver if it repeatedly sees a certain license plate or human. Kain's

motivation for the project was to provide extra security for the user in the case that a perpetrator is preparing to steal or break into the vehicle.

In the case of the average day-to-day driver, they will not need excess power or capabilities required for law enforcement personnel or the luxuries of a Tesla. If we consider the price points of a high grade ALPR used in law enforcement and a Tesla alone, both are sold at exorbitant prices that are not feasible for a majority of people. Throughout this project, we will design an affordable license plate scanner that will meet the needs of an everyday driver and can be utilized in any vehicle. Existing ALPR technologies will be reduced in this project to lower the market cost while meeting minimum requirements to satisfy the needs of a daily commuter.

The driver will most likely need a device that will capture license plate information in the case they find themselves in a dangerous situation involving another vehicle. This will be achieved by developing necessary computer vision software while installing the required (and minimum) peripherals. Since the driver is not going to need to remember every vehicle encountered for an extended period of time, there will be no need for massive storage space (either local storage or in the form of cloud storage) or constant data flow found in law enforcement ALPRs. Due to the lack of advanced camera features and sensors installed in a majority of vehicles, there will also be a mobility factor, giving users the ability to configure the system based on their vehicle layout. This project will take the basic functionality of an ALPR used in law enforcement while taking the similar computer logic integrated into the Tesla with additional enhancements to increase performance and further decrease cost and power.



Figure 2.1: Example of License Plate Detection Using Computer Vision

Shown above is a visualization of the behavior of an ALPR. A device capable of being able to instantly record a vehicle's license plate information provides the primary goal for creating a portable and low-profile license plate scanner for the average driver. The device will have the ability to log the date and time and obtain a vehicle's license plate information using computer vision (*Figure 2.1*) powered by openCV or Yolov5. Preliminary research seems to show the Yolov5 would be better at object recognition compared to openCV. However, upon further research, it was discovered that this isn't an either or question, instead, they can both be used in conjunction with one another to power the license plate

recognition software. This would most likely be the better option since in this manner we would potentially be able to take advantage of the benefits of both while mitigating their shortcomings.

2.2. Goals and Objectives

In designing our project, our overall goal was to produce a low-cost, portable license plate scanner that can be used by the average day-to-day driver. CSS will be able to capture license plate and vehicle information and store collected data into local storage while being as power efficient as possible. This will then extend to the development of the mobile component that will display the collected data to the user on their mobile device. In the event of an accident, data obtained will be recoverable. CSS will achieve the objectives listed below in Table 2.1.

The nature of ALPR leads it to have many potential applications, not just for law enforcement and daily drivers. Therefore, CSS will be developed to be versatile in order to accommodate the addition of future stretch goals and the ability to be applied to numerous scenarios. A potential application would be tracking the movements of a vehicle within a given area, such as an airport or theme park. To ensure that CSS remains versatile, the codebase will be programmed to be modular and readable so that it can be expanded upon.

#	Objectives
2.2.1	Plug-and-Play functionality
2.2.2	Lightweight & Portable Design
2.2.3	Crash Survivability
2.2.4	Prevent Obstruction in Driver View
2.2.5	Scan license plate information for enclosed, 4-wheel civilian vehicles
2.2.6	Accurately identify vehicle attributes (Model and Color)
2.2.7	Build to IEEE/IEC/UL Standards
2.2.8	Modular codebase

Table 2.1: List of Objectives

2.3. Requirement Specifications

There will be two forms of requirement specifications that will be relevant throughout this project: hardware and software. These requirement specifications will act as the foundation to produce an accurate, viable license plate scanner that can be used by the standard driver.

The hardware requirements specifications relate to the needs of the physical components and overall construction. Hardware requirements include battery life, camera capabilities, unit size and weight, sensor implementation, and enclosure protection. CSS will meet the hardware requirement specifications described in the table below.

#	Hardware Requirements
2.3.1	Battery life will last several days (3 - 5 days LPM)
2.3.2	Camera of at least 1080p and greater than 20 fps to read vehicle attributes
2.3.3	Troubleshooting light indicators to indicate storage capacity and battery life
2.3.4	Overall design must be < 2lbs
2.3.5	Will not exceed a size of (5" x 4" x 4")
2.3.6	Contain an accelerometer to monitor speed of vehicle
2.3.7	Bluetooth 5.0+ that is backwards compatible to sync to variety of devices
2.3.8	Have a GPS Module to determine vehicle location*
2.3.9	Enclosure is able to protect the local storage device so that it is still readable 90% of the time after surviving a two story drop

*Table 2.2: List of Hardware Requirements / * Stretch Goal*

Software requirement specifications identify what the system should do and services it provides. CSS must be able to recognize and store license plates and other attributes. Not only will this information be stored on the local storage unit, but it will be accessible to CSS users on a mobile application. To increase efficiency, several features will be implemented in the software component. The table depicted below states the software requirement specifications of CSS.

#	Software Requirements
2.3.10	Have the ability to recognize/read license plate information
2.3.11	Ability to detect an accident (impact vs. hard braking)*
2.3.12	Have the ability to offload data from local storage device
2.3.13	Have the ability to wirelessly backup data
2.3.14	Features multiple power modes to provide users with flexibility: <ul style="list-style-type: none"> • Low Power Mode (LPM) - Activates when vehicle is stationary and only stores data when collision detected • Default/Active Mode - Always keeping all components on and storing data
2.3.15	Will have the ability to write vehicle attributes onto local storage device
2.3.16	The software application will be synced to the hardware via wireless network*
2.3.17	Have the ability to write location coordinates onto storage unit*
2.3.18	Features multiple security modes to provide users with flexibility:* <ul style="list-style-type: none"> • Essentials Mode - Only stores license plate information, vehicle model, vehicle color, and time stamp. • High Security Mode - Stores video as well

Table 2.3: List of Software Requirements /* Stretch Goal

2.4. Marketing and Engineering Requirements

The marketing requirements were picked based on what a user would prefer. These characteristics correspond to what a user would value as being important. As can be seen in the house of quality diagram, a user values lower cost the most. This has a strong positive correlation with costs for engineering requirements.

Engineering requirements were determined based on what was required to design and construct the device at hand. To make an efficient, low cost, and compact device, only a few of these characteristics strongly align with what a user would prefer. However, these correlations provide a foundation as to where to design and build off of. The additional constraints will have to be taken into consideration for the design and component picking of the device.

The marketing and engineering requirements were measured using a house of quality diagram. This matrix depicts the impact of each marketing and engineering requirements against one another while identifying what can be achieved. Shown below is the house of quality diagram (*Figure 2.2*) for CSS.

		Engineering Requirements					
		Cost	Dimensions	Power Output	Setup Time	Weight	
		+	-	+	-	+	
Marketing Requirements	Durable	+	↓				↓
	Low Cost	+	↑↑		↓		↑
	Easy to Use	+	↓	↓		↑↑	
	Easy to Install	+	↓	↓↓		↑	↓
	Small Form Factor	-	↓↓	↑↑	↓	↑	↑↑
	Long Battery Life	-	↓↓	↓↓	↓↓		↓
			< \$150	4"x4"x4"	< 30 Watts	< 5 Mins	< 5 lbs.

Legend			
Correlation		Polarity	
Strong Positive	↑↑	Positive	+
Slight Positive	↑	Negative	-
Slight Negative	↓		
Strong Negative	↓↓		

Figure 2.2: House of Quality Diagram

3. Research and Part Selection

Several factors are to be considered when identifying and selecting parts required for this project. Components are categorized based on the microsystems of the overall CSS unit. Categories include electronics, power, and software. Electronics contribute to CSS's functionality. These parts will primarily focus on data collection and transmission. Parts related to the power system will help provide the necessary power for the CSS unit to operate. Hardware components will then be narrowed down based on efficiency. Although there are no physical software components, we will consider various software tools to program the applications and algorithms.

Parts to be selected must contribute to the functionality of CSS. This unit must be able to obtain data related to the driver's vehicle and vehicles nearby. Data includes speed, location, physical attributes, and license plate information. For this data to be collected, the team must research and select a camera, accelerometer, and GPS module. Users must also be able to access this information. A storage unit and a transferring module will be required. This will be achieved using a specified SD card and Bluetooth LE 5.0 module. Most of the components mentioned will then be connected to a project controller of the team's choice to facilitate the inputs. The chosen project controller will then be connected to a single board computer that will handle the computer vision algorithm based on the camera input.

The CSS unit must be able to operate in a certain time frame. This means that there are power specifications to be considered. The system will need a charger, voltage regulators, batteries, AC/DC adapters, and DC/DC adapters. The specifications will be investigated and components will be narrowed down based on its capabilities and the optimizations that can be made. Due to the current shortage and limitations, it may be difficult to optimize the power system. Thus, we will consider power optimization as a stretch goal.

Automating the performance of identifying license plates, storing, and transferring information will require programming tools. The computer vision algorithm, firmware, and mobile application will require the use of libraries, various software, development strategies, and developmental technologies. There are many online tools available, however, these tools will be selected based on the team's current knowledge, learning curve, and overall convenience.

CSS comprises all three subsystems. Components for each subsystem will be researched, analyzed, and selected.

3.1. Current Market Comparison

Initial research of automatic license plate readers (ALPR) currently on the market reveals that they tend to be large, expensive, and exclusive to law enforcement agencies [2]. These enterprise solutions are more than \$500 per camera and some may even include software subscription fees on top of the initial cost. They also need to be hooked up to the car and mounted permanently. Additionally, a majority of these ALPR solutions are only available

to law enforcement. The aim of the project will be to create a solution that is less expensive, portable, easy to set up, and potentially available to the public, either for individual use or for use by businesses. The latter will need further research into, as Florida has laws and regulations [3] in place with regards to ALPR. Preliminary reading of the law seems to suggest that it would be fine under Florida law. However, a lawyer may be needed as they would be best in determining the legal impact and responsibilities of making such a project available commercially or open to the public.

3.2. Hardware

The hardware of CSS will comprise of electronics and power components. Power components pertain to the flow of energy. These components will provide a sufficient supply of energy to keep the electronic components powered, in order to achieve the objective of having a portable device. More details regarding power will be discussed in the “Battery Life and Power” section. The components to be powered contribute to the exchange of data. The electronics system consists of several modules and peripherals, a project controller, and a single board computer. The following figure shows the block diagram for the electronics portion of the hardware.

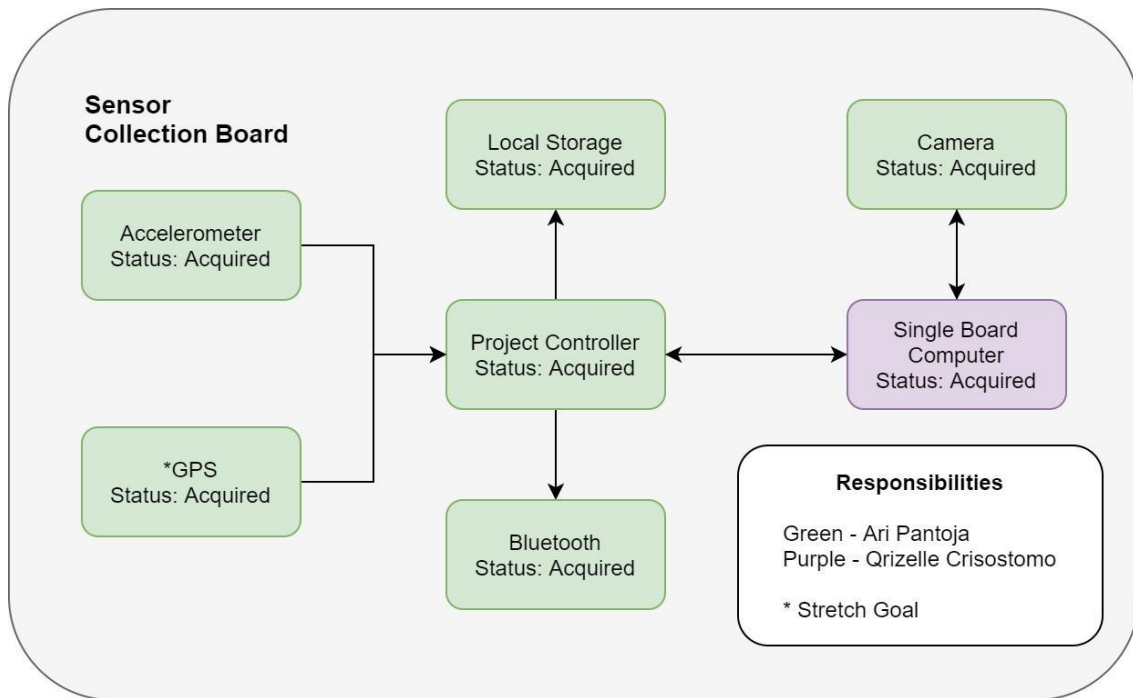


Figure 3.1: Electronics Block Diagram

Power and electronics components will be researched and discussed. Each product considered will be analyzed and compared to other products. Through comparison, the best component will be selected and implemented in the design.

3.2.1. Single Board Computer

Computer vision is an essential and necessary component in the functionality of the CSS, therefore this project will require a computing device that is capable of running an intensive machine learning algorithm or the computer vision process that will enable the data collection. A single board computer will be implemented to execute the computer vision algorithm that will capture, interpret, and store license plate information and vehicle attributes. Due to the demands of the algorithm, the board must have a substantial amount of processing power and memory. The availability of resources such as libraries, community forums, and support will be a significant factor in the determination of a single board computer. As these resources will accelerate development. Additionally, the size and extensiveness of programming libraries must be taken into account as to whether the board will be able to support them. Furthermore, a community forum will be useful in the process of debugging any issues that arise, either with development or with getting started with using the resources. For CSS to obtain input data and produce the proper output, the system must make the appropriate hardware connections.

The single board computer must have the proper I/O ports to make all the necessary hardware connections and establish communication across the entire system. It will require the attachment of a camera to obtain the necessary input for the algorithm to function. In addition to the camera, there must be a connection to the project controller. After the data obtained by the single board computer is finalized, it shall be transferred to the project controller for it to be stored on the local storage unit. In this manner, the data obtained will be modular and can easily be removed to be accessed by the user. Besides the hardware connections, another hardware feature to consider is board optimization.

Although there is higher priority for software integration and hardware connections, we must consider optimization. One way to optimize the overall system is determining the physical size. The board must not exceed the physical size constraints. Established in the “Requirement Specifications” section, the overall system must not exceed the dimensions 5” x 4” x 4”. With this requirement and the incorporation of other hardware elements, the board size must be less than the overall system size. Decreasing the physical size means further optimizations can be made by reducing the energy needed for the board. Energy consumption plays a major role in the development of the system. A decrease in energy consumption allows for energy efficiency. However, this must not strain the board’s ability to execute the computer vision software in a timely manner. One other optimization to make on the development side is cost. We must consider the cost of the single board computer in correlation to the features provided. Despite the lowest cost being ideal, we must ensure that the low-cost single board computer has all the tools and qualities needed to provide an accurate, functional system.

There are four potential boards being investigated to determine which of the four best meets the criteria and needs of the project: Arduino Nano 33 BLE Sense, ASUS Tinker Board S, Nvidia Jetson Nano Developer Kit, and Raspberry Pi. Considering the role of the single board computer in CSS, we must take into account the following properties: processing power, memory size, energy consumption, physical features, resources, and cost. These

will be the determining factors of which single board computer will work best for the overall system.

3.2.1.1. Arduino Nano 33 BLE Sense

Arduino is most known for providing microcontrollers that handle basic input and output functions. Their products and their capabilities have expanded to handle more complex data sets with the integration of machine learning and artificial intelligence. The Arduino Nano 33 BLE Sense is an extension of the Arduino Nano that is suitable for machine learning algorithms. The board features a combination of the ARM Cortex-M4 microcontroller with onboard sensors. These include a microphone, and proximity, color, and movement sensors. A product like this is ideal when used in end applications such as wearable/motion technology.

The team has considered the Arduino brand due to its simplicity, price, size, machine learning capabilities, and its large and active community of developers. Due to the shortage in electronics, the microcontroller's price ranges from \$10 to \$50. Based on availability and shipment dates, the team has looked at a board that is listed for \$22.50 on the retailer Amazon. The listed Arduino Nano 33 BLE Sense has a form factor of 3.07 x 2.4 x 1.18 inches, making it ideal for the system build. Additional features include the board's advanced on-chip interface. This unit supports UART, I2C, and SPI, allowing for flexibility when it comes to communication between the Arduino Nano and the selected project controller peripheral. Specifically, it features 2x UART and up to 4x SPI master/3x SPI slave with EasyDMA and up to 2x I2C compatible 2-wire master/slave. Nordic SoftDevice also supports concurrent use of multiple protocols. There are 48 general purpose I/O pins for necessary peripheral hardware attachment.

Machine learning algorithms are executed through the ARM Cortex -M4 32-bit processor, running at 64 MHz. The code and data stored on the Arduino Nano are located on the available 1 MB of flash memory and 256 KB of RAM. While running core processes, power consumption averages to approximately 19 mAh in normal mode, but can be optimized in low power mode.

For further information in regards to debugging, diagnostics, and technical specifications, there are many resources available. The Arduino Nano 33 BLE Sense comes with a datasheet detailing various hardware and software aspects of the board. Arduino also has a dedicated website containing tutorials, guides, forums, and reference libraries for each of their Arduino products. These available resources make the Arduino Nano easy to use and troubleshooting simple.

3.2.1.2. ASUS Tinker Board S - 2 GB

The Asus Tinker Board S provides ease-of-use for a new user and offers increased durability and stability, featuring onboard microSD and 16 GB eMMC storage for better performance and a microSD slot. This also features an ARM-based processor – the

Rocketchip RK3288 – providing the power needed to execute the license plate recognition algorithm.

On Amazon, the Asus Tinker Board S is listed for \$199.99. The dimensions listed for this board is 3.37 x 2.125 inches, which is under the size requirement specification. Included in the build is a heatsink attached to the board, which is ideal for hot ambient environments, such as a warm stationary vehicle. There are several standard connectivity options, such as a 40-pin header with a 2-pin I2C with both master and slave modes, 4-pin UART, and 2-pin SPI. These interfaces will allow for the connection of the project controller, but are also capable of handling several sensors and modules needed for CSS. A CSI MIPI connection is available for compatible cameras. In the case the system is to be enhanced for user interaction using a display, a DSI MIPI connection is available. Depending on the peripherals attached and the usage of the CPU and GPU, the board may use around 700-1000 mA.

Programs will run through the Cortex-A17 Quad-core 1.8 GHz CPU. The board includes a Mali-T760 MP4 GPU with four cores and a clock speed of 600 MHz. Considering the number of cores and the speed/frequency, these specifications ensure fast performance and response time across our main application. The Tinker Board S also offers a 2 GB dual-channel DDR3 to provide flexible memory bandwidth.

Asus provides many resources to assist those working with the Tinker Board S. These resources include video tutorials, forums, blogs, and documentation. Tinker OS source code and several third party code is accessible to the public, which is a good starting point for those first working with the board.

3.2.1.3. Nvidia Jetson Nano Developer Kit - 2 GB/4 GB

NVIDIA Jetson Nano Developer Kit is a small, powerful computer dedicated to run machine learning applications such as image classification and object detection. Priced at \$62.84 on Amazon, the board has a form factor of 2.72 x 1.77 x 1.77 inches and features high-performance components. Memory utilized by the system is a 2 GB LPDDR4 at 25.6 GB/s. Included onboard is both a GPU and a CPU. It is equipped with a quad-core ARM A57 that runs at a clock speed of 1.43 GHz and a 128-core Maxwell GPU.

As computer vision is one of the primary focuses of the project, the Jetson also features video encoding and decoding specifications that can be useful. It can encode 4x at a quality of 1080p at 30 fps and decode 8x for the same video quality at the same frame rate. A camera connection can be made through the 2x MIPI CSI-2 DPHY lanes located on the Jetson Nano. To allow communication between the project controller and the Jetson, there are several communication protocols that can be utilized. These protocols include GPIO, I2C, I2S, SPI, and UART.

Energy consumption varies on the number and types of peripherals attached to the board. When no peripherals are attached, the minimum power drawn is 5000 mW. If the system

is fully operational and working every aspect of the board, power consumption can reach up to 10000 mW.

Similar to that of the previous boards, NVIDIA provides all the resources and tools needed to learn and troubleshoot the Jetson Nano. In search of answers to a problem related to the board, there are many video tutorials, forums, communities, and documents out there to reference. NVIDIA also includes an extensive library that not only goes over the hardware specifications of the board, but also software related items that relate to the interface.

3.2.1.4. Raspberry Pi 4 Model B - 2 GB

Raspberry Pi is another brand notable for tiny, portable computers. The Raspberry Pi 4 Model B provides mobility and power in its compact form of 3.94 x 2.76 x 1.18 inches. It offers increases in speed, performance, memory, and connectivity. Pricing starts at \$35. However, due to recent shortages, available Raspberry Pi boards are listed for \$83.95 and up. Installed on the system is 2 GB LPDDR4 memory. This product's main features include a quad-core ARM-based Cortex-A72 with a clock frequency of 1.5 GHz. A GPU is not included in the build.

Despite the lack of a GPU, a 2-lane MIPI CSI port is available for a camera connection and video encoding and decoding can be done as well. Other physical features of the board include the standard 40-pin GPIO header. According to the pinout for the Raspberry Pi 4 Model B, the GPIO has pins designated for interfaces such as SPI, I2C, and UART. Based on the physical connections made, the energy consumed in the system can range from 2700 – 6400 mW.

The tools and resources available are similar to those listed previously. Users of Raspberry Pi's products get access to videos, documentation, and forums relevant to the board.

3.2.1.5. Single Board Computer Comparison

The four options discussed in the previous section were narrowed down to the one that will be integrated in the CSS system. The selection process was done through direct comparisons made based on each product's processing power, memory size, energy consumption, onboard features, available resources, and cost. Each aspect was a determining factor to select the most suitable single board computer for CSS.

3.2.1.5.1. Processing Power

Having enough processing power is essential for the execution of the computer vision software. Processing power indicates overall performance when the program is being run. Processing power can be split between two different units, if they are presented on the board: the CPU and GPU. The clock frequencies of all the processing units on each board were recorded. The following table shows the CPU and GPU clock frequencies.

Single Board Computer	CPU Clock Frequency (MHz)	GPU Clock Frequency (MHz)
Arduino Nano 33 BLE Sense	64 MHz	N/A
Asus Tinker Board S	1800 MHz	600 MHz
NVIDIA Jetson Nano	1430 MHz	640 MHz
Raspberry Pi 4 Model B	1500 MHz	N/A

Table 3.1: Single Board Processor Power Comparison

Both the Arduino and Raspberry Pi do not include a GPU. However, the Tinker Board and Jetson Nano come equipped with GPUs that run at nearly the same speed. The absence of a GPU in the Arduino and Raspberry Pi puts both at a disadvantage. Analyzing the clock frequencies themselves, the Tinker Board's CPU performs slightly better compared to the Jetson Nano's CPU. The opposite can be said in regards to their GPU clock frequencies.

3.2.1.5.2. Memory Size

Random access memory is another key component in identifying system performance. It will provide our application a place to temporarily store and access data generated from the algorithm. An increase in memory can allow for more complex programs. Shown below are the specifications of memory on each board.

Single Board Computer	Memory (GB)
Arduino Nano 33 BLE Sense	0.001 GB FLASH 0.000256 GB SRAM
Asus Tinker Board S	2GB Dual Channel DDR3
NVIDIA Jetson Nano	2 GB LPDDR4
Raspberry Pi 4 Model B	2 GB LPDDR4

Table 3.2: Single Board Computer Memory Size Comparison

Three of the four boards have a version of DDR while having the same memory capacity. The Asus, NVIDIA, and Raspberry Pi come in 2 GB forms. The dual channel DDR3 found on the Asus provides RAM with two channels, one dedicated for reading and the other for writing. Low-power double data rate (LPDDR) on the NVIDIA and Raspberry Pi consumes less power and is often used in mobile devices.

Based on the data collected, the Arduino is the most lightweight product. Although it is capable of machine learning and artificial intelligence, the capacity is not substantial enough for the expected load. As for the remaining boards, they are all relatively similar.

NVIDIA and Raspberry Pi are favored as these devices are dedicated for mobility. The LPDDR4 featured on both boards can contribute to the decrease in energy consumption. Which would be desirable in order to achieve the objective of being portable as the battery can last a longer time without recharging.

3.2.1.5.3. Energy Consumption

Battery life is crucial as it directly affects CSS’s ability to operate. Drivers should have no concerns related to the system running out of power during a trip. According to the Federal Highway Administration, American drivers average around 13,476 miles annually. To satisfy the national average, CSS is required to function for approximately three to five days. Table 3.3 below depicts the minimum and maximum power consumption across all four single board computers.

Single Board Computer	Operating Voltage (V)	DC Current Min - Max (mA)	Power Consumption Min - Max (mW)
Arduino Nano 33 BLE Sense	3.3V	15 - 330 mA	49.5 - 1089 mW
Asus Tinker Board S	5 V	500 - 1000 mA	2500 - 5000 mW
NVIDIA Jetson Nano	5 V	1000 - 2000 mA	5000 - 10000 mW
Raspberry Pi 4 Model B	5 V	540 - 1280 mA	2700 - 6400 mW

Table 3.3: Single Board Computer Energy Consumption Comparison

Power consumption was calculated by multiplying the operating voltage and the minimum/maximum current. Both the minimum and maximum power consumption is considered. The maximum power consumed shows how much power can be drawn if the board is used to its full capacity. The minimum depicts its potential in optimization. The Arduino Nano has the lowest minimum and maximum values, which is ideal to further reduce the amount of power being used. However, the range of the Arduino Nano would not be able to function based on the workload of the entire system. The Arduino would not be a good contender in comparison to the other boards listed.

3.2.1.5.4. Physical Features & Resources

The main features being looked at are the connection pins and the product dimensions. An ideal system would contain as many connection protocols as possible to provide flexibility when attaching necessary peripherals. The system would also have the smallest dimensions due to the size requirement that must be satisfied. The table below shows the available connection pins and protocols for each system.

Single Board Computer	Product Dimensions (inches)	UART	I2C	I2S	SPI
Arduino Nano 33 BLE Sense	3.07 x 2.4 x 1.18 inches	✓	✓		✓
Asus Tinker Board S	3.37 x 2.125 inches	✓	✓	✓	✓
NVIDIA Jetson Nano	2.72 x 1.77 x 1.77 inches	✓	✓	✓	✓
Raspberry Pi 4 Model B	3.94 x 2.76 x 1.18 inches	✓	✓		✓

Table 3.4: Single Board Computer Physical Features Comparison

The Raspberry Pi is the largest with the dimensions set to 3.94 x 2.76 x 1.18 inches. If we compare that to the build requirement, the board takes up more than 50% of the size requirement. With the implementation of the project controller and modules, we may exceed the size limit. On the other hand, the Jetson Nano has the smallest build in addition to having all UART, I2C, I2S, and SPI connections. It is clear that the Jetson Nano is favorable as it occupies less than 50% of the size requirement and has more connections to allow for the use of various protocols.

While working with the boards, we must refer to different resources that may be useful when troubleshooting issues or learning about the board's capabilities and features. All listed boards offer the same types of resources from their respective companies. NVIDIA, however, provides an extensive, detailed library about every aspect of the Jetson Nano. With NVIDIA dedicated to providing products with high-performance, there is a great deal of information available to make the development process easier.

3.2.1.5.5. Cost & Overall Consensus

System cost is important as we want CSS to be accessible to as many drivers as possible by being affordable for them. The percentage of board cost was calculated based on our current budget of \$497.52. While keeping costs low is significant, we must make sure that the costs line up with everything the board must provide. Table 3.5 below displays the cost per board and the percentage cost of the board compared to our overall budget.

Low cost will be a major deciding factor in determining which board will be picked. However, it will be taken into consideration whether or not the team is confident in the board's ability to achieve a sufficient level of performance to execute the computer vision algorithm. As well as being able to interface with all the other components required for the device. Another consideration will be the ease of development and the resources available to assist in developing on that platform. Taking these factors into account and after obtaining data from analysis of the board, it was not too difficult when it finally came down to picking a board.

Single Board Computer	Board Price (\$)	Percentage of System Cost (%)
Arduino Nano 33 BLE Sense	\$22.50	4.5%
Asus Tinker Board S	\$199.99	40%
NVIDIA Jetson Nano	\$62.84	12.6%
Raspberry Pi 4 Model B	\$83.95	16.9%

Table 3.5: Single Board Computer Cost Comparison

Table 3.5 shows that the Arduino is the cheapest option, but through careful analysis, the Arduino does not meet the requirements needed for this project. The Asus Tinker Board is shown to be the most expensive and consists of 40% of our overall budget. Although it does contain better specifications in some areas, the differences between it and the NVIDIA Jetson Nano are marginal. While the price savings on the other hand, are significant..

The team has made the decision to purchase the Jetson Nano for the SBC component of the project due to its performance, features, and price. Although it falls short in some categories to the Asus Tinker Board S, the performance differences are small relative to the price difference. Despite the much larger power consumption of the Jetson Nano relative to the other boards, it contains all the functionalities desired to achieve the objectives laid out for the project. All while only taking up 12.6% of the budget compared to the much higher 40% by the Asus Tinker Board S or the slightly higher cost of 16.9% by the Raspberry Pi 4 Model B. Due to this, the Jetson Nano becomes the clear choice as the CSS's dedicated single board computer.

3.2.2. Project Controller

To consolidate the number of inputs on the single board computer GPIO pins, and for the sake of offloading some processing power, a microcontroller will act as a project controller between the sensors and the single board computer. An inspection of the sensor data communication methods was required to pinpoint an appropriate microcontroller model. Research on TI and Atmel brands revealed capabilities in line with the project requirements and still fit the cost and power consumption budget.

3.2.2.1. Sensor Communication Requirements

First and foremost, to narrow down a microcontroller, it was helpful to know how many communication features in total we will need to connect to our sensors. Based on the knowledge that SPI is the fastest communication protocol available through the GPIO pins, a preliminary review suggests that SPI might be the most appropriate communication method between the single board computer and the project controller. For each selected sensor, the communication scheme is listed below in Table 3.6.

Part	Communication Scheme
Accelerometer	I2C
SD Card	SPI*
Bluetooth	UART, I2C, or SPI*
GPS (Stretch Goal)	UART
Single Board Computer	SPI
* - can use the same pin for both connected in an SPI daisy-chain configuration	

Table 3.6: Summary of Sensor Communication Method

3.2.2.2. Atmel

Atmel is the brand of processors used for Arduino microcontrollers. Arduino has a wealth of source code and community support to help make any project a reality. Arduino has an IDE (Integrated Development Environment) with most libraries included and an easily accessible menu to get started with already-written source code to get the user talking to sensors. Our team considered Arduino for the CSS project because at least one team member had experience using their microcontrollers. Additionally there is a wealth of resources for getting started and developing on an Arduino platform. The Atmel ATmega2560 microcontroller was considered because it has comparable features to the TI MSP430 family. Atmel is similarly very appealing due to its low cost.

3.2.2.3. TI MSP430

Most of our coursework experience involved the use and programming of a TI MSP430 microcontroller. We are already familiar with TI's Code Composer Studio to program TI's microcontrollers, so they seem most appealing. Using the software the team is knowledgeable with will save valuable time and encourage the team to tackle every facet of this project. Another big draw to the TI MSP430 family is the ultra-low power consumption (on the order of μW in Standby mode). Battery life is a big concern for this project. It is desirable to use as little power as possible outside of the single-board computer, the project's biggest energy consumer.

3.2.2.4. Comparing Models

An additional factor paramount to this project is time. Though the MSP430 can transmit via I2C in milliseconds, how long will it take to receive data from the accelerometer and then wake from a sleep mode? Which TI or Atmel model is the fastest? Which microcontroller has the most significant throughput and adequate accessible memory while

remaining the most energy-efficient? The table below compares each of the considered microcontroller's technical specifications discussed side by side.

	ATmega2560	MSP430BT5190	MSP430FR698x	MSP430F249
Architecture	8-bit RISC	16-bit RISC	16-bit RISC	16-bit RISC
Clock Speed	16 MHz	25 MHz	16 MHz	16 MHz
Voltage Range	2.7V – 5.5V	1.8V – 3.6V	1.8V – 3.6V	1.8V – 3.6V
Active Power Mode	500 μ A	230 μ A/MHz	100 μ A/MHz	270 μ A/MHz
Standby Power Mode	11.85 mA	1.2 μ A	0.4 μ A	0.3 μ A
Wake Time from Sleep/LP Mode	Unknown	< 5 μ s	6 μ s	< 1 μ s
Non-Volatile Memory (kB)	256	256	128	60
RAM (kB)	8	16	2	2
UART	4	0	2	2
I2C	1	4	2	2
SPI	1	4	4	2
Bluetooth Version	N/A	2.1	N/A	N/A

Table 3.7: Microcontroller Specifications

On an interesting note, TI offers an MCU with built-in Bluetooth capabilities. However, it is merely Bluetooth version 2.1. Bluetooth versions may be backward compatible, but they are not forward compatible in the sense that older versions cannot send data to a newer version device. Though it has a high clock speed and relatively low active mode power consumption, it has zero UART capabilities. If we hope to realize our stretch goal of GPS implementation, we need at least one UART connection.

While the ATmega2560 has a memory size equivalent to the MSP430BT5190, it pulls more than twice the current of any other MSP430 model. Its standby mode current draw is laughable in comparison to any MSP430. The ATmega2560 can also only process 8 bits at a time, which is exactly half the size of the MSP430.

Though the MSP430F249 packs a competitive punch in terms of standby power consumption and wake time from sleep mode, overall the active power mode consumption

pales in comparison to the MSP430FR668. For the sake of cost and convenience, the MSP430FR6989 is preferable since said part is already in our possession, though it may have capabilities above our needs. We can test our sensor hardware connections and programming on the TI Launchpad also acquired from previous coursework.

3.2.3. Accelerometer

Considering our power constraints, we intend to have the option of entering a sleep mode for the CSS that will wake when a sudden jolt is detected. One way to accomplish this is to employ an accelerometer with at least a 2-dimensional axis. The accelerometer needs to withstand the most violent force possible in the event of an accident. A 75 mph car accident experiences an acceleration upwards of 60g. The accelerometer module must be able to survive the acceleration experienced in a crash without catastrophic failure. It is the one sensor that will always need full power to detect an accident even when the vehicle isn't in motion. As such it is desired to find a part with the lowest power consumption. Table 3.8 shows the comparison of accelerometer specifications.

	SMT LIS2DS12TR	Memsic MXC4005XC	NXT MMA8491QR1
Acceleration Range	2g, 4g, 8g, 16g	2g, 4g, 8g	8g
Voltage Supply Range	1.62 V - 1.98 V	1.8 V - 3.6 V	1.95 V - 3.6 V
Output Type	I2C, SPI	I2C	I2C
Highest Current Draw	150 μ A	1.6 mA	980 nA/Hz
Absolute Maximum Acceleration	10,000 g	200,000g	10,000g
Cost	\$1.94	\$1.49	\$2.69

Table 3.8: Accelerometer Specifications

In addition to the above considerations, availability is a major motivator in deciding which module to use. The price, voltage supply range, and absolute acceleration survivability leads one to believe that the Memsic MXC4005XC is the best choice for the CSS.

3.2.4. Camera

The camera is perhaps one of the most important sensors for the CSS. It functions as the eye from which the primary functional data will be extracted. As opposed to the other sensors used, the camera will be directly connected to the single board computer by way

of a MIPI CSI-2 port. For the purposes of license plate recognition, the field of view and image resolution are a priority. The lowest required image resolution is at least 1080p. This translates to a frame of at least 1920 pixels wide by 1080 pixels high or 2.1 megapixels. Since the CSS will only be taking in images from the front of the vehicle it would seem on the surface that the biggest field of view is wanted. However, the wider the angle in the field of view the more likely there is to have distortion at the edges of the frame, creating a fisheye effect. A full fisheye field of view is 180 degrees. A field of view of 130 degrees was voted upon as a compromise to still obtain as much data as possible in the image. Figure 3.2 provides a visualization of the camera's field of view.

As can be seen in Figure 3.2, a 75-degree field of view would provide a narrower cone for the camera to capture. By having a field of view of at least 120-degree or greater would be preferable as this enables the camera to view adjacent lanes that are in front of the vehicle as well. Increasing the field of view too much to something such as 180 would distort the image as previously mentioned. A field of view within the range of 120 to 160 degrees would be preferred and this led the team to selecting the goal of about 130-degrees for the camera.

