Senior Design 2
Backup Buddy
Wireless Backup Camera for Automobiles

UNIVERSITY OF
CENTRAL FLORIDA

University of Central Florida
Department of Electrical Engineering & Computer Science
Dr. Lei Wei

Group 10
Dylan Ortiz | Computer Engineering
Coleman Rogers | Computer Engineering
Luca Silvester | Computer Engineering
Zachary Slakoff | Electrical Engineering

Contents

List of Figures

List of Tables

1.0 Executive Summary

Thanks to many advances in technology the standard options available to people looking to buy new cars has improved in recent years. But going back just a few more years and premium options like a back-up camera were often out of reach for anyone except those who were willing to pay thousands of dollars extra. While many people can get by just fine by looking behind them as they back out of a parking spot, nobody is perfect and there still exists room for error and mistakes can be made. The dangers of backing out are clear and present, according to the National Highway Safety and Traffic Administration, more than 18,000 backup-related injuries occur in the United States each year with more than 200 of these injuries being fatal **[1]**. An unaware or distracted driver operating a 2000-lb vehicle creates a significant safety hazard for those walking nearby and behind their backing-up vehicle. What if the driver looks away for a second to check their phone? In those few seconds, someone could appear that the driver didn't see initially. Thinking they are still clear they continue to back-up, running the risk of unknowingly hitting the pedestrian. Without a clear view of what's behind them, many drivers will just start to back out slowly. This makes backing-up out of a parking spot a needlessly dangerous guessing game for everyone involved. Fortunately, this potentially dangerous situation can be made safer by providing the driver with more information about their surroundings through the use of a backup camera on the vehicle.

Having more information available to them can assist drivers in making the right decisions and reduce the dangers of backing up. Take for example, the poor field of view for drivers, often some of their vision looking back is blocked by parts of the car's frame or cars that are parked next to it. With a backup camera using a wide-angle lens, the field of view behind the vehicle could be greatly expanded to see much further beyond what the driver could see on their own. By taking advantage of technology available today our group is looking to develop a simple, cheap, and effective universal backup camera for those who bought their car during a time where back-up cameras either didn't exist in cars or were too premium an option.

2.0 Project Description

The product we are striving to deliver on this project is a small attachment for the back of anyone's motor vehicle. This device would be capable of streaming a video feed of the back of the vehicle with minimal latency to the user's smartphone. This stream could be viewed using an app that the user would download from the something like the Google Play store. On the app they could also choose to display information from a series of ultrasonic sensors built-in to the device that provide precise distances to objects behind them, and can provide audible warning through the app, should the distance between the vehicle and an obstruction behind it get too close. Once a user has successfully pulled out of the parking spot and begins to drive away, they will push the exit button on the camera view to stop the video transmission. Once they are no longer backing out, and leave their vehicle, they can turn on the security feature. This will allow the camera to see and record footage if the accelerometer detects any motion, and store it for viewing later.

2.1 Motivation

Our motivation for working on this project is to make use of the skills that we have all acquired over the last 4 years of taking engineering classes in a way that closely resembles the industry that we are about to enter. We are graduating at a time when improvements in mobile chip computing power and power efficiency are allowing us to build incredibly small and powerful devices that can be utilized in ways previously unimaginable only a few years ago. One of the best ways for the device we create to impact the greatest number of people was to try and improve something that is a part of everyone's daily lives, in the case of this project, motor vehicle safety.

While there have been many advances in vehicle safety utilizing the previously discussed advantages of mobile computing power and efficiency, many of these safety features are locked behind owning a particular brand of vehicle, some of which are very expensive and cost-prohibitive to most average consumers. To bring some of these newer safety features to the masses, we wanted to create a device that could integrate some of these safety systems and do so on a universal platform that was independent of the kind of vehicle you drove. Whether a user's car just rolled off the lot yesterday or its been driven for the last 20 years, everyone should have access to enhanced vehicle safety. One of the simplest yet effective safety features that we determined could be done was a backup camera with collision detection that helped you keep an eye on what was going on behind your vehicle as you backed up. We created Backup Buddy to make something that

would help make a daily part of people's everyday lives better and safer for both drivers and pedestrians.

2.2 Objectives

Our vision for the project would involve offering a small, unobtrusive, box for users to attach to the back of their vehicle, near the license plate. The box would be powered through two options: one would be a connection for users who are more open to modifying their car slightly, offering a connection for them to hook the camera up to the back-up lights of the car, only powering the device when the backup-lights are turned on. The second option would be for those who are less adventurous, an internal battery that could be swapped out every month or so. Once the device is mounted and receiving power, the user would download an app to their phone that would help them connect to the camera. When they want to backup they can open the app and the video stream from the camera will be sent to their phone.