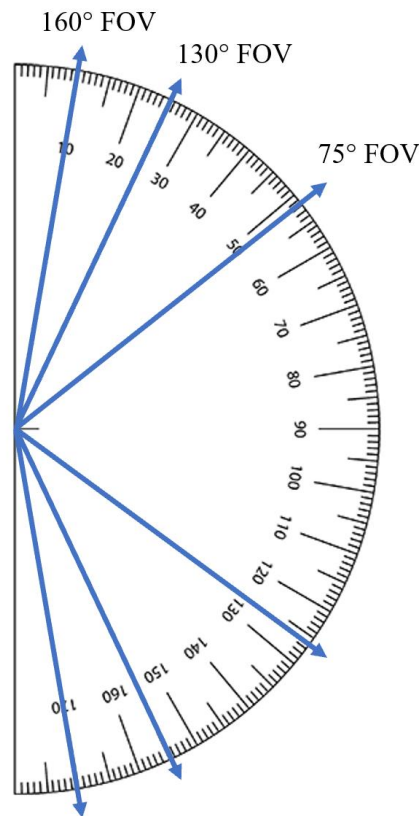


Figure 3.2: Field of View Example

There are many cameras available made to be specifically compatible with the NVIDIA Jetson Nano. Most of them at or above 8-megapixel resolution. Once cameras surpassed the 8-megapixel mark the field of view decreased substantially. Nightvision was also

considered but ultimately decided against since the cost increased significantly for the feature of any deterministic quality and lower-priced versions were unavailable. Shown below is a table containing the camera comparisons.

Camera	Waveshare IMX219-160	Waveshare IMX219-130	Arducam IMX477
Image Resolution	8 MP	8 MP	12.3 MP
Field of View	160°	130°	75°
Cost	\$29.95	\$19.90	\$64.99

Table 3.9: Camera Comparisons

3.2.5. SD Card

Our project will utilize a microSD card outside of the one used by the NVIDIA Jetson Nano to store license plate information. This will allow more permanent storage of the license plate data than the mobile app alone. The microSD card writer will need to be protected enough to withstand significant impact as this will be a main source of license plate information in the event of a serious accident. The mobile app will display suspect license plates and information, but the microSD card will be our main non-volatile storage.

The decision to use a microSD card was realized after considering the factors of weight and size on the board. Our goal is to be as light and small as possible, especially since the single board computer and power system will be so heavy and bulky. In designing our circuit for the microSD card writer, we must take caution to protect the microSD card from variances in voltage. This situation could corrupt the data and render the card useless in data retrieval. A secondary failsafe should be implemented at the microSD card pin level in the form of another 3.3V regulator, in addition to the regulator in the power circuit.

For the purposes of testing a prototype, we will need a breakout PCB to connect to a breadboard. We found 3 different breakout boards to choose from on Digi-Key. One by a company called Pololu, one by Adafruit, and one by Sparkfun. These are compared side by side in Table 6.11. As far as selection goes, the microSD card connection hardware is pretty much the same across suppliers. Differing options observed are with or without a spring-loaded design. The main difference between the 3 is the presence of a 5V to 3.3V converter. We will have 5V and 3.3V regulated power rails in our designed power circuit available to utilize so a 5V to 3.3V converter on-board is not necessary for our purposes. The next refinement to our selection is the obvious cost. We chose the cheapest option since we just need the microSD card pin connections easily accessible. Table 3.10 shows each PCB breakout board's cost and converter ability.

	5V to 3V3 Converter	Cost
Pololu	No	\$3.49
Adafruit	Yes	\$7.50
Sparkfun	No	\$4.75

Table 3.10: PCB Breakout Comparison

Outside of the hardware, there are programming options to consider. Data transfer can take place using SD mode or SPI mode. In SD mode there is either a 1-bit transfer or 4 bit transfer scheme. Though our microSD card writer may have the capacity to perform SD mode, it is a "proprietary transfer format" to which we do not have access. Our intention of using a small, low-power microcontroller limits our ability to use the SD mode.

In previous coursework, we learned about the SPI (Serial Peripheral Interface) mode and how to implement it on the TI MSP430FR6989 MCU. Because of this, and since SPI is the most common form of communication between device and host in SD cards, we will be using SPI mode.

3.2.6. Bluetooth LE 5.0

Our team chose Bluetooth 5.0 because it is the newest version that we currently have access to, and it is backward compatible with all prior Bluetooth versions. Being backward compatible would enable our device to communicate with a vast number of different phones. Shown in the table below, Bluetooth 5.0 has a greater range, faster response times, and a decent maximum message size compared to previous versions.

	Bluetooth 3.0 or Earlier	Bluetooth 4.x	Bluetooth 5.x
Range	100 m	100 m	200 m
Latency	< 100 ms	< 6 ms	< 3 ms
Message Size	<= 358 bytes	31 bytes	255 bytes

Table 3.11: Bluetooth Version Comparison

It is vitally important that we comply with all regulations surrounding wireless communications. Specific frequencies are restricted to government or regulated use, and we want to make sure we don't accidentally break any laws. We also need to be sure that our project does not produce any harmful emissions. The best way to go about this is to use a prepackaged Bluetooth transceiver that is already FCC certified. The Federal

Communications Commission, or FCC, regulates all interstate and international communications in the United States [4]. To be FCC certified means that the technology adheres to correct frequency usage and does not emit harmful radiation [4].

We have decided on an FCC-certified surface mount Bluetooth module, capable of transmitting data using Bluetooth 5.0, with a built-in PCB antenna. There seems to be only one module available meeting the requirements of surface mount, internal antenna, and Bluetooth 5.0. The Laird BL651 was in stock a few days ago when added to the Digi-Key cart. We hesitated on purchasing right away, and because of the current electronic shortage caused by the pandemic, Digi-Key was sold out when the decision was made to purchase. Searching around different supplier sites for a comparable module turned up nothing. Every supplier has sold out. Digi-Key had the fastest projected availability with a date of 02/28/2022. We are currently back-ordered.

Speaking with an Electrical Engineer working in the field yielded some leads on the Bluetooth module search. He stressed that with an anticipated arrival date well into next semester, it was time to abandon ship on the originally chosen Bluetooth modules. A Bluetooth 5 Low Energy Module featuring Nordic nRF52810 is apparently what we are looking for. That was the key we needed. Once the Nordic brand was thoroughly researched, it was clear we should go with this brand for more reasons than just availability. Compared to the Laird BL652 module we were initially going to purchase, a Raytac MDBT42Q-P192KV2 similarly operates using Nordic 2.4 GHz Bluetooth technology and an ARM Cortex M4 32-bit 64 MHz processor.

Nordic has an entire software suite dedicated to programming their modules called nRFConnect. It can be interfaced with Microsoft Visual Studio Code for a more familiar programming experience. In order to program an external Nordic module like those available for purchase online, a Nordic Development Kit is needed. We have found that a demo board is also needed to be able to break out the pins from the tiny surface mount device. Programming the Raytac MDBT42Q-P192KV2 device requires either a JLINK debugger/programmer (\$500+) or the Dev Kit by Nordic (\$39) with a built-in JLINK debugger/programmer. The Dev Kit is then connected to the demo board with the target device via JLINK and programming can commence. Unfortunately, it doesn't seem that there is any easier way to program the Raytac MDBT42Q-P192KV2 module that's available for purchase. It needs to be soldered to the demo board, programmed, desoldered, and then resoldered on our custom PCB. It seems like there are too many opportunities for something to break off or become unusable, but we must soldier on.

3.2.7. GPS

As mentioned, a stretch goal would be to equip the CSS with a GPS sensor for additional features. This would enable the ability to obtain the location of the CSS device at all times if the user desires to do so. The addition of a GPS sensor would expand the capabilities of the CSS greatly. Logging a vehicle's location is incredibly powerful and would allow for that data to be analyzed. This has applications in law enforcement for scenarios such as tracking a vehicle's whereabouts and also has applications in traffic management. As previously mentioned, this could be applied at theme parks or airports. This could be used

to determine how much time people spent at a location before moving on to another location. An airport could use this to find out information such as how long it takes people to drop off or pick up someone. This data could then be used to influence airport design or construction. Along the same vein, theme parks and other large businesses could use this data to analyze how customers move around the premises.

However, at the moment, this feature is not crucial to individual users, nor does it add much benefit to the user besides being able to confirm the location in which they saw a specific license plate. For that reason, the addition of the GPS has become a stretch goal so that focus may be concentrated to more time sensitive features. If time permits, the addition of a GPS module will not be overly difficult. However, development of the associated features to utilize this capability may take a significant amount of time, as it would require setting up a database to store all the information and analyzing the data to track vehicles and length of time spent in a specific location.

3.2.8. Battery Life and Power

The goal is to have the CSS last a minimum of 3 days of a typical American daily commute as determined by the Census Bureau. Which is approximately 40 minutes to an hour a day [5]. The on-board accelerometer will come in hand in extending battery life as it can be used to determine the speed of the user's vehicle. The speed could then potentially be used to set the frames per second (FPS) of the camera. The lower the speed, the less FPS necessary to obtain a suitable image to analyze. This can then be optimized so that the CSS is able to save on power consumption at lower speeds as it does not have to process as many frames as when it is at higher speeds.

Additionally, since a license plate scanner will not have to be recording a continuous video to process its data, it has the added benefits of an extended battery life over that of a dash cam, which would be constantly writing the video to a local storage unit. In fact, the goal is to have it run primarily on battery power rather than requiring users to supply it with power from the car like most modern day dash cams, however that will be an option for the user. Most dash cams currently have a small battery or use a super capacitor for the event of a sudden power loss so that the dash cam can have the opportunity to save data before shutting down. Since a continuous video doesn't need to be taken and constantly be saved, the CSS will be able to run solely on battery power if the user so chooses to do so. That way the user will not be required to constantly charge it.

The benefit of the increased battery life is that it provides users with flexibility in placement. Currently, most common dash cams require being wired in or plugged directly into a car charging port. In addition to the loss of a charging port, a user has to figure out a way to route the wires of the dash cam in such a way that the wires are out of the way and not distracting. Even then, bumps could knock wires out of place unless they are strapped down. As previously mentioned, the battery life of the license plate scanner would provide users with the flexibility of choosing whether they want to have it constantly plugged in or simply run it off the battery only and charging it only when necessary. Figure 3.3 below displays a block diagram of CSS's power system.

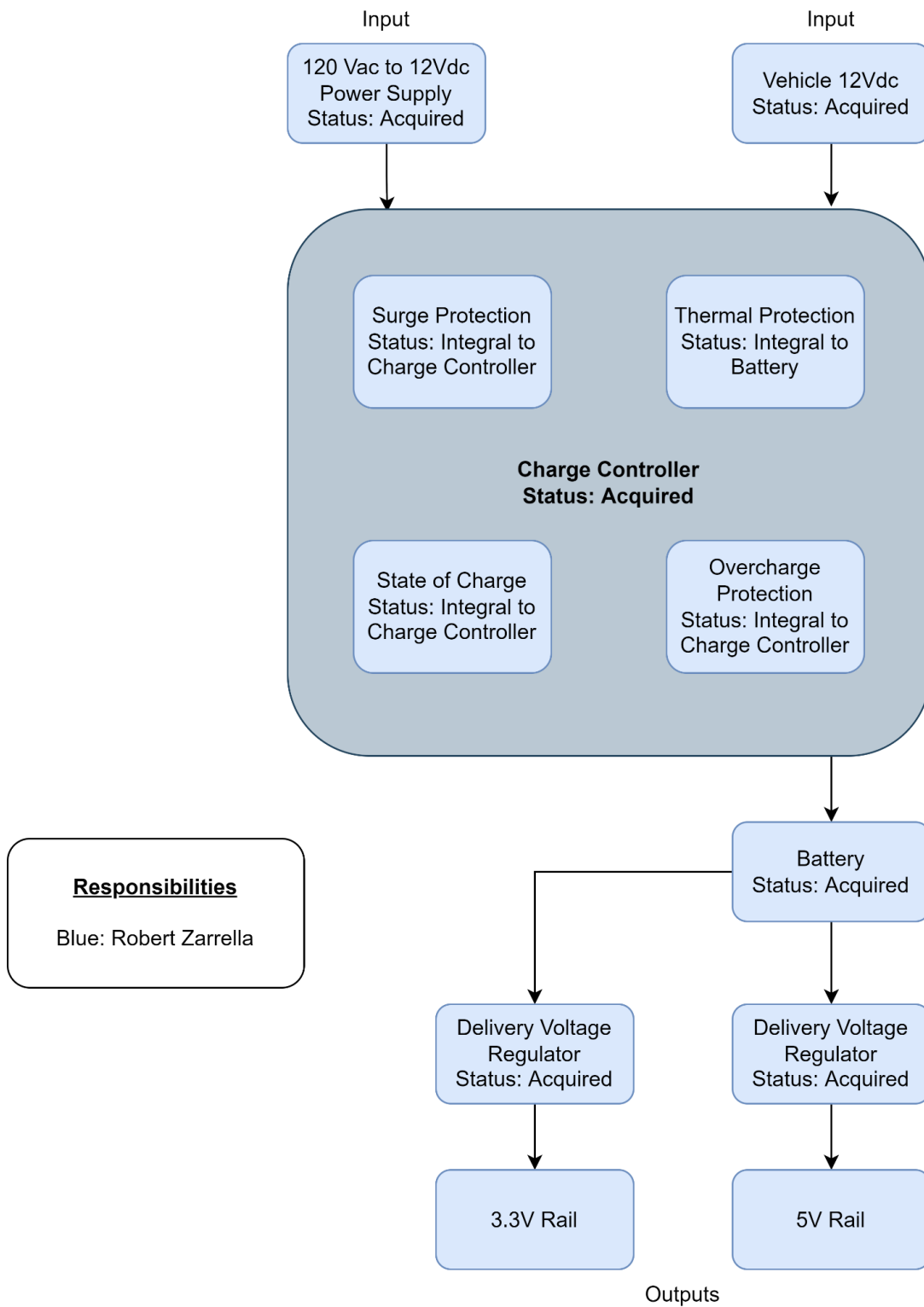


Figure 3.3: Power Block Diagram

In order to achieve the objectives of the device there are numerous battery factors to take into consideration. One of which is the constraint of the environment. Such as heat; since there will be situations in which the car is sitting out in the sun. Therefore, the battery will have to be able to withstand such conditions. Additionally, there is also the size constraint. A battery of enormous size is not preferred as it would conflict with the objective of keeping the device portable.

Because the device is portable in nature, multiple battery charging modes must be supported to provide maximum usability. The primary charging and operating power supply mode will see the device connected to a 12 Vdc power port in the vehicle. Since the vehicle power supplied fluctuates from 12.5 Vdc (Engine Off) to 14.75 Vdc (Engine on, Cruise RPM), the charging circuit of the CSS must be able to accept these fluctuating voltage values as well as protect from voltage sag and spike during the high current draw of starting the engine. Additionally, the CSS should be able to be charged from 120 Vrms mains voltage using a commonly available AC/DC converter. This gives the user intent on operating solely on battery power the option to either charge the CSS using an always-on 12V port in the vehicle while parked or to bring the device indoors with them and charge it from a wall socket. By facilitating both options, the primary goal of operating under battery power without cords is maintained while minimizing the possibility of the device running out of power.

Additional considerations for the charging circuitry is avoiding overcharging the battery, because overcharging damages the battery by (depending on the battery chemistry) reducing capacity, reducing lifespan, rapidly increased pressure, thermal runaway, or possibly combustion. Any power storage technology that is under consideration (Li-Ion, Li-Po, LiFePo₄, Ni-MH, etc.) is sensitive to overcharging and careful consideration must be taken to ensure that overcharging is prevented. The charge controller must therefore be designed to incorporate some method of measuring the state of charge such that an accurate accounting of the battery capacity is recorded by the charge controller to measure the net charge being delivered to the battery. Offline methods are those which require the device operation be interrupted in order to measure indicators of the remaining charge capacity, but the system must be ready at any moment which renders any offline State of Charge (SoC) methods undesirable. This leaves only the online SoC measurement methods to be considered.

3.2.8.1. Technology Comparisons

3.2.8.1.1. Regulators

There are two technology modes available for DC to DC voltage regulators: Linear and Switching. Linear regulators provide voltage regulation through power dissipation, using either a variable resistance in series with the load or a variable resistance with current shunted to ground. Because the resistive element is in series with the load, all current passing through the resistor passes into the load and all available current is used. Although the current is used by the load, the linear regulator has taken a large input voltage and dissipated power to reduce the output voltage to the desired level, rendering the efficiency of the series linear regulator incredibly low, especially if there is a larger input/output

voltage differential. The shunt-type linear regulator controls the output voltage by using a variable resistor in parallel with the load, effectively shunting current through the resistor to reduce current into the load such that the load effective resistance times the load current produces the desired load voltage. This operation wastes current through the shunt resistance to ground and is even less efficient than the series linear regulator and is only a valid design choice in extremely low power systems. Since both technologies regulate the voltage through a variable resistor, all unused power is transformed into heat.

In contrast to the low efficiency linear regulators, switching voltage regulators use internal clock signals to modulate the input voltage by rapidly (KHz to MHz range) opening and closing the circuit between the input and output ports. Given this high switching rate, even though the voltage is alternating from 0V to V_{in} , the effective V_{out} of the regulator is regulated to the desired level by changing the pulse width of the modulation in reaction to load changes. The act of the voltage regulation causes no power loss and theoretically results in 100% power efficiency, but the switching regulator consumes a small amount of quiescent power to operate. Despite this power consumption, the switching regulator is much, much more efficient than either type of linear regulator and is desirable in designs where power consumption is critical. Switching regulators require additional external components to program the regulator for the desired voltage output. Despite the fact that these external components are commonly transistors, resistors, capacitors, and inductors, their combined power consumption is still far less than that of the linear regulator.

Due to the large power consumption of the computing hardware in the CSS design, switching voltage regulator technology is chosen so as to reduce unnecessary power consumption wherever possible. In addition to low power consumption, the switching regulator produces far less heat than the linear regulator, which is a critical consideration as the CSS will be exposed to the high temperatures present in the direct sunlight of an automotive dashboard.

3.2.8.1.2. Battery: Li-Po, Li-Ion, LiFePo4, Ni-MH, Ni-Cd

Available battery technologies span the gamut from lead-acid to LiFePO₄. As stated in the requirements specifications, the appropriate battery technology for the CSS would: have energy density high enough to contain approximately 10,000 mAh, be small enough in volume to fit in the 4 x 4 x 5 inch housing, be able to operate in temperatures approaching 150° Fahrenheit, have low self-discharge to maximize run time per charge, provide sufficient supply current to allow proper operation at full power consumption, and accept charging currents high enough to allow full charge in 4 to 5 hours. Regardless of the technology selected, the battery pack shall comply with UN Manual of Tests and Criteria, Section 38.3 in order to satisfy the constraints put forth by the IEEE 1725-2021 standard [6] such that the design is eligible for the IEEE Student Grant [7].

Immediately eliminated from consideration is the lead-acid battery (including absorbent glass mat technologies), due to its immense size, immense weight, and unsealed water-acid solution.

The next battery technology under consideration is Nickel-Cadmium (Ni-Cd). Ni-Cd batteries are the least expensive of the available technologies and have simple charge/discharge profiles such that complexity and cost could be saved when choosing an appropriate charge controller. Ni-Cd batteries are also extremely rugged in that they have high impact tolerance and a large range of allowable charge and discharge temperatures which make them well suited for the high temperature environment of the CSS. Ni-Cd batteries also have high self-discharge currents (approximately 20% of maximum charge per month above 20° C).

There are many disadvantages to the Ni-Cd technology which make it less than ideal for deployment in the CSS, namely: the lowest energy density of any battery technology considered. In addition, Ni-Cd batteries lose effective energy capacity if they do not receive periodic full discharge-charge cycles, which would be critical as the energy capacity is already quite low compared to the other technologies under consideration. Additionally, Ni-Cd batteries provide low usable output current, on the order of 0.1C which, for a 10,000 mAh battery would only output 1000 mA for a power output of

$$7.2V * 1000 \text{ mA} = 7.2 \text{ W}$$

while a single board computer such as the nVidia Jetson Nano consumes between 5 and 10 Watts. This low output would force the CSS to throttle the compute power of the SBC, which could cause critical performance issues.

Nickel-Metal-Hydride (Ni-MH) batteries have largely replaced Ni-Cd batteries in common usage due to their higher energy density for a similar low cost. Ni-MH also has higher usable discharge rates of approximately 0.2C, which would provide an effective output power with the same battery capacity of 10,000 mAh. This exceeds the base power requirements of the SBCs under consideration, but the rest of the system (system controller, sensors, etc.) could push the power consumption beyond these bounds and again induce compute throttling.

The largest detractor for the Ni-MH technology is that it has a high self-discharge current on the order of 4% to 20% of total capacity lost per day at around 45° C and increasing with temperature. Given that the CSS is expected to experience sustained temperatures around 60° C, Ni-MH effective capacity would actually be far less than the nominal 10,000 mAh and not ideal for our purposes.

Lithium-Ion (Li-Ion) technology has vast improvements beyond Ni-MH and Ni-Cd technologies. It has a much higher energy density (185 TO 220 WH/L) than either of the previously investigated battery technologies, as well as a much higher discharge rate of 1C, with some batteries reaching 10 C. Given a Li-Ion battery with capacity 10,000 mAh discharging at 1C, the output power provided would be which is far more than the power requirements of the CSS design. Additionally, Li-Ion has an extremely low self-discharge rate of only 0.3% per month, lending off-charger stability to the system.

The major drawback of the Li-Ion technology is that it is extremely sensitive to high temperature and has strictly controlled charge and discharge temperatures. If the

environment exceeds the limits, Li-Ion batteries can leak electrolytes or even explode. Since the intended environment will consistently reach 65° C, the Li-Ion is not well suited to our purposes.

Lithium Polymer (Li-Po) batteries provide many of the same benefits as Li-Ion, while being vastly more stable in high temperature environments. The self-discharge rate is similar to Li-Ion at 0.3% per month and has similar design discharge rates of 1C giving similar power output of 72W for this application. Li-Po outperforms Li-Ion in high temperature conditions, with a stable charge temperature up to 50°C, as well as being far less likely to experience catastrophic failure such as combustion of electrolyte leakage. Additionally, Li-Po technology allows for flat and even flexible battery form factors, which provides maximum usable volume. Li-Po has a lower energy density than Li-Ion at only 135 to 150 WH/L, but this is still significantly higher than Ni-Cad or Ni-MH.

Lithium Iron Phosphate (LiFePO4) batteries continue the trend set by Li-Po batteries in that they are even more stable in high temperature environments, present little danger if they do fail while maintaining a high discharge rate, high energy density, and low self-discharge rate. Unfortunately, they have lower specific energy such that a LiFePO4 battery pack of sufficient voltage and capacity (6 to 7 volts at 10,000 mAh) would require a battery pack weighing 1.4 pounds. This weight would require an excessively large mounting system to keep the CSS secure in environments experiencing deceleration inherent in an automotive accident. Shown below is a table of battery comparisons and the technology selected.

Technology	Energy Density (WH/L)	Charging Rate (C)	Power Output (W)	Self-Discharge (%Capacity/Month)
Ni-Cd	100	.85	7.2	20
Ni-MH	140	Up to 1	14.4	40 up to 90
Li-Ion	250	Up to 10	72	.3
Li-Po	150	Up to 10	72	.3

Table 3.12: Battery Technology Comparison

3.2.8.1.3. Charger

The battery charge controllers utilize one of three available technologies: linear, switching, and pulse. Linear and switching operation operates on the same principles discussed earlier for voltage regulation, while the pulse charging technology is unique to charging circuits. Pulse charging technology permits full current to the battery until battery voltage nears full regulation voltage, the charge pulses (at much lower frequency than a switching charger) to reduce current flow and prevent excessive voltage at the battery.

Switching charge controllers are the most power efficient topology of the three. Similar to switching regulators, they have low power consumption providing for high efficiency, and low quiescent power consumption which is critical in obtaining the CSS battery life specifications. The primary drawbacks of switching charge controllers are cost and complexity. The external components required to program the controller and provide for

current delivery to the battery are more numerous than either the linear or pulse controllers, while the switching controller itself is more costly than either of the other controllers.

Linear controllers are the least power efficient topology of the three. The pass transistor consumes the input/output voltage difference and dissipates it as unused heat, which drastically reduces efficiency as well as increasing the heat in the CSS system which is already near the practical limits of the specified battery technology. The advantages of the linear controller are that it requires far fewer external components, and the IC is less expensive than either other option.

The pulse mode controller provides a middle ground between the linear and switching charge controllers, in that it is much more efficient than the linear controller while being less expensive and requiring fewer external components than the switching controller. The largest drawback of the pulse mode controller is that it requires an accurately current limited voltage source, which is difficult to source if it is even available in the voltage and current needed. Further complicating this matter, is that a DC/DC voltage source for use with the automotive 12V system is even more difficult to source than a current limited AC/DC voltage source.

3.2.8.1.4. AC/DC Adapter

The CSS will utilize an AC/DC converter for charging the battery indoors off of 120V AC circuits. Available AC/DC converters utilize a transformer to achieve rectification with the addition of flyback voltage regulation to provide steady DC output at the rated voltage. Although nearly all available AC/DC converters utilize this technology, the design must adhere to IEC/UL 60950-1 or IEC/UL 62368-1 and IEC 62368-3 as stipulated in the IEEE 1725-2021 standard [6] in order to meet the constraints of the IEEE Student Grant [7]. The XP Power VET30US120C2-JA meets the required industry standards while providing sufficient power output capabilities at low cost.

3.2.8.1.5. DC/DC Adapter

The CSS will also utilize a DC/DC adapter to provide a voltage source while installed in the vehicle when desired by the end user. This DC/DC adapter is only required to either

1. Pass input voltage and current from the DC charging port (cigarette lighter) of the vehicle to the CSS
2. Regulate the voltage supplied by the DC charging port (cigarette lighter) to a voltage within the input bounds of the CSS charge controller

Regardless of the voltage and current regulation requirements of the DC/DC adapter, the selected device also must comply with IEC/UL 60950-1 or IEC/UL 62368-1 and IEC 62368-3 as stipulated in the IEEE 1725-2021 standard [6] in order to meet the constraints of the IEEE Student Grant [7]. The MPD ZA1030 meets the required industry standards and provides full 12 volt DC power at up to 10 amps, and is more than sufficient for the DC/DC power needs at an acceptable price.

3.2.8.2. Component Selection

3.2.8.2.1. Battery

The decision to employ Li-Po battery technology presents a wide range of battery pack choices. Although Li-Po is much safer than Li-Ion, it is still necessary to prevent overcharging and under discharging, as well as monitoring the internal temperature of the battery pack to prevent damage. Most, if not all, available Li-Po battery packs incorporate a PCB containing integrated circuits to provide this over/under charge protection. Much less common, but absolutely necessary for the high temperature environment the CSS will operate in, is the inclusion of a temperature monitoring thermistor or thermocouple. To supply the minimum 5 volts required by the SBC, two 3.6 volt Li-Po packs will be utilized in a series configuration to provide 7.4 volts at full charge. This requires that 1) each series battery pack supply the full 10,000 mAh charge capacity or 2) multiple low-capacity packs will be connected in parallel to achieve 10,000 mAh, and these parallel packs would then be connected in series to achieve 7.4 volts at 10,000 mAh. Individual pack dimensions are minimally dictated by the capacity within the range of 100 mAh to approximately 10,000 mAh, where the height of a 100 mAh pack is 22 x 14 x 5.5 mm for an effective energy density of

$$100 \text{ mAh} / (22\text{mm} * 14\text{mm} * 5.5\text{mm}) = 0.059 \text{ mAh/mm}^3$$

and the 10,000 mAh pack is 100mm x 60mm x 12mm for an effective energy density of

$$10000\text{mAh}/(100\text{mm} * 60\text{mm} * 12\text{mm}) = 0.138 \text{ mAh/mm}^3$$

This shows that despite identical manufacturers (and therefore battery chemistry specifics), there is significant volume overhead per battery pack (due to packaging, insulation, PCB, etc) that does not scale with the battery capacity. Given this overhead, it is far more volumetrically efficient to utilize as few battery packs as possible to implement the 7.4 V 10,000 mAh specified for the CSS. In addition to volumetric concerns are the cost per battery pack: if low capacity, feature-matching battery packs from the same manufacturer are inexpensive enough, these benefits would offset the negative effects of the resulting volume increase. Utilizing a similar approach to the volume to capacity comparison, three battery packs are investigated: 1000 mAh, 4000 mAh, and 10000 mAh. To achieve the required capacity, 10 1000 mAh packs would be required at the cost of \$2.80 per pack, 3 4000 mAh packs would be required at \$5.19 per pack, whereas a single 10000 mAh pack can be purchased for \$7.09. The total cost, therefore, is \$28 for 10 1000 mAh packs, \$15.57 for 3 4000 mAh packs, and \$7.09 for 1 10,000 mAh pack. Investigating available battery pack costs yields similar results to those of capacity: it is far more efficient to purchase a single battery pack at the specified capacity than to purchase lower capacity battery packs and combine them in parallel. Table 3.13 below shows the battery pack configuration comparison with the best option highlighted in green.

Configuration	Model	Integrated Thermal Sensor	Charge Protection	Total Volume (mm^3)	Total Cost
2 + (10 1000 mAh)	DEAH 102050	NTC @ 25°C	Over/Under/Transient	200800	\$ 56.00
2 + (3 4000 mAh)	DEAH 606090	NTC @ 25°C	Over/Under/Transient	194400	\$ 31.14
2 + (1 * 10000mAh)	MakerFocus 9065115	NTC @ 25°C	Over/Under/Transient	144000	\$ 14.18

Table 3.13: Battery Pack Configuration Comparison

3.2.8.2.2. Voltage Regulator

Switched mode voltage regulation technology was chosen for both the 5 Volt rail and the 3.3 Volt rail. The regulation circuit topology is chosen such that source voltage from the battery or power adapter is provided to the 5 Volt rail, and the 3.3 Volt rail is supplied by the 5 Volt rail. This increases overall system efficiency by decreasing the input/output voltage delta of the 3.3 Volt regulator (5/3.3 vs 12/3.3). Further increasing power, cost, and area efficiency is the battery configuration: the voltage was chosen such that even the minimum charge would provide higher voltage than required by the SBC and other components, therefore buck regulators can be utilized in place of costlier buck/boost regulators.

The 3.3 Volt regulator is first under consideration, since the 5 Volt regulator must account for the efficiency of the 3.3 Volt regulator. A survey of voltages and currents consumed by the SBC, system controller, camera, sensors, and other peripherals results in 0.533 Amps maximum consumption at 3.3 Volts. To avoid possible brownout and to allow for possible expansion of the 3.3 V rail, the 3.3 V regulator should be configured to provide 0.750 Amps. Texas Instruments' WEBENCH Power Designer was utilized to obtain an array of suitable regulator circuits, which were then analyzed with efficiency as the primary concern, followed by total Bill of Materials (BoM) cost (not including the regulator IC) and the regulator IC cost. Initially, the top ten most efficient regulator circuits were analyzed, and a component choice was made.

Unfortunately, the semiconductor shortage massively affected this investigation as more than half of the regulators under consideration were not in stock at any reputable vendor and were back ordered well into 2022 at best. Given this unfortunate reality, component availability became the primary concern, followed by efficiency, BoM cost, and IC cost. There was a clear delineation between efficiency and BoM cost at 96% to 98% efficiency, in that the additional 2% efficiency drove the BoM cost up 317% with respect to the 96% efficient designs. Although efficiency is important to the design, such a high cost for such a small gain in efficiency is unacceptable, so 96% efficiency in the target circuits became the maximum. Among the 96% efficient designs was a wide variety of circuit area and cost, but the TLV62568 regulator provided a comparable efficiency (96.26% vs 96.5% of costlier topologies) with an extremely cost and area efficient external circuit. The BoM

cost was only 25% of the mean while the IC cost was 39% of the mean and the area was 29% of the mean, while only sacrificing 0.123% efficiency. Table 3.14 below depicts the comparisons made across all volt regulator ICs.

Model Number	Efficiency %	Area (mm ²)	IC Cost (\$)	BOM Cost (\$)	Total Cost (\$)
TLV62568	96.26	60	0.22	0.42	0.64
LM60440D	96.5	219	0.9	2.18	3.08
LM63615-Q1	96.4	265	0.95	2.46	3.41
TPS565208	96.4	712	0.54	2.85	3.39
TPS62823	96.4	61	0.54	1.28	1.82
TPS62827A	96.4	57	0.65	1.42	2.07
TLV62595	96.4	57	0.41	1.18	1.59
TPS56221	96.3	195	0.25	1.19	1.44

Table 3.14: 3.3 Volt Regulator IC Comparison

With the 3.3 Volt rail selected, the 5 Volt rail is selected to provide regulated voltage for both the 5 Volt loads as well as the 3.3 Volt rail. At 96.26% efficiency, the 3.3 Volt regulator demands 0.779 Amps to provide the rated 0.750 Amps output, and this load is combined with 5 Volt loads demanded by the SBC and peripherals. The total current output demanded of the 5 Volt regulator sums to 2.934 Amps, so the regulator shall be configured to 3.5 Amps to allow for brownout protection as well as possible expansion of 5 Volt components. Similar to the options available for 3.3 Volt regulators, there was a drastic cost increase when efficiency increased from 96% to 98%, so 96% efficiency was the target parameter. Even within this range, there were drastic differences in costs. The most cost-effective option in this range is the LM3150 at only 61% of the mean total cost, and the only drawback to this regulator is its dramatically larger area requirement at 209% of mean area. Depicted in the table below are more comparisons made in regards to the volt regulator IC.

Model Number	Efficiency %	Area (mm ²)	IC Cost (\$)	BOM Cost (\$)	Total Cost (\$)
TPS54J060	96	239	3.51	3.53	7.04
TPS54J061	96	239	3.51	3.53	7.04
TPS51397A	96.2	193	4.07	4.18	8.25
TPS566235	96.5	139	3.56	3.53	7.09
TPS546A24A	95.8	167	6.91	4.18	11.09
TPS546B24A	95.8	167	9.02	4.18	13.2
LM3150	96.1	489	1.57	3.53	5.1

Table 3.15: 5 Volt Regulator IC Comparison

3.2.8.2.3. Charge Controller

Switched mode technology was decided on due to its superior efficiency despite the excess surface area and complexity of the external circuit compared to the linear and pulse charge controllers. There is an extremely wide range of charge controllers available which implement switched mode charging, so the component choice will be dependent on the additional features offered by the controller. Foremost among these features is low quiescent power consumption since the CSS will operate primarily off of the charger and on battery power, so the charge controller must consume negligible power to optimize battery life. Additional features under consideration are intelligent source switching, load/charge demand control, state of charge monitoring, thermal protection control, and system controller communication. The operation and benefits of these features are presented, then component selection is made based on the completeness of the feature set on comparable charge controllers.

Intelligent source switching is the operation whereby the charge controller monitors the connection status of the AC/DC or DC/DC power adapter and provides source voltage to the system dependent on this. While the power adapter is not connected, the charge controller provides power to the system from the battery packs, but when the power adapter is connected the controller checks for any raised flags to indicate that charging should not proceed, such as battery overvoltage, temperature faults, or short circuit detection. If no flags are raised, the controller will switch source voltage to the system from battery power to the external power adapter and begin charging the battery as well.

Load/Charge demand control is a feature available on some charge controllers whereby the power adapter supply capability is programmed into the charge controller, and if the total load demanded by the system combined with the power being supplied through the charge controller to the battery approaches the power supply limit, the charge controller throttles the power being supplied to the battery to ensure that system operation is not impacted. Since the CSS is specified to work at complete efficacy while charging, this feature is a necessity.

State of charge monitoring utilizes a current sensor in series with the voltage source to the battery to constantly monitor the current into and out of the battery. By utilizing a two-fold approach of

- 1) A look up table of predetermined voltage-charge relationships
- 2) Integrating the current into and out of the battery, the charge controller will determine the maximum charge of the battery pack, as well as monitor the total charge contained at any point in time

Additional functionality available in some charge controllers, is to monitor the battery for maximum discharge state. Once this is observed, and the battery has been recharged to its maximum state, the charge controller stores the current delivered so as to reevaluate the maximum charge capacity of the battery pack, since this will change over time even though Li-Po chemistry is not subject to memory conditions the way older battery chemistries were.

Thermal protection is provided by the charge controller on multiple fronts: the charge controller accepts as an input the thermistor or thermocouple temperature sense from the battery as well as thermistor(s) internal to the controller IC and external ambient temperature thermistor. By monitoring these inputs, the controller modulates the charge rate to prevent damage to the battery, and in extreme conditions will raise a flag to disable charging entirely if permanent damage or catastrophic failure is possible.

Lastly, the desired charge controller will directly communicate with the system controller to provide information on the state of charge, indicate that battery charging is occurring, and indicate that any alarm parameters are set.

The charge parameters of the controller are determined by the system power draw and the battery specifications. Li-Po batteries require multiple charging modes to attain maximum charge state and battery life, so the charge controller must be specific to Li-Po chemistry or have a Li-Po specific mode selectable. The batteries selected are configured to provide 7.4 Volts at a maximum charge current of 1.6 Amps, so the charge controller must supply this current plus an additional 0.5 Amps for a buffer against transient brownout conditions. Shown below is a table comparing all charge controller ICs considered.

Model Number	Cost	Quiescent Current	Source voltage Switching	State of Charge	Thermal Protection	Outputs to System Controller	Additional Features
TI bq24703	\$5.71	<20 uA	Adapter/Battery with adaptive charge rate for system power demands	Differential OpAmp current sense and V/C LuT	External Sense Input and Internal Thermistor	SoC, power source, depleted battery, short circuit	Capacity relearn, battery conditioning, zero volt operation
MAXIM MAX1772	\$7.70	2.7 mA	Adapter/Battery with adaptive charge rate for system power demands	LuT, not communicated to system controller	None, System controller must provide control logic	Current from power source, charge current, AC adapter present	
Linear Technologies LTC4100	\$12.22	3 mA	None, system controller must control	Current sense resistor	Internal thermistor	All system parameters	SMBus communication

Table 3.16: Charge Controller IC Comparison

3.2.9. Enclosure Survivability

Since the device is intended to provide evidence in the case of a traffic incident, it shall be rugged enough so that the local storage medium is able to survive the incident and remain capable of supplying the recorded data. A suitable enclosure for the device would be able to provide for simple shock resistant internal mounting points for the device, while the enclosure itself would be secured to the windshield or dashboard of the vehicle with a suction cup or other non-invasive mounting device and able to withstand the forces present in a traffic incident.

Due to the prices of components currently and the cost of crash testing working components, a dummy prototype will also have to be built for the purposes of testing the survivability of the enclosure and PCB to see whether a local storage device would be able to survive in the event of a crash. Further research is required to determine the best way to simulate a crash on the enclosure, however a drop test is suspected to be sufficient enough to determine the survivability of the CSS by replicating the force sustained in an incident, the value of which would be chosen using traffic incident data from the National Highway Traffic Safety Administration (NHTSA) [8] or other safety agency. Simple dynamics would permit the theoretical calculation of the parameters for the drop test, and the experimental forces could be measured by the accelerometer recording the data over probe cables long enough to traverse the drop height. More expensive components, such as the processor and single board computer, can be swapped out for dummy replacements in order to stay within the budget.

One team member has experience using Autodesk Fusion 360 to create 3D designs as well as home access to an Anet A8 3D printer. Though not the best printer, it should work for the purposes of design and 3D printing the CSS enclosure. The Anet A8's maximum print size is 220mm x 220mm x 240mm or roughly 8.6in x 8.6in x 9.4in. This is plenty of room if the final product adheres to the 5" x 4" x 4" final project size requirement. Though PLA is the most common 3D printing material, it is incredibly brittle. Previous experience has shown that printing with ABS filament and smoothing/hardening the plastic through an acetone vapor polishing process yields parts that withstand prolonged outdoor exposure and automotive applications.

3.3. Software

The software flow will comprise three different components: the single board computer, mobile application, and the mobile application's interface. The single board computer software will utilize computer vision to obtain data through the camera attached to the single board computer. Data collected by the single board computer will then be transmitted to the mobile application's cloud database. The data sent to the database will then be displayed on the mobile application's user interface to provide users the ease and flexibility of accessing their CSS data. This is the general process of how data will move from application to application. To achieve this flow, several computer vision and web technologies will be investigated.

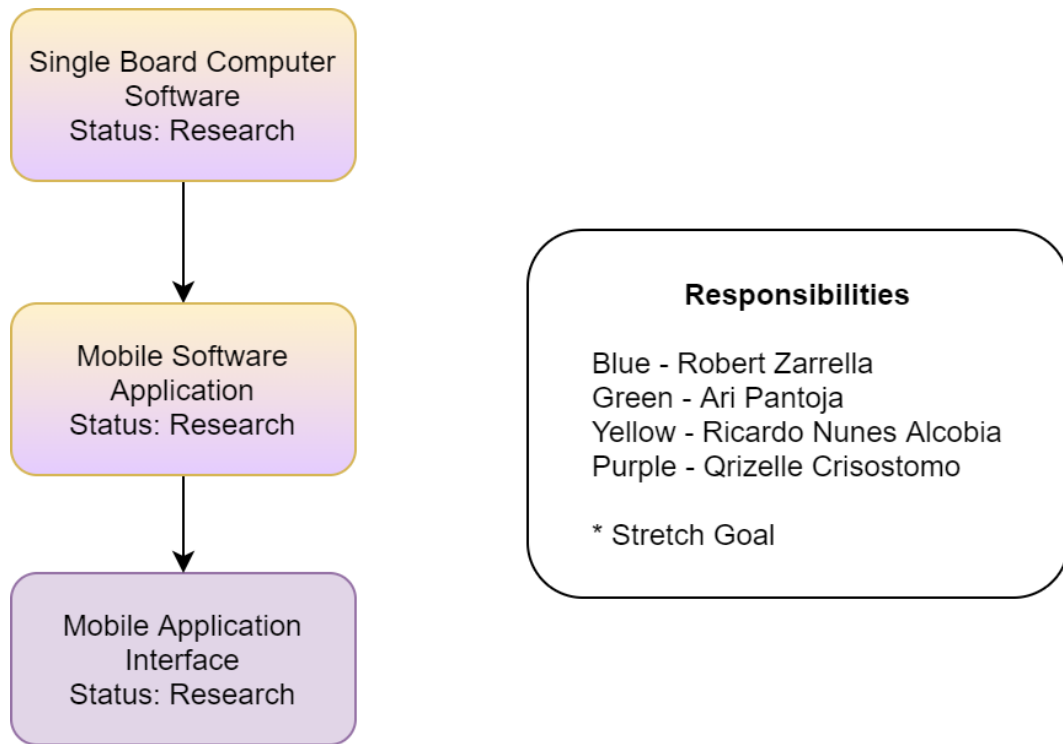


Figure 3.4: Software Block Diagram

3.3.1. Computer Vision and Machine Learning

Through the use of computer vision, the capabilities of the scanner can also be extended to include logging a vehicle's color, and model, in addition to the license plate information. This will make it easier to distinguish data later when it is being reviewed. In this manner, the device will only be storing essential data about a vehicle rather than continuous video streams, thereby significantly reducing the amount of storage space taken up and greatly extending the amount of relevant data that can be stored on a local storage device. This will decrease the number of manual backups required in addition to freeing up the need for cloud storage services. Further saving on costs post-purchase.

Computer vision software will be running on a single board computer. There will be a camera peripheral connected to the board and it will have a quality of at least 1080p and greater than 20 fps to read vehicle attributes. The camera attached to the board will take images that will then be processed by the software. This will essentially convert the images of a license plate into pure text. Text extracted from a license plate image will then be stored in a local storage device located on the single board computer. Similar processes will apply when trying to record other vehicle attributes tied to a specific license plate. Furthermore, data will be collected from the peripherals connected to the project controller. Peripherals include the GPS and accelerometer. Not only will the software log information regarding vehicles within the field-of-view, but it will also document the current location and the speed of the vehicle the device is in.

Additionally, the camera will have the ability to read license plates up to 60 feet away and have a minimum field of view (FOV) of 90 degrees. Two options to achieve the 60 feet goal are digital zoom and physical zoom lens for the camera. The former would yield poorer image quality results, but would be cheaper in terms of components. Due to part scarcity, it may be impractical to purchase a zoom camera of sufficient quality. A compromise using both digital zoom and physical zoom may be required to achieve the objective of range while saving on costs. Furthermore, the FOV of the camera would have to be taken into account. A low FOV would require a pan and tilt mechanism so that the device is able to sufficiently scan all vehicles within view. This would not necessarily be required for a higher FOV camera, however it would need to have a high enough resolution to clearly obtain the license plate information. Otherwise, a zoom feature would need to be added, which would once again add to the cost. A technical and price comparison of different cameras would reveal which would bring the best balance in order to achieve the objective.

3.3.2. Investigation of Computer Vision Solutions

There exists many computer vision solutions, offered by both the industry and the open source community. The top contenders include, but are not limited to, Microsoft Computer Vision API, Amazon Rekognition, Google Cloud Vision API, OpenCV, Yolov5, SimpleCV, and Scikit-image. These solutions were singled out for being the top in terms of performance, costs, and support. The three industry solutions offer complete solutions for computer vision while the open source solutions will need to be developed in order to achieve the same level of services as the industry cloud based solutions currently available.

It is also important to take into account the resources and support offered by the three cloud based solutions. Despite the costs associated with the services, each company has taken the time to provide thorough documentation of how to access and use their API. In addition, they offer various forms of support for assistance, such as phone and chat support. Which could potentially speed up the time in which it takes to get the application ready.

Furthermore, analysis of cloud-based services revealed that they would not render the design and construction process of CSS to be pointless. It has already been determined that a GPU would be needed with a CPU to run the intensive tasks required for computer vision. The cloud provided would simply take the role of the GPU. The device would still need to take pictures or video, transmit data to the cloud via a wireless communication method, and store the data either locally or in the cloud.

3.3.2.1. Microsoft Cognitive Services Computer Vision API

Microsoft Computer Vision API is an easy to use computer vision solution that is able to achieve the license plate recognition capabilities required. This is possible through the intuitive API calls provided by Microsoft. Compared to its industry competitors, Microsoft also has some of the fastest API calls, beating Google's Cloud Vision API by a narrow margin.

The downside comes in the form of pricing. Even though Microsoft generously offers 5,000 transactions for free each month, with an initial \$200 upon sign up, OCR transactions costs \$1 per thousand transactions. Therefore, initially, the application will be able to perform roughly 200,000 transactions. After that, it will have a significant drop off to 5,000 transactions per month. This is more than ample for testing purposes and very small scale applications, however at scale this would incur a lot of charges.

Furthermore, as a cloud based application, the device will require a constant wireless connection, which would also require a wireless service plan. This would alleviate the need for an onboard GPU powerful enough to run the computer vision algorithms necessary. However, the costs over the lifespan of the device would quickly add up between the API transaction charges and the wireless service plan.

3.3.2.2. Amazon Rekognition

Amazon Rekognition is another cloud based computer vision solution offering access to Amazon's powerful servers to perform computer vision analysis of images or videos for a range of purposes. It has a similar pricing plan as Microsoft for the image recognition services, with 5,000 free monthly API calls. However, it does not appear to offer any initial signup bonuses and charges for any storage related fees as well. As a cloud based application, it also suffers from the same downsides as Microsoft's Computer Vision API. That is, the lifetime costs will continue to incur charges.

3.3.2.3. Google Cloud Vision API

Out of the three industry computer vision services, Google Cloud Vision API is one of the most accurate. Although not as fast as Microsoft's Computer Vision API, it has the most consistent speeds. Its pricing breakdown runs differently from the previous two as it only provides 1,000 API calls per month for free. However, rather than a collective pool of calls, it is 1,000 free API calls per month for each type of call. This would be useful for a wide range of services, however, for the CSS, only one or two types of API types will be needed. Resulting in Google's solution providing the least amount of calls. As a cloud based solution it suffers the same cost issues as the previous cloud based solutions, the recurring costs.

3.3.2.4. OpenCV and YOLOv5

OpenCV is one of the largest open source computer vision libraries currently available for free. It is well documented, has a forum for user's to discuss and troubleshoot, and has a plethora of online videos on how to use the service. It can perform objection detection and recognition on both images and videos similar to the cloud based services.

Rather than being a service, it is an open source library available to all. The advantage of this is that it is free and well documented. However, the downside is that it has to be developed for the specific application. OpenCV also offers pre-trained networks to eliminate the need for user's to dedicate time to train their own networks. These pre-trained networks have utilized top deep learning algorithms such as Convolutional Neural

Networks (CNNs). If these pre-trained networks do not fit the needs of the application, there is also the ability to train your own network through the use of OpenCV's deep neural network (DNN) module. Furthermore, OpenCV can integrate with other deep learning frameworks. Normally, it would be a feat for a small team to gather the necessary data to train a deep learning network, however, there are open communities such as kaggle that provide free datasets.

Yolov5 is a powerful deep learning object detection tool. As previously mentioned, it can be used in conjunction with OpenCV to enhance object detection. OpenCV uses the Efficient and Accurate Scene Text (EAST) algorithm. Which differs from Yolov5 in the way it detects text in a video or image. Both algorithms are highly accurate, however it appears that Yolov5 is faster and relatively more accurate over the EAST algorithm. No concrete data or studies were found to support this, besides a few home project videos showcasing that Yolov5 scores more hits over OpenCV alone. Further testing will need to be done to verify this. However, the fact still remains that Yolov5 is a very powerful tool that is oftentimes used in conjunction with OpenCV to perform object detection and recognition purposes.

The popularity of both OpenCV and Yolov5 provides a lot of flexibility. Both of them provide extensive documentation and have active communities surrounding them. They are not only widely used in everyday projects, but are also used by the tech industry. This is an important consideration since even though it may not be as simple to use as the cloud based solutions available, it does provide a lot more flexibility and options for development.

3.3.2.5. SimpleCV

SimpleCV is also an open source library for computer vision applications. As the name implies, it is simple and straightforward to use. It uses Python rather than Java or C++ like OpenCV. So it is friendly to new programmers. The time to develop and get a working product is significantly faster through using SimpleCV, simply by design of it. Although SimpleCV uses OpenCV, it is not as comprehensive nor does it provide all the features that OpenCV does. SimpleCV does provide documentation as to how to get set up and use it, however the support forums are not as active as OpenCV. So it would be more difficult to get help or try new things through SimpleCV. Regardless, it could prove to be beneficial to get a prototype up and running quickly to test components before switching over to a combination of OpenCV and Yolov5 and optimizing it.

3.3.2.6. Scikit-image

Scikit-image is another open source solution that is designed to be simple to use and utilize Python to lower the barrier of entry into computer vision. It does feature more advanced capabilities over SimpleCV, however, it does not have all the features that OpenCV has. One of the key features it does not have is real-time analysis of a video feed. Although, it is possible to instead feed the algorithm images only, it does limit the scope of

development. The use of Python does lower the bar of entry, however, its documentation is not as extensive and does not cover as many topics as OpenCV.

3.3.3. Comparison of Computer Vision Software and Libraries on the market

Figure 3.5 below depicts the costs of each available computer vision solution based on the prices shown on their respective websites. The in house solution represents a solution developed using one of the open source tools available. It is important to note that the in house solution does not count the cost of man hours required for it to be developed. It also does not account for the cost of the device and its components, which would be more expensive due to the need for a GPU powerful enough to run it.

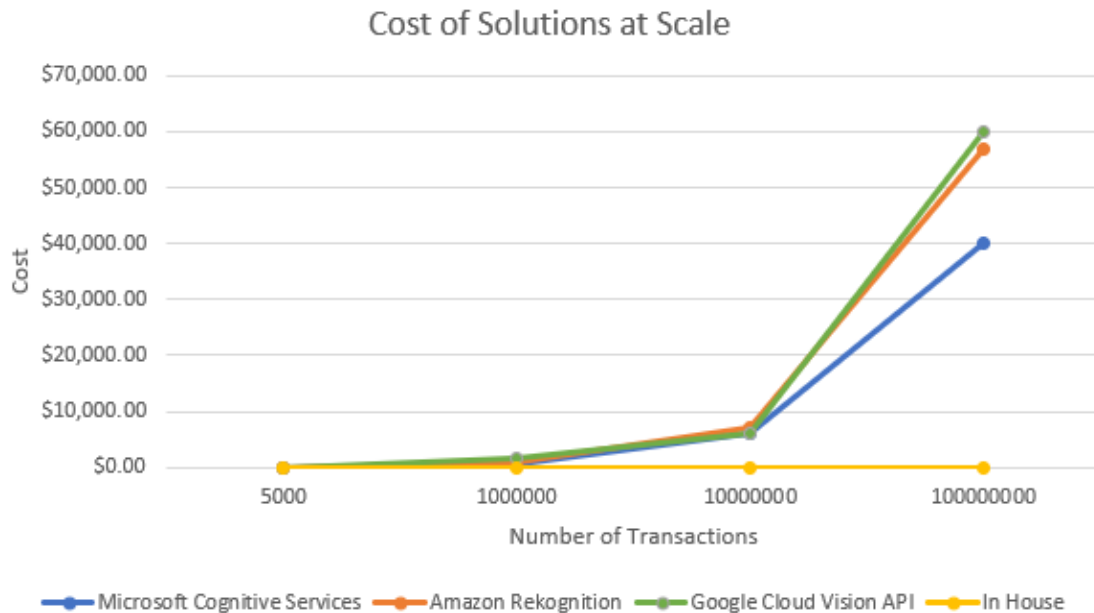


Figure 3.5: Costs of Computer Vision Solutions at Scale

Despite this, the cost of the device and the initial cost of development, would still be insignificant compared to the recurring transaction costs from using one the cloud based solutions. Even when considering substituting cheaper components in place of the relatively more powerful system controller and GPU required for the in house solution. The initial savings would be dwarfed by the costs at scale. It made the most sense to go with a cloudless solution that would save on costs later down the road. Therefore, for the purposes of this project, an in house solution will be pursued.

Furthermore, examining Table 3.17 below reveals that the open source libraries align closer with the objectives of this project, to keep costs low and implement a learnable solution. No team member currently has intimate knowledge regarding development of a computer vision application, therefore, one of the highest factors was picking a solution in which

there were ample amounts of resources to learn from. Although the cloud based solutions had this, the cost of them resulted in them having to be ruled out.

	Well Documented	Active Community	Low Cost	Easy to Integrate	Open-Source	Flexible
Microsoft Cognitive Services	✓	✓				
Amazon Rekognition	✓	✓				
Google Cloud Vision API	✓	✓				
OpenCV + Yolov5	✓	✓	✓	✓	✓	✓
SimpleCV	✓		✓	✓	✓	
scikit-image	✓		✓	✓	✓	

Table 3.17: Comparison of Computer Vision Solutions and Desired Characteristics

Out of all the open-sourced solutions, only OpenCV with Yolov5 met all the requirements. It provides a large and active community of developers in the event that questions need answers and it provides flexibility through the numerous functionality and customizability available simply by nature of the repository. SimpleCV and scikit-image had their advantages in terms of low bar of entry. However, since they were derived off of OpenCV, it simply made sense to go straight to OpenCV so that it was easier to customize down the road. Additionally, it provides access to the largest community of computer vision developers and resources already publicly available. Therefore, we'll be going with OpenCV in conjunction with Yolov5 for the final computer vision implementation.

Examination of the different computer vision solutions available led to the conclusion that OpenCV and Yolov5 would be the best solution for powering the computer vision application on the CSS. The SBC will act as an independent module from the rest of the device. It will be in charge of parsing the input feed from the camera, extracting the license plate information, and sending that data to the system controller to store.

3.3.4. Mobile Component

For user convenience, data stored in local storage will also be accessible via mobile device. A Bluetooth module linked to the project controller will provide users the ability to connect their mobile device to the license plate scanner. When the license plate scanner detects that the vehicle is not in motion, the single board processor will begin to relay the stored data to the project controller. The range of the Bluetooth module does not need to be very far, since the device will automatically power down after it has remained idle for a while. So it is assumed that the user would be in the vehicle with the device. The data would then get sent to a mobile application on the user's mobile device via Bluetooth. Data obtained by the mobile device then gets stored locally until it can be transmitted to the database. The mobile application will be developed to allow for information to be extracted from the database and displayed on the graphical user interface of the application. It will also have the responsibility of syncing with the cloud database.

3.3.5. Investigation of Mobile Application Development Strategies

As previously mentioned, the mobile component will be a lightweight application designed to give user's access to their data and update their database on the go. It is also important to note that for the mobile phone market, the two dominant operating systems are iOS and Android. For this purpose, there are two main avenues available to pursue, development of a native application or a progressive web application.

3.3.5.1. Native Applications

Native Applications have been the main method of mobile application development for many years. It involves the usage of SDKs and development tools provided by Apple and Google to take full advantage of the device. This allows for enhanced performance due to being able to optimize the application for the specific platform or device. This results in faster and more stable applications. Additionally, the UI and user experience blends naturally with the rest of the platform. The disadvantage is that, to reach the largest target audience possible, two applications must be developed and maintained for the two separate platforms. This would result in longer development time unless the decision was made to exclude a platform.

There are cross-platform solutions for native application development such as React Native. This allows for the development of one code base that can be deployed to both platforms. Being able to reuse code would significantly speed up development time. However, even then, the user experience would not be the same across both platforms. So specific components may have to be developed to target a platform in order to achieve the desired experience.

3.3.5.2. Progressive Web Applications

Progressive Web Applications (PWA) are a relatively newer approach to developing a mobile app. The idea of a PWA is that there is one code base for the web application and the mobile application. When the application is accessed on either platform, it loads the appropriate experience. This achieves the task of creating a truly cross-platform application. Additionally, the mobile application has the option to bypass the need to be on a store front such as the IOS App Store or the Android Play Store, making deployment and updating a faster process. Furthermore, PWA's benefit from faster loading times over a user going to a browser due to the ability to pre-cache content.

The appeal of having to only develop and maintain one code base, meaning that PWA's have a faster time to market over native applications, has led to the rapidly growing popularity of PWAs. For large companies, this means not requiring two teams for mobile app development. This popularity has resulted in a plethora of resources on how to quickly get started and develop a PWA.

Analysis of the development strategies available reveals that PWAs are the best route for this project. This is due to the fact that the mobile application is designed to be a lightweight application, not requiring a lot of work to design or maintain. Therefore, it makes little sense to develop two versions of it. Instead, one version that can be deployed simultaneously to all platforms would be greatly beneficial. Additionally, it will not deter attention from the main goal of the project, developing the CSS device.

3.3.6. Investigation of Mobile Application Development Technologies

There are a great deal many technologies available for mobile application development. Technology stacks bundle highly popular and compatible technologies together into one accessible for development purposes. Two of these highly popular technology stacks to be considered are the MERN and FERN stacks. The names are an acronym for the technologies that make up these stacks. MERN stands for MongoDB for the database component, Express as the back end framework, React for the front end development, and Node.js also for the back end. FERN includes the same technologies with the key difference being Firebase for the database instead of MongoDB. Firebase has a range of different APIs aimed at getting apps up and running as soon as possible, which makes it desirable for small scale applications.

As previously mentioned there are a plethora of technologies available to facilitate development and numerous stacks that bundle them into an accessible manner. There are similar technology stacks such as MEAN and FEAN, which are similar to MERN and FERN, with the key difference being the inclusion of Angular for the front end framework rather than React. There is no wrong answer with picking any of these. However, this team, having previously had personal experience with React, it was determined to use a stack that includes React to speed up the mobile development process.

3.3.6.1. MERN Stack

The key aspect of the MERN stack over the FERN stack is the use of MongoDB for its database component. MongoDB is an increasingly popular NoSQL database that is continually growing. It now provides MongoDB Atlas, which is their all in one service for database management. Recently, MongoDB also offers MongoDB Realm, a service designed to accelerate the time it takes to set up a database for a mobile app. This doesn't change the fact that MongoDB was built to fit many different needs. Although having just recently shifted focus to expanding its mobile development aspect, MongoDB still provides a fast and robust database service with a large amount of features available to be deployed.

3.3.6.2. *FERN Stack*

Firestore was built from the ground up with mobile support in mind. It has a well established community and a wealth of resources available. The key attraction to Firestore is the speed in which it takes to get set up and deployed. Especially for mobile app development. However, it does not offer as many customization options as MongoDB, therefore it is harder to optimize. Additionally, MongoDB has better performance due to the nature of the service being to cater to all services and platforms, not just a mobile focus.

Once again, there is no wrong choice when picking between using Firestore or MongoDB for the database of the technology stack. Both are strong choices. Since this project focuses mainly on the development of the device rather than the mobile component intended to be lightweight and being a bonus quality of life feature for users, it makes greater sense to pick Firestore. Although it is relatively worse in terms of performance and configurability, the speed and ease of set up make it the preferable option. Especially since it was designed specifically aimed at small-scale mobile development. Which fits the needs of the project perfectly.

3.4. Overall Structure

The culmination of both hardware and software components will produce the functionality for CSS. All hardware components will be integrated within the unit. The software will provide the necessary logic for the flow of data.

The camera attached to the CSS unit will gather input for the computer algorithm to extract. CSS will then collect all data from input peripherals and store them. This data then gets emitted to the mobile application via Bluetooth for the user to access. This is the structure we are going to attempt to model. Figure 3.6 below shows a high-level overview of how CSS will work.

The previously mentioned structure can be seen reflected in Figure 3.6. Which provides a high-level overview of the different interactions between the components of the project. As can be seen, the primary source of input is the camera feed which will provide the frames for the computer vision software to capture and analyze to extract data from. CSS represents the microcontroller, the PCB, the single board computer, and all attached peripherals. All these components will work with one another to ensure that the device is able to obtain and store license plate information so that the user is able to retrieve it. Either by extracting the local storage device or by using the mobile application.

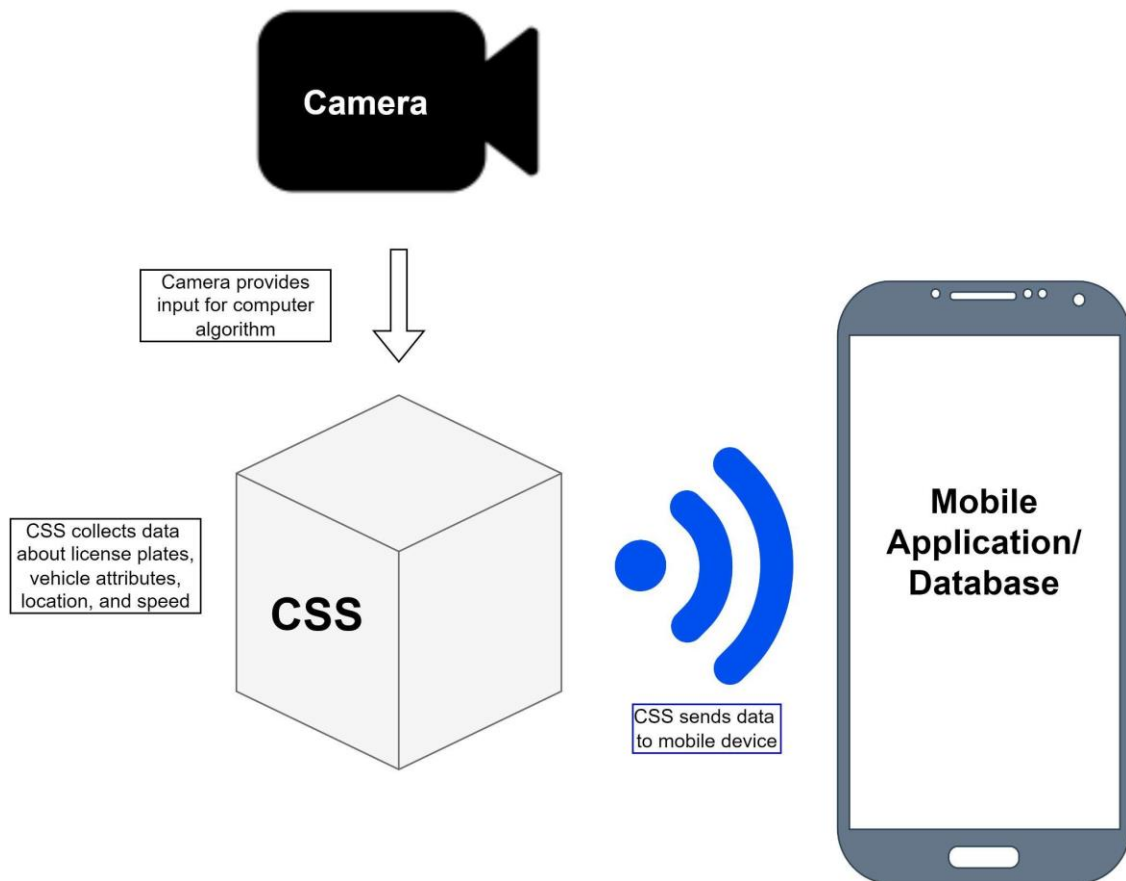


Figure 3.6: High-level Overview of CSS Functionality

The mobile application will be able to connect with the CSS device through Bluetooth. This will allow it to extract data from the device. Not only does this add convenience to the user so that they are able to access and view the data anytime they want, but it will also allow them to sync it to the cloud.

By providing a database in which the user is able to sync with the cloud provides data redundancy and protection in the event that the CSS is lost. This would be achieved by periodically pinging the CSS device; however, this would drain the battery. Therefore, the user will have the option of whether to automate this process or to manually sync it themselves. Thereby allowing them to cater it to their personal use case.

4. Related Standards and Design Constraints

The CSS project will abide by a set of standards with the impact of hardware and software design constraints. Standards declared will contain the criteria that will be used as a definition and be present throughout development. This will ensure that the construction and evaluations of CSS are done in a consistent manner. As well as preventing feature creep and deviations from the core goals of the CSS device. Thereby keeping the project focused and on pace.

Project constraints will be the limiting factors that may impact the project and its success. Bringing awareness to all possible constraints will provide a better understanding of how each constraint may relate to one another and how the team can be prepared to handle their impact.

4.1. Project Standards

CSS must comply with multiple project standards and regulations. The project must be able to recognize standard U.S. license plates and possibly expand to European license plates. It must also satisfy certain power and energy standards. Shown below is a table containing the standards to be implemented in this project.

#	Project Standards
4.1.1	IEEE Standard 1725-2021: Battery Storage, Charging, Power Supply Sub-standards
4.1.2	U.S. License Plate Standards
4.1.3	European License Plate Standards *
4.1.4	PCB Standards

*Table 4.1: List of Project Standards / * Stretch Goal*

4.1.1. Power Standards

The IEEE Standard 1725-2021 establishes “design approaches for reliable operation of mobile phones, power banks, and similar rechargeable battery-operated systems” specifically encompassing Lithium-Ion and all derivative Li-Ion battery technologies, including the Li-Po technology used in the CSS [6]. The standard provides a litany of design procedures and tests to guide the design and production of battery cells, battery packs, and power adapters, however these particulars are outside the scope of the CSS design project. The design team will ensure that the battery pack and power adapters selected are in compliance with IEEE 1725-2021 standards which will satisfy the respective sub clauses of the overall project compliance, as stated by IEEE in Appendix D.4.1.4 “It is a requirement in the normative validation portion of this standard that a

system component cannot be certified unless all component-defined product types have been certified.” Therefore, if the battery pack and power adapters have been IEEE 1725-2021 certified, then every component in the construction of the battery pack and power adapters are confirmed to be likewise certified and their uses validated in the design of the CSS with the view to IEEE 1725-2021 certification.

As directed by IEEE 1725-2021 and stated previously in Section 7.1 of this document, for an AC/DC or DC/DC adapter to conform to IEEE 1725-2021 certification, the adapter shall conform to the more widely adopted Underwriters Laboratories and International Electrotechnical Commission collaborative standards IEC/UL 60950-1 or IEC/UL 62368-1 and IEC 62368-3. Similarly, IEEE 1725-2021 states that a battery pack conforming to UN 38.3 and/or UL 1642 shall be certified as conforming to IEEE 1725-2021.

The main concern with design of the CSS as it pertains to the IEEE 1725-2021 certification is Clause 7: Host Device Considerations. The CSS is powered by the battery, and when connected to an applicable AC/DC or DC/DC adapter the CSS charges the battery, thus the CSS is designated as a host device by IEEE 1725-2021 in that it is “The device that is powered by a battery and/or charges the battery.” [6]. Clause 7 includes provisions for electrical and mechanical connections between the host and battery which, in the design case of the CSS, are already constrained to the use of an IEEE 1725-2021 certified battery. In designing the CSS to use this certified battery, subclauses 7.8.2.2, 7.8.2.3, 7.8.2.4, and 7.8.2.5 are satisfied since the Host Device interface is a mirror of the battery pack interface and thus pin mating, separation, polarity, etc. protocols are satisfied, else the battery and Host Device simply could not physically interconnect.

Clause 7 is primarily concerned with ensuring that the Host Device provides safe and reliable methods of charging and monitoring the battery pack. Loosened provisions are made for devices with embedded batteries, designated in 1725-2021 as “A battery that is not intended to be replaced or serviced by the end user/consumer.”, in that utilizing an embedded battery renders battery identification protocols of the standard unnecessary.

The remaining applicable protocols [6] of 1725-2021 Clause 7 specify constraints on the safe operation of the Host Device battery charging operation. Of primary note are subclauses 7.3.1, 7.3.7, and 7.3.8 encompassing safety, temperature validation, and voltage range validation. Subclause 7.3.1 Safety requires that no part of the Host Device shall disable or override any safety features built into the certified battery pack in order to maintain the integrity of the battery pack certification to prevent any unsafe conditions, regardless of whether or not the Host Device would be operated under those conditions. Subclause 7.3.7 requires that the Host Device monitor battery cell temperature before, during and after the charging process and if there are any temperature excursions out of temperature ranges specified for the battery pack in its certification, the Host Device shall disable charging until the temperature measurements are within parameters again.

Lastly, subclause 7.3.8 presents an array of protocols to ensure that the battery pack is operated by the Host Device within the voltage thresholds specified for the battery pack in its IEEE 1725-2021 certification. The charge controller of the Host Device shall prevent

the initiation of the charging algorithm if the voltage on the battery pack is either below the minimum, or depletion level, voltage. Likewise, the charge controller shall not initiate charging if the battery pack is at or above the specified battery charge regulation voltage. Provisions are made to allow for a safe, low current pre-charge profile to be applied to a battery below the specified depletion voltage in order to safely float the battery into the voltage range in which the full charging algorithm can be applied by the charge controller. Similarly, provisions are made to allow for the charge controller to apply a safe, low current trickle charge to the battery pack once it has reached the specified regulation voltage in order to keep the battery safely at its maximum charge level until the power adapter has been removed from the Host Device.

4.1.1.1. Power Standards Impact

The primary impact of designing the CSS to the IEEE 1725-2021 standard is that the design team, advisors, and end users are assured that the system is designed to perform within the strictest safety standards. Although the team could have devised effective safety constraints in the design of the CSS power system, safety is assured in every aspect of the power system from the wall power adapter to the voltage rails on the PCB. Although the design team could not reasonably perform certification tests for the AC/DC adapter, DC/DC adapter, battery, charge controller, and overall Host Device, the standard provides for interoperability between existing standards. This allows the design team to utilize existing production components that are widely available from reputable vendors without having to independently test and certify third party components.

This leaves the design team free to expend the time necessary to ensure that the charge controller, voltage regulation, and system controller integration conform to the IEEE 1725-2021 standard via independent testing. Although it is certainly possible to design a safe system without consulting or conforming to any existing standard, any time or cost benefits gained by such an approach would surely be lost in the devising and implementation of independent testing procedures. Similarly, since the standard provides a wide range of interoperable standards for commercially available subsystems (power adapter, battery, etc.), the constraints of conforming to the standard are easily satisfied. Considering both of these aspects, the adoption of such a standard introduces no truly negative impact for the design team and the design is guaranteed to be safe and effective. The table below shows the components, models, certifications, and standards in relation to IEEE standards.

Component	Model	Certification	IEEE 1725-2021 Compliance
Battery Cell	Integrated in Battery Pack	Implicit in Battery Pack Certification	Implicit in Battery Pack Compliance
Battery Pack	MakerFocus 1260100	UN Manual of Tests and Criteria, Section 38.3 and IEC/UL 62133	IEEE 1725-2021 Clause 4.3.1
AC/DC Adapter	XP Power VET30US120C2-JA	IEC/UL 60950-1 and UL 1310	IEEE 1725-2021 Clause 4.3.1
DC/DC Adapter	MPD ZA1030	IEC/UL 60950-1 and UL 2089	IEEE 1725-2021 Clause 4.3.1
Charge Controller Circuit	N/A	Self Test	IEEE 1725-2021 Clause 7

Table 4.2: Components, models, certifications, and standards based on IEEE

4.1.2. PCB Standards

Although the IEEE does not provide a single unified standard with regards to PCB design, it does offer a set of guidelines which represent the most design critical elements of PCB layout. These guidelines are intended to minimize unintentional electromagnetic compatibility (EMC) and electromagnetic interference (EMI). To note, EMI in general is the radiation of electromagnetic waves caused by inductive loops created by the conductor networks created in the PCB. EMC is EMI which specifically affects components and networks on the device it is generated on, and in which the interference caused is severe enough to disrupt the proper operation of the design circuit. As stated above, the conductor loops created in the circuit networks create current loops when the device is under operation. Any such loop induces a magnetic field which will affect currents in any conductor passing through this field. The field is dependent on the current passing through the loop as well as the physical dimensions of the loop. To mitigate these, the IEEE guidelines provide four major components of PCB design that should be adhered to.

- 1) Since the induced magnetic field is dependent on the size of the conducting loop that generates it, the designer should minimize the size of any current conducting loop. Primarily of note by the guidelines is the reminder that, although extremely small in magnitude, digital signals do have current flow. The small magnitude of current is magnified by the switching nature of the signal, but these parameters are bound by the operation requirements of the device. Thus, the only way to mitigate

the EMI from digital signals (without external shielding) is to minimize the area of any signal current loop.