To further enhance safety, the app would also offer more features than just the video feed to the user. Through the use of distance finding sensors, like ultrasonic sensors, exact distances to objects behind the backing up vehicle would be known. This information would be used to alert the driver both audibly and visually if they come too close to an obstruction behind the vehicle. The app would also provide an overlay on the video feed marking distances from the back of the car to 5 ,10, and 15 feet away, to further assist the driver. Once the vehicle has successfully backed out and is driving away, an onboard accelerometer sensor, integrated into the PCB of the camera system would trigger a shutdown of the video feed when a sufficient positive acceleration is measured, and the app would close itself.

The features being offered by our project would allow it to stand out on its own in the market of universal back-up camera solutions. What we've noticed is that most solutions involve a hardwired connection to a monitor to display the video feed, being connected by a 15 to 20-foot-long wire. This seems like a major inconvenience to the user and far too much time needed to setup properly, which is why our project would provide them with a quick and easy setup wirelessly using their phone. Another issue we noticed with many of these back-up cameras were just cameras with a distance marker overlay and that was it. With all the technological advances in recent years, we want to take advantage of the processing power available in smart phones to also implement tracking using computer vision. This would make our camera smarter, safer, and all around better than other options available to buy online.

Our ultimate goal is to make a wireless, mountable backup camera, that takes advantage of the processing power of the modern smartphone by using it as a

monitor and hub to keep both drivers and pedestrians safer. The use of not just a rear facing camera, but sensors, such as ultrasonic, keep this from being a generic add on for motor vehicles, but a genuine investment people can make for their own well-being.

2.3 Requirement Specifications

Given the nature of being an assembly that will be outside of the car at any given time, our specifications for the integrity of the structure are tailored to it being durable enough to withstand both the cars own velocity and various weather conditions. The performance of the system then relates to both the Android application performance on its own, how the app performs when video is being streamed, and the hardware performance. Hardware performance comes down to latency between the feed and actual placement of the car, the accuracy of external sensors, and eventually for the consumer, ease of installation. Each of these is designed in a minimum situation, where there is always room for improvement following our continued research and prototyping of our design. Below are the specifications and requirements of the system based on those parameters:

Our design came from a need of better car safety, and not just for brand new cars that have all of the bells and whistles added onto them. These requirement specifications are aimed precisely at adding the functionality of a rear view camera to cars after they have left the lot, but also for the consumer that isn't interested in paying someone to install hardware to their vehicle. The requirement specifications are as follows in table 1.

	Hardware Performance
1	The system will require no more than 12V from its power source
2	The system will weigh less than 10 pounds
3	The face of the system will take up a space no larger than 16in x 10in
4	The system will be able to accommodate up to 2A of current draw under load
5	The size of the PCB shall be no larger than 2.5in x 5in
	Software Performance
1	The Android application will provide a visual graphic to indicate an obstruction the car is approaching
2	The Android application will provide an audible tone, with increasing speed as the car becomes closer to an obstruction
3	The system will alert the driver when obstructions that are behind the vehicle are within 5 feet and the size of 1 cubic foot or larger
4	Using an accelerometer, the system will detect when the car is in motion, and if not in use will arm the security state of the app
5	The video feed of the rear facing camera will have a framerate of at least 15 fps at any given time

Table 1: Requirements Specifications

2.4 House of Quality

From table 2 below, the House of quality, cost seems to only have an impact when it comes to the video quality, battery life, and durability. This is due to having direct consequences with some of the components we would have to buy (ex. A larger battery, a better camera, etc…). Some of the other aspects, for instance the dimensions, would have other effects that wouldn't necessarily make the cost rise. If we had a bigger casing, it would cost more to make that material, but we would have room for a component that might be larger and cheaper than a faster, smaller component.

			Engineering Requirements							
+ Positive Polarity - Negative Polarity ▲ Positive Correlation ▲▲ Strong Positive Correlation ▼ Negative Correlation ▼▼ Strong Negative Correlation			Ultrasonic Detection	Dimensions	Weight	Power	Compatability	Cost	Video Quality	Accelerometer Accuracy
			+	-	-	-	+	-	+	+
User Requirements	1. Cost	-	▼▼						▼	▼
	2. App Speed	+								
	3. App Performance	+							▲	
	4. Battery Life	+	▼	▼	▼	▲▲		▼		
	5. Durability	+	▲▲	▼	▼			▼		
	6. Installation Ease	+		▲			▲▲			
Targets for Engineering Requirements			Detect Max of 150cm	<=16in x 10in x 4in	<= 10 pounds	12V source	At least 3 car models	~$225	Up to 720p at 30fps	Up to 0.5 G's

Table 2: House of Quality with Legend

3.0 Relevant Technologies & Competing Products

In this section we will discuss the research in regard to the current and past market availability of backup camera systems. The research below will discuss the many features sets of not only existing market backup camera solutions as well as the enclosure. These systems were chosen to provide a broad market range of what is available showing not only the hardware design and feature set but including the software companion applications. These products chosen below range from $99 to $500 showing a range in pricing and components that go into that price differential.