- 2) Despite minimizing the area of signal current loops, it is impossible to eliminate them completely. This is exacerbated by the larger currents and/or high frequency switched signals of input/output connectors. To mitigate this, the IEEE PCB Guidelines recommend keeping circuitry away from I/O connectors. Given the common practice of an input flowing from I/O on one end of the board while flowing out from the I/O on the opposite end of the board, care must be taken to isolate components from this largest of conducting loops.
- 3) EMI is dependent on the change in current of a conducting loop, thus steep transitions in digital signals greatly increase the radiation of the board. By reducing the slope of the switching signals via longer rise and fall times, the harmonic radiation induced by the digital signals is mitigated.
- 4) Gapped signal return planes should be avoided since the return plane is critical in allowing conducting loops to be as compact as possible. If the return plane continuity is disrupted, signal loops may be forced to close through unintended pathways creating interference in circuits which are not intended to be interconnected.

The remainder of the guidelines present specific scenarios in which these four primary guidelines can be implemented, and shall be implemented in the CSS design where applicable.

4.1.2.1. Impact of PCB guidelines

The PCB Guidelines [9], while not a true standard, adhere to the spirit of a standard in that they are a set of rules provided through industry experience and collaboration which constitute a best practice approach to the topic they govern. The guidelines for PCB design do not require testing, but they are an overarching system to control design such that EMI and EMC are reduced to acceptable levels. This allows the CSS design team to leverage the design experience of the collective IEEE in designing the CSS by simply being mindful of component placing and routing when transitioning from schematic to board designing and layout.

Although the requirements for solid signal return planes may add to overall copper area and increased price, it is a small price in the overall cost of the design since many board houses have set price per board for wide ranges of board sizes. Additionally, the circuit isolation guidelines require added area to the PCB to prevent intrusion from one circuit into another. This may render the board area large enough that added cost is incurred by the board house, but the counter cost is that in doing so, the guideline provides the best possible opportunity for the first PCB design to be acceptable per EMI and EMC concerns. A higher board cost on one board run is significantly less costly than multiple iterative runs of less expensive boards. With these considerations in mind, the IEEE PCB guideline negative impact is negligible time cost, while the benefits ensure that the design is most

capable of achieving interference free operation. Shown below is a summary of PCB design standards.

#	PCB Design Standards
4.1.2.1.1	IEEE PCB EMC design guidelines
	1. Never route signals over split reference planes
	2. Keep current loops as small as possible
	3. Decoupling: use low-inductance capacitors and planes
	4. Use ground planes for shielding
	5. Route high-frequency signals adjusted to a plane
	6. Control rise and fall time
	7. Add ceramic capacitors close to each pin of a connector
	8. Fill top and bottom layers with GND and matelize edges
	9. Add stitching vias
	10. Connect circuit GND to chassis

Table 4.3: List of Project PCB Design Standards

4.2. Constraints and Operation

There will be limiting factors that will impact the functionality and design of CSS. By considering these constraints, we can determine how the project will operate. Project operation will describe portions of the implementation and how the system will be designed around the constraints mentioned.

4.2.1. Constraints Overview

As previously mentioned, the device will be portable. This is so that it can be placed on the windshield or dashboard in any vehicle with an enclosed space. With this feature, users are able to make adjustments based on their vehicle’s dashboard layout.

This objective also results in a size constraint as the device must be able to fit on a user's dashboard or close to their rear view mirror. The system being developed will not cater to “open” vehicles, such as motorcycles, scooters, etc. The unique size and configurations of these vehicles will necessitate the need for smaller sizes and more mounting methods for the device. The tradeoff of accommodating these vehicles and the increase in design challenge is currently not worth taking. For safety reasons, when the device is installed in

the vehicle, it should not be so large that it obstructs the driver's view or distracts them from driving.

Most of the constraints for the project arose from the environment in which the device would be subject to, a vehicle and for this project, a vehicle that is in the state of Florida. Therefore, the device must be portable and have a low profile so that it can be situated on the windshield of the vehicle or the dashboard. Additionally, a stationary vehicle in direct sunlight could reach temperatures up to ~150 degrees fahrenheit [10]. In addition to the extreme ambient temperature conditions, the processing power required for computer vision would add even greater excess heat that would compound the heat constraint. This means that the components of the vehicle must be able to withstand the high temperatures and continue to operate. This limits the types of batteries that can be used.

Super capacitors are oftentimes used in dash cams. This is to prevent memory loss in the event of a power failure and allow the device to safely shut down. Super capacitors would also be able to withstand the heat. However, it would not have the charge capacity required to power the device over the desired use case. Preliminary research reveals that lithium polymer batteries would be the best solution for a high temperature environment.

Cooling solutions such as heatsinks, and fans will also have to be considered and most likely tested to see if they make a significant impact on heat reduction. Heatsinks would increase the size, and fans would also increase power draw. Therefore the solutions mentioned would be limited by the size constraint, as there is not much room to work with behind the rear view mirror. If placed in a different location, then a slim profile will still have to be considered as to not interrupt the driver's field of view. Testing would reveal which solution would be the best.

Additionally, it is important to note that sitting in a car in either direct or indirect sunlight, the temperature of the device would rise considerably. It is possible that the device could get to the point where it is unable to power on in the first place. Or that it may reach the temperature maximum bounds rather quickly. It may just be necessary to consider the inclusion of a small fan to cool the device upon start-up to ensure it is able to function properly.

Regardless of the eventual choice of battery, the heat of the operating environment will necessitate rigorous monitoring of the thermal state of the battery especially during charging and high current discharge such as active image processing. This constraint might necessitate throttling based on the thermal condition of the battery, voltage regulators, or the single board computer, which will need to be implemented in such a way as to not interfere with the ability of the system to execute its functions in real time and therefore remain reliable. Once again the accelerometer can be used to determine prioritization. If a jolt is detected then the throttling may be temporarily overridden to prioritize scanning for a potential getaway vehicle.

For the purpose of demonstrating electronic design, we conclude that most of the peripheral sensors will be connected and controlled from a PCB of our own design containing a

project controller. This controller will appropriately route data from the sensors to the single board computer for processing through the machine learning algorithm. Data will also be sent from the single board computer to the project controller to write to the SD card or transmit over Bluetooth for example. Taking the size constraint into consideration, we challenge ourselves to accomplish power and peripheral controls on one shield that can connect directly and sturdily to the I/O pins of the single board computer. This would realize the components on the same board. Shown below are the set of constraints relevant to this project.

#	Constraints
4.2.1.1	Weight
4.2.1.2	Environment (Car interior and temperature)
4.2.1.3	Time
4.2.1.4	Cost of System Controller
4.2.1.5	IEEE Student Grant Specifications
4.2.1.6	Image Processing Capability of Controller/Processor
4.2.1.7	Compatibility of Components
4.2.1.8	Ethical Constraints
4.2.1.9	Social and Political Constraints
4.2.1.10	Economic Constraints

Table 4.4: List of Constraints

4.2.1.1. Ethical Constraints

When designing and developing a product that is able to scan and store the license plate data of all the vehicles it encounters, it is also important to discuss the ethical implications of such a device. For instance, the privacy concerns that may arise from other drivers on the road. Digital privacy is currently a very hot topic with many people concerned about how large companies and government entities are using the data they collect. As it currently stands, the Driver Privacy Protection Act, 18 U.S.C, Section 2721 [11], protects personal information and motor vehicle records in possession by a U.S state department of motor vehicles. This restricts who can have access to a person’s personal information and what information they can access. Exemptions include but are not limited to auto manufacturers for recalling vehicles or law enforcement agencies.

This is relevant, because it means even if a citizen is able to acquire a large database of license plate information, they cannot, without permissible reason, use that data to obtain

even more information regarding a person, from a state department. This ensures that the privacy of other drivers on the road is maintained and keeps in spirit of the intention of CSS, to assist drivers in obtaining license plate information in the event of a hit and run scenario and hand over the data to law enforcement to handle.

Therefore, a CSS device owner would not be able to abuse the information they have obtained. As there is very little, they can do by obtaining a person's license plate information. Besides the intended use of aiding law enforcement. However, the inclusion of a GPS module, a stretch goal previously mentioned, would allow a CSS device owner to log the location of license plate information from other drivers. This raises another interesting privacy concern. As it would allow a user to see the location where a license plate was detected and potentially map out where a specific driver has been.

A malicious user may use this information to commit nefarious deeds upon another driver. This raises the question of whether or not to give CSS users the ability to log the locations in which plates were scanned. The downside of this would be potentially enabling a malicious user, and the upside would be that it would further collaborate a CSS user's story in the event of an accident. In this scenario the harm outweighs the benefits it would have to drivers. A potential solution would be to hide and encrypt the location obtained only so that law enforcement could request access to the information. This then leads to whether or not a user would want law enforcement to access this information. Perhaps by allowing a user to disable this feature would overcome this obstacle. Yet again, this leads to another issue, this would include a redundant GPS module, increasing overall costs, when some users would not even take advantage of the feature.

This has not yet touched the topic of what if a malicious user were to manually note a victim's location or tamper with the device to gain access to secure data such as the location if a GPS module were to be included. This then begs the question of how far can one go to keep people safe while simultaneously providing a useful product for the end user? Additionally, securing a user's privacy adds to the design and development time of the device. As well as the costs.

Initially, the inclusion of a GPS module makes sense. Devices ranging from phones, tablets, cameras, and even desktop computers already log your location. Applications that are given permission can then access this information to use. Therefore, why not include this ability as well for the CSS?

The key difference is that these devices are often manufactured at scale. Possibly years of research and development has gone into securing and isolating sensitive data on such devices. Additionally, rather than individual users having access to this data, companies have access to this data and have privacy policies in place informing users as to how their data is being used. Perhaps the solution then, is to store this data in a centralized location that is only handed over to the proper authorities. This, however, would increase costs drastically as cloud storage would be required.

Upon reflection, the inclusion of a GPS module has much larger ethical implications than originally imagined. Although there are several possible solutions, all these solutions add a significant amount of time and costs to only one feature of the overall device. It would be best to leave this till last while it is determined whether the solutions are permissible or not. Additionally, the device alone, without the GPS module, does not have as large of a privacy concern. Therefore, this would not affect the development time scale or cost.

4.2.1.2. Social and Political Constraints

Usage of ALPRs are currently mainly by law enforcement and a few businesses. The release of a feasible device to the retail market may disrupt the political landscape. If it is even allowed to do that. Would the U.S government, either at the state or federal level, allow the usage of such a device.

The disruption of the political landscape would come from the fact that any individual would be able maintain their own database of license plate information. This alone has privacy implications for individual drivers. Now factor in the consideration about whether a CSS user or non-user would want a government entity accessing this database, further adding to the information they currently possess. The introduction of ALPRs into the arsenal of law enforcement was a heated topic in itself. The possibility that anybody would be able to possess would stir the political landscape.

Would the government even allow the introduction of CSS into the market? The answer varies from state to state. Only a few states even have rules and regulations regarding the usage of ALPRs. According to the research, “Automated License Plate Readers: State Statutes”, done by the National Conference of State Legislatures (NCSL), at least 16 states have laws regarding the usage of ALPR. Florida being one of them [12]. This limits the market for the CSS and also raises the question of how long that market would even exist before state intervention occurs.

As specified by Florida Statute Section 361.0777, information obtained by an ALPR, may only be disclosed in a very select few scenarios. Such information must be confidential. Therefore, it is assumed that the CSS would not be permitted to use such information. Only the exempted agencies as specified by the stature. Vagueness results from whether anyone is able to obtain this information as long as it is confidential or whether only law enforcement and permitted agencies are allowed. As mentioned before, it may be best to consult a lawyer regarding this, as no team members are well versed in the law.

The social constraint of the CSS is low. The slow adoption of ALPR across the nation by states has resulted in decreased backlash. Even universities across Florida, such as the University of Central Florida and the University of Florida, have already adopted the usage of license plate readers for enforcement [13]. It is unlikely that a large amount of resistance to the CSS would occur.

4.2.1.3. Economic and Time Constraints

It goes without saying that the global pandemic has had a significant impact on the economy of the world, including the shipping industry. Due to lock downs and the shutting down of ports, manufacturing times of parts have drastically increased as manufacturers are overwhelmed by orders and limited by availability of resources being shipped to them. By extension, the time constraint of an available part arriving within the estimated time has become unpredictable and thus unreliable

This has had a major impact on the design constraint of the CSS. Desired and optimal parts for the device have either become unavailable or delayed till an unknown date. This is not ideal as the time constraint of the project limits it to one semester of design and one semester of development and building. The effect is having to make design and part selection decisions significantly earlier in the process than ideal. This was in order to protect the project from the possibility of parts going out of stock and mitigate the potential damage from parts being delayed. This could potentially lead to making less informed decisions due to the urgency in obtaining parts. Not only that, but also increase in prices of parts due to ordering them in less-than-ideal situations.

To add onto this, there are supply chain issues as a result of the lockdowns at the beginning of the pandemic. Not only are components scarce, the components that are available have increased in price. As previously seen, this has led to the increase of prices of single board computers and microcontrollers among other parts. Some instances, the price isn't just reflected in the cost of the components, but also the cost of shipping. Which many smaller vendors charge. Once again, resulting in a less than ideal situation for sourcing parts to build the CSS.

This has resulted in the team pivoting to making design decisions prioritizing which components are available and readily able to ship in a reliable amount of time. Examples of components that are scarce include, but are not limited to, microcontrollers and system controllers as previously mentioned, but also affordable high-quality accelerometers. Many of which are no longer in stock or on back order. The ones that are affordable for the project are not as accurate or sensitive and have odd sizes, requiring design reconsiderations for the PCB.

To add onto this uncertainty there is also the delay in shipping time. As container ships are in high demand and some ports are still backed up, this has led to uncertainty regarding when components would arrive from the manufacturing plant. Most of which are located in China. To overcome this obstacle, the team has decided to go ahead and order multiple copies of certain parts in order to ensure there are plenty when they arrive in case any are defective or broken later down the road. This way if a situation does occur in which the parts can not be used, development is not completely halted while a replacement is ordered. This is essential, because as previously mentioned, there is the time constraint of having to have a complete project in a semester. Therefore, any delays could potentially prevent the delivery of a complete product and any issues that could result in this has to be mitigated as much as possible.

The economic constraint of the pandemic has resulted in increased costs and a restrictive time frame, since it would be best to order components sooner, rather than later. These constraints put a significant amount of pressure in the early stages of the design phase. Additionally, it results in increased costs due to having to purchase parts for redundancy. This leaves less room for experimentation without further increasing costs.

Besides the pandemic, the other time constraint is the length of the Senior Design course. There is only one semester to design and prototype and another semester to build and finalize the finished product. Combined with the time constraint of the pandemic, this means that time is of the utmost urgency. The project will be running on a tight schedule while at the same time plans have to be made to allow time for delays. Otherwise, unforeseen circumstances could result in missing deadlines and the delivery of a finished product, which would be a disaster.

The economic constraint of the pandemic and the price increases that have ensued, means a more careful analysis of the price of components must be done. Due to the increased price of components, it is more difficult to determine if the price is justified or not. Factors to aid in determining this would be cost of comparative competitors and availability of the product. As well as when the product is needed during the development process. If a product is inferior to a competitor, but it is currently in stock and the competitor currently has it on a six to eight month back order, then there is no other option than to go with the inferior product in order to stay within the time constraint. Besides that, not much can be done without taking a gamble on parts not currently in stock or on back order.

The best way to ensure that the team is on track to meet the deadlines previously sent is to keep in communication with each other. Not only with regards to how much each team member has done, but also with regards to the status of components a team member has ordered. Additionally, if an unforeseen circumstance were to arise, team members will communicate this with one another to keep each other in the loop. That way, everybody is aware as to where everyone is currently at and how much work is left to be done to reach a deadline.

Project management tools and techniques such as software development approaches such as agile or Kanban boards, can ensure the team is on track. Tools such as the website Trello can be used to set and track deadlines. There are also communication platforms such as discord and slack to keep the team up to date with one another and maintain a good communication precedence.

5. System Design

This section will discuss the system design details for the CSS unit. Design has been split between hardware and software. The hardware design consists of how breadboard testing will occur, schematics of hardware components, the MSP430FR6989 circuit, and the sensors. Software design includes the flow of software, construction of the computer vision software and mobile application, and communication protocols to be implemented.

Within the schematics portion, the power system architecture has been described in detail. The power system architecture will contain the charge controller primary block, charge controller output block, and the 5 volt and 3.3 volt rail circuits.

5.1. Hardware

In this section, the CSS hardware design configuration will be demonstrated. All designed schematics are given with detailed explanations of each component connection. Schematics are provided of the power system and the MSP430 to sensors circuit.

5.1.1. Schematics

Featured here are our final schematics for the CSS project. First presented are the power supply schematics followed by the sensor schematics. For the power supply schematics we first present the primary charge controller, then the power supply connections, and finally the 3.3V and 5V rails. In the sensor schematics presented, the MSP430FR6989 connections are shown, then how the sensors are connected in the circuit, and we conclude with the jumper and header pin port connections.

5.1.1.1. Power System Architecture and Operation

The CSS power system consists of an AC/DC or DC/DC (depending on user location) power adapter feeding power to the charge controller. The charge controller monitors the presence of the power adapter and communicates this to the system controller. When the power adapter is not detected, the charge controller is placed into low power mode and system power is supplied by the battery. When a power adapter is present without any fault indication, the power adapter supplies power both to the charge controller as well as the system. The charge controller provides current to the battery in accordance with the preprogrammed profile specific to the battery chemistry. If the system power demand in addition to the charge power demand exceeds the programmed adapter power limits, the charge current is limited to allow the system to continue at full power operation. Whether the system is running from battery or adapter power, the source power is supplied to the 5 volt regulator circuit and is output to the 5 volt rail, and the 3.3 volt regulator receives its supply power from the 5 volt rail. Now that component selections have been finalized, the total system current demand must be verified to ensure that the selected power system parameters are sufficient for operation. Table 5.1 sums the current demands for verification of each voltage rail.

Device	Voltage Rail	Current Rating
nVidia Jetson Nano	5 volt	2 A
TI MSP430FR6989	3.3 volt	2.7 mA
Memsic MXC4005XC	3.3 volt	1.6 mA
Waveshare IMX219-130	3.3 volt	0.3 A
uSD Port	3.3 volt	0.1 A
Bluetooth Module	3.3 volt	5 mA

Table 5.1: System Current Consumption Verification

The system therefore consumes 2 amps at 5 volts and 0.4093 amps at 3.3 volts, for a total power demand of 11.351 watts. Given that the 3.3 volt rail is fed from the 5 volt rail, the 3.3 volt power demand is actually a function of the 3.3 volt regulator efficiency (96.26%). The 3.3 volt power demand is therefore $(3.3 \text{ V} * .4093 \text{ A}) / 0.9626 = 1.4032 \text{ watts}$, which increases the system demand to 11.403 watts, or 2.281 amps supplied by the 5 volt rail. Given that the system was designed to provide up to 3 amps on the 5 volt rail and 0.750 amps on the 3.3 volt rail, the system is sufficiently scoped to supply the system power demands.

5.1.1.1.1. Charge Controller Primary Block

The charge controller circuit is composed of two functional blocks: the primary block and the output block. The primary block contains a 5 volt system reference matrix that provides an output from the internal 5 volt reference voltage as well as the first resistor of a voltage divider circuit. Each individual sense or control circuit contains a second resistor to complete the voltage divider, which is used to: sense the presence of the AC adapter, control signal to select direct the charge controller to output to isolate the battery for charging or connect the battery to supply system power, output signal to provide a programmable alarm to the system controller, current sense output to the system controller, input to program the voltage threshold of the battery, and input to program the maximum charge current of the battery. The primary block passes power from the AC/DC or DC/DC power adapter to rectification diodes to provide primary filtering, control MOSFETS which isolate or connect the power adapter current to the system output, differential sense Op-Amps to provide voltage sense inputs to the charge controller, as well as filtering capacitors to filter any latent AC noise from the power source. The figure below shows the schematic for the charge controller primary block.

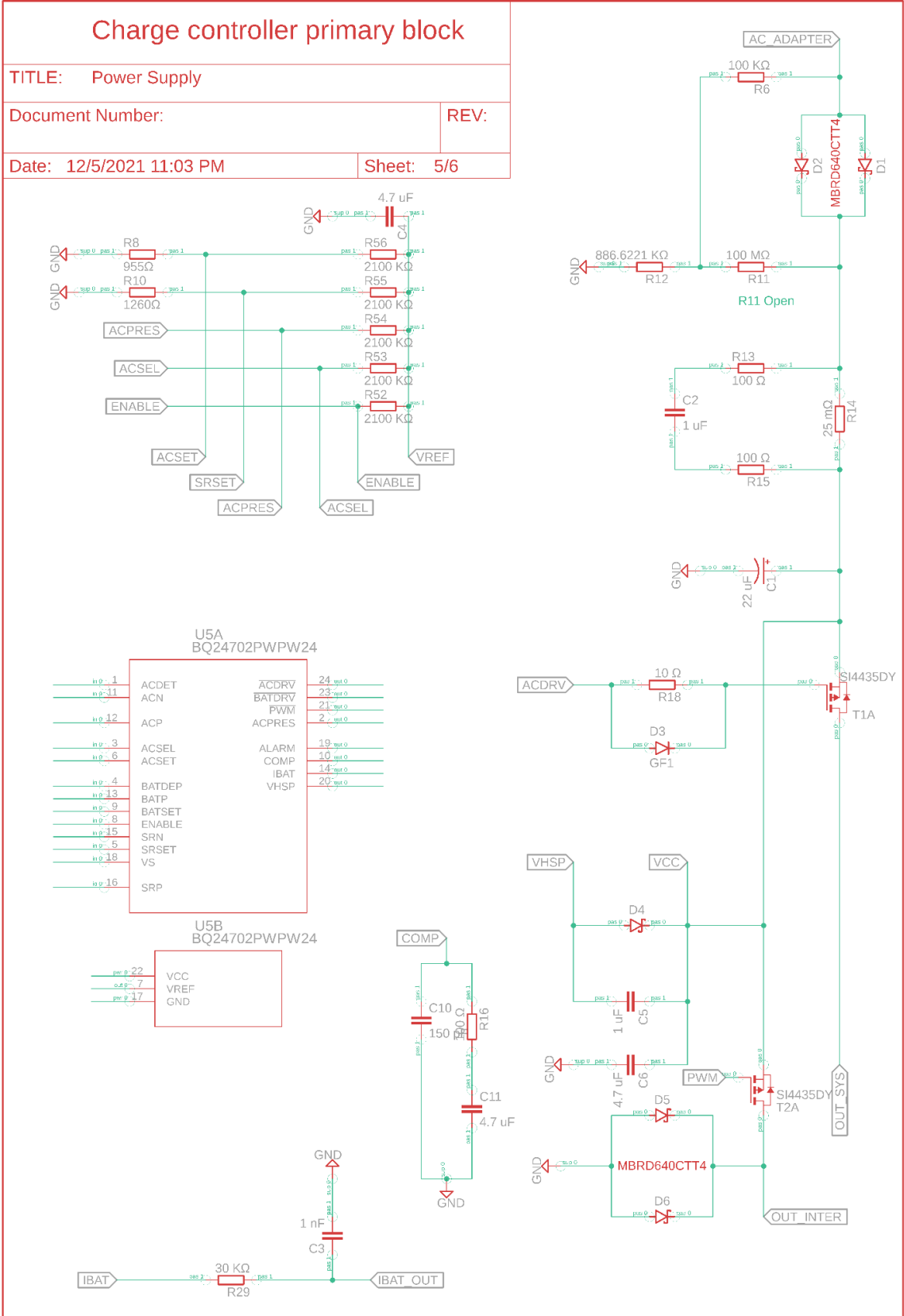


Figure 5.1: Power System Schematic: Charge Controller Primary Block

5.1.1.1.2. Charge Controller Output Block

The output block of the controller circuit provides the external components to control the charge current to the battery as well as isolating or connecting the battery to the host system. The charge current is controlled by the PWM MOSFET, which the charge controller switches based on the voltage sensed by the differential sense Op-Amps of SRN and SRP. The inductor is used to filter and store the switched current while the diode pair provides protection and filtering from oscillations inherent in inductor operation. B ATP is a voltage divider circuit which normalizes the battery voltage to the reference voltage provided by the charge controller and a comparator signals the batteries at charge threshold when the comparator inputs match. BATDEP provides battery voltage sense to provide input to the charge controller of the battery depletion level using a similar scheme to B ATP (battery voltage is normalized by voltage divider and compared to reference voltage), which is used to protect the battery from damaging charge under depletion scenarios in which an undervolted battery would be charged at full charge current. Lastly, BATDRV and VS are used to decouple the battery from the host system to enable battery charging while the system power is supplied by the power adapter of the OUT_SYS net. This operation is performed in response to the presence of the Power Adapter and direction from the system controller. The input and output signals from the bq24702 to and from the MSP430FR6989 are routed to the voltage level shifters, where the outputs are routed to the gate of the level shifters supplied by the 3.3 volt rail to allow the shifter to step the signal down to 3.3 volts while the inputs are routed to the gates of the level shifters supplied by the 5 volt rail to step the signal up to 5 volts. These input/output signals, along with IBAT, BATT1_TEMP, and BATT2_TEMP are then routed to a ten pin connector, which represents the connection to the MSP430. The integrated system shall not have a connector at this point, but the input/output signals between the bq24702 and MSP430 will be routed directly to the GPIO pins of the system controller. The two battery packs are connected in series utilizing one three pin connector each. The positive terminal of battery pack one is connected to the charge output net, BATT1_POS, of the charge controller, while the negative terminal of battery 1 is routed across the connector pins to the positive terminal of battery 2. Shown below is the schematic for the charge controller output block.

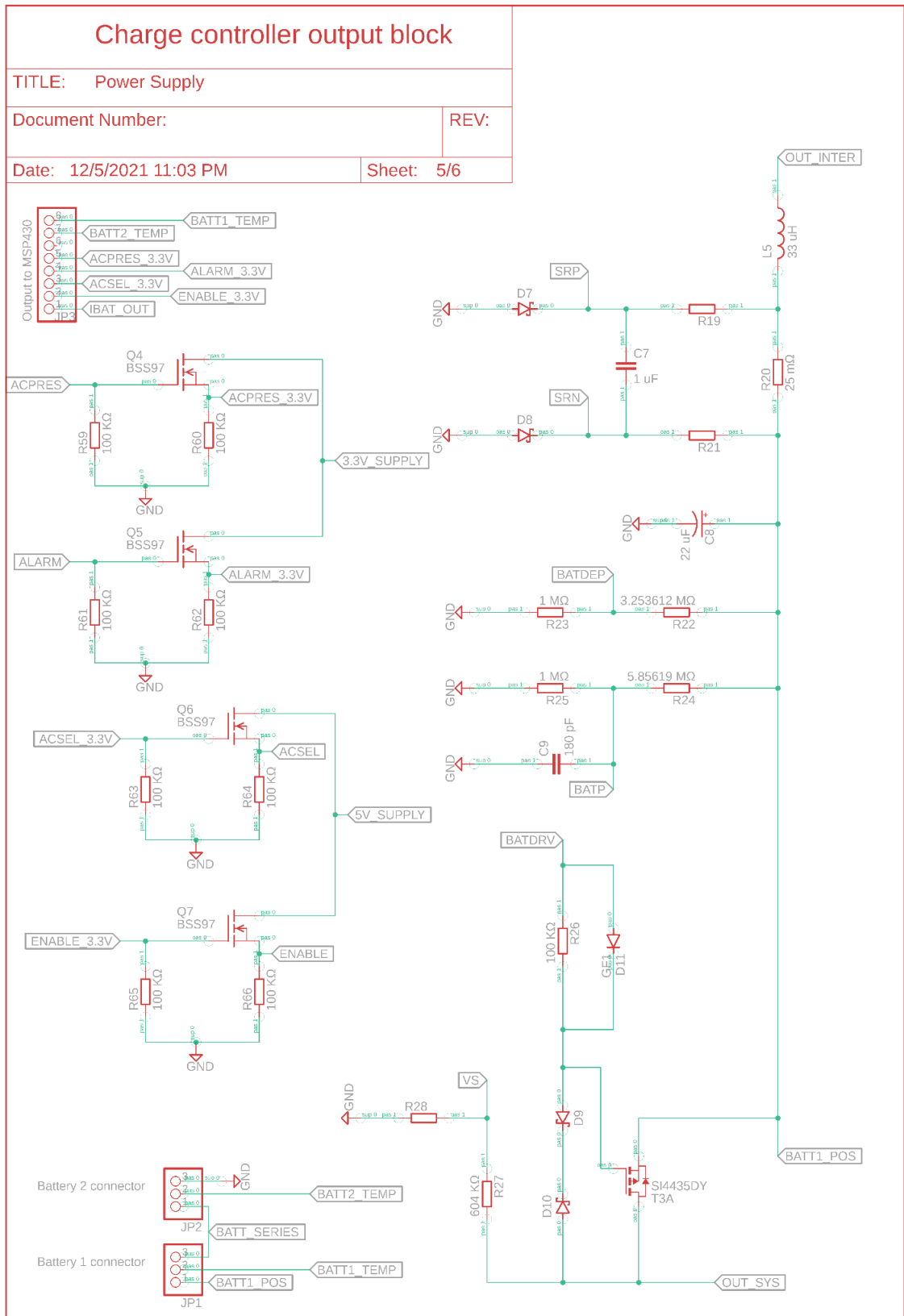


Figure 5.2: Power System Schematic: Charge Controller Output Block

5.1.1.1.3. Programming The Charge Controller

Each of the following input and output circuits is a voltage divider circuit to normalize the desired threshold voltage to the internal reference voltage for proper comparator operation. To prevent power loss wherever possible, these voltage dividers shall have very large magnitudes, but the compFor BATDEP, ACDET, B ATP: where the leakage current of each pin is 0.2 microamps

$$V_{Input} = V_{REF} * (1 + Gain_{Divider}) + (R_{Div} * I_{leak}) \gg Gain_{Divider} = \frac{V_{Input} - (R_{Div} * I_{Leak})}{V_{Ref}} - 1$$

BATDEP programs the voltage at which the battery is considered depleted, and the charge controller outputs the ALARM signal. For the MakerFocus 9065115 the stated depletion level is 2.75 volts therefore the BATDEP target voltage shall be 5.50 volts since the 2 battery packs are in series. Reference voltage for BATDEP is 1.246 volts. For a 1 MΩ primary divider resistor, this gives a voltage divider gain of 3.253612 for a second divider resistor of 3.253612 MΩ.

BATP is the threshold voltage of the battery at which active charging stops and only a small maintenance charge continues to flow. For the MakerFocus 9065115, the maximum charge voltage per battery is 4.20 volts for a target BATP value of 8.40 volts. The reference voltage for pin BATP is 1.196 volts and the pin leakage current is 0.20 micro amps. For a 1 MΩ primary divider resistor, this gives a voltage divider gain of 5.85619 for a second divider resistor of 5.85619 MΩ.

ACDET senses when the power adapter has been connected to the system to inform the system controller. The voltage provided by the power adapter ranges from 12 volts to 14.4 volts, so ACDET must be set to recognize the lower value of 12 volts to ensure the system does not fail to detect the adapter. Care must be taken to ensure that the maximum input value does not exceed the allowable pin voltage of ACDET, thus the voltage divider will be calculated based on a 12 volt input and pin voltage will be verified to not exceed the pin maximum voltage with these calculated parameters and an input of 14.4 volts. For a 1 MΩ primary divider resistor, this gives a voltage divider gain of 8.866221 for a second divider resistor of 8.866221 MΩ. To test, the higher value, the input equation is rearranged to

$$\frac{V_{Input} - V_{Ref}}{1 + Gain_{Divider}} = V_{Pin}$$

Testing an input value maximum of 15 volts, the pin voltage only increases to 1.39912 V, which is far below the pin maximum of 5 volts, and the circuit can operate over the desired range without issue.

The MakerFocus 9065115 batteries have a charge current of 3C, which for the 10,000 mAh capacity results in a charge current of 3 amps. This is programmed into the charge controller by a voltage divider referenced to the internal 5 volt V_{REF} given as

$$I_{BAT} = \frac{V_{SRSET}}{25 * R_S}$$

where R_S is the current sense resistor, V_{SRSET} is the charge current programming voltage with a maximum value of 2.5 volts, and I_{BAT} is the programmed charge current. R_S is also shown in datasheet as R20 with a given value of 0.025Ω , so the only unknown is the value of V_{SRSET} and the equation is rearranged to

$$V_{SRSET} = I_{BAT} * 25 * R_S$$

which results in an V_{SRSET} value of 1.875 volts. Given this target value, the voltage divider is calculated with $V_{in} = V_{REF} = 5v$, $Z_1 = 2100\Omega$, $V_{OUT} = V_{SRSET} = 1.875v$ and the standard voltage divider equation is rearranged to solve for

$$Z_2 = \frac{Z_1}{\frac{V_{IN}}{V_{OUT}} - 1}$$

which results in a value of $Z_2 = 1260\Omega$.

The adapter current must also be programmed such that the charge controller can effectively control the power supplied to the battery without compromising the power supplied to the rest of the system. The XP Power VET30US120C2-JA AC/DC adapter utilized for the CSS provides 2.5 amps at 12 volts, while the power port in a vehicle is commonly 10 amps or greater. This sets the current constraint for the CSS at 2.5 amps. Voltage at the V_{ACSET} pin programs the adapter current and references the same 5 volt V_{REF} internal voltage reference provided by the charge controller as V_{SRSET} discussed above. Where R_{S2} (also shown as R14 in the datasheet) is a 0.025Ω resistor used to sense the current flowing from the AC adapter and is used to program the comparator referenced by the ACSET opamp. The equation to set V_{ACSET} is the same as for V_{SRSET} and reduces to

$$V_{ACSET} = I_{adpt} * 25 * R_{S2} = 1.5625$$

Z_1 of the voltage divider is predetermined as 2100Ω , R_{S2} is 0.025Ω , and I_{adpt} is 2.5 amps, therefore the voltage divider equation is rearranged to solve for

$$Z_2 = \frac{Z_1}{\frac{V_{IN}}{V_{OUT}} - 1}$$

which results in a value of $Z_2 = 954.54\Omega$.

To prevent the transient currents passed by the generation of the input power as well as the ripple current produced by the switching operation of the charge controller, the input and output capacitors must be designed to shunt such currents away from the battery and system. Both capacitors must have a current capacity that is calculated with respect to the duty cycle of the switching operation of the charge controller and the frequency of the switching oscillator, but the duty cycle varies from a minimum of 0% to a maximum at

100% while the oscillator frequency varies from 240 to 350 KHz. This provides a broad operating range, but the maximum current capacity will be sufficient to shunt lower ripple currents from the battery while providing protection from input transient conditions from hot-plugged conditions. This maximum current capacity is derived from the maximum ripple current of

$$I_{Ripple} = \frac{(V_{in} - V_{BAT}) * DutyCycle}{F_S * L}$$

where *DutyCycle* is a decimal ranging from 0 to 1.0 representing 0 to 100%, F_S is the oscillator frequency in Hertz, and L is the output inductor inductance in Henry. With the maximum stated duty cycle as 100% or 1.0, the maximum frequency of $350 * 10^3 \text{ Hz}$, maximum V_{in} of 14.5 volts, a minimum V_{BAT} of 5.50 volts, and an inductance of $33 * 10^{-6} \text{ H}$, the input and output capacitors are calculated to require a current capacity of 6.905 amperes. This is much larger (230%) than the designed and programmed steady state charge current, but the switching nature of the charge controller allows such transient current spikes.

5.1.2.1.4. 5 Volt and 3.3 Volt Rails.

The 5 volt rail is controlled by the TI LM3150 with source power provided by the OUT_SYS net from the charge controller circuit. The power is provided to the VIN net of the regulator which is filtered by the CIN and CBYP capacitors. RON provides a voltage presence sense for the regulator. Source power is passed through VIN to the switching MOSFETs controlled by HG and LG nets while the switched current is fed to the filtering inductor. The feedback net is comprised of a filtering capacitor and a voltage divider to program the desired output voltage as sensed by the LM3150 to control the pulse width modulation of the switching MOSFETs, and COUT3 is the final output voltage filtering capacitor to remove any added noise from the switching process.

The 3.3 volt rail is built around the TI TLV62568 and is fed by the 5 volt rail. The 5V_SUPPLY net is connected to the VIN net of the TLV62568 which holds the Enable pin high and provides power for the regulator. This net is decoupled with CIN2 to provide isolation from latent voltage ripple. The output from the regulator is the SW net which provides the internally switched and regulated current to the output inductor L3 which provides the filtering regulation of the output current. The 3.3V_SUPPLY net provides the output of the 3.3 volt rail and feeds the feedback voltage divider of RFBT1 and RFBB1 which are used to program the output voltage threshold via the FB pin on the TLV62568. Lastly, the output capacitors COUT1, COUT2, and COUT4 provide the final filtering and decoupling capacitance for final filtering of the 3.3 volt rail current. The following figure depicts the schematics for the 5 volt and 3.3 volt rails.