3.1 Competing products

The market for backup cameras are quite vast as the safety concerns that are rising from more and more people getting on the roads today. As of March of 2014 the U.S. National Highway Traffic Safety Administration made it a requirement that any cars built after May 2018 have to include a backup camera system (quote this baby). This is due to the added safety features that a backup camera system adds to a car with these systems it eliminates blind spots. Since there is a growing trend of Americans keeping older cars for longer and longer not all of the market will be able to take advantage of the newer cars with a built-in backup camera system. So, the market demand and need for a backup camera system is definitely in place and will have a life span for another decade. The current competition has a wide pricing model from the more luxury model doing everything in a compact sleek form factor to the symbol utility model that may need to user interaction but gets the job done and does not causally blend in with the car.

3.1.1 ZUS Wireless Smart Backup Camera

In this section we discuss the Zus wireless backup camera. This camera is a relevant competing product as it is wireless and within the market range, it is shown below

Figure 1: Existing backup camera design (Courtesy of Nonda. Permission Pending)

Figure 1 shows a relatively inexpensive wireless backup camera solution available on Walmart this system features a 170 angle view from a single 2MP camera. The installation of the product uses the top of the license plate and is attached using a security screw to protect against theft. The system runs on a rechargeable battery that has a battery life of 2 months with an average use case. The system allows any smartphone device to connect to it wirelessly using a Bluetooth connection sending a real time video the the users phone. The device features a water-resistant enclosure to protect the camera system from the elements. The device has a weight of 1.42 ounces and size of 12.20 x 5.71 x 1.73 inches. This system retails for 99.99 **[2]** and is shown below.

Figure 2: Existing backup camera application (Courtesy of Nonda. Permission Pending)

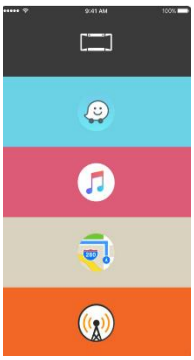
As seen in figure 2 the application for this product shows a view of the back of the car as well as guide lines that are user set. The application also as a night vision mode that will allow it to use the ir lens to get a better picture at night or in low level conditions. The system also features a battery indicator showing you how much charge you have left on you back up camera system before you need to remove the system and charge it. The application lets you pair with the backup system via a Bluetooth connection to control the device.

3.1.2 Pearl RearVision Wireless Car Backup Camera

In this section we will be discussing the pearl rear vision wireless backup camera. This device is a competing product as it is a full featured device. Having the ability to charge itself with a built in solar panel dual camera lenses to provide distance draw and an IR lense for better low light sensing. Though this device does lack any other sensors than the cameras it is priced at around $499.

Figure 3: Existing backup camera (Courtesy of Pearl RearVision. Permission Pending)

Figure 3 shows an existing backup camera solution that is no longer being manufactured but is still for sale. This backup camera system features two cameras to provide depth perception. This system features a waterproof enclosure allowing it to withstand the elements. This system also contains a small solar panel on the lower part of the assembly, so the system no longer needs to be charged. The system comes with a OBD adapter which provides the Wi-Fi service for a smart cellular device to connect to **[3]**. The system contains an infrared sensor for helping with seeing in low light environments. The camera has a wide-angle lens allowing views of 140 degrees both left and right. These cameras are also attached to small servos allowing the user control and move the angle either up or down.



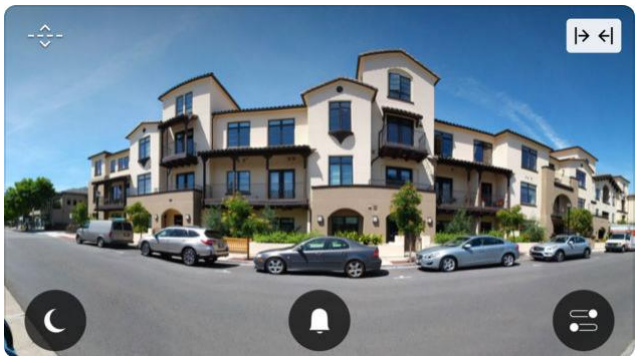


Figure 4: Sensor Application Design
Figure 5: App View (Courtesy of Pearl RearVision. Permission Pending)

The companion app show in figure 5 shows the view it gives the users when backing up it. As well as another screen that is triggered when the user is no longer backing up which allows them to either go back to the backup camera open the Waze app which is a navigation app that shows turn by turn directions as well as locates cops and accidents, the third app is the music app on the device allowing

the user to change the songs that are playing, the app after that is another mapping application for directions. The last application on the row is a podcasts app allowing the user to turn on or switch podcasts playing. In the backup view of the app on the top shows the backup view which allows the user to toggle night time mode toggle alerts and then go into the user's settings for customization.

3.1.3 FenSens Smart Wireless Parking Sensor

In this section it will discuss the FenSens smart wireless parking sensor. This product is a completely wireless sensor solution without cameras.

Figure 6: Existing backup sensor design
Figure 7: Application Demo (Courtesy of FenSens. Used with permission from Andy Karuza)

This backup sensor system follows the same design as the last design with a housing that goes completely around license plate acting as a frame. This device claims an install time of five minutes and includes both an iOS and Android application giving users the ability to get visual, audio and vibration alerts best on the detection of objects in front or behind the vehicle. This device also features an anti-theft screw that is proprietary to this device.

They also include anti-theft software that allows the user to track the car if it is lost or stolen. The system weighs 1.7 pounds and has the dimensions of 4.7 x 0.4 x 2.4 inches. The system does not include

a built in rechargeable battery but instead requires two AA batteries to power the system though they claim that with the battery system the user can get a battery life of 5 months.

The system connects to the user's device via a Bluetooth 4.1 connection, but the system is not relaying video just information from the sensors. The application for this system gives the user a status of the battery level and the ability to toggle or control the volume of the phone as the phone makes beeps altering the user how near they are to an obstacle. The application shown in figure 3.1 also gives the user a generic car with a radius showing the levels illuminating how which zone an obstacle for the car is in

3.2 System Enclosure

To house both the outside camera system and the inside system we need to design something that protects the electronics from the elements. For the camera system on the outside it is behind the car but still needs to withstand wind speed in excess of 60 miles per hour, the summer heat, and cold weather conditions. The casing for our outside camera system must be made of the appropriate material that is adaptable to all of these weather and external conditions preventing the casing from exposing the PCB and any important electronics to the harsh weather. As for the causing on the inside it still needs to handle the temperature extremes but has the benefit of not needing to be so water resistant and wind resistant. This indoor system may need to have vents to handle the head generated by the microcontroller affixed with in it allowing it to cool passively with the cars climate control. While waterproofing is within the needs of a device such as this, given our limited budget and time, testing for this would be too costly and would not give us proper time spent on development.

3.2.1 Laser Cutting

Another option for housing the system would be to use a laser cutter provided in the TI innovation lab. This laser cutter can be used to cut wood, acrylic, paper, cardboard. This would be useful for a more prototype based exterior as it allows us to quickly make an external casing for all of our components with a fast turnaround. With the turnaround time for 3D printing being as slow as it is it would be beneficial to make use of the laser cutter to make a quick housing for the system. Using just the laser cutter would most likely be used as a complement to 3d printing as the 3d printer would allow the components of the system to have a more secure hold than just attached to a flat surface but one that contours the component itself. As paper and cardboard can be extremely brittle they are out of consideration for this prototype but acrylic and wood would be the two options to consider. Acrylic would be more costly than wood but comes with the benefit that

it can be easily cut and allows for a higher level of detail. Woods downside would be great for beginning prototyping but the ability to keep out water is not there so as the design comes to the final prototype that's when it would be best to make an acrylic enclosure.

3.2.2 Types of 3D printers / printing services

There are an abundance of 3D printing services that allow us to print using various types of printers with a general turnaround time of one to three days. There is an option to use the on campus 3d printer located in the Texas Instruments innovation lab at the University of Central Florida without an extra cost of the using a 3d printer and the use of free material if they have it in stock which can reduce the turnaround time and cost of rapidly prototyping our designs.

3.2.2.1 Printing material

Once the backup buddy is finally assembled both using the PCB, the associated sensors and the battery the best way to prototype it would be to 3d print the housing as it gives us the flexibility to make the casing and quickly adapt to the different shapes that may occur from the design and assembly stage. The 3d printing filaments that best suit this system having the qualities to handle the outdoor temperatures and rainy weather conditions would be an Acrylonitrile Butadiene Styrene (ABS) filament as this material is less brittle and can handle the higher temperatures of the outdoors. The ABS filament would be a great filament for the exterior of our device to protect against the elements, but its downsides is the ability to be precise with it. A Polylactic Acid (PLA) filament would be a better choice as it is designed for more precise prints but its known as being a biodegradable plastic which would be a great for a more disposable application, but this system is designed to have a long life. A PET(G) filament would be the best bed for the outdoor camera portion as it is what is used to make water bottles, so it has both sturdiness and water-resistance. Though that is the case for this project will be using the ABS filament for prototyping as it is the most accessible filament available on campus inside of the TI innovation lab and is the best to use cost wise.