3.3 Volt Rail and 5 Volt Rail

TITLE: Power Supply

Document Number:

REV:

Date: 11/18/2021 9:37 AM

Sheet: 2/5

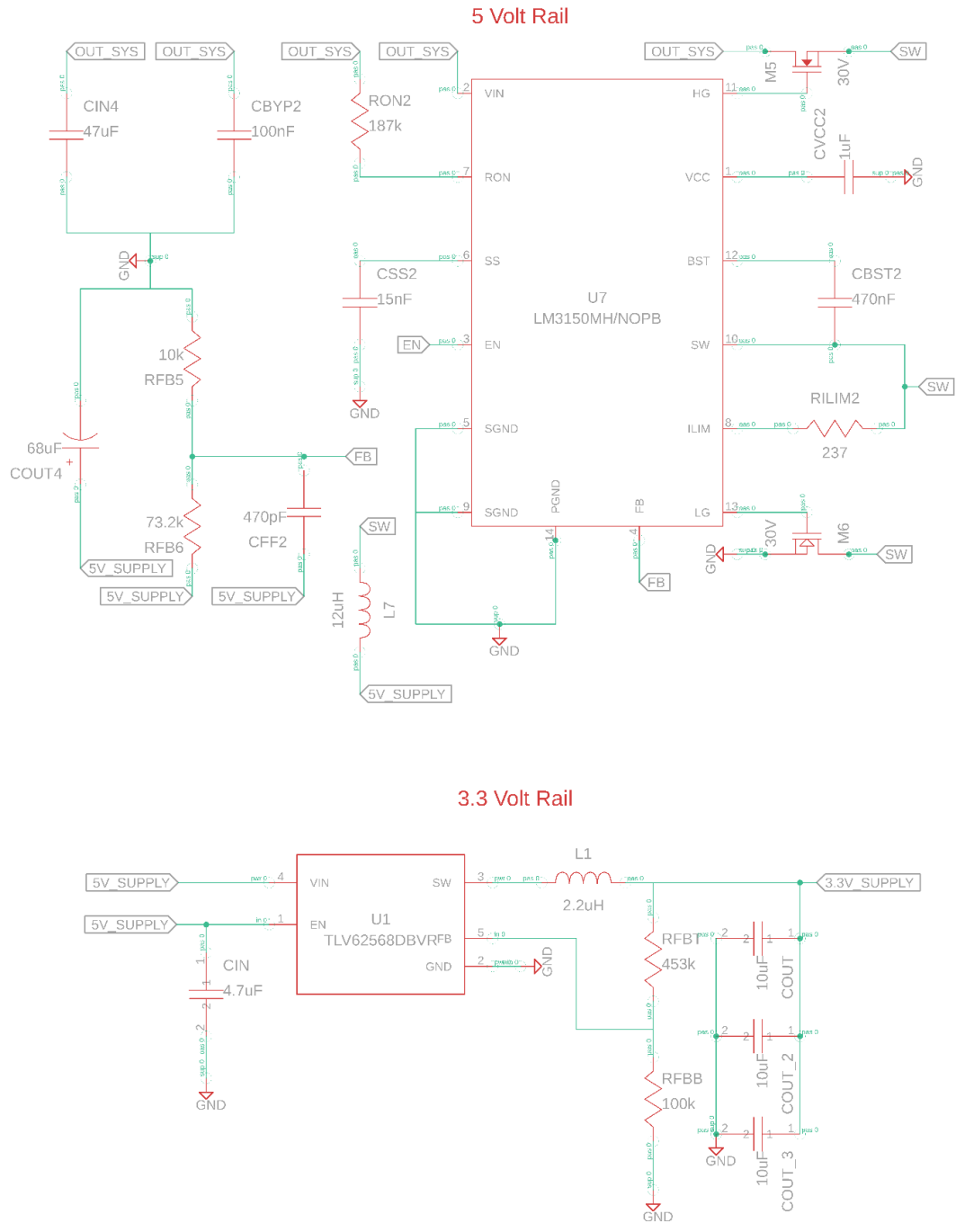


Figure 5.3: Power System Schematic: 5 Volt Rail and 3.3 Volt Rail

5.1.1.1.5. Power System PCB Considerations

The signals PWM, BATDRV, ACDRV, and VHSP of the charge controller and SW, FB, HG, and LG of the 3.3 and 5 volt rails are all high frequency switched and therefore must be isolated from any signals sensitive to the EMF induced by such a high dv/dt circuit. ACDET, BATDEP, VS, BATP, SRP, SRN, ACP, and ACN are the signals most sensitive to such induced EMF as they are extremely dependent on precise current flow to provide feedback of crucial voltage threshold and charge current systems. A small current induced by EMF would result in a large error voltage at the sense pin of the charge controller.

5.1.1.2. MSP430 Circuit

For overall clarity the MSP430FR6989 schematic only shows the associated net connection labels for each peripheral connection. Trying to fit everything in one image obscures meaningful connection information. Labels show all connections for sensors, power, and the Jetson Nano input to the MSP430FR6989 in Figure 5.5.

To provide power to all parts of the MSP430FR6989, the local 3.3V source from the power circuit is connected to each VCC. Analog and Digital. Each connection has its own bypass capacitors to smooth out any noise on the wire as dictated in the datasheet [14]. They are closest to the input of the MSP430 to limit any additional noise development by traveling through the circuit and to further stabilize the input voltage.

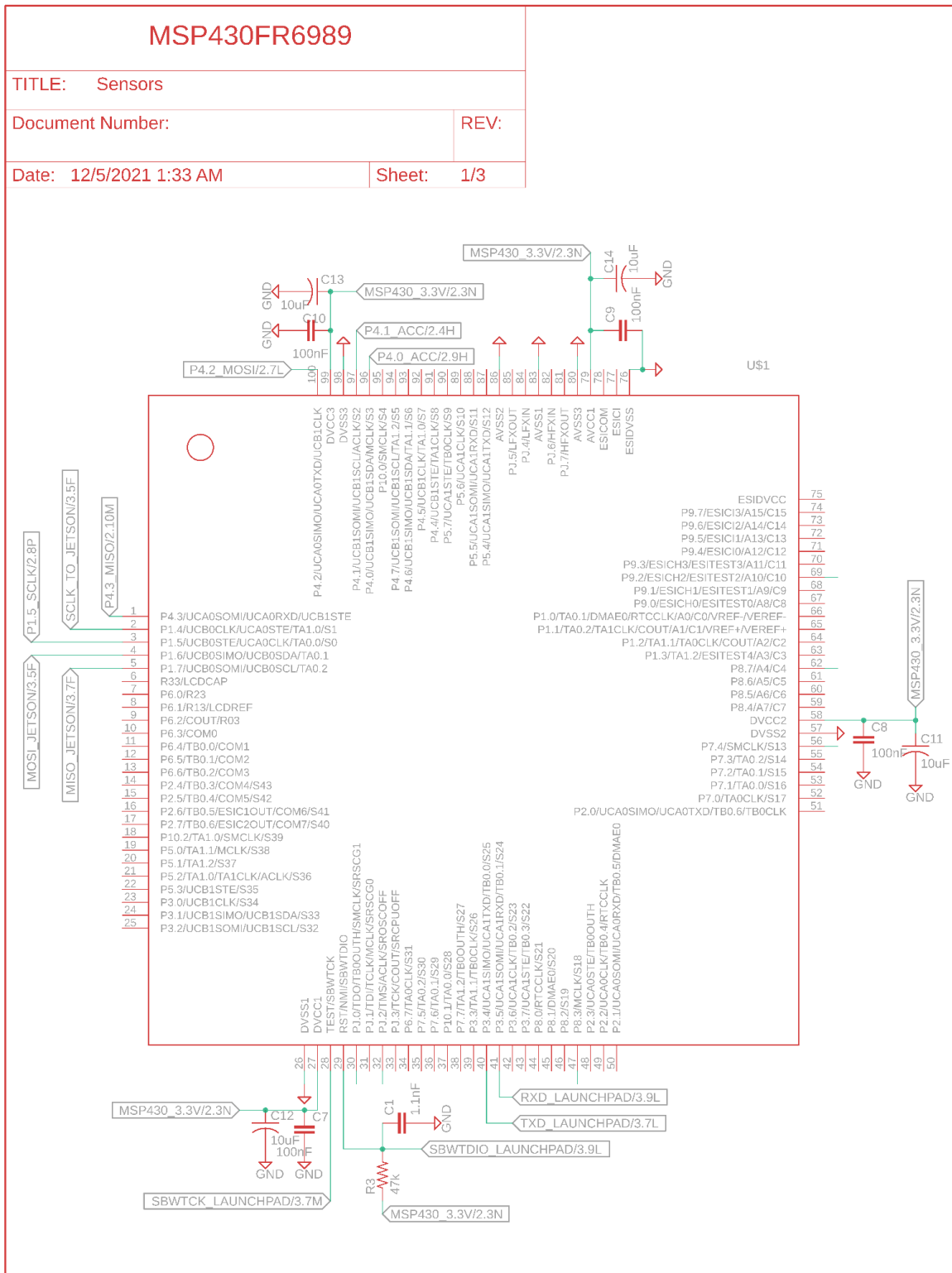


Figure 5.4: MSP430FR6989 Net Label Connections

5.1.1.3. Sensors

Figure 5.8 displays the connections to the MSP430FR6989, Accelerometer, MDBT42Q-P192, and the MicroSD socket using a SPI daisy chain scheme.

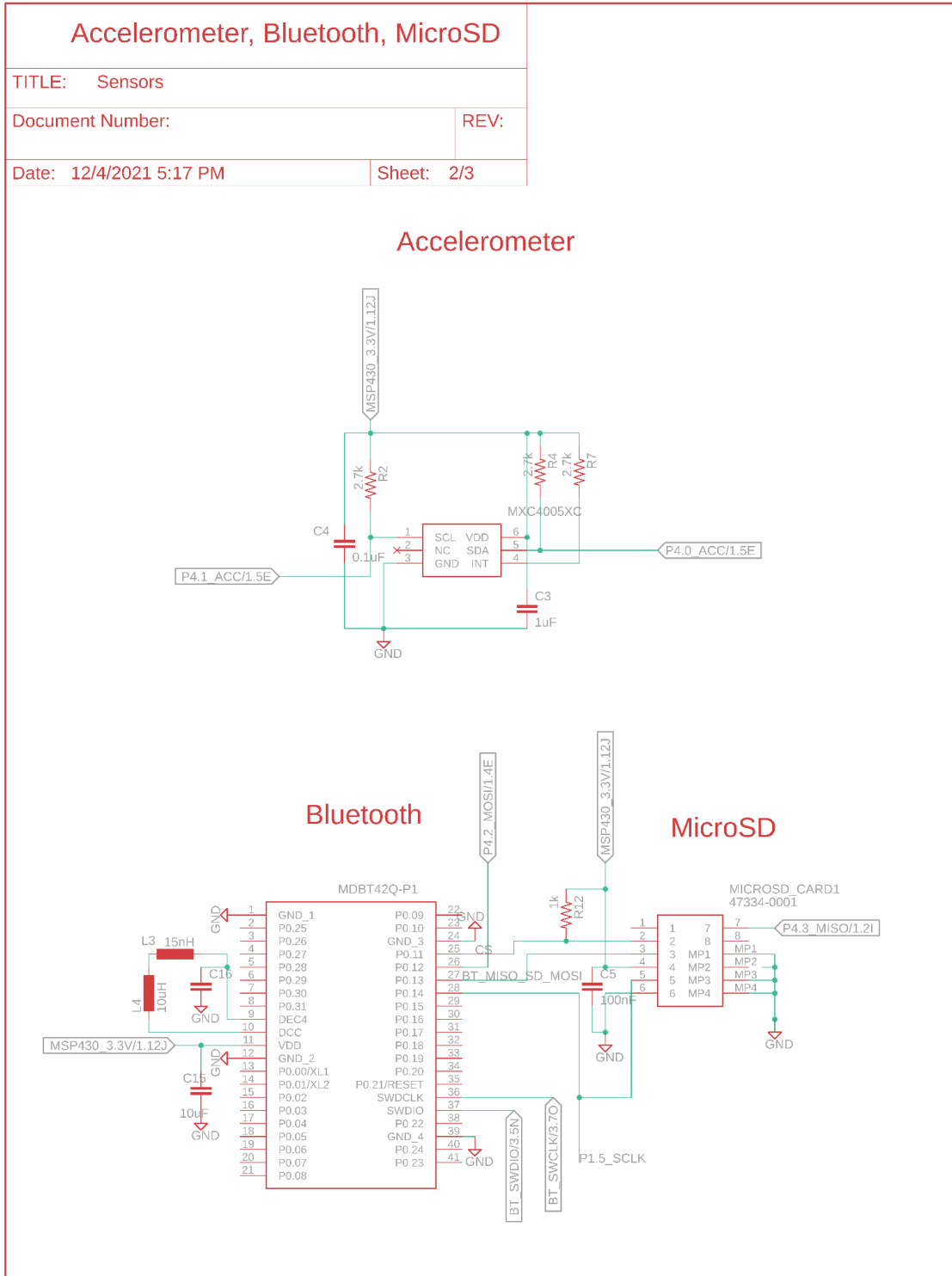


Figure 5.5: Accelerometer/BT/MicroSD/MSP430 Schematic

5.1.1.4. Sensor Connections

The following schematic shows all connection ports necessary for the operation and programming of the MSP430FR6989 and the Bluetooth module. The Jetson Header shows the female port that will connect to the 40-pin header on the NVIDIA Jetson Nano. Pins 19, 21, and 23 connect to the MSP430FR6989 by way of SPI. The 5V supply on pins 2 and 4 shall provide power to the Jetson Nano.

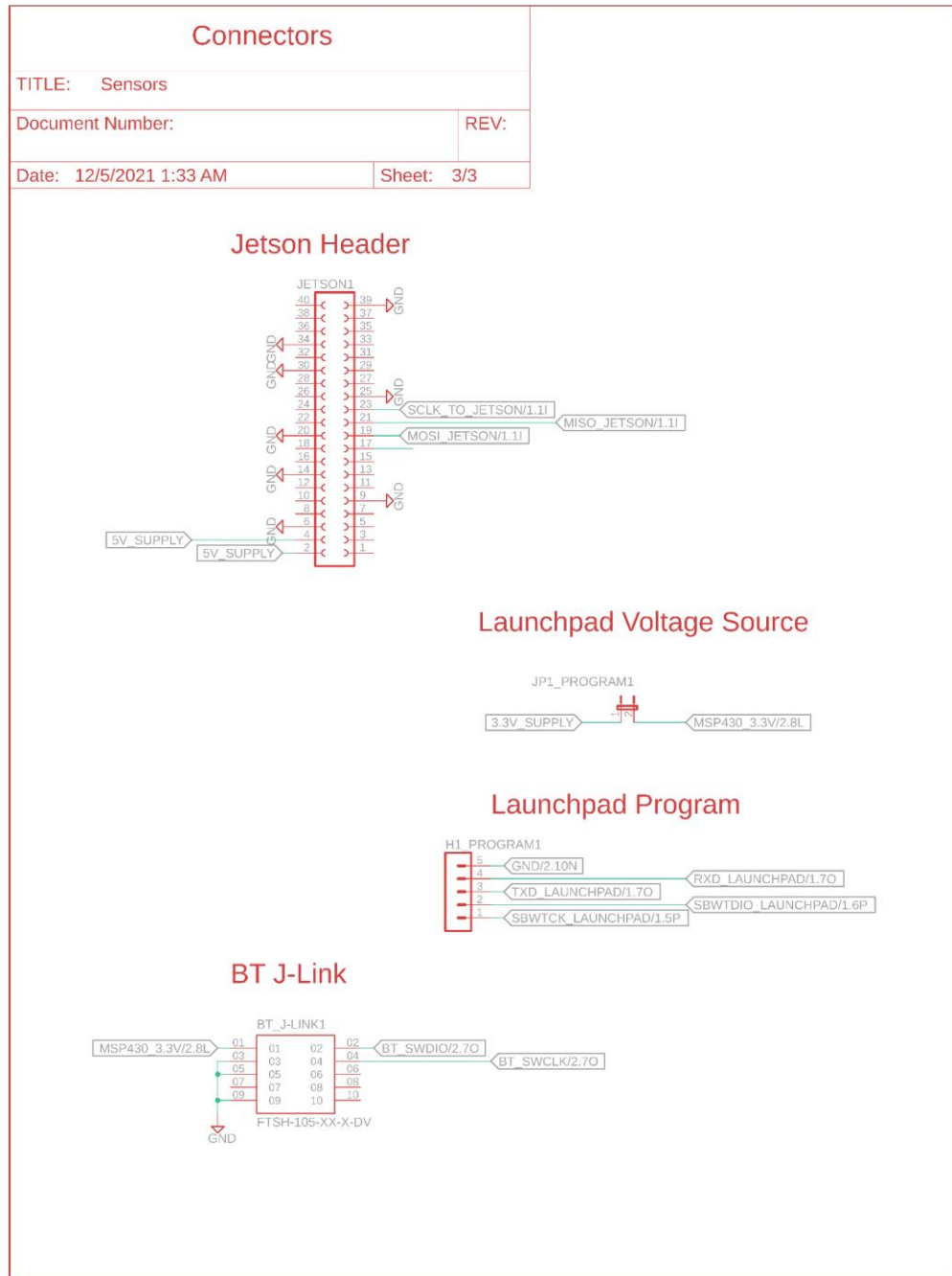


Figure 5.6: Headers for Programming on the PCB/Jetson Communication

Disconnect jumper at “JP1_PROGRAM” to power the external MSP430 from the Launchpad for programming. A female socket jumper wire can be used to connect the external MSP430FR6989 with the 3.3V output on the Launchpad eZ-FET output. Connect Launchpad pins to appropriate header pins on CSS PCB for programming as shown under Launchpad Program. The BT J-Link connector will be used for pretty self-explanatory purposes. This will be the point of connection by 10-pin J-Link header wire to the nRF52-DK for programming of the Bluetooth module.

5.2. Software

Three different forms of software will need to be designed for this project. The single board computer will require a computer vision application that is capable of recording license plate data. Data obtained from the calculations made by the computer vision program will then be sent to a mobile application for display. All inputs made on the project controller must have proper communication protocol interfacing to allow for the transmission of data across all devices.

Both the computer vision and mobile application will each have a specific software flow. These flows will be the basis of each application’s design. The mobile application’s user interface must also be designed to be visually appealing and easy to use. Depending on what pins are available on the MSP430FR6989 and each peripheral, each communication protocol must be configured. Designing all three software components will allow for a fully functioning CSS unit.

5.2.1. Flow of Software on Single-Board Computer (SBC)

Upon start up, the CSS license plate recognition algorithm will identify which one of the two different power modes featured is currently active. If it is the low power (passive) mode, then the SBC can remain off. Only the system controller and the accelerometer will remain on. The system controller itself will also enter low power mode until a jolt is detected. By reducing the number of powered components, power consumption of the system can be optimized and greatly extended. Since the computer vision algorithm will run on the SBC and the SBC is the most power consuming device, this would have the biggest impact on extending battery life. Once the accelerometer detects a collision like jolt, it will wake up all components, including the SBC and begin collecting data. In active mode, the entire system will remain on and continuously obtain as much data as possible.

Once the SBC logs data, it will periodically send data to the system controller to log it onto the interfaced local storage device. In this manner, the data is easily accessible to the user and the local storage device can continue working uninterrupted. Depicted below is the flow diagram of the SBC. As can be seen, the power hungry computer vision algorithm will on be running when the SBC is on, which is when the device is in active mode.

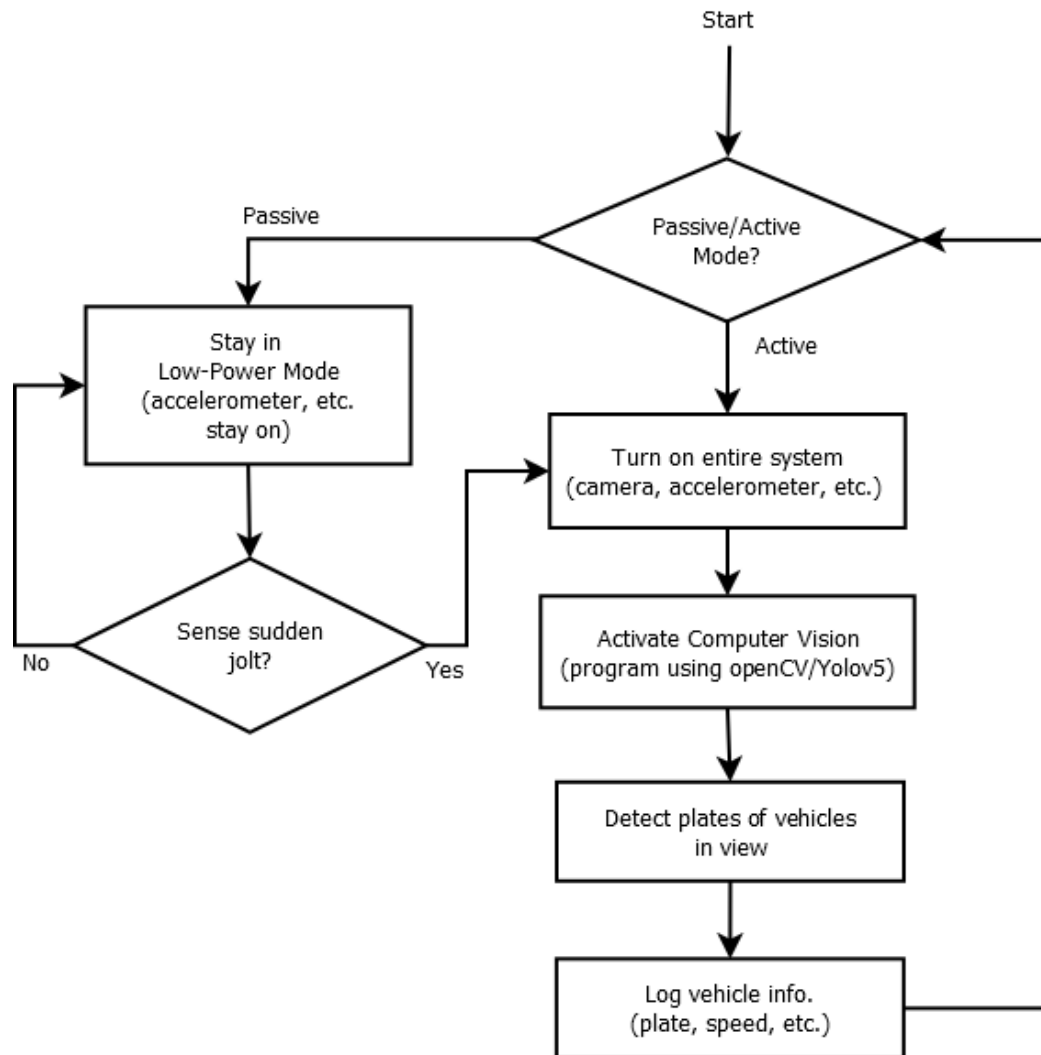


Figure 5.7: Single Board Computer Software Flow Diagram

5.2.2. Computer Vision/Single-Board Computer Design

The Jetson Nano, the SBC, is an entire computing system in of itself. This is the environment in which the computer vision application will be developed and deployed. Once the computer vision application has been developed, the program will also be built to receive input from the MSP430FR6989. This input will determine what state to put the entire Jetson Nano into. Such as a low power state or being fully powered.

It should also be noted that the camera will be directed connected to the Jetson Nano. This is so that the computer vision application can more seamlessly communicate with the camera. This will require less setup for the Jetson Nano to communicate with the camera as it is using the integrated method which will have documentation provided to setup.

Additionally, the Jetson Nano will be configured so that when it is powered on, it will automatically run the computer vision application. It will also be set up so that the Jetson

Nano does not need an external input to go about its operation. The Jetson Nano will be able to run autonomously in conjunction with the rest of the system.

Once the MSP430FR6989 indicates to the Jetson Nano that it should be on and running the computer vision application, then it will be scanning and logging data. The necessary data will be extracted from the camera feed and sent to the MSP430FR6989 via SPI communication protocol. There the MSP430FR6989 will handle the storing of the information recorded.

There are two approaches to this. The first is to continuously send data to the MSP430FR6989 to continuously write to the local data storage unit. The second is to collect the data on the side of the Jetson Nano and send it in batches to the MSP430FR6989. The advantage of the first method is that as soon as the data is obtained, it can be sent and received by the MSP430FR6989. The disadvantage is that it is constantly waiting to receive and write data.

The second method limits data transmission to only certain intervals. That way, the MSP430FR6989 does not need to leave the write to file open constantly. Possibly preventing any data corruption from taking place. The downside, however, would be when the system powers off. Some data may be cut off.

The solution to overcoming this also has two possibilities. The first is to override the shut off procedure for the Jetson nano and have it save the data before shutting down. The second option would have the Jetson nano store the data locally on the Jetson nano and send the data over in batches. The first method has the risk of failing in the event of a battery disconnect. Therefore, the second would be the better alternative so that the data can be sent over safely. Also, it avoids having to override the shutdown procedure.

Therefore, the best course of action would be to develop the computer vision application to store batches of data as it is obtained. Then send them over to the MSP430FR6989 at set intervals. Upon startup, the Jetson Nano will run the CSS application, check with the MSP430FR6989 to see which state it should be in, and enter the appropriate state. When recording it will store the data locally to limit the risk of data loss and data corruption. At set intervals, this data will be sent over SPI to the MSP430FR6989. Since the data is sent over in batches, if there is an incomplete batch from the last time, the application will fill out that batch before sending it over.

As mentioned in the initial conception of the project, the device will also feature troubleshooting LEDs. These LEDs will be set by the MSP430FR6989. However, the Jetson Nano will be responsible for alerting the system controller which LEDs to set. For instance, if the battery is low or if writing data over fails, it will alert the MSP430FR6989. Which will then either light the low battery or storage full LEDs respectively.

An overview of the software flow can be seen in figure 5.9. As noted in the software research, it was decided to use OpenCV and YoloV5 to develop the computer vision application. This will use the Python programming language which can easily interact with

the Jetson Nano and the MSP430FR6989. Which is perfect for the CSS. Python has an abundance of resources and libraries to assist and make the development process smoother. Using OpenCV the program can interface with the camera and settings such as video input resolution can be set. This can also be easily changed and also fine tuned during the testing phase to optimize for accuracy.

Furthermore, built-in sensors on the Jetson Nano can be used to monitor the hardware. One important factor to monitor would be the temperature. Given the environment and the resources that the application will be using, it can get very hot. It would be best to use this temperature data to throttle the Jetson to ensure it can run for as long as possible before built in safety features kick in and take over. That way, control of the behavior of the Jetson Nano can be predictable.

5.2.3. Flow of Mobile Application

The aim of the mobile application is to be lightweight in nature. It will not be bogged down with excessive features. Instead it will have a simple and streamlined experience for users. A user will be prompted to create an account or log in to an existing account. The mobile application will then perform local checks to verify the integrity of the data before sending it to the user to verify whether they are a user or not before logging them in.

Once logged in, a user will be able to pair the CSS with their mobile device. This will enable the transmission of data from the CSS to the mobile app. The mobile app will then perform periodic calls to sync with the database. This will ensure that the database has an up to date copy of the data and that the user can, at any given time, view information obtained from anywhere. Shown below is the general flow of the mobile application software.

This flow can be seen in Figure 5.8. This allows both systems to easily interact with one another while being compartmentalized. This will allow that either system to be expanded independently of one another in the event that a stretch goal is achieved. This reduces the need for refactoring. Not only does this reduce the development work later on and the redundant work, but it also satisfies the goal of maintaining a modular code base. Which is vital for later expansion of the device to fulfill additional stretch goals.

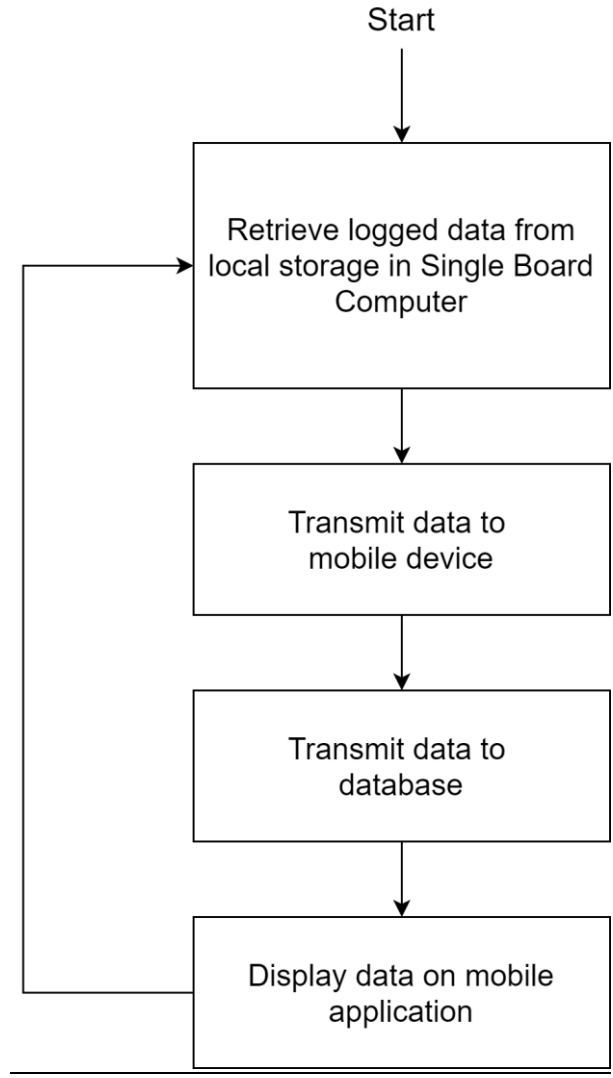


Figure 5.8: Mobile Software Application Flow Diagram

Since the main objective is to provide the user with a view of their data, their interaction with it will be limited. Beyond simply viewing the data, the user will not be able to create new data or modify the data. As it would not make sense to provide this functionality. This is depicted in the use case diagram in Figure 5.9. A user going to such lengths would have the data on hand and have use of it immediately. Therefore, the user will only be able to delete information from the database deemed irrelevant, old, or wrong. It is important to note that this would only apply for the database side of the application and would not interfere with the device itself. When syncing, the database will only request data since the last request was made. This is in order to prevent a user having to repeat any modifications they have made.

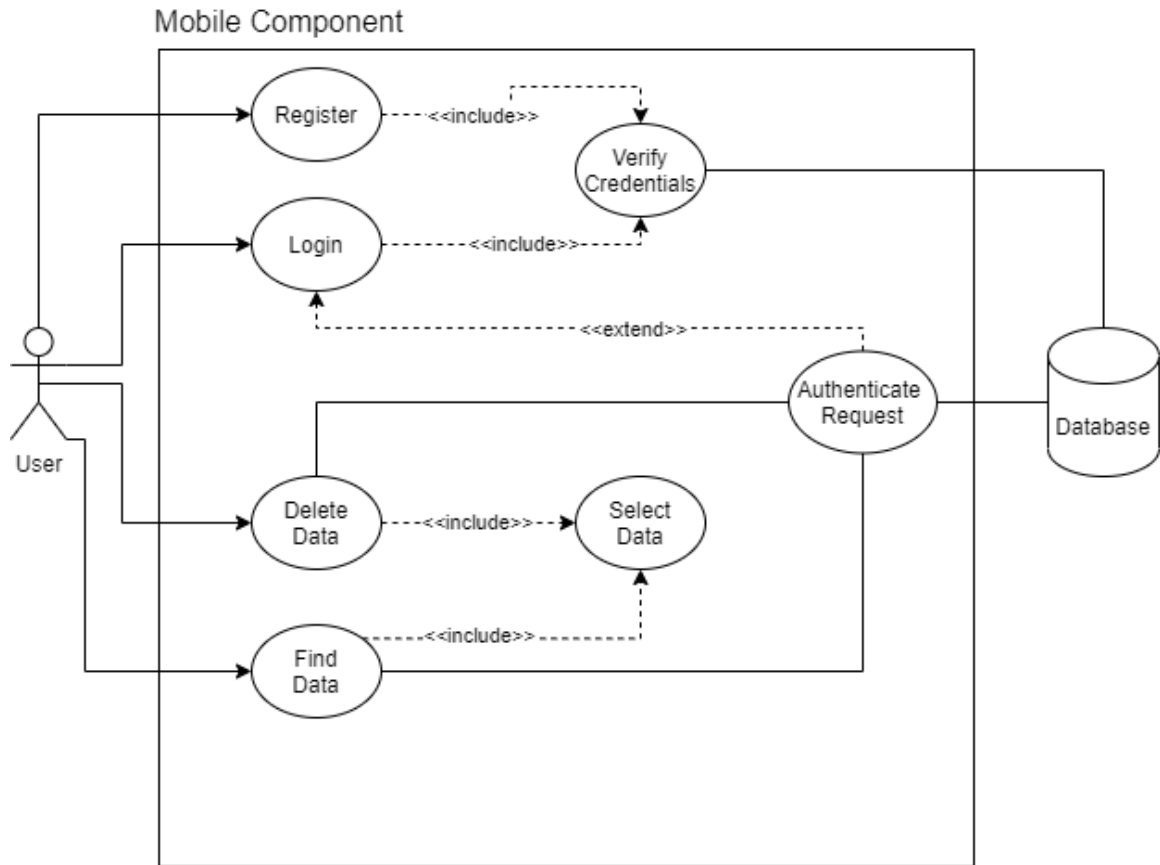


Figure 5.9: Mobile Application Component Use Case Diagram

5.2.4. Mobile Application Design

For the mobile application to be lightweight and user friendly, an extensive amount of programming and design is required. The mobile application will be a progressive web application that will implement some technologies from the FERN stack. All code will be stored in GitHub and accessed/modified using Visual Studio Code. Bootstrap will be utilized to extend functionality. These tools (and many more) will allow for the project to meet application requirements in an efficient and timely manner.

Prior to code development, mobile mockups will be created to establish the website's look and aesthetic. Mockups will be designed using Adobe XD. They will act as a guide in the development of the application's frontend. The user interface of the PWA will primarily use HTML, CSS, and JavaScript. To increase functionality and the speed of the development process, Bootstrap will be implemented. Bootstrap's vast library provides a variety of prebuilt components that can enhance the user experience. When designing the user interface, Bootstrap provides convenient and responsive components, which is ideal for a PWA. To ensure that the application conforms to PWA standards, the application will be tested using Google's Lighthouse.

CSS will consist of four main pages consisting of text fields and buttons that require user interaction and communication across the unit and the database. Other components will showcase data obtained by the CSS unit. There will be four primary pages. A login page will allow for users to get access to their data. The registration page will enable new users to create their own accounts and begin saving data to the cloud database. A “forgot password” page will give users the ability to reset their login credentials. And finally, the main page will display all the data collected from CSS. The layout of these pages will be minimal and straightforward. It will also follow accessibility guidelines, such as having an appropriate amount of contrast. Maintaining these requirements and guidelines will grant multiple users access to data being collected on their CSS device.

With the frontend established, the backend must be configured. Data obtained by CSS will be sent to the micro processing unit and the mobile application. The data sent to the mobile application will be stored in a cloud-based database. Firebase will be used to manage and store vehicle information. This no-SQL database will also handle user authentication and user login credentials. Implementing email confirmation will be considered, but as of now, it will be a stretch goal for the project.

Backend configurations also include the setup of application programming interface (API) endpoints. These API endpoints will provide a set of definitions and protocols to establish a connection between the application and the resources needed to give it proper functionality. Several API points need to be established. Basic API endpoints include login, registration, and logout. A service worker API must also be included to run scripts in the background of the application that do not require user interaction.

A JSON file must be created as a request and response package to enable communication. The file can be created from the frontend using HTML and JavaScript. Incoming data from CSS and from user input will come in as a JSON. Depending on the type of data, the JSON will be deserialized into an object or data structure. The incoming data used will often be used as a database query. Then the payload data will be serialized into a JSON and it will be sent to the client. With the API endpoints established, they will then be documented and tested using SwaggerHub.

Overall, the application must satisfy the needs of the user in a timely manner. While it is important for the developers to create the application in a short amount of time, enough time must be put towards the construction of the application to ensure fast loading time and high performance. As this is a lightweight application that only displays data obtained by CSS, the application should have a very short response time. It is up to the developers to implement the code and design that does not increase response time by a significant amount.

5.2.5. Mobile User Interface

The user interface will be minimalistic in nature to facilitate development and streamline the user's experience. For this purpose a preliminary wireframe was created. This wireframe will serve as the foundation for future development regarding the mobile

application. However, like all wireframes, it is not set in stone. This will provide flexibility later down the road for updates and changes.