3.2.3 Final Enclosure Material

During the prototyping stage ABS gave us the results we wanted, unfortunately our final 3D design was too large for the printers on campus. We looked into out sourcing the print to a 3rd party company, but this proved to be too expensive and would cause us to go over our budget. Since laser cutting was the only other option to do on campus, and would have given us the precise cuts we needed, we turned to this for our enclosure. Again, due to time constraints and our budget we chose

not to move forward with this. We would have had to go through multiple iterations of the laser cut and t5his was not feasible due to the time remaining before our deadline.

The final enclosure used was made from wood and constructed and cut by hand. This gave us a model that had very similar dimensions to the 3D print created with precise software, but also allowed us to make quick changes on the fly without having to do any extra printing or modification of the model.

3.2.4 Attachment to the Car

To attach the system to the car there are three options, one being to use the back-license plate and use a license plate frame to house and hold the system, to use an adhesive to adhere the system to the back of the car or to use magnets to attach the system to the back of the car.

From the previous research of competing products, the system will have a weight to around a maximum of ten pounds. So, for the system being attached via the license plate frame weight will not be an issue. To attach the system via magnets it will allow the user to easily is mount and remount the system for recharges or any need to take the system off. For the adhesive solution it will be best for the concern of security as it will be heavily affixed to the car and cannot be easily removed.

Now a downside of the magnets will be security and turbulence if there was a rough road patch there would be a chance the system would fall off as well as with the ease of use taking it off to readjust means that someone can steal it. With adhesive the downsides would be when the user affixes it, removing the system or readjusting will not happen without hassle as they would have to use a solvent to remove the system then apply a new adhesive strip to reattach the system. The downsides of the license plate frame system would be to fall in line with any regulations regarding as well as taking away any custom frame the user may have had before.

In the end we designed our enclosure to have holes to be mounted around the license plate, with enough space for the license plate number to show through for proper identification of the vehicle. As mentioned in section 3.2.3, our final enclosure was not as structurally sound as ordinally designed, so we were not able to test the mounting of all of our hardware.

3.3 Wireless communication

Wireless communication will allow the system to not only be easy to install but ease of use for the user. This section covers the types of wireless communications we will or may be using in the system. This section will serve not only as a pro or cons list but will talk about the tech built for all of these communications. The wireless communication of the system will allow the system to communicate with the user's smartphone as well as allow the system to communicate with another sub system that will handle storage of the video.

3.3.1 Wi-Fi

Wi-Fi is a common communication that is used by smartphones, computers and almost all of today's current technology. Wi-Fi is an IEEE 802.11 standard. This communication technology is a low powered one allowing us to minimize the amount of power that our system is drawing and using.

With Wi-Fi we can achieve a range of over 100 meters and speeds up to 1Gbps **[5]**. Wi-Fi can be used in a low powered mode and provide fast transfer speeds to multiple devices. With this speed we can use this fast transfer speed to transfer high resolution video from the backup camera module to the user's smartphone.

3.3.2 Bluetooth

Bluetooth is a low powered communication that has a range from 10-100 feet. Bluetooth works by transmitting UHF waves on the 2.4 to 2.485 Ghz band **[5]**. Bluetooth was standardized by IEEE but is currently maintained by SIG. From Bluetooth works on a system where there is no central node for the network, but each device interchanges a slave master state. Once a slave the device is set in a receiving mode and once in the master state the node is set in the sending state. The Bluetooth 4.0 standard has the speed up to 25 Mbps. This standard supports smartphones from back in 2010 which gives the backup buddy the ability to utilize the later Bluetooth technology to take advantage of the faster speeds. The Bluetooth 4.0 standard supports a low energy mode that is called Wibree. The usual use cases of the Bluetooth technology is speakers and headphones. But as the growing market and demand of the Internet of things has created a widespread adoption of this close-range communication technology.

3.3.3 ZigBee Wireless

The ZigBee wireless technology is another contender for wireless communication that may be used in this system. This wireless system is used to make personal area networks using low power chips. This technology favors low power but, in a

tradeoff, it has low data transfer and a minimal transfer distance. ZigBee is currently being used in some IOT devices as it benefits the users with security and the low power draw. The range for a ZigBee system can range from 10 -100 meters depending on power draw and any obstacles in the way **[6]**. ZigBee makes up for the range though as it can span even further as long as there are other ZigBee nodes that are a part of the network in the system because of this it would be more than capable to cover even the largest of vehicles. Which would be tracker trailers moving wide loads or even the trucks where they double the load and have a double trailer system. Though for a single car and most consumer vehicle only two nodes would be needed with not much need of a mesh. The security provided by the Zigbee communication protocol is a 128-bit symmetric encryption but being that this system is both encrypted and low power to make it secure the speeds are around 250 kbit/s. In our case to have it as a way for the system to pass on just the data from our it would be more than enough any video transfer would have to be supplemented by the Wi-Fi chip.