Figure 5.10: Mobile Application Component Wireframe

Development of PWAs will require knowledge of HTML, CSS, and JavaScript. Most of these languages will contribute to the construction of the application's user interface. The user interface must abide by accessibility standards and have a consistent and simple appearance. These key components will make the application easy to navigate on a mobile device. Four pages will be designed for the mobile application: login, registration, forgot password, and the main page.

All four pages will have a clean, cohesive look that follows website accessibility standards. The CSS logo will be implemented for users to be able to identify and represent the CSS brand. The logo will be placed at the top of the user interface for each page. Visual aesthetics will then be derived from the logo. Colors used in the logo are gray, red, and black. These colors will be used in the theming of the mobile application. Additional colors such as dark gray and white will also be included to increase contrast between the background and the text that appears. The background will consist of vector-like lines and

blocks of the color scheme to increase visual appeal. The next few figures will showcase the visual design along with each page’s functionality.

The login page will have a simple layout containing text fields for the user to enter their login credentials. Beneath those fields will be a login button, register button, and a “forgot password” link. The “login” button will verify user credentials and transfer the user to the main page upon successful verification. An error message will appear in the case there is a failed login attempt. Shown below is the user interface design for the login page.

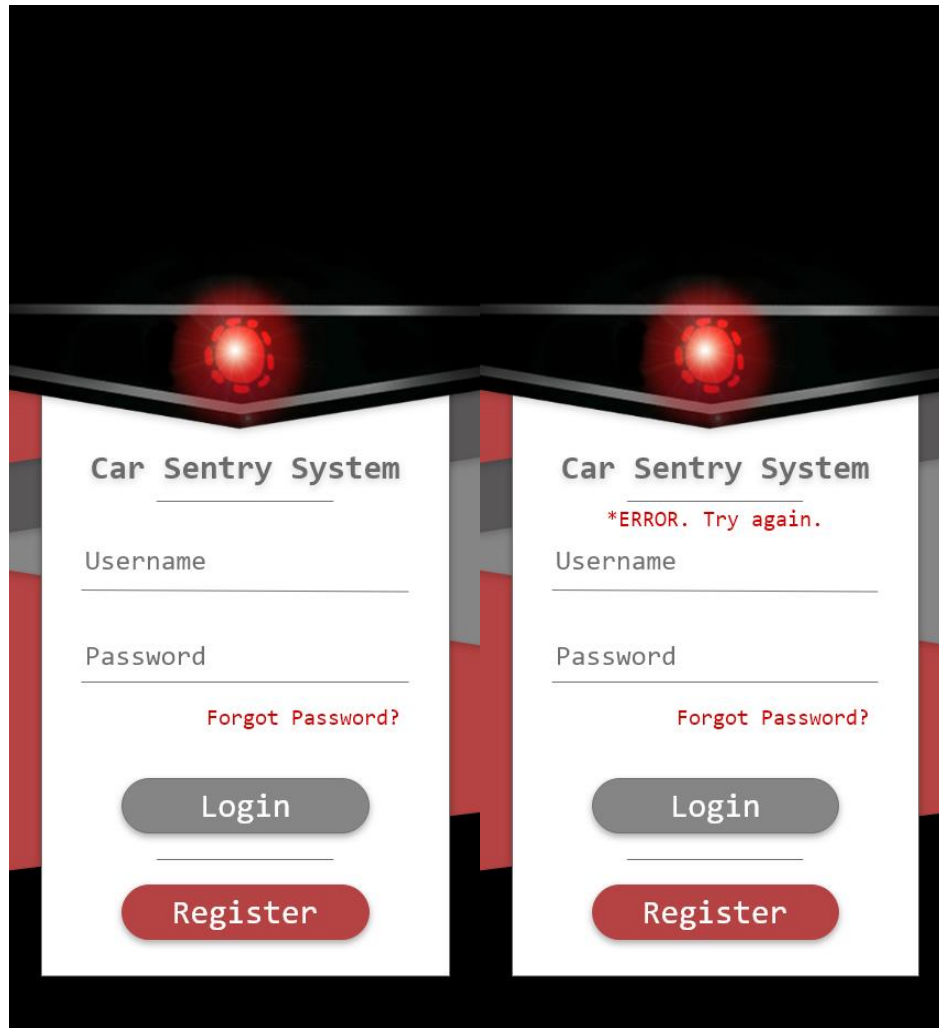


Figure 5.11: Login Page and error message on iPhone 12

The “register” button will send the user to the registration page where they can set up a new account. If the user chooses to register, they will be presented with five text fields. The user will be required to input their first and last name, a password, email, and serial number pertaining to their CSS unit. From there, the user will select the “register” button to create their account. If any text field is missing or the serial number is not valid, an error message will appear. When all text fields are completed and a valid serial number is inserted, a new account is generated, and all their data is sent to the database. The figure below shows the layout of the registration page.

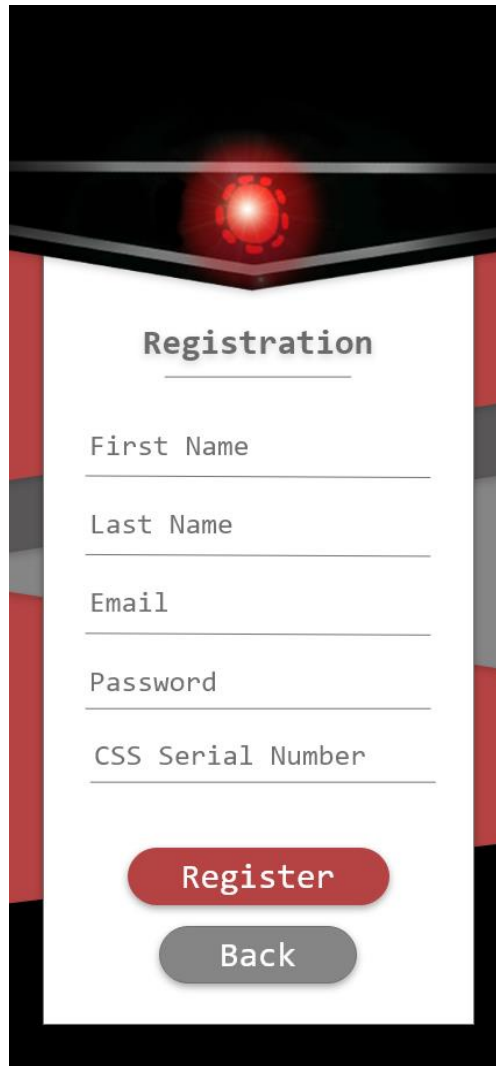
The image shows a mobile application registration page on an iPhone 12. At the top, there is a decorative header with a glowing red virus-like icon. Below the header, the word "Registration" is centered in a bold, sans-serif font. Underneath, there are five text input fields, each with a label above it: "First Name", "Last Name", "Email", "Password", and "CSS Serial Number". At the bottom of the form, there are two buttons: a red "Register" button and a grey "Back" button. The entire form is set against a white background with red and grey decorative elements on the sides.

Figure 5.12: Registration Page on iPhone 12

People often forget passwords, so the “forgot password” button located on the login page will direct the user to a “forgot password” page. This page will prompt the user to input their email and their CSS serial number. If a valid email and serial number are tied together, an email will be sent to the user to reset their password. If an error occurs, a message will appear and the user will be asked to input the correct data. The following figure shows the “forgot password” page and its three different states.

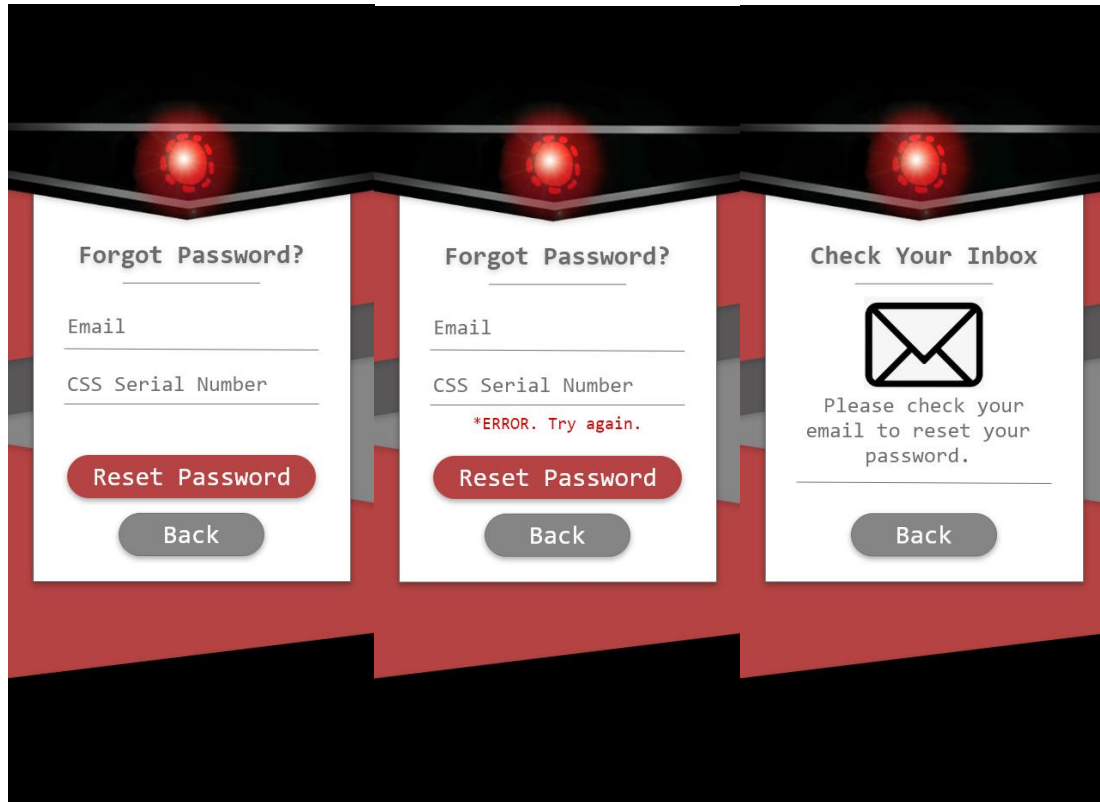


Figure 5.13: “Forgot Password” Page with error message and notification on iPhone 12

A verified user will then have access to the main page. The main page will be a visualization of the database with easy-to-use features. The primary components of the main page will be the search bar and a list of data obtained by CSS. The search bar will help users narrow down vehicles based on their attributes or license plates. A card component will contain a summary of the data obtained for a certain vehicle. If a card is selected, the card will expand and provide all details regarding the vehicle in addition to a “delete” button. If a user would like to discard data on a vehicle, they will have the option to delete it and be prompted with a verification pop-up to make sure that they want to proceed. If the same card is selected again, the expanded card will collapse to its original state. To logout of the account, the user can select the “logout” button and they will be redirected to the login page. Depicted below is what the main page will look like.

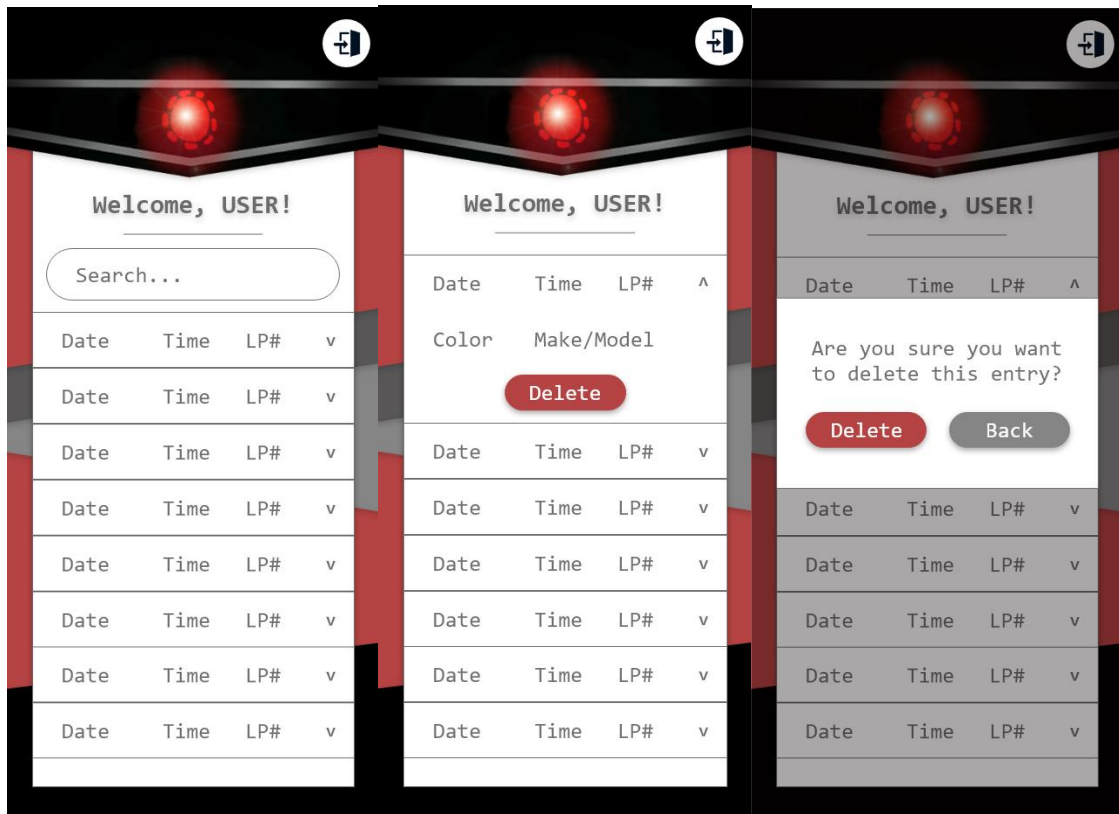


Figure 5.14: Main Page on iPhone 12

5.2.6. Communication Protocol Interfacing

Data transmission between the project controller and single board computer is crucial. This process is necessary for the computer vision software to run properly. The project controller is responsible for connecting all peripherals. Specifically, the MSP430FR6989 must have communication buses to the Jetson Nano, accelerometer, SD Card, Bluetooth, and GPS. The accelerometer and GPS modules will collect data based on the vehicle's speed and location, respectively. Data related to license plates and vehicle attributes will be acquired via the Jetson Nano. All data will then be directed to the SD Card for the information to be stored. The Bluetooth module will act as a gateway to send all data stored in the SD Card to a cloud database. For successful transmission of data across all peripherals, communication protocol interfacing must be implemented.

Multiple communication protocols must be put in place to exchange data between the project controller, sensor modules, and single board computer. The MSP430FR6989 will be the master that will communicate with the accelerometer, SD Card, Bluetooth, GPS, and the single board computer. Three protocols are supported by the selected project controller: I2C, SPI, and UART. Code Composer Studio and Visual Studio Code will be used for all interfacing.

5.2.6.1. Single Board Computer: Jetson Nano

Communication between the Jetson Nano and the MSP430FR6989 will be set up with serial peripheral interface (SPI) ports available on both devices. The SPI master device will be the MSP430FR6989 and the SPI slave will be the Jetson Nano. This means that the clock signal will be generated by the project controller and will be used to keep both devices in sync. MISO and MOSI will be utilized as data will be exchanged in both directions. Information relating to vehicle speed and location from the accelerometer and GPS will be sent from the MSP430FR6989 to the Jetson Nano. That data will then be tied to specific vehicles and their attributes obtained by the Jetson Nano. The consolidated data will then be transferred back to the MSP430FR6989 so it can be stored in the local storage unit. Considering the flow of data, SPI slave selection will be based on the regular independent slave configuration. The figure below shows the SPI interface and the flow of data.

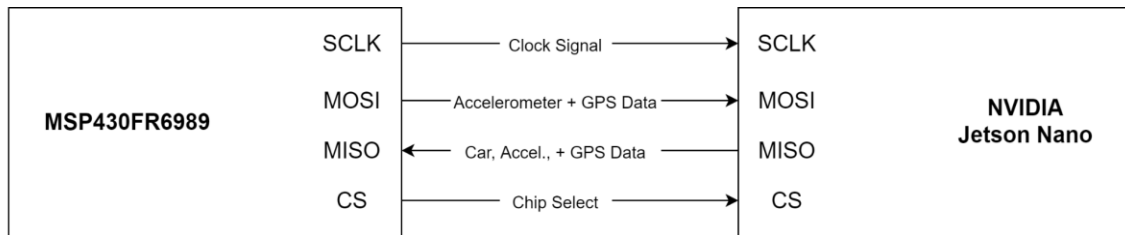


Figure 5.15: SPI Interface between MSP430FR6989 and Jetson Nano

SPI provides various modes that alter data transmission. These modes are notated by their corresponding mode number. Each mode is based on the clock polarity, phase, and edge in reference to the SCLK. There are four possible modes. The modes for SPI are shown in the table below.

Mode	Polarity	Phase	Description
0	0	0	SCLK: Active High Data capture: Rising Edge
1	0	1	SCLK: Active Low Data capture: Rising Edge
2	1	0	SCLK: Active High Data capture: Falling Edge
3	1	1	SCLK: Active Low Data capture: Falling Edge

Table 5.2: SPI Mode Comparisons

By default, both devices will operate on SPI mode 0. This mode will set both the polarity (CPOL) and phase (CPHA) to zero. A polarity of zero will make the clock idle at low. A phase of zero will have the devices latch at the trailing edge, and communicate at the

leading edge. The figure below is a visual of the timing that occurs depending on the mode selected.

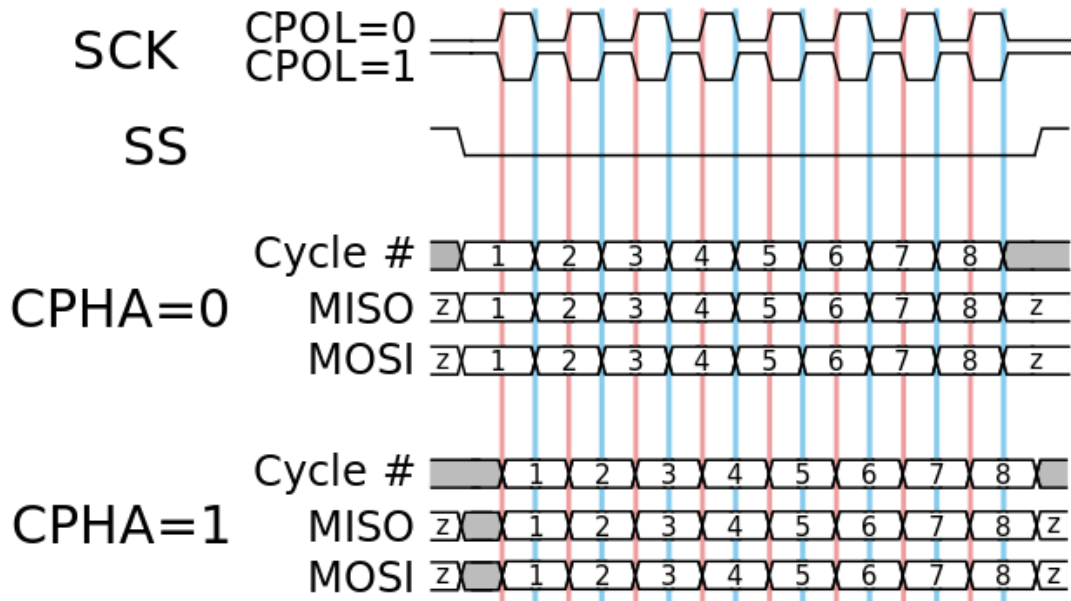


Figure 5.16: SPI Timing Diagram. The use of this image is governed by the GFDL. [15]

The SPI pins will be configured using the datasheets for the Jetson Nano and MSP430FR6989 and Code Composer Studio. The SPI pins must be diverted to SPI functionality. Several functions will be developed to read and write the data that must be transmitted. Libraries will be implemented and utilized within those functions to extend their read and write capabilities.

5.2.6.2. Accelerometer

The accelerometer will only be outputting data to the project controller. I2C is the only protocol supported by the Memscic MXC4005XC accelerometer. While the protocol supports full-duplex, the two devices will only utilize half-duplex as data will be moving in one direction. As mentioned previously, the MSP430FR6989 will act as the master device and the accelerometer will be the slave.

With I2C interfacing, only two pins are required from each device. The serial data (SDA) pin will be the line that will enable data transmission from the Memscic accelerometer to the MSP430FR6989. To ensure data is being sent in real time and at the proper time interval, a clock signal is sent from the MSP430FR6989 to the Memscic using the SCLK pins. This allows for synchronization. No other modules will require the I2C protocol, so there will be no need to design the system where the project controller must handle multiple slave devices using the same protocol. The figure below is a visualization of the I2C interface.

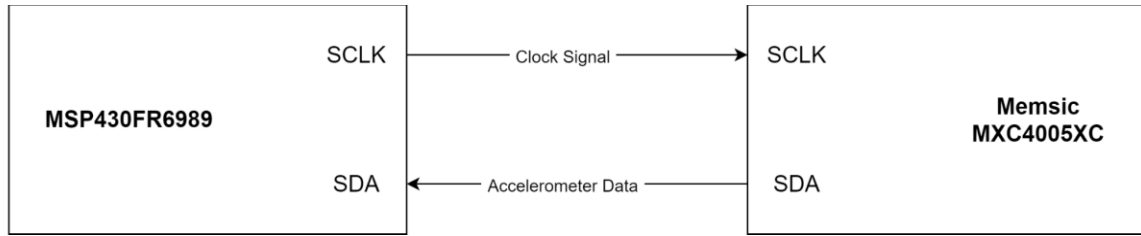


Figure 5.17: I2C Interface between MSP430FR6989 and Accelerometer

There are various I2C modes that the accelerometer and MSP430FR6989 can use. These modes relate to the maximum speed, capacitance, drive, and flow direction. I2C modes are displayed in the following table.

Mode	Max. Speed	Max. Capacitance	Drive	Direction
Standard	100 kbit/s	400 pF	Open drain	Full-duplex
Fast	400 kbit/s - 1 Mbit/s	400 - 550 pF	Open drain	Full-duplex
High-Speed	1.7 Mbit/s - 3.4 Mbit/s	400 - 100 pf	Open drain	Full-duplex
Ultra-fast	5 Mbit/s	-	Push-pull	Simplex

Table 5.3: I2C Mode Comparisons

With data flowing in one direction, the I2C interface can utilize ultra-fast mode. This mode can provide a maximum transmission speed of 5 Mbit/s. In this mode, data can only be written. Thus, data related to the speed of the card will be written to the project controller's connected local storage unit. However, in the case that errors would arise in the movement of data due to the fast transmission of data, the other modes such as high speed mode, fast mode, and standard mode can be used.

All modes will be following the same timing pattern. The primary difference lies within how fast the data transfers. Data starts the transmission process with the starting condition that is signaled by the SDA pulled low as SCL remains high. When the SCL is low, the SDA sets the first bit of data and keeps the SCL low. When the SCL rises, the data is obtained. Validation of the bit is done when the SDA does not change within the rising and falling edge of the SCL. This process continues until there is a clock pulse that pulls the SDA low for the stop condition. Data transmission then stops once the SCL and SDA rises. Shown below is a visual of I2C's timing diagram that indicates the start condition (S), setting (blue bars), bits (B1, B2, ... BN), and stop condition.

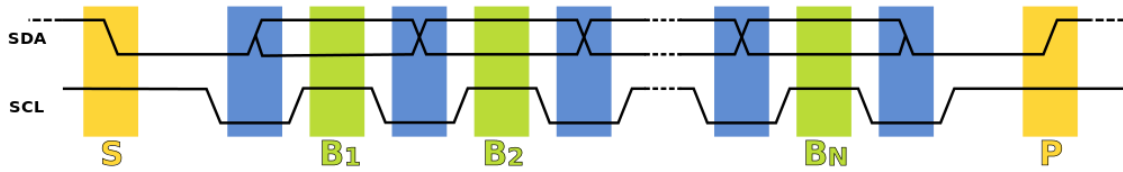


Figure 5.18: I2C Timing Diagram. This image has been released to the Public Domain. [16]

5.2.6.3. Bluetooth & SD Card

Since the MSP430FR6989 can at most implement 2 SPI communication channels and we need one of those for the Jetson Nano communication, the Bluetooth module and MicroSD card slot need to be connected in a daisy chain configuration. The data can be shifted from the Bluetooth module to the MicroSD card where data privacy is not needed because they will be receiving the same data. Figure 5.19 below shows the SPI interface using the daisy chain configuration.

Data will begin to be shifted from the MSP430FR6989 to the Bluetooth MDBT42Q-P192. The MDBT42Q will need to be configured as a slave device with the MSP430FR6989 as a master. The firmware on the MDBT42Q-P192 will need to be written such that the data received from the MSP430 is compiled and transmitted via Bluetooth in the form of a single packet. This can be done by incorporating an SPI code on the Bluetooth device to temporarily store the received license plate with a Bluetooth transmit code to send the data packet. The license plate information will then be received and displayed by the mobile app.

It is also important to note that transferring text files via Bluetooth will be easy and seamless. The addition of images and videos would require a more stable connection and indication that the transfer is in progress. As these are much larger files that could take a significant amount of time. Therefore, either the user would have to manually wait for the upload to complete, or the device would have to automatically do it in the background while the device was running. To reduce the risk of file corruption or data loss, the device would be alerted if a user went beyond a given distance to prevent sudden shut off. If the device was plugged in, then when the car is turned off could also be used as a signal to halt and power down.

Depending on the level of permission given to the final version of the mobile application, the user may or may not be given the ability to update the SD card to correct wrong information or update data. However, as previously discussed this may face legal issues as a user would be able to wrongly update data. This may be addressed by keeping a log of the data that the user cannot modify.

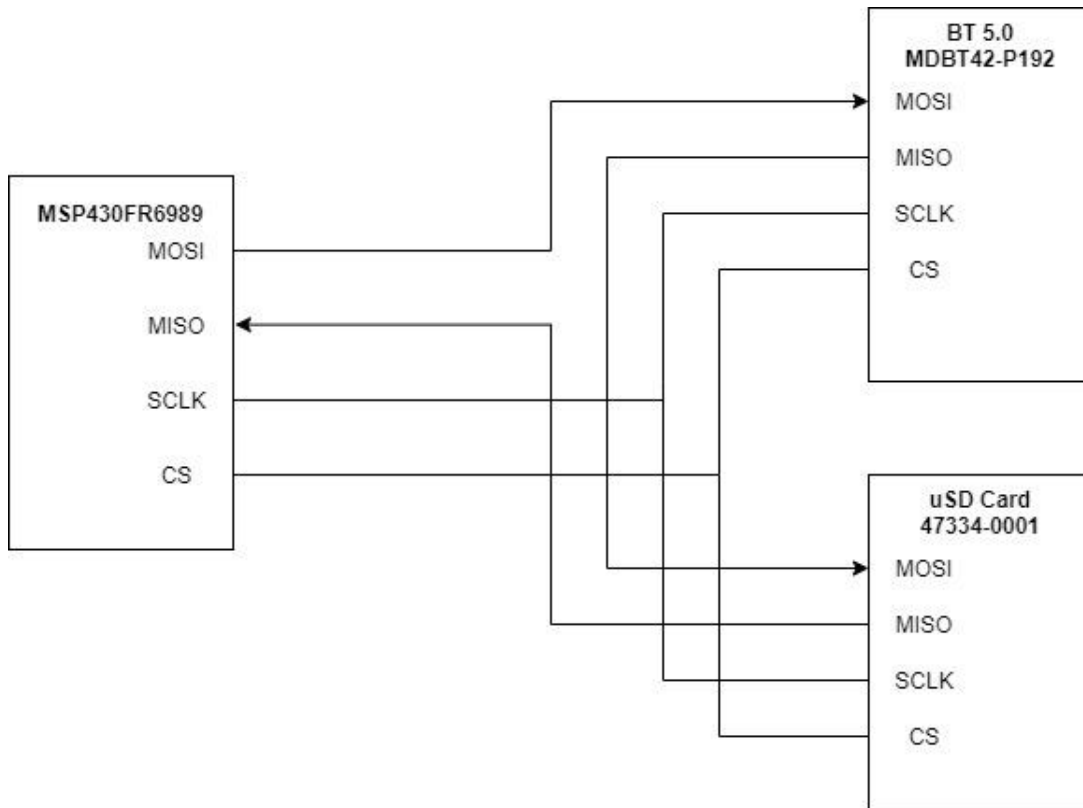


Figure 5.19: SPI Daisy Chain Interface between MSP430, BT, and MicroSD

5.2.6.4. GPS

While the GPS module is considered a stretch goal, we will continue to discuss the design and implementation of the module. The GPS module will be the only peripheral utilizing the UART protocol. As the GPS is independent in terms of its interfacing, it will not interfere with the data exchange occurring across all other peripheral devices. This makes it convenient for module implementation and removal.

Data related to the user's location will be received by the MSP430FR6989. The transmitting UART on the GPS module will be connected to a data bus that sends the information in parallel. This will allow for data to be transmitted through the wire to the receiving UART. When interfacing, the Tx pin of the GPS module will be connected to the Rx pin of the MSP430FR6989. No data will be transmitted from the MSP430FR6989 to the GPS module. Thus, the transmitting UART of the project controller will not be connected to the receiving UART of the GPS. Shown below is the bus that allows for GPS data to travel to the MSP430FR6989.

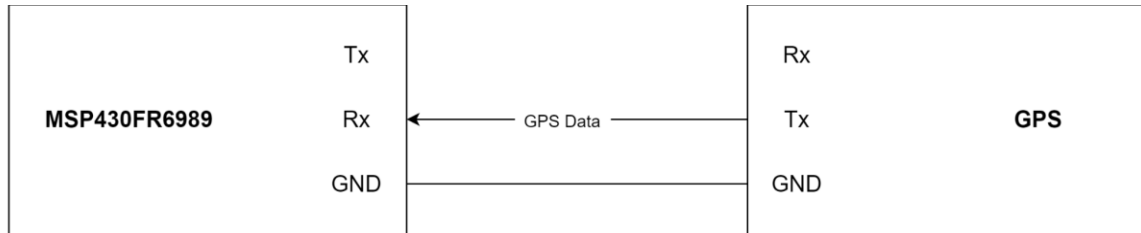


Figure 5.20: UART Interface between MSP430FR6989 and GPS

5.2.6.5. OLED Display

One stretch goal is to incorporate a small OLED screen we already have to display battery state and other important messages on the backside of the CSS, facing the user. Though the MSP430FR6989 has the capacity for 2 different I2C communications, there are not enough channels to incorporate 2 I2Cs with all of our other sensors separately.

Consequently, to proceed with testing by isolating the MSP430 code from the possible hardware connection issues, the Launchpad does not contain enough pins with separate clocks for 2 SPI and 2 I2C connections. In addition, running more transmissions separately would greatly decrease our speed and more than likely impose some significant and unnecessary lag.

To use both I2C devices they can be connected to the same SDA bus and SCLK bus. The firmware will communicate with each device independently by first transmitting the appropriate address. We can reuse and modify some code from earlier projects to get the OLED up and running. The code we have utilizes Adafruit's GFX Library for basic graphics. It gets a little tricky since it is originally coded for Arduino but we can hopefully edit it to work for the MSP430. This will need to be further researched and tested to interface with the MSP430.

5.2.6.6. Charge Controller Communication with System Controller

The TI bq24702 charge controller requires communication with the TI MSP430FR6989 system controller both for basic operation of the charge controller as well as communicating system parameters for communication to the end user. The bq24702 outputs are either analog outputs that must be routed to the Analog-to-Digital converter (ADC) ports of the MSP430 or they are digital logic interface over 0 to 5 volts which must be level shifted down to 3.3 volts because the MSP430 GPIO pins can accept an absolute maximum value of 3.6 volts.

The level shifting is accomplished by using the 5 volt output from the charge controller as the input to an N-Channel MOSFET gate terminal, with a shunt resistor to ground to dissipate charge when the logic level transitions to low. A current branch from the 3.3 volt rail is connected to the drain terminal of the transistor, while the level shifted output is branched from the node connecting the Source terminal and a source resistor to ground. The inputs from the MSP430 to the bq24702 likewise must be level shifted from 3.3 volts to 5 volts utilizing the same concept utilizing the 5 volt rail.

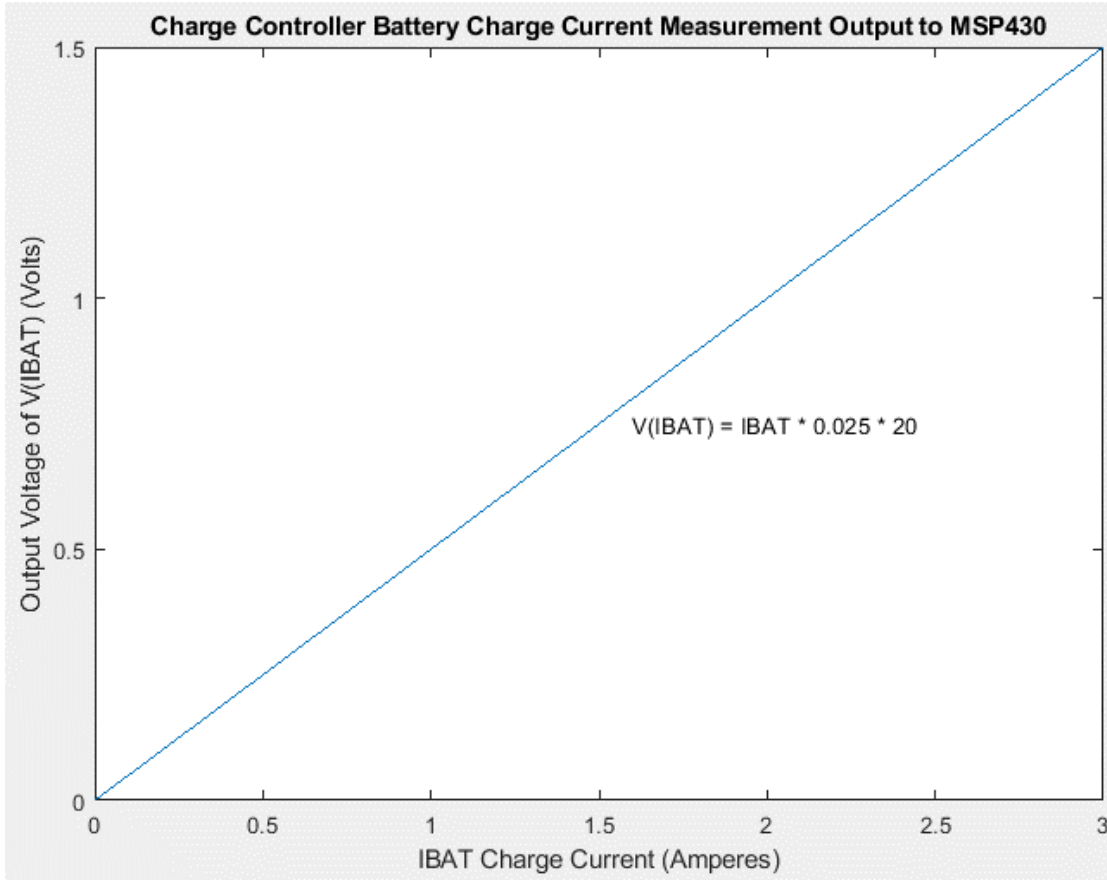


Figure 5.21: Charge Controller Input and Output Level Shifters

IBAT, BATT1_TEMP, and BATT2_TEMP are analog outputs ranging from 0 to 5 volts which are generated by a differential op-amp circuit with gain of 20. The inputs to the amplifier are the current sense inputs SRP and SRN, which measure voltage across the 0.025Ω current sense resistor in the battery charge circuit. For the CSS, the output generated by the amplifier is:

$$20 (V_{SRP} - V_{SRN}) = 20(IBAT_{Actual} * 0.025)$$

where $IBAT_{Actual}$ is the measured current in the battery charge circuit.



Figure 5.22: Charge Controller Battery Charge Current Measurement Output to MSP430

The MSP430 primarily controls the bq24702 through the following signals, for which a brief overview is given.

Signal	Source	Description
ACPRES	bq24702	Logic High if Power Adapter is connected to charge controller.
ACSEL	MSP430	Logic High powers system from Power Adapter, while Logic Low powers system from battery even though Power Adapter is available.
ALARM	bq24702	Battery is depleted. Firmware will force the system into a low power mode until a power adapter is connected.
ENABLE	MSP430	Logic High communicates that PWM battery charging is to take place. Charging only occurs if ACPRES is also high.

Table 5.4: Charge Controller and System Controller Communication Signals

5.3. Stretch Goals

The task of scanning license plates seems easy enough at a glance, however upon further analysis, it is actually a much more challenging task. Not only does each state within the United States have a different license plate, but each nation also has a different license plate. This would add time to training the computer vision software. To complicate things, vehicles can also have bumper stickers that could confuse the algorithm. Furthermore, there are different types of license plates for motorcycles and four or more wheeled vehicles.