In comparison to Bluetooth ZigBee technology draws less power and is less expensive that Bluetooth. ZigBee beats out Bluetooth in the distance ability as Bluetooth suffers a performance deficit further source. Though Bluetooth doesn't require another smart device for the user as the ZigBee needs a main ZigBee device to be the coordinator which the user can then interface with the ZigBee network and devices this is a pro for Bluetooth as this allows the user to use their smartphone to interact directly with the device and lowers the design redundancy for the system to take in the ZigBee signal then have a Bluetooth module to interact with the user's device.. Both ZigBee and Bluetooth run on the 2.4 Ghz band. The ZigBee technology and Bluetooth are both low power which is suitable for this system as it will be battery powered and may be charged with solar. In a more secure based system ZigBee would beat out Bluetooth but in this case we are just passing along sensor data to the user as a complementary safety feature and this information does not need to be secure in the model.

In comparison to Wi-Fi the ZigBee system is low powered which would definitely be a pro over Wi-Fi as battery time is critical in this system as there will be no such way to hardwire the system into the car with ease. Though the battery gain from using ZigBee would be insignificant in the power draw of the micro-computer that will be handling the video encoding and transfer. With ZigBee having a speed of around 250 kbit/s it transfers a video feed though being of a super low quality, but it would leave little to no overhead for the sensor data to transfer. Both ZigBee and Wi-Fi require there be a router like device to handle the connections for the others. Another advantage of ZigBee over Wi-Fi would be security as well as ZigBee is 128-bit encrypted and not normally picked up by devices as an easy to connect to device meaning the network that would transfer data would also be hidden through obscurity. Though since user devices don't pick up ZigBee natively there will still

need to be a hub that converts from ZigBee to a transmission technology that user devices can use.

Spec	Bluetooth	Wi-Fi	ZigBee
Transfer rate	3 Mbps	150Mbps	250 kbit/s
Power Consumption	5-25mA	50-180mA	5-25mA
Range	25m	50m	10m-100m

Table 3: Difference between Wi-Fi and Bluetooth and ZigBee

From table 3 it can be seen that the Bluetooth technology and Zigbee both have the least amount of power consumption but the differentiating factor between them is the transfer rate of the ZigBee device which to pass along sensor data is enough but for a video feed it is quite lacking. A negative for ZigBee is the need for a receiver to capture the ZigBee transmission and then have the signal become something usable by user electronics but both Wi-Fi and Bluetooth are implemented in almost all smartphones on the market meaning there will be no middle man device passing along the data. ZigBee would be needed in a larger distance setup, but Wi-Fi could also be used in that setup as well. In the end we decided on using both Wi-Fi and Bluetooth but using a Bluetooth module for the microcontroller and Wi-Fi being part of our microcontroller.

4.0 Standards and Design Constraints

This section covers both road blocks for our design as well as stepping stones. Some of the standards set in place by the various standards organizations give us some guidance to our approach, but the constraints can be a bit more restrictive. Considering we are developing a system to be used in a car, a lot of our constraints come in the form of motor vehicle reports and rules. Nonetheless we use both standards and constraints to find the best way to design our system to both conform to the norms set in place and avoid future dilemmas.

4.1 Standards

With the basis of our design being universality, standards are of the utmost importance to us for this device. We want to make sure that not only is the final product one that can be used by a variety of car owners, but also the manufacturing of it could be done universally as well, considering if we were to get this idea to market. There are no aspects of our design that are specific to our region, being the United States, so all considered standards are universal. Some of these standards are aimed specifically at university, but others, such as the programming conventions, are more of a practice that make transferring code easier for collaborators. Ethical dilemmas also have to be put into consideration, when asking users for app permissions. The standards that apply to our design give us not only conventions to make development more streamlined, but when the final product is finished, gives it a wider target market.

4.1.1 Lithium Ion Battery Standard

While no standards were found for the use of lithium ion batteries, some information regarding the manufacturing and testing of them was found. As it turns out, there are some regulations in order for safety testing to be made. However, according to an article written by UL, despite these tests, the rate of failures among lithium ion batteries is considerably high, so during the testing and prototyping stages of our design, we will have to place this into consideration. It should be noted that some batteries that we found online, had no information in terms of their safety testing. Based on the information gathered from the standards found for lithium ion batteries, we are exercising caution when handling and testing our hardware with these types of batteries

4.1.2 Java Programming Language Code Conventions

The language that we will be writing the Android app in is Java. When it comes to standards related to the language, a better term would be code conventions. Some

of these are not necessarily specific to Java, but all of them are necessary when sharing code with other people. These conventions are aimed towards the developers, as the consumer will never see the code running the application and looks to aid in the development and maintenance process. According to the official Sun Microsystems' Java code Conventions document, "Code conventions improve the readability of the software, allowing engineers to understand new code more quickly and thoroughly," and following these guidelines will make the development process run a lot smoother **[7]**.