Therefore, the initial objective will be to train the program to recognize and read Florida license plates on a standard commercial four wheeled vehicle for individuals with no bumper stickers. This will greatly decrease the learning time for the algorithm. Fortunately, each state has standards that license plates must be held to. The usage of these standards would help in refining the algorithm. A stretch goal could be to expand this to vehicles with bumper stickers and plates from other states. The project could also be expanded to plates from other nations, if time permits. Additional stretch goals could include motorcycles, business plates, government plates, or even further, license plates from other countries.

Extra features can be implemented to further enhance data security. In addition to the different power modes, we can consider the implementation of two different modes in regards to the type of data that can be saved as a stretch goal. The first mode is essential mode. This mode will enable CSS to store basic license plate and vehicle information, location, and timestamps needed in text format. The second mode would be an extension of essential mode. This mode will be a high security mode. With high security mode activated, CSS will not only store essential information from essential mode, but it will also capture and store video from the camera.

For users who feel the need to have substantial evidence in the event of an accident, they can use text and video stored on the device. The only downside to the incorporation of high security mode is the demand for storage and excess power consumption for processing and storing footage. Additionally, these modes can also regulate the status of the fan. Depending on the mode selected and the temperature of the device, the decision can be made to have the fan on or off to either conserve battery life or be on to lower temperatures. This was made as a stretch goal, because the current two modes supported would only need to have the fan on (default mode) or off (LPM mode). However, with additional modes of operation, it may be useful to further optimize the usage of the fan to best fit the user's needs in a particular mode.

Another stretch goal would be the full implementation of a software platform to interact with and utilize the CSS. This software platform would go hand in hand with the mobile application to allow a user to access their data from a desktop computer or mobile device. The software will allow them to see a log of all the data they've collected so far, the number of occurrences they've seen a particular license plate, and even a map of the locations in which license plates were detected. Due to the scope of such a platform, this will not be high on the list of priorities as it could potentially result in missed milestones. It will remain in consideration as this would improve a user's experience with the device.

While storing location is not vital to the effectiveness of CSS, it can be beneficial to the user. Time permitting, a GPS module will be installed to capture the driver's current location. The location will be denoted on a set of vehicles that were detected in that area. Although the module does not enhance performance, the information provided by the GPS module will increase data accuracy. By knowing the locations of vehicles that were encountered, the user will be able to narrow down the data set of vehicles.

Neighboring vehicles can surround your vehicle at any given point. Issues may arise if a nearby vehicle is located at a blind spot. A front-facing CSS unit is capable of capturing vehicles that can be seen in a 130 degree field of view. It will not recognize vehicles that lie beyond these constraints. To increase the number of vehicle captures, a front and rear module can be integrated into the vehicle. The 4GB Jetson Nano has the extensibility to utilize two cameras which makes this goal possible.

An additional rear module will allow for extra security as the back of the vehicle is being surveyed. The development of the rear module would be identical to the development of the front. Despite this simple duplication process, we must ensure that both modules are in-sync and are not obtaining duplicate information. Another issue that may arise when designing two modules is the doubling of project costs and limitations on time. Also, not all states in the United States require front mounted license plates. This would diminish the usefulness of the rear camera module and render it to simply capturing footage or vehicle information. One potential use could be face capture, however the privacy concern of this would have to be taken into consideration as well. For these reasons, this idea currently remains a stretch goal if time and budget permits. It was also considered to extend the tracking capabilities of the GPS module by adding in additional metadata. The idea behind this was that an organization with a large premises could position CSS around the premises and label these devices. The meta data would then allow them to track the number of vehicles traveling around the premises, how long they spend in certain locations, and where they go afterwards. Table 5.5 below lists all the possible stretch goals of this project.

List of Stretch Goals	
5.1	Recognizing international license plates and alternate vehicle license plates (motorcycles, etc.)
5.2	Active, online mobile application
5.3	Power Modes: Default and LPM
5.4	Ability to record video in addition to license plate information
5.5	Expanded GPS capabilities such as tracking vehicle locations
5.6	Front and rear module able to survey front and rear of the vehicle
5.7	Ability to store images in conjunction with license plate information

Table 5.5: List of Stretch Goals

6. Project Integration and Testing

6.1. Integration

All portions of the project must now be unified through the process of integration. Both hardware and software must come together in order to provide a functioning CSS unit. In terms of hardware, all electrical components mentioned in the previous sections must be interconnected. Shown below is the integrated system schematic.

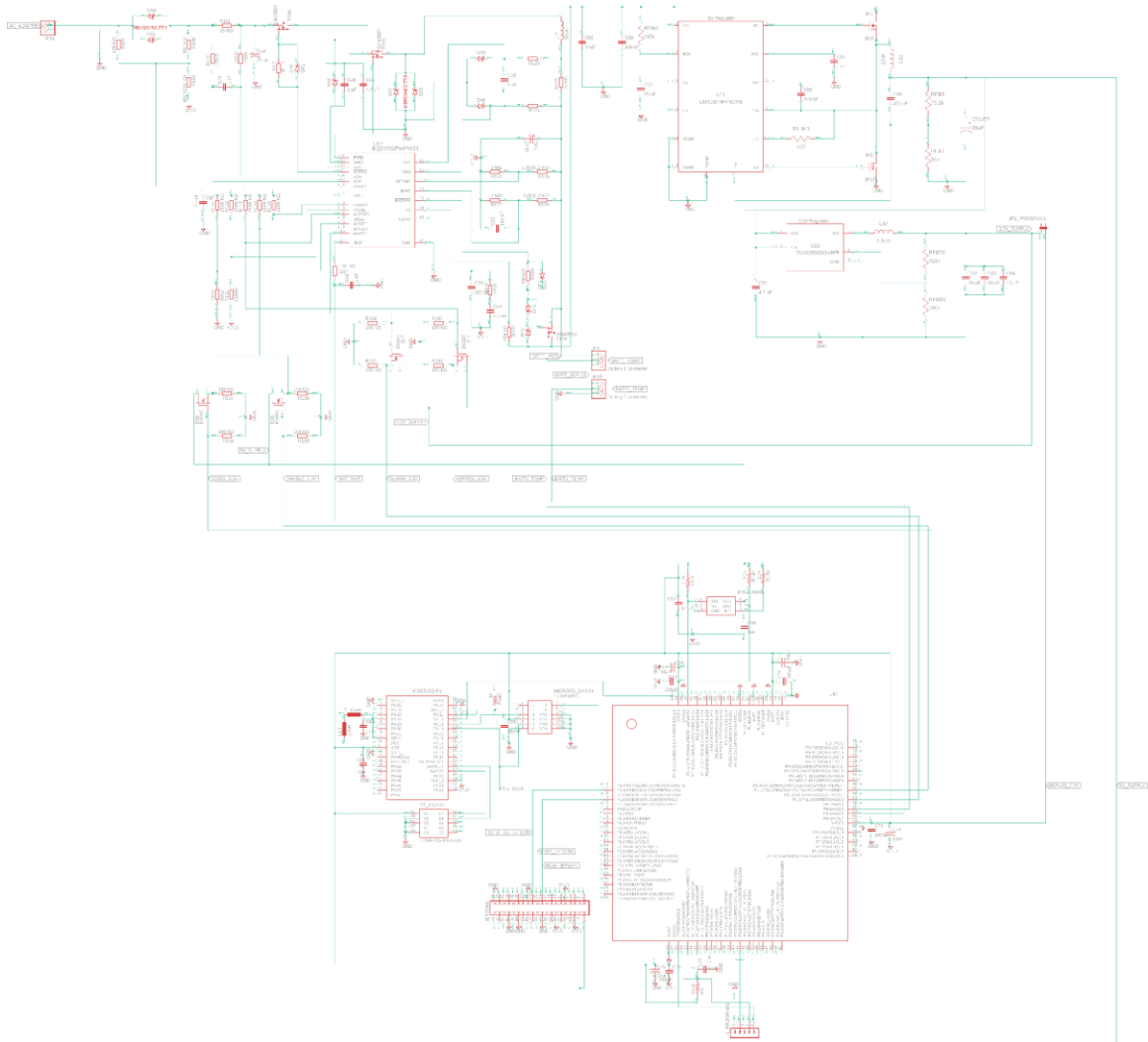


Figure 6.1: Integrated System Schematic

6.1.1. PCB Fabrication

PCB fabrication can be easily achieved by putting in an order with one of several major vendors. These include, but are not limited to, JLCPCB, PCBWay, or PCBgogo. The PCB itself can be designed through using software such as EAGLE or Fusion 360. Once tested and verified, a Gerber file can be generated and provided to the manufacturer. Due to the global situation, the main issue is not with the manufacturer, which is quite quick, but lies with the shipping. Despite a recovering economy, global supply lines are still suffering

from the effects of the pandemic. Some ports are still backed up and container ships are still in high demand. For these purposes, it would be best to get the PCB Design completed as early as possible to submit the purchase order. That way it can arrive as soon as possible from the factory.

6.1.2. Final Coding Plan/Design

The CSS device features multiple unique components that will be interacting with each other. The Jetson Nano will need to communicate with the MSP430FR6989, which will then need to communicate with the different sensors such as the accelerometer, as well as the local storage unit that will be storing the data obtained. The plan is to divide the work up amongst the team to speed up the coding process.

One team will work on the embedded system. This includes the MSP430FR6989 and the attached components. This team will be using C to program the MSP430FR6989 in order to make sure it is able to properly obtain data from the sensors and write data to the local storage unit as well. The communication protocol that will be used is SPI. This is because it is fast and able to interact with multiple components simultaneously. Furthermore, the MSP430FR6989 will need to be able to send and receive data from the Jetson Nano. So, the embedded team will also have to handle this.

A second team will deal with programming the Jetson Nano. This involves development of the computer vision application and configuring the Jetson Nano appropriately. Python is the best solution for programming on the Jetson Nano. This will allow easy use of OpenCV and YoloV5, as well as easily interfacing with the connected camera. Additionally, Python will make communication between the Jetson Nano and the MSP430FR6989 significantly easier as there are available libraries to speed up this process.

With the coding work broken up amongst two separate teams, work on each component can begin independently of one another. Once each team has completed their portion, the two components can be interfaced with one another. At this point the interactions between the two systems should be tested to ensure they are working as intended. Additionally, any final developments and fine tuning of the systems can be finished up.

The final process would involve software testing the code to ensure proper development. Also, that everything is working and running as intended. Software testing will not require a lot of manpower. Therefore, freeing up a team to work on another component while the software undergoes testing. Once the testing is completed and the CSS device is functioning within the given specifications as stated in the design, the coding process is finished.

6.1.3. Prototypes

6.1.3.1. Enclosure Prototype

For visual aid, a first draft of the CSS enclosure was created in Fusion 360. It has overall dimensions exactly matching our final size constraints. It measures in at 4” tall by 4” wide by 5” long at the furthest outside edges of the enclosure. It is a bit boxy but subsequent revisions will involve modifications to achieve our envisioned sleek design. Shown below is the first prototype of the enclosure.

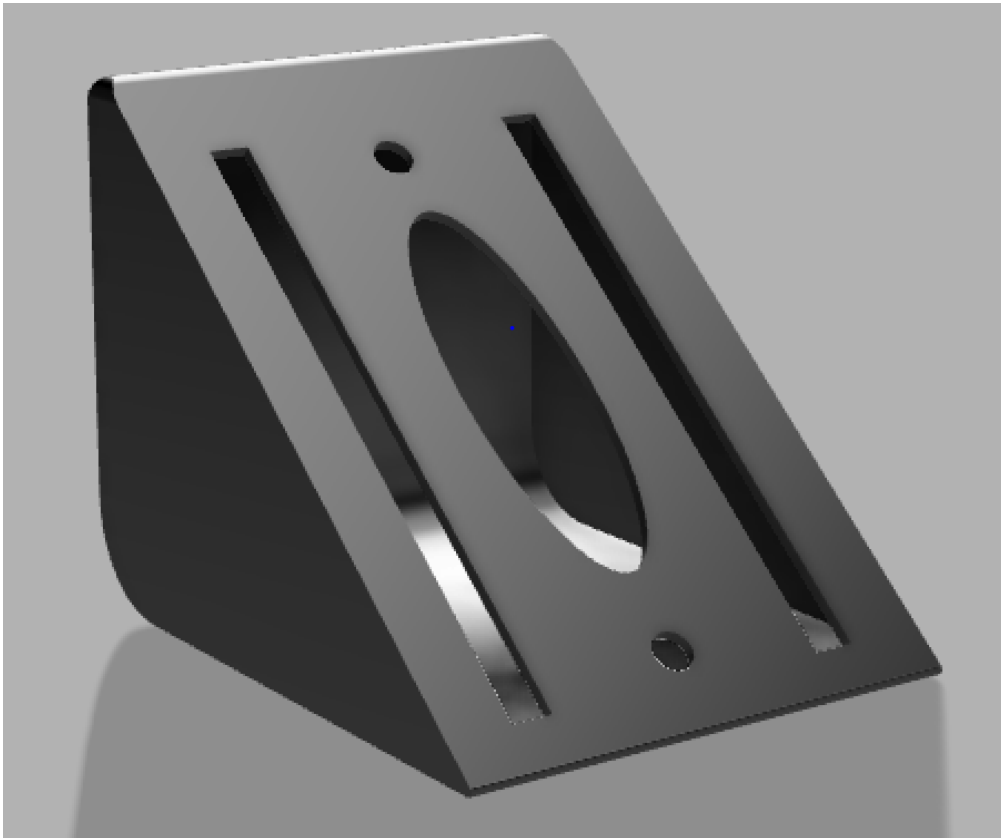


Figure 6.2: 3D Rendering of CSS Enclosure First Draft

The ellipsoidal shape on the front will be flush with the windshield and will serve as the eyelet for the CSS camera. The eyelet was made big enough so that we might adjust the camera position inside the enclosure throughout testing. The rectangular slots are made as a reference to show how we will be designing the rest of the enclosure in terms of ventilation. Likely thinner vents will be strategically positioned on the walls of the enclosure to achieve the best airflow. The two smaller holes on either side of the ellipse are for the large suction cups to be positioned into for adherence to the windshield. These

will be edited into keyhole slots placed so that gravity pulls the suction cup anchors into a locked position. The angle of descent of the windshield plane is currently set at 40°. This was chosen as a generic average but windshield angles will need to be measured and compared for optimum fit.

6.2. Testing Procedures

The section details how the power portion of our circuit will be physically tested as well as testing methods for the sensors.

6.2.1. Breadboard Testing

For prototype and testing purposes, the components can be tested by using a breadboard. Although not as compact or efficient as a PCB, for testing purposes a breadboard will work. This will allow for testing of the components, ensure they are able to interface with one another, and allow for a prototype to be built. PCB design, manufacturing, and shipping, would add significant time and potentially delays to the project deadline. Therefore, a breadboard would be more efficient and allow for concurrent work, design, and testing to ensue. Breadboard testing will be executed based on the schematics and designs further discussed in this section.

6.2.2. Power Testing

The power system is to be tested on the Digilent Analog Discovery 2 by utilizing its oscilloscope and waveform generator capabilities. The waveform generator will be used to superimpose AC noise onto the primary DC input signal in order to simulate AC noise being passed to the power system through a non-ideal AC/DC adapter. An ATX power supply will be utilized to provide high wattage 12 volt and 5 volt input power for the 5 volt and 3.3 volt rails. The output of the 3.3 volt rail and the 5 volt rail will be measured by the oscilloscope of the Discovery 2 under a full design load, and will be required to provide a voltage regulation level such that the output is within no less than 2% minimum allowable input voltage for the least tolerant device on the respective rails. Voltage regulation is determined by measuring output voltage in an open circuit, no load condition and again at a full load condition in which the maximum design load is applied to the respective voltage rail output utilizing resistor loads specified to handle the resulting power dissipation.

For the 5 volt rail, a successful voltage regulation is considered to be within $\pm 3\%$ of 5 volts (4.85 volts) at 3 amps, as demanded by the nVidia Jetson Nano's absolute minimum source voltage requirement of 4.75 volts and the maximum system design power consumption. To test the load capacity, the 5 volt rail will receive an input of 12 volts from the ATX power supply (simulating the AC/DC adapter) and output to a

$$5V / 3A = 1.666 \Omega$$

resistor with a minimum

$$5V * 3A = 15 W$$

power rating as the load.

The 3.3 volt rail is subject to less critical regulation values, due to the wide input voltage range of 2.7 to 3.6 volts for the Pololu 2597 MicroSD card reader. All other components being fed by the 3.3 volt rail have wider input voltage ranges, thus the MicroSD voltage requirements provide the voltage regulation constraint. The 3.3 volt rail, therefore, is to be subjected to a load of the design maximum 0.750 amps and required to provide no less than $\pm 15\%$ of 3.3 volts (2.8 volts). To test the load capacity, the input voltage to the 3.3 volt regulator is supplied by the 5 volt rail from the ATX power supply (simulating the output from the 5 volt regulator) and output to a

$$3.3V / 0.750A = 4.4 \Omega$$

resistor with a minimum

$$3.3V * 0.750A = 2.475 W$$

power rating as the load.

To simulate a full test of the voltage regulation system, the 3.3 volt regulator will be fed by the 5 volt rail, and the 5 volt rail load will be adjusted such that the total output of the 5 volt rail is 15 W while feeding 2.475 W to the 3.3 volt rail. This requires a new 5 volt load to be

$$15W - 2.475W = 12.525 W \Rightarrow (5V)^2 / 12.525W = 1.996 \Omega \approx 2 \Omega$$

and thus the test will be performed on a 2 Ω resistor with a minimum power rating of 13 W.

The testing scheme for the battery pack and charge controller is three-fold: 1) the battery will be connected to the maximum design load so as to draw the maximum necessary current from the battery until the battery is depleted, 2) once the load has depleted the battery the charge controller shall be connected to the AC/DC or DC/DC power adapter and the battery shall be charged to its full charge voltage and remain connected for 3 hours, 3) The AC/DC or DC/DC power adapter shall be removed and the power system shall be connected to the battery with no load for 1 hour, then the maximum design load shall be placed and the battery run back down to depletion. The reasoning for and insights gained by such testing are thus:

Step 1) shall verify the battery pack solution is sufficient to provide the power necessary for the unthrottled operation of the CSS as well as providing an initial verification of both the battery capacity and the operation of the current sense capabilities of the battery charger. The test should be repeated later in the design

phase to verify the current integration operation of the system controller for external state of charge calculation.

Step 2) shall verify source selection operation of the charge controller as well as the charging profile necessary for the Li-Po battery chemistry, the proper charge current, as well as ensuring that voltage regulation is maintained over the long term (~3 hour) to prevent damage to the battery. The test should be repeated later in the design phase to verify the autonomous source selection implemented by the system controller.

Step 3) shall verify the battery pack self-discharge rate as well as the quiescent power consumption of the charge controller, 5 volt regulator, and the 3.3 volt regulator while disconnected from the power adapter and the load. When reconnected to the load, the test shall provide additional data for the charge controller to calculate the maximum battery capacity, as well as a second opportunity for the oscilloscope and later system controller to measure the capacity.

6.2.3. Hardware Testing

This section details how we will test the hardware connections of the CSS. We first go over the MSP430FR6989 testing methodology then follow with the individual sensors. Firmware flashing is referenced here only as it pertains to troubleshooting raw hardware connections.

6.2.3.1. MSP430FR6989 Breadboard Testing

The following details methods to be implemented to test the MSP430FR6989 and all of the necessary sensors. All parts will be connected via breakout boards and tested on breadboards. This is to ensure proper functionality of the hardware connections prior to the final PCB design.

6.2.3.1.1. External MSP430FR6989 Target

For the purposes of hardware testing an unprogrammed MSP430FR6989 will be sacrificed by soldering it to a surface mount to through hole connection converter or Schmartboard. We can then connect jumper wire to break out each individual surface mount pin from the MSP430FR6989. Each desired pin will be connected to a row on a breadboard then connected to the appropriate pin of the sensor/device. A 2 wire JTAG (Spy-Bi-Wire) connection can be made from the eZ-FET emulator on the Launchpad to the external MSP430FR6989 to be programmed. It is important to note that the 3.3V power should come from the eZ-FET side and not the target MSP430FR6989 power supply.

We can begin our testing procedure by flashing a simple LED blink code to the MSP430FR6989 and connecting it to a simple LED on the breadboard. Success will indicate that we not only powered the MSP430 correctly, but the JTAG breakout pins work as desired.

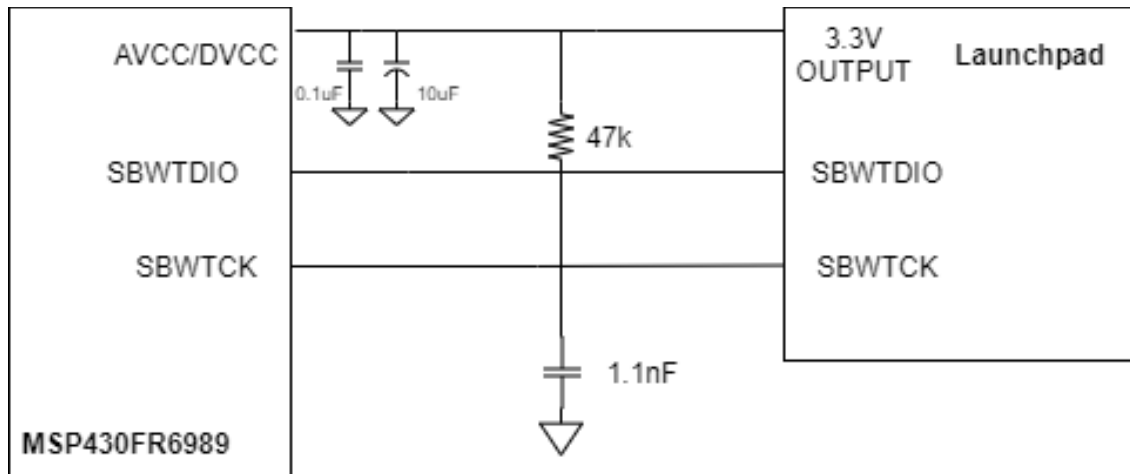


Figure 6.3 SBW Connection for External MSP430 Programming

6.2.3.1.2. Sensor Hardware Testing

Each sensor or device will be primarily tested for functionality on its own via connection on a breadboard. Code can be flashed to the MSP430FR6989 in consideration of the attached device where the part of the final project code pertaining specifically to the device under test will be utilized.

Our testing method for each sensor or device will be threefold. The sensor/device shall be determined as functioning by connecting it individually to the MSP430FR6989 on the Launchpad. The sensor/device shall then be connected to our external MSP430FR6989 circuit via the breadboard and tested to ensure that our hardware configuration for the device is correct. The final test will involve incorporating all external sensors/devices to the external MSP430FR6989 and verifying complete function.

6.2.3.1.3. Accelerometer

For the MEMSIC MXC4005XC accelerometer, a breakout board was obtained that allows for connection to a breadboard. This is crucial for testing as the SMD is far too small to hand solder jumper wire and it allows for us to test the exact same SMD part to be used on the final PCB.

Firstly, the accelerometer breakout board will be connected to the Launchpad. Data output to verify function can be sent to a terminal application like Tera Term on the PC via backchannel UART over USB. Next the accelerometer will be connected to the external MSP430FR6989 on the Schmartboard. Code to obtain axis rotation information will be used and will be crafted to output readings to Tera Term via the backchannel UART system communication setup on the MSP430FR6989 eZ-FET. The Schmartboard MSP430 will have to have RXD and TXD connected to the eZ-FET. The accelerometer breakout board will then be set aside for later full integration testing.

6.2.3.1.4. MicroSD Card

A generic breakout board for a MicroSD connector tray will also be used for testing. It will utilize SPI communication protocol to converse with the MSP430FR6989. The Bluetooth module and MicroSD card can both keep up with clock speeds up to 400 kHz. At data blocks of sizes 512 bytes, this translates to upwards of 10.2 ms per transmission. Though not the greatest, this should be sufficient for the CSS. It should also be noted that we will first need to format the MicroSD card as a FAT32 file system. Header files and functions provided for the MSP430 to SD Card communication will need to be updated to accommodate this.

As before the MicroSD card breakout board will first be tested against a DOA scenario by connecting it to the Launchpad. Code can also be more easily edited in this fashion as well since VCC will not have to be rewired on each flash of the program as would be the case if we were to attempt wiring to our external Schmartboard MSP430FR6989 right out of the gate. Data will be written from the MSP430FR6989 (originally from the Jetson Nano) to the MicroSD card using functions outlined in the TI Application Report slaa281c [17]. The MSP430FR6989 code should verify that the data was written successfully by a ‘read’ command; however this can easily be checked by inserting the MicroSD card into a PC and checking the files. Once it is determined that the testing code is operating sufficiently, we can proceed to our second testing phase of connecting to an external MSP430FR6989. We will examine the output to the MicroSD card as was done in the preliminary test.

6.2.3.1.5. Bluetooth Module Testing

So far, the Raytac MDBT42Q-DB development board has been powered up and examined. The Bluetooth chip soldered onto the board currently is almost exactly the same as the chip to be used on our PCB with the exception of the type of antenna. The already soldered-on Bluetooth chip uses a ceramic antenna on-chip and the chip to be used has a PCB printed antenna also on-chip.

The Bluetooth chip registers on a cell phone but when trying to connect it with a serial terminal mobile app it returns an error “No Serial Profile Found”. Preliminary research suggests that this can be mitigated by updating the Bluetooth profile auto selected in the mobile app to match desired device Bluetooth profile.

After some research and concluding that the target MSP430FR6989 would have 2-wire JTAG breakout pins on the final PCB for programming, it would seem that perhaps we can do something similar for the MDBT42Q-P192 Bluetooth Module. The Raytac “development board”, MDBT42Q-DB, is very simplistic. It’s not very far away from a simple breakout board with minor components and a J-Link header. Looking into the pinout of the 10-pin J-Link it would appear that the J-Link is eerily similar to what we already know as JTAG. Further, there are breakout header pins directly on the Nordic Development Kit, nRF52-DK, that can be used to connect directly to the target MDBT42Q-P192, bypassing the need for the J-Link connector altogether. The pins of course will first be tested on the Raytac Development Board breakout pins to verify functionality before mounting the SMD to the prototype PCB.

6.2.4. Firmware Testing

Our methods for testing the functionality of our firmware code shall include utilizing 2 different development environments. One for the MSP430FR6989, which is TI's Code Composer Studio and one for the Raytac MDBT42Q-P192, which uses Nordic's Software Development Kit (SDK). Though different, both fundamentally use C programming so translation shouldn't be terribly difficult. Program testing for both will be outlined here.

6.2.4.1. MSP430 Code Testing

The MSP430FR6989 Launchpad will be used to test the written code to be flashed to an external MSP430 device. This can be an efficient way to isolate the program from possible hardware connection issues and ensure that the program is fully operational. The Launchpad has 40 header pins that can be utilized by connecting female socket jumper wires to our sensors under test. The sensors will be connected directly or via breadboard if additional connected components are necessary. The code shall be written specifying the necessary pins to be used on the Launchpad output. It has been determined that the code will not need to be changed when interfacing with an individual MSP430FR6989 chip. The same pins can simply be used on the MSP430 to be programmed and soldered to the CSS PCB.

The 40 header pin out on the Launchpad also contains enough pins with the capabilities necessary for our entire project. All necessary sensor connections to the MSP430FR6989, including from the battery charge status in the power circuit, can be connected via the Launchpad pins. In this way the fully functioning code can be tested while minimizing the possibility of hardware errors. This will enable us to finalize a working code and eliminate the firmware as a source of failure.

In order to program the external MSP430 to be used on the PCB, there shall be header pins necessary for the 2-Wire JTAG interface integrated into the final PCB. We will solder the MSP430FR6989 to the PCB without being programmed. Then the JTAG header pins will be connected to the Launchpad eZ-FET to program the external MSP430 while it is already in place on the PCB.

6.2.4.2. Raytac MDBT42Q-P192 nRF52810 Code Testing

Pictured in Figure 6.2 are all of the development boards we have collected so far and will be using for this project. The leftmost board, pictured here for comparison, is the illustrious MSP430 Launchpad we've spoken so much about and are familiar with. In the center is Nordic Semiconductor's nRF52-DK, which is a learning board for Nordic's technology capabilities. One can upload programs and manipulate many different sensors and devices on the board, including a built-in Bluetooth module.

The Bluetooth capability on the development kit can be manipulated to test if the code is in working order prior to testing on an external module. We can do this by programming a serial Bluetooth "Hello World!" transmission and retrieve it on another Bluetooth device

such as a phone. This will further verify that we can successfully send data to our phone terminal.

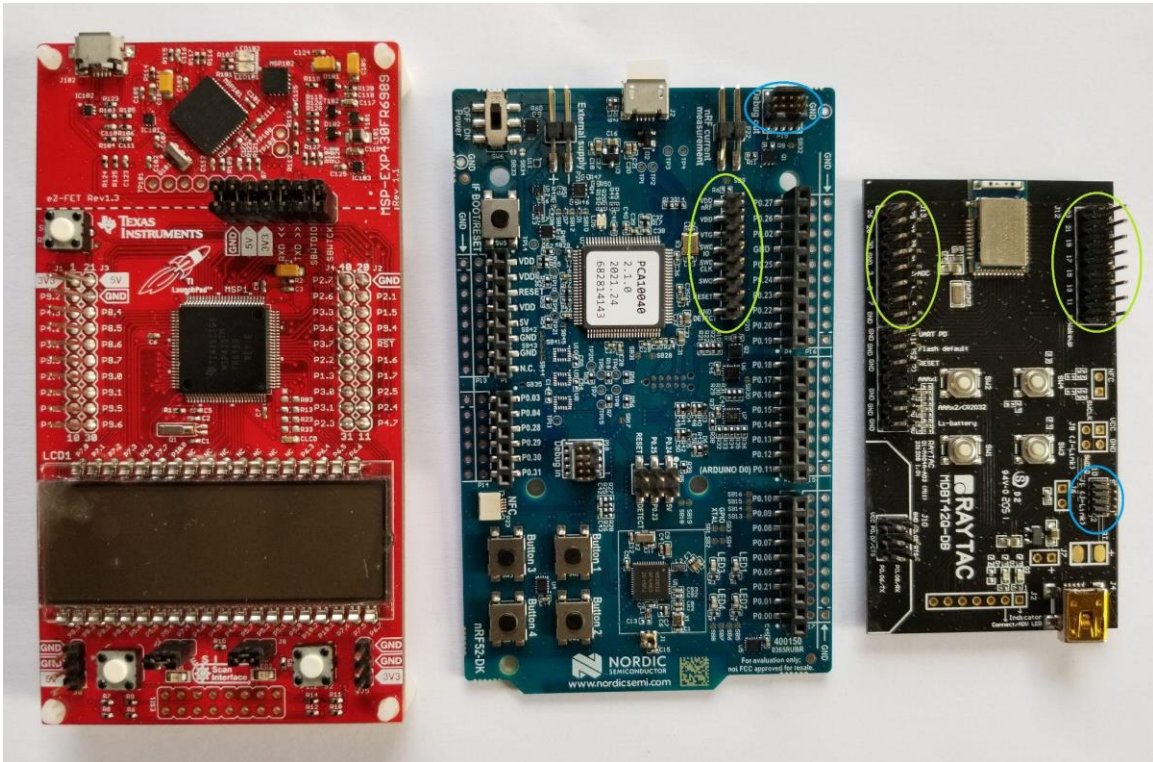


Figure 6.4: Development Boards Reference: Green - Direct Pin Program, Blue - J-Link

The reason we amassed the Nordic development kit was purely so that we could program an external nRF52810 module like our third party Raytac MBDT42Q-P192. Purchasing the Nordic nRF52-DK was hundreds of dollars cheaper than using a SEGGER brand J-Link programmer. A Raytac development board was also obtained so that we could test out our code and hardware connections while taking the guesswork out of soldering straight to our PCB and hoping for the best. The Raytac MBDT42Q-P192 can be programmed pretty easily via a J-Link connection Nordic development kit's SEGGER J-Link debugger and Bluetooth development software in Nordic SDK. It is difficult to separate firmware troubleshooting and hardware troubleshooting as they are so intertwined. Once a working sample code is established and data can be sent successfully via Bluetooth, we can conclude that it is possible and favorable to put a J-Link breakout header similarly on the final CSS PCB. The Bluetooth module on the CSS PCB will then be programmed over J-Link via the Nordic nRF52-DK.

Similarly to the MSP430FR6989, the Raytac Bluetooth module can be flashed with test code obtained from the Nordic DevZone and modified to our needs. Our second test will be to connect and program the Raytac Development Board using either the "Debug Out" jumper (J19 - Blue circles) on the Nordic DK or the 12 pin header (P20 - Green circles). We will start by attaching an LED to one of the module pins on the Raytac Development Board. We can flash a preliminary "Blink" code and test that we have a successful setup and hardware connections for programming from the Nordic nRF52 Development kit.

Then we will program the Raytac module with the final code and remove it from the Nordic Development Kit.

As with other sensors/devices we will first connect the Raytac development board to the Launchpad individually and attempt to send data via SPI and then retrieve it via the mobile application. Once this is successful, we will move onto connecting to and communicating with the external MSP430FR6989. After this success, we can then begin to practice sending ASCII code data from the MSP430 to the MDBT42 to simulate license plate data. Then we will attempt to retrieve the information via Bluetooth.

So far it seems that the “Serial Bluetooth Terminal” mobile app will be quite useful in initial testing to verify the Bluetooth is able to send data. Commands can be sent directly to the MDBT42Q-P192 from the terminal open in the mobile application. This can be used to mimic the eventual goal of the CSS mobile app obtaining data from the MSP430FR6989 by sending a request for and receiving the license plate information.

6.2.4.2. Bluetooth/MicroSD Troubleshooting

Unlike in previous tests, we will lastly join the MDBT42Q-P192 and the MicroSD card into their final circuit with the MSP430FR6989 and test for data output using the same previously mentioned configuration as shown in Figure 5.19. This is to ensure that the daisy chain setup is working properly. We know that the daisy chain configuration won't be easily compatible with the commands needed to initialize the MicroSD file writing. The order of the Bluetooth and MicroSD devices may be reversed in the case of total failure. The MSP430 can mount, write to, and read the MicroSD card pretty straightforwardly on its own. The Bluetooth device may need to be programmed to sample data only after certain commands to the MicroSD card are completed.

6.2.5. Software Testing

The main software component that will require testing, is the computer vision application itself. In order to ensure that it hits the targeted accuracy, both the actual license plate information and the obtained license plate information from the CSS will have to be recorded. To do this manually would be a laborious and lengthy task.

This process can instead be sped up by setting up an automated testing rig. The CSS device will be set in front of a monitor in which license plate information will be displayed. License plate information for testing purposes may be collected manually. However, this would also be slow and time intensive. Instead, license plate information may be freely obtained online at Kaggle.com. This website provides numerous datasets for testing purposes. These license plate data can be displayed on the monitor.

With everything in place, it will be a simple matter of recording the accuracy of the characters obtained by the CSS device against what is being displayed. This may have to be done manually to ensure that the testing is done correctly. Alternatively, datasets from Kaggle.com may come with the license plate information prerecorded. Which would

significantly speed up testing as the pre recorded information can be easily compared against the information obtained from the CSS device.

Alternatively, the CSS can be taken out to a parking lot and simply go up and down each lane recording data and verifying whether it is correct. This method is not as ideal as it is more time intensive and requires creating a portable testing rig. Therefore, this will be the backup alternative in the event that there is not enough data provided from Kaggle.com.

Once the CSS has been tested, the accuracy can be recorded. The goal is to obtain the highest accuracy possible with an average accuracy of 90% or greater. In a real-life scenario, an incorrect license would at best be a waste of time and at worst, incriminate the wrong person. Therefore, a high accuracy rating needs to be achieved. Through testing, the accuracy can then be optimized and also fine tuned for different scenarios such as low light scenarios or far away distances.

Another software component or rather components, that needs to be tested, is the flow of the application. That is, after being turned on, does the software automatically and correctly follow the intended flow as depicted in figure 5.9. As can be seen, when the device is powered on, the MSP430FR6989 will alert the SBC as to which state it should be in. The SBC will enter the appropriate state. Then at the right time, scan and record the data.

There are several ways in which this can be tested. The first is by defining a preset flow that the program should be following given a specific scenario. Turn on the CSS and see if the final output matches what is expected. For instance, defining a scenario in which the CSS should be in low power mode and not turn on until a jolt is detected. If not jolt, does the CSS record anything? Check the data storage unit to determine if any was recorded, if nothing was recorded given the scenario, the CSS has behaved correctly. This would require numerous scenarios in order to check that each software component is functioning as intended. Therefore, it may be better to devise a more efficient manner.

The more efficient manner would be the use of the troubleshooting LEDs. These LEDs could be used to specify which state the CSS is currently in. Therefore, as it goes about executing the flow of the program, the appropriate LEDs should light up and indicate in which state the CSS is in. This is a simpler and faster approach to testing the various software components of the CSS. Since this would be able to indicate all at once whether the device is running as intended.

Additional software features that will need to be tested would be data recording onto the local storage unit. This would be a simple matter of taking out the local storage unit and verifying the data written to it. The communication between the SBC and the MSP430FR6989 will also have to be tested to ensure the correct transfer of data. This will have to be done during the development process. As it would be difficult to develop the rest of the system without knowing whether or not the MSP430FR6989 and the SBC are communicating correctly.

To do this, debugging measures can be implemented on the software side. Such as outputting test data before sending it between the two systems. Then output the data received to the terminal. In this manner, it can be determined whether the intended message was received correctly or not. This will make it so that it is easy to verify the data and proceed forward with the rest of development.

Implementing all these different testing measures, will ensure that the software side of the system is working as intended. This will reduce the length of time required to debug the system as well, by following the defined testing measures. Additionally, it will allow optimization of the system to achieve the desired goals.

6.2.6. Survivability Testing

6.2.6.1. Collision Testing

To ensure the CSS is able to survive the forces involved in an average car crash, those forces will be simulated by a drop test. The drop test is also a chance to test the accuracy and communication of the accelerometer, as testing this by driving in a vehicle would only simulate pre-crash conditions.

Assuming a severe rear end impact at highway speeds, the speed differential between the two vehicles is to be approximated as 30 MPH. A car crash can be considered an inelastic collision, and assuming a worst-case scenario where the victim mid-size vehicle was rear-ended by a fully laden over-the-road tractor trailer, the victim would be accelerated to nearly the velocity of the striking car. According to the NHTSA study on impact pulse widths [18], the time delta of an impact was between 77 and 109 mS. Applying Newton's Second Law, $a = \frac{dV}{dT}$ where dV is the differential in victim vehicle initial speed and victim vehicle post collision speed, and dT is the lower range of NHTSA measured impact time of 77 mS. This calculation presents a collision acceleration of

$$a = \frac{13.4112 \frac{m}{s}}{77 * 10^{-3} s} = 174.17 \frac{m}{s^2}$$

To simulate such an impact on the CSS, a drop test is designed utilizing data provided by PCB Piezotronics [19] wherein dense foam is used as the impact surface to provide an impact pulse width of 100 mS. The assumption is made that the impact on the plastic surface is elastic, such that the velocity after the impact is equal to the velocity before the impact albeit in the opposite direction. These parameters allow for the calculation of impact height from the second law in the form

$$a = \frac{2 * g * h \sqrt{2}}{t_{pulse}} \gg \frac{a * t_{pulse}}{2 * g \sqrt{2}} = h \gg \frac{174.14 * 100 * 10^{-3}}{2 * 9.792 * \sqrt{2}} = 0.6288 \text{ meters}$$

Thus, a drop test onto dense foam from a height of 62.88 cm shall simulate the acceleration the CSS would undergo if the host vehicle were hit from behind by a fully laden tractor-trailer travelling 30 MPH faster than the host vehicle.

The chosen accelerometer has a maximum detection value of 8g or $78.34 \frac{m}{s^2}$, so the test shall first be run at a height of 25.274 cm (derived from the same equation as above) to observe the test fixture and accelerometer operations at less than saturation conditions.

Once both have been verified, the test shall be run at its maximum height with a sparsely populated board of only the accelerometer, which shall be connected to power and communication lines via a cable to the MSP430 launchpad. Once the sparsely populated board has passed the drop test to verify PCB substrate and solder joint integrity, the test shall be rerun from the maximum height using a fully assembled CSS control board and battery packs. nVidia provides information on the survivability of the Jetson Nano [21] which states that the module is tested to survive impacts of 50g or $489.6 \frac{m}{s^2}$ which far exceeds the impact constraints of the CSS and therefore is excluded from drop testing.

6.2.6.2. Heat Testing

As stated in the constraints section, the CSS must survive regular exposure to temperatures of 150 °F (65 °C). Full capability HALT or HASS testing equipment is not available to the CSS team for testing, but a conventional toaster oven can be made to operate as a test chamber. A thermocouple input to the MSP430 Launchpad shall be used to control a 120 VAC relay to provide power to the oven when temperature reaches the minimum allowed, as programmed into the MSP430.

A clay brick can be used as a temperature buffer by absorbing large amounts of heat energy when the oven is powered and releasing that energy to maintain the temperature when the power is off. A clay brick has an average mass of 2.27 Kg and a specific heat of approximately $1 \frac{KJ}{Kg K}$ would allow for 2.27 KJ of energy leakage from the oven while before the local temperature in the oven would decrease by 1°C. Utilizing this test fixture would allow for long term testing by applying the Arrhenius equation for reliability as stated by JEDEC [20] to test the CSS at increased temperature to simulate accelerated lifespan of the device.

Utilizing an Apparent Activation Energy for the electronic components of $0.7 \frac{eV}{K}$ as stated by Wyrwas and Bernstein [22], the Arrhenius acceleration factor is calculated as

$$A_t = e^{\left(\frac{-E_{aa}}{k}\right)\left(\frac{1}{T_1} - \frac{1}{T_2}\right)}$$

where E_{aa} is the Apparent Activation Energy, k is the Boltzmann constant, T_1 is the temperature of the test fixture in Kelvin, and T_2 is the temperature of the system under normal operation. This gives an acceleration factor of

$$A_t = e^{\left(\frac{-0.7}{8.62 \times 10^{-5}}\right)\left(\frac{1}{150+273.15} - \frac{1}{65+273.15}\right)} = 124.46$$

which means every hour tested at 150 °C (easily achieved by any easily procurable toaster oven) simulates 124.46 hours at 65 °C. Assuming that the system temperature is only the extreme value considered in the constraints (65 °C) for 6 hours or one quarter of a day, the accelerated test can simulate

$$\frac{A_t}{6} = 20.744 \text{ days}$$

for each hour in the test fixture, meaning a reasonable 24-hour test simulates 1.36 years of real world operation.

6.3. Project Operation

With all the technicalities established, we will discuss abilities and features that the user will be able to utilize. CSS will feature plug and play functionality to give users the ability to physically move the entire unit and make needed adjustments. There will be a set of modes of operation to give users the option to use CSS at maximum power or conserve energy when it is not being actively used. The user can be notified with certain indicators placed on the unit to identify and troubleshoot issues that may arise. Data obtained by the system will then be accessible via the local storage unit and mobile application.

6.3.1. Plug and Play Functionality

A great feature of dash cams is the plug and play functionality. Out of the box, all it needs is an SD card and once plugged in and set up, it is immediately recording. A user does not have to play with any settings unless they want to. CSS will also incorporate plug and play functionality to make it as simple and convenient as possible for the user to use the device. Once powered on and with an SD card slotted, it will automatically begin running in the default mode or whatever mode it was last set to. A user can change settings if they so desire but will not be required to do so in order to use the device.

6.3.2. Modes of Operation

The battery life of the device can be further extended by programming additional modes of operation into it, so that a user can select the mode that best fits their needs. The default mode would have all components active and the device is scanning and logging the information of all vehicles with a license plate that comes into view. This would be detrimental to battery life, however, it would provide the user with the most amount of information.

The other mode would be a low-power mode (LPM) that would turn off all unnecessary components and only have the accelerometer and relevant components on. Then, when a jolt or collision-like movement is detected, the device will power up all components and begin recording data for several minutes before returning to LPM. In this manner it will increase battery life and reduce storage consumption even more. The circumstances in

which to wake up the device can be determined later in testing. Shown below is a comparison table between the default and low power mode.

<u>Default Mode</u>	<u>Low Power Mode</u>
Components that Stay On the Entire Time	
<ul style="list-style-type: none"> ● Project Controller ● Bluetooth ● Camera ● Single Board Computer ● GPS ● Accelerometer 	<ul style="list-style-type: none"> ● Accelerometer
Components that Turn/Stay On When a Collision is Detected	
<ul style="list-style-type: none"> ● Project Controller ● Bluetooth ● Camera ● Single Board Computer ● GPS ● Accelerometer 	<ul style="list-style-type: none"> ● Project Controller ● Bluetooth ● Camera ● Single Board Computer ● GPS ● Accelerometer

Table 4.5: Comparison between default mode and LPM

6.3.3. Troubleshooting

Additionally, troubleshooting light indicators will be included on the device to assist the user in determining what the issue could be if an issue arises. Such as an indicator that lights up if storage is full or in the wrong format. Another indicator for battery status. These indicators will aid the user in quickly debugging the issue so that the device can go back to being used. They will also be helpful for testing the device.

The device is equipped with both high capacity on board storage as well as wireless communication in the form of Bluetooth, therefore more in depth but less essential troubleshooting or device health data can be provided to the user. Basic battery statistics useful for long term use such as number of charging cycles, maximum capacity degradation, temperature, critical temperature events, and remaining running time would be useful but infrequently accessed data for the user. Since the prominent online SoC measurement method will constantly measure the current entering and leaving the battery, the troubleshooting features are relatively simply implemented in the charge controller firmware as functions performed with the SoC measurement as input. The charge controller shall have non-volatile memory with enough capacity to store these values over the life of the device, and thus the integrity of the measurements can be retained in the event of off-

controller storage change, such as formatting of the local storage. In the interest of power savings and extending battery life, the device health data will remain in on-controller non-volatile memory and a report shall only be generated and distributed on user request.

6.3.4. Data Access

There will be two ways to access information collected by CSS. The most basic way to access data is through the SD card. The SD card on the project controller can be removed by the user and inserted into their own computing device. From there, a file will be available for them to see license plate information and vehicle attributes. The output file containing all the collected data will be a basic text file to optimize storage space.

As previously mentioned, this can later be expanded upon. In the event that additional time is available and stretch goals are completed. In the that images of videos are added to the capabilities of the device, a basic file system will be implemented. One way this could be accomplished is by having a folder with the license plate information and associated images or videos inside. However, a downside of this is that there would quickly be a plethora of folders which would make it difficult for the user to identify and extract data. An alternative would be to simply have the user select which option they want, text, image, or video, and then name the associated file with the license plate information. The text file will contain data on the vehicle such as make and model, while the image and video won't, as the user can identify this manually. As time permits more advanced methods could be implemented so that all the relevant data a user would want to be seen can be accessed from a GUI (graphical user interface) so that a user would not have to worry about file structure.

Another method of accessing data is utilizing the mobile application. The mobile application will be available on any handheld touchscreen device. When the user enters the application and logs in, the application will automatically request to update the database with newly collected data. If no new data is collected, the database will maintain its current state. If a new entry is detected, then the entries shown on the mobile application will be updated with new entries at the top of the table. This method of accessing data will provide more convenience due to the clean layout of the design.

7. Project Administration

This section discusses the organization and management of this project. A budget analysis was done to consolidate all components and item costs. This included a projected cost prior to component selection and the actual cost. The actual cost specified specific items, their unit price, quantities obtained, and overall total cost. To reduce out-of-pocket costs, the team has researched the necessary qualifications for the IEEE Student Grant.

With all the components and costs covered, the work distribution has been declared. Tasks would be delegated based on each person's discipline and capabilities to increase efficiency and reduce stress. These tasks are then to be completed based on the timeline established in the milestones section.

7.1. IEEE Student Grant Eligibility

One of our goals is to qualify for the IEEE Student Project Grant. To be eligible we must implement one or more IEEE standards with an IEEE Standard as the primary standard sought. The closest applicable standard found was IEEE 1725-2021: Standard For Rechargeable Batteries for Mobile Phones. This standard also specifies the use of rechargeable Lithium Ion batteries for a "host device" other than a mobile phone. In our case this host device would be the single board computer and/or the project controller.

In Figure 7.1, the interoperability of different parts of the power circuit are shown. The battery pack communication methods and electronic and mechanical protections must be developed. This will lead to requirements of the charge controller. Next the host device needs to be considered as far as it's requirements for software interfacing and power constraints. Then the adapter components, such as AC/DC and DC/DC conversion, shall be designed. The user notification system will either consist of indicator LEDs or notification messages on the stretch goal of the OLED screen.

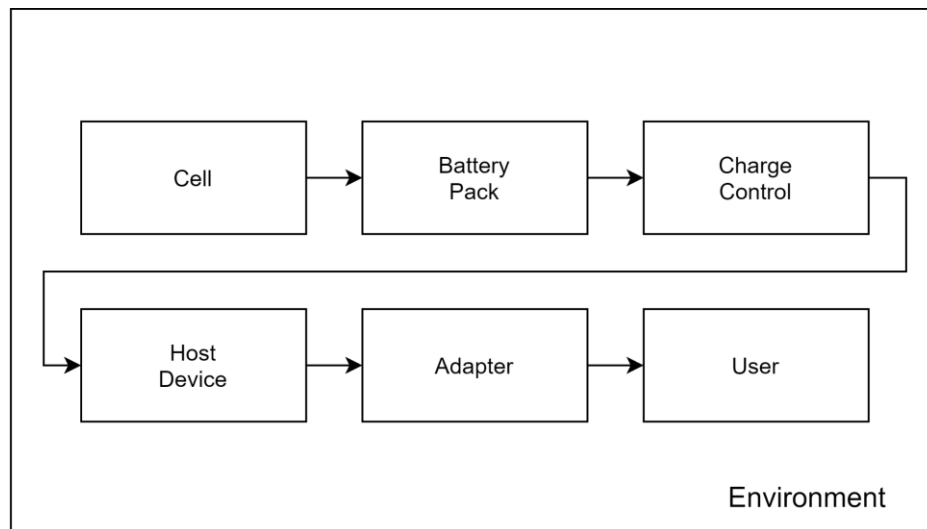


Figure 7.1: Achieving IEEE 1725-2021 Standards

One major limitation for the CSS, as previously stated, is the environment. It encompasses all other components for this reason. The battery system of our device needs to be capable of withstanding and operating in temperatures in excess of 150°F. Cooling methods may be further researched. The user of the device will receive indicator messages on battery state and/or warnings.

The adapter, host device protections, and charge controller intends to be designed and must adhere to subsystem standards cited in IEEE 1725-2021. The battery pack may be designed using one or more cells that will be purchased. The subsystem standards must be followed in order to qualify under the 1725-2021 standard. The requirements are listed below in Table 7.1.

System Component	Standard
AC/DC adapter or charger	IEC/UL 60950-1 or IEC/UL 62368-1 and IEC 62368-3 or appropriate safety standard of destination country
DC/DC adapter or charger	IEC/UL 60950-1 or IEC/UL 62368-1 and IEC 62368-3 or appropriate safety standard of destination country
Battery pack	UN Manual of Tests and Criteria, Section 38.3 Lithium Batteries and appropriate safety standard of destination country
Cell	UN Manual of Tests and Criteria and appropriate safety standard of destination country

Table 7.1: Minimum Subsystem Requirements for IEEE 1725-2021 Standard

Upon further investigation it was unfortunately determined that our constraints for the CSS rendered us ineligible for the IEEE Student Grant. Our main complication is the environmental limitation of the battery needing to be able to survive and operate in extreme temperatures. The IEEE Standard 1725-2021 specifically states using a Lithium Ion battery. Our analysis led us to decide on Lithium Polymer batteries because of their ability to fare better in high temperatures.

The IEEE Student Grant is geared more towards the design of the individual parts themselves. If our team was designing just a power supply, or other related project, we might be able to fit within the requirements of an IEEE standard or working group. For our overall purposes, the micro-level design of components is outside the scope of our project. The student grant requires that an IEEE standard be the main focus of the project and we were looking to apply it to only a part of our project. Though we may have components used in our design that meet IEEE standards we are not designing those parts.

7.2. Budget Analysis

7.2.1. Projected Cost

The budget will be self-funded with the anticipated assistance of the IEEE Student Grant. Based on current estimates, we may have an excess amount of funds considering the lower bound of the cost estimate. Excess funds will allow for further enhancements, cover the costs of broken/malfunctioning parts, or allow for duplicate purchases in case a part fails. Shown below are the estimated calculations and overall budget for one-time purchasing.

Item	Quantity Estimate	Cost Estimate
Camera	1	~\$5 - \$30
Local Storage (circuit)	2	~\$20
Bluetooth transceiver	2	~\$40
Battery	1	~\$80
Battery Regulator	2	~\$30 - \$40
Accelerometer	2	~\$10 - \$40
GPS Module	2	~\$10 - \$60
Printed Circuit Board	4	\$50
Case	1	\$22
Single Board Computer	2	\$125
Project Controller	2	\$30
Total Estimated Cost		\$422 - \$537

Funding	Assistance
IEEE Student Grant	- \$500

Estimated Credit/Debit	+\$78/- \$37
-------------------------------	--------------

Table 7.2: Project Purchasing and Financing

7.2.2. Actual Cost

Necessary components for the project have been quantified and purchased. In comparison to the overall projected cost, the actual cost falls within the projected cost range. Shown below is a table of all the components purchased and the actual cost of the project.

Item	Price	Quantity	Shipping Cost	FL Tax	Total
MSP430FR6989	\$10.10	2	-	1.07	\$29.50
NVIDIA Jetson Nano	\$59.00	1	-	1.065	\$62.84
5V 4A Power Supply	\$12.59	2	-	1.065	\$26.82
2.5 to 2.1mm Adapter	\$8.96	2	-	1.065	\$19.08
4GB NVIDIA Jetson Nano	\$169.95	1	-	1.065	\$181.00
GY-521 MPU-6050 MPU6050 Module 3 Axis analog gyro sensors	\$2.64	1	-	1.00	\$2.64
WAVGAT Micro SD Storage Expansion Board	\$1.83	1	-	1.00	\$1.83
Accelerometer & uSD expansion	\$3.25	1	-	1.00	\$3.25
10000mAh LiPo Batteries	\$12.00	2	-	1.00	\$24.00
Cooling Unit	\$24.38	2	\$1.50	1.00	\$50.26
uSD Module	\$0.20	1	\$1.80	1.00	\$2.00
GPS Module	\$2.60	1	\$1.65	1.00	\$4.25
Camera Module	\$19.90	1	-	1.07	\$21.29
Accelerometers	\$1.49	2	-	1.07	\$3.19
Mouser Shipping	\$7.99	1	-	1.00	\$7.99
10000mAh LiPo Battery	\$13.82	2	\$0.99	1.07	\$30.56
BL651/BL652 Breakout PCB	\$8.00	2	\$9.98	1.065	\$27.02

Grand Total	\$497.52
--------------------	-----------------

Table 7.3: Actual Cost

7.3. Work Distribution

To complete this project in a timely manner, each team member must be assigned tasks. Tasks for this project have been distributed based on each person’s discipline and strengths. Those with electrical engineering as their major will oversee tasks related to schematic design and integration of electrical components. Others with a focus in computer engineering will conduct software-based tasks.

All members will contribute to the research portion of this project. After the research portion, the build tasks of the project will be dispersed. Robert and Ari will conduct tasks related to electrical components of the unit. Qrizelle and Ricardo will handle software-related tasks. When putting all components together in the enclosure, all team members will participate in the integration of the CSS unit. Shown below is a general work distribution table for both Senior Design I and Senior Design II.

Tasks	Task Distribution (Senior Design I)	Task Distribution (Senior Design II)
Research	All members	-
Power Schematics/Testing/Integration	Robert	Ari, Robert
MCU & Peripherals Schematic/Testing/Integration	Ari	Ari, Robert
PCB Fabrication/Implementation	Robert, Ari	All team members
Firmware: MSP430	Ari	All team members
Software: Mobile	Qrizelle	Qrizelle, Ricardo
Software: Computer Vision (Jetson)	Qrizelle, Ricardo	Qrizelle, Ricardo
Soldering/Connections	-	All members
Enclosure	-	All members

Table 7.4: Work Distribution

7.4. Milestones

The milestone dates were established based on the required tasks for Senior Design I and Senior Design II. The dates listed were used as “targets” to properly pace ourselves and plan accordingly. Senior Design I was primarily focused on project research and system testing. Table 7.3 shows the timeline of Senior Design I.

Senior Design I	
Objective	Time - Date of Completion
September	
Initial D&C (<= 10 pages)	12:00 PM - 09/17/21
October	
Updated D&C (20-25 pages)	12:00 PM - 10/01/21
Order Parts/Resources	By 10/04/21
Acquire Parts/Resources	11:30 AM - 10/18/21
Develop/Test Version 1 of Software	10/18/21
November	
Individual Part Testing (Breadboard)	10/18/21 - 11/05/21
Senior Design I Report (60 Page Draft)	12:00 PM - 11/05/21
IEEE Application	11/08/21
Develop/Test Version 2 of Software	11/10/21
Prototype #1 Designed	11/12/21
Senior Design I Report (100 Page Draft)	12:00 PM - 11/19/21
December	
Prototype #1 Testing	11/19/21 - 12/03/21
Dummy Prototype Crash Testing	11/19/21 - 12/03/21
Final Senior Design I Report	12:00 PM - 12/07/21

Table 7.5: Project Timeline of Fall 2021

Senior Design II requires overall system integration and construction. In the second half of the course, the software must be refined, and the PCB must be in development. With these stages completed, a working prototype must be ready to be presented to the public. A set of projected milestones and their corresponding dates have been listed in the table below.

Senior Design II	
Objective	Time - Date of Completion
January	
Develop/Test Version 3 of Software	01/03/22
February	
PCB Design Testing Done	02/19/22
PCB Ready To Ship	02/21/22
March	
PCB Delivery	03/25/22
Begin Solder Process	03/28/22
April	
Have Working PCB	04/08/22
Final Testing/Revisions	04/01/22 - END
Peer Presentation	TBD
Final Report	TBD
Final Presentation	TBD

Table 7.6: Project Timeline of Spring 2022

8. Final Thoughts

Thorough research was conducted on every part to be integrated into the final design. For the single board computer, comparisons were made on four different models and detailed reasoning for the selection of the Nvidia Jetson Nano was provided. Four different microcontrollers were analyzed, and the CSS team chose the TI MSP430FR6989 based on robustness and peripheral capabilities. Choice of sensors were heavily debated and chosen based on cost and compatibility with the microcontroller. The power system is the cornerstone of the project. Many different battery compositions and regulatory devices were compared and evaluated to achieve the required heat tolerance and electrical specifications. A thorough circuit design was presented which centered around a lithium polymer battery with enough capacity to power the device for an acceptable amount of time between charges. Extensive work went into researching six different computer vision solutions, of which OpenCV coupled with YOLOv5 was the final choice. This was because of the extensive library, documentation, and community support, a byproduct of being open source.

Project standards and constraints were identified to establish workflow and goals for the team. This led to hardware subsystem design including the power system, choice of microcontroller, and configuration of sensors. This further dictated the design of the mobile application, which will communicate with the hardware. The requirements of the computer vision software enabled the team to make an informed decision on the single board computer. Communication methods for all major components were presented in detail. Stretch goals were also identified and evaluated in terms of feasibility within our time constraint.

All preliminary hardware block schematics were incorporated into one overall layout to provide a view of how the design will be laid out. This will be the guide to creating our final PCB design. Programming and testing will occur independently at first and as a team in overall project integration. Guidelines were established for testing of the power circuit and sensors, as well as testing for the firmware and software. Overall project survivability in the relentless Florida weather and in the event of the crash will be evaluated. This is paramount to final product functionality.

Finally, the CSS team presented research into eligibility for the IEEE Student Grant, a detailed budget analysis, work distribution table, and projected milestones to be met throughout the project. It was unfortunately determined that the project does not meet the criteria needed to qualify for the IEEE Student Grant. Our design will include parts that meet IEEE standards, but we will not be designing a system to be qualified as passing a particular standard as desired for the grant. An estimated and final budget was presented for all major components necessary for project completion. So far, the final expenditures are actually within range of our estimate considering the absence of the IEEE grant.

A major concern and challenge at the onset of the project was the scope and legality of the project. The inspiration for the creation of the Car Sentry System arose from the desire to help victims in hit and run situations to be able to identify the perpetrator. This would be

achieved by creating an affordable low-profile device that users can place on the dash of their vehicle and forget about it until the need to recharge the battery or access the data is necessary. After conception many great ideas on how to expand the abilities of the device were brought up, such as scanning license plate information for other countries, storing images and videos, and adding location data to license plate information obtained. Additionally, the idea to add location tracking and analysis was also brought up, which would be useful for businesses and organizations that are interested in tracking the whereabouts of vehicles within the premises. All of which are practical applications for the device that can easily be expanded upon, however the major constraint is time.

Therefore, the team reached the consensus that, in order to prevent feature creep and missing deadlines, a modular foundation would be developed. Having this flexibility won't only be limited to the hardware components of the project but will also apply to the software aspect as well. The intention is to create a code base that reflects the same modular attribute as the hardware. This would be done by creating genericized interactions, such as APIs (Application Programming Interface), between software components so that they can remain modular and be reused. So that in the event of achieving deadlines earlier than expected, additional functionality could be built on top of it to expand the capabilities of the device. This led to the decision to keep the CSS primarily geared towards the individual and making it as affordable as possible. Since this provides a foundation that can be built on top of.

Also, throughout the project, the question of, "is this legal?" was brought up several times. Is it legal to automatically obtain and store a person's license plate information? What about associating additional data such as car brand and model with it or even location data? The conclusion the team reached was that although the possibility of a retail outlook for such a device was vague without a lawyer, the applications and benefits of the device were worth developing and exploring as it has more than just one application. This further reinforced the idea of creating a core product that could be expanded upon in the future. The intention is to create a code base that reflects the same modular attribute as the hardware. This adds versatility to the project by adding the ability to shift the focus and end goal of the product if necessary.

The decision to create a modular core has so far been a great idea and has already shown benefits. As previously mentioned, the main intention of this was to prevent feature creep, while still giving the team the ability to expand the functionality of the device through stretch goals. One of these stretch goals was originally the mobile component. During the technology investigation, the discovery of PWAs led to the decision to include the mobile component as a feature in the final product.

It should also be noted that, it was also greatly beneficial to have team members with a diverse set of experiences as each team member was able to identify parts of the project in which they wanted to focus on. This made the division of labor very easy and reduced the amount of time needed for project management. There was also overlap in skills and experiences between members as well. Which was reassuring since in the event that any member ran into a problem, they would not have to be alone in finding a solution.

As the end of the semester draws near, the team feels confident in the amount of effort put into the investigation of the technologies and components necessary to go ahead and develop the CSS. As previously mentioned, due to the global economic situation with uncertainty with regards to obtaining parts, several components have already been purchased and are currently in possession. These include the Jetson Nano for the single board computer and the MSP430FR6989 microcontroller as well as various sensors such as the camera and accelerometer. This has allowed the team to familiarize themselves with the components and perform more thorough investigation components.

Not only have the parts been obtained, but a majority of the design work has been completed. This includes the power system, the integration of the MSP430FR6989 with the various sensors and the single board computer, the mobile application, and the computer vision application. This allows the team to immediately begin working on the development at the beginning of the next semester. We all have our work cut out for us next semester, but the division of labor will enable the hardware/firmware team to work concurrently with the software/Jetson team. If the CSS team, at the very least, adheres to the milestone deadlines delineated in this report, a successful working prototype will be achieved by the end of Senior Design 2.

9. Appendix

The appendix is used to show the referenced content used throughout this paper. Following the references section, a permissions section is placed to showcase the permissions for each outsourced image used.

9.1. References

- [1] A. Greenberg, "This Tesla mod turns a model S into a mobile 'surveillance station'," *wired.com*, Aug. 9, 2019. [Online]. Available: <https://www.wired.com/story/tesla-surveillance-detection-scout/>. [Accessed Sept. 30, 2021].
- [2] Nelly's Security, "FullHD 1080p (2MP) license plate recognition LPR Weatherproof Bullet IP security camera with a long range 8-32mm zoom lens and silky smooth 60fps video (NSC-LPR832-BT1)," *nellyssecurity.com*, May 19, 2021. [Online]. Available: <https://www.nellyssecurity.com/license-plate-recognition-lpr-security-camera-w-long-range-8-32mm-lens-high-powered-infrared-illuminators-silky-smooth-60-fps-at-1080p-for-plate-capture-nsc-lpr832-bt1.html>. [Accessed Oct. 1, 2021].
- [3] Criminal and Juvenile Justice Information Systems Council, *Guidelines for the Use of Automated License Plate Readers*. Florida Department of Law Enforcement , 2016.
- [4] Federal Communications Commission, "Commission Documents," *fcc.gov*, Nov. 10, 2021. [Online]. Available: <https://www.fcc.gov/documents>. [Accessed Nov. 10, 2021].
- [5] Census Bureau, "Census Bureau estimates show average one-way travel time to work rises," *census.gov*, Mar. 18, 2021, [Online]. Available: <https://www.census.gov/newsroom/press-releases/2021/one-way-travel-time-to-work-rises.html>. [Accessed Sept. 26, 2021].
- [6] IEEE, "IEEE Standard for Rechargeable Batteries for Mobile Phones," IEEE Std 1725-2021, 23 Aug., 2021.
- [7] IEEE Standards University, "IEEE Student Grants," *standardsuniversity.org*, Jun. 5, 2021, [Online]. Available: <https://www.standardsuniversity.org/grants/>. [Accessed Oct. 1, 2021].
- [8] National Highway Traffic Safety Administration, "Air Bags," *nhtsa.gov*, [Online]. Available: <https://www.nhtsa.gov/equipment/air-bags>. [Accessed Oct. 1, 2021].
- [9] T. Hubing, "PCB EMC Design Guidelines: A Brief Annotated List," University of

- Missouri-Rolla, [online document], 2003. Available: <https://cecas.clemson.edu/cvel/pdf/EMCS03-034.pdf>. [Accessed Oct. 26, 2021].
- [10] L. Geggel, “How long does it take a parked car to reach deadly hot temperatures?,” *livescience.com*, May 24, 2018. [Online]. Available: <https://www.livescience.com/62651-how-hot-cars-get.html>. [Accessed Sept. 30, 2021].
- [11] Florida Highway Safety and Motor Services, “Driver privacy protection act,” *flhsmv.gov*, Apr. 19, 2021. [Online]. Available: <https://www.flhsmv.gov/privacy-statement/driver-privacy-protection-act/>. [Accessed Nov. 19, 2021].
- [12] National Conference of State Legislatures, “Automated License Plate Readers: State Statutes,” *nctl.org*. [Online]. Available: <https://www.nctl.org/research/telecommunications-and-information-technology/state-statutes-regulating-the-use-of-automated-license-plate-readers-alpr-or-alpr-data.aspx>. [Accessed Nov. 19, 2021].
- [13] J. Marina Lee and C. Gilmartin, “New UCF parking policy to take effect July 1,” *ucf.edu*, Jun. 20, 2021. [Online]. Available: <https://www.ucf.edu/news/new-ucf-parking-policy-to-take-effect-july-1/>. [Accessed Nov. 19, 2021].
- [14] Texas Instruments, “MSP430FR698x(1), MSP430FR598x(1) Mixed-Signal Microcontrollers,” SLAS789D datasheet, Jun. 2014 [Revised Aug. 2018].
- [15] Jordsan, “SPI Timing Diagram,” *commons.wikimedia.org*, Sept. 7, 2010. [Online]. Available: https://en.wikipedia.org/wiki/Serial_Peripheral_Interface#/media/File:SPI_timing_diagram2.svg/. [Accessed Dec. 1, 2021].
- [16] M. Floryan, “I2C Timing Diagram,” *commons.wikimedia.org*, Feb. 7, 2007. [Online]. Available: https://commons.wikimedia.org/wiki/File:I2C_data_transfer.svg/. [Accessed Dec. 1, 2021].
- [17] Texas Instruments, “Interfacing MSP430™ MCUs With MMC or SD Flash Memory Cards,” SLAA281C report, Nov. 2005 [Revised Aug. 2018].
- [18] M. Avery, M. Krafft, A. Kullgren, and A. Linder, “CHANGE OF VELOCITY AND PULSE CHARACTERISTICS IN REAR IMPACTS: REAL WORLD AND VEHICLE TESTS DATA,” The Motor Insurance Repair Research Centre, Thatcham, United Kingdom, Tech. Report. Paper No. 285, 2, Apr. 2002.
- [19] PCB Piezotronics, “Impact and Drop Testing,” PCB Piezotronics Company,

- Depew, New York, Tech. Report, 20, Apr. 2018.
- [20] JEDEC, “Home | JEDEC,” 2021. [Online]. Available: <https://www.jedec.org/>. [Accessed Nov. 30, 2021].
- [21] NVIDIA, “Jetson Download Center,” *developer.nvidia.com*, 2021. [Online]. Available: <https://developer.nvidia.com/embedded/downloads#?search=Jetson%20Nano/>. [Accessed Nov. 23, 2021].
- [22] J. Bernstein and E. Wyrwas, “Quantitatively Analyzing the Performance of Integrated Circuits and Their Reliability,” in *IEEE Instrumentation & Measurement Magazine*. [online document], 2011. Available: https://www.dfrsolutions.com/hubfs/Resources/Quantitatively_Analyzing_Performance_Integrated_Circuits_and_Reliability.pdf/. [Accessed: Nov. 29, 2021].
- [23] Arduino, “Arduino Products,” *arduino.cc*, 2021. [Online]. Available: <https://www.arduino.cc/en/Main/Products/>. [Accessed Oct. 1, 2021].
- [24] M. Cefaci, “My "quick" Journey creating license plate recognition on Android with tensorflow 2,” *medium.com*, Oct. 6, 2020. [Online]. Available: <https://medium.com/swlh/my-quick-journey-creating-license-plate-recognition-on-android-with-tensorflow-2-51e89387fac4/>. [Accessed Oct. 1, 2021].
- [25] C. Hallman, “Average commute to work by state and city,” *titlemax.com*, Apr. 14, 2021. [Online]. Available: <https://www.titlemax.com/discovery-center/money-finance/average-commute-time-by-city-and-state/>. [Accessed Oct. 1, 2021].
- [26] F. V. Henderson, “What causes an airbag to deploy?,” *findlayvw.com*, May 11, 2021. [Online]. Available: <https://www.findlayvw.com/what-causes-an-airbag-to-deploy/>. [Accessed Oct. 1, 2021].
- [27] Insurance Institute for Highway Safety, Highway Loss Data Institute, “Airbags,” *iihs.org*, Mar. 2021. [Online]. Available: <https://www.iihs.org/topics/airbags/>. [Accessed Oct. 1, 2021].
- [28] JetsonHacks, “Front Page - JetsonHacks,” *jetsonhacks.com*, Nov. 7, 2019. [Online]. Available: <https://www.jetsonhacks.com/>. [Accessed Oct. 1, 2021].
- [29] B. Lipton, C. Quintin, A. Schwartz, J. Kelley, M. Guariglia, and N. Sheard, “Automated License Plate Readers (alprs),” *eff.org*, May 15, 2020. [Online]. Available: <https://www.eff.org/pages/automated-license-plate-readers-alpr/>. [Accessed Oct. 1, 2021].
- [30] NVIDIA, “NVIDIA Jetson Nano For Edge AI Applications and Education,”

- developer.nvidia.com*, 2021. [Online]. Available: <https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano/>. [Accessed Oct. 1, 2021].
- [31] OpenCV team, “About,” *opencv.org*, Nov. 4, 2020. [Online]. Available: <https://opencv.org/about/>. [Accessed Sept. 30, 2021].
- [32] Raspberry Pi, “Buy a Compute Module 4,” *raspberrypi.org*, 2021. [Online]. Available: <https://www.raspberrypi.org/products/compute-module-4/?variant=raspberry-pi-cm4001000/>. [Accessed Oct. 1, 2021].
- [33] TensorFlow, “Tensorflow Website,” *tensorflow.org*, Nov. 9, 2015. [Online]. Available: <https://www.tensorflow.org/>. [Accessed Oct. 1, 2021].
- [34] Texas Instruments, “Arm-based processors – Products,” *ti.com*, 2021. [Online]. Available: [https://www.ti.com/microcontrollers-mcus-processors/processors/arm-based-processors/products.html#p2192=Vision Analytics/](https://www.ti.com/microcontrollers-mcus-processors/processors/arm-based-processors/products.html#p2192=Vision%20Analytics/). [Accessed Oct. 1, 2021].
- [35] Texas Instruments, “MSP430FR6989,” 2021. [Online]. Available: <https://www.ti.com/product/MSP430FR6989/>. [Accessed Oct. 1, 2021].
- [36] Texas Instruments, “MSP430™ Hardware Tools,” *ti.com*, 2021. [Online]. Available: <https://www.ti.com/lit/pdf/slau278/>. [Accessed Oct. 1, 2021].
- [37] TutorialsPoint, “OpenCV - Storing Images,” *tutorialspoint.com*, n.d.. [Online]. Available: https://www.tutorialspoint.com/opencv/opencv_storing_images.htm/. [Accessed Sept. 30, 2021].

9.2. Permissions