Most of the code conventions are well known practices in the code community, and our developers were already used to. This includes, but is not limited to: import statements, indentations, and good commenting. Our team not only follow these conventions from years of practice but believes in their reliability and effectiveness in making code easier to read when a part of a team.

4.1.3 C Language Standards

The use of the C language is limited to our use of the microcontroller, however nonetheless it is an important role in the system. Just like the Java language, C has its own standards and code conventions that must be followed not only to make maintenance on the code more streamlined, but to allow the collaborative effort to flow more smoothly. We will be following the standards expressed in the document titles ISO/IEC 9899, and while there are over 500 pages of information related to this standard, the most important points in terms of what we will be coding will be expressed outright **[8]**.

Section 3 of this standards goes into the syntax of C and how certain libraries shall work when using this language. These are all conventions that we are familiar with, and after analyzing this there are no discrepancies in the way we program and what these standards are. Section 4 then goes into the conformance of the C language, relating to using what C has to offer and making sure the code is transferable.

Section 5 and 6 go into the compiling and syntax of a C program, which is more or less out of the control of us as programmers. It goes into detail of the nature of return types, the main function, and what characters are recognized by the C language. The syntax element of the language is, as stated before, not a new concept to this team. The libraries that we will be utilizing are standard to the C language, such as <stdlib.h> and <math.h>, but we also hold the right to use some custom libraries, provided we give a source of where said library came from. The fact that there are different compilers in C should not affect us, considering the scope of our project and how the base functionality of compilers remains the same.

The standards related to C are a bit more intricate than the Java standards, but always boil down to the syntax, code conventions, and running of the specific language. Our combined experience with the C language along with some of the information regarding the use of libraries in these standards give us a good foundation for programming the microcontroller.

4.1.4 App Permission Practices

While not an official standard, gathering permissions from the user when developing an application is not an easy topic. Fathering info from Android devices is sometimes a necessary thing to do for the functionality of the application. It does force the user to allow this access, however, and make them feel like their privacy has been invaded by offering up this access. Following the practices set by the Android developers' website can narrow does the permissions we need to program into our app, and limit the privacies the user has to give up when using our application **[9]**.

Of the 4 tenets of permission requests, 3 are specifically aimed at the developers. The first is only asking for permissions that are necessary for the functionality of our application. The only permission we will need from the user is to be able to access their devices storage. This is so we can store metadata regarding the connection to the hardware, so setup is not necessary every single time. The second tenet asks to pay attention to the permissions that certain libraries require. We would not continue to use a library that required the use of a library that was not of use to us. The fourth tenet asks that our system make explicit indications anytime a feature needs a new permission, which won't be an issue considering our use of only 1 permission.

The remaining tenet is geared to the user experience when asked to allow permission, and it is to for us as developers to be transparent. Making sure that the user of the app knows when we are collecting data or saving data to their device is an ethical necessity, and one we are sure to implement. One component to this is letting the users know why the app needs certain permissions. From personal experience, when loading an app for the first time, all that is displayed is an explanation of which permission I need to give, and options for yes or no. We will give the user a brief description of the permissions we need from them before giving them the chance to accept or not, not only complying with these permission practices, but also generating a better user experience from the initial launch of the application.

Since the release of Android 6.0, the way that app permissions are handled has changed. Prior to 6.0, the permissions were asked upon the applications initial install, but since that release the user can allow and revoke permissions at any time, even after the initial launch. For us this means that we would have to test our

app under the conditions assuming permissions may or may not be allowed. If we only ask the user to allow us permission to save data, we would have to test the condition of not having the link to the hardware saved, causing a manual startup every time.

These standards do not affect the design of our application but give us some pointers in the functionality of our app in the event the user does not want to give up the permissions that we need.

4.1.5 Software Testing Standards

While the next section in this report goes into depth the processes we will follow to test our system as a whole, specifically regarding software, there are standards followed by the IEEE. In the report ISO/IEC/IEEE 29119, standards set regarding software testing have been established that apply to any of the software development life cycles, which in our case is the agile method. There are 5 main aspects of the standards, with the main ones being test processes, documentation, and techniques. An overview of the standard written by students at King Faisal University serve as the basis for our establishment of these standards for our design **[10]**.

4.1.5.1 Software Testing Standards: Test Processes

The second section of the standard, aptly named “Test Processes”, refers to the actual methods gone into the testing of software. Test monitoring and control processes allows testing to stay up to date with the flow of the project and whatever entity is in control of said project. Keeping testing up to date with the progress of the project is just as important as the software itself. The test completion process then refers to the data found at the end testing and passing it on to the stakeholders in the project. Since our group are the main stakeholders for this project, this aspect is not as vital as we are going to be in contact with the entire software lifecycle.

Dynamic test processes are where the details of actual testing come from. There are 4 components to dynamic testing, starting with creating of actual test cases. We will develop test cases as the software grows, making sure that each new feature added has at least one test to go alone with it. The next component refers to the testing environment, which will more or less be our own computers, and once the app has more functionality, will move to a smart phone running the Android operating system. Test execution, of course, refers to the act of running these tests, with our main take away being that we need to re test anytime the code is updated. This will allow us to immediately notice if we change something critical that caused some functionality issues before continuing to write more code.

The last piece is test incident reporting, which gives us the opportunity to update our code requirements or update the tests when we run into unexpected behavior.

4.1.5.2 Software Testing Standards: Test Documentation

With the various functionality that we expect out of the application, keeping track of everything being worked on is critical, especially the tests we run and their progress. Figure 8 below shows the various test documentation titles. Considering the nature of this project, and that the Android application is only a part of the overall design, we kept track of all the test we ran through the messages sent as a group, and fixed them as issues came up.

Seq.	Documentation name	Seq.	Documentation name
1	Test policy	8	Test procedure specification
2	Organizational test strategy	9	Test data requirement
3	Test Plan	10	Test environment requirement
4	Test status report	11	Test data readiness report
5	Test completion report	12	Test environment readiness report
6	Test design specification	13	Test execution log
7	Test case specification	14	Test incident report

Figure 8: Test Documentation breakdown (Courtesy of King Faisel University permission pending)

4.1.5.3 Software Testing Standards: Test Techniques

The testing of our software with the hardware components that go along with it are detailed in the following section, “System Testing”, however the standards related to test techniques have more to do with the application’s performance on its own. There are 3 techniques expressed in this section. Specification based testing refers to the main source of information to device and write the test cases. Our team is most familiar with this component, having written test cases in the past based solely on the functionality of the software. Structure based testing refers to using the code itself as a basis for testing. Coming into contact with bugs and unforeseen programming errors is to be expected, and we intend to devise test cases revolved around our code specific issues. The final component, experience based testing, allows us as programmers to develop tests simply from our own knowledge and experience with coding.

4.1.5.4 Software Testing Standards: Our Approach

These standards expand our knowledge on testing, giving us an insight to the creation and organization of testing. Separating the tests based on which technique was used will help us to distinguish flaws in our design versus flaws in our code. The documentation standards will also give us an efficient means of keeping track of everything. We will have an efficient means of tracking a test from its conception to its results

4.2 Constraints

In this section the topics covered will be the constraints and the impact they have on a feasible design that can be completed for this project. The constraints for this project range from economic, environmental, social, political, ethical, health, manufacturability, sustainability, time and testing constraints. Though there are many constraints on this project it allows for true engineering to happen with these constraints it means that we have to work harder to make and be more innovative to get our project to meet our goals.

4.2.1 Economic Constraints

With the limited number of sponsors for the summer semester this project is going to be fully financed by us the students in the group limiting the projects ability to use higher end hardware. This lack of funding hurt the groups means of acquiring and testing with higher quality hardware. This means that our purchasing decisions have to be well researched in order to minimize on the cost factor. Since we don't have the funding that would allow us to try out different products in person this means we have to make our buying decisions based on specification sheets and the listed compatibility of the parts. Having to buy different components that will go unused would severely hurt the team budget.

With the team economic constraint in mind now we have to look at the markets economic constraint. With the current market pricing we are aiming towards having a product with in the middle of the market price range around $200 this allows it to be a competitive. If we have to increase the cost this limits the ability for our product to be marketed towards the crowd with older cars or truck systems who could benefit from this system.

4.2.2 Environmental constraints

For this system the environmental constraints we have to keep in mind the societies ever changing wastefulness and to do this we have to focus on being as renewable and environmentally friendly as possible. Not only does being

renewable help the environment but can serve as a way to market to a different clientele. This system needs batteries to work to achieve the wireless aspect of it. At this time, we are researching the possibility of having a solar panel implemented in the design, so the system will be completely self-sustained and not need the user to charge this system or for it to leech off of the cars power source. If this does not affect our other constraints, then it will be implemented.

4.2.3 Social Constraints

The social constraints for this project fall under making this system as available as possible by doing this is done by working towards making the product to be as price friendly as possible and keeping in mind regulations that may fall in other states of countries. This is done by working towards making the system attach to the vehicle in a manner that doesn't interfere with any laws. As well as making it so that those the accessibility of this system is there for everyone.

4.2.4 Political Constraints

The political constraints for this project would be limited to any changes in laws that would have a decision on changing the laws on mounting external devices to the vehicle and the use of a cellular device during the operation of a vehicle. If those don't change then there should not be any political constraints that will have an effect on this system.

4.2.5 Ethical Constraints

The ethical constraints for this project would be to not cut corners or use inferior products on purpose. This system cannot choose to use material that are not only harmful to people but the environment just to cut costs. To accomplish this our research and part selection will be based off of if the product is actually a better fit in more than one aspect over another product.

There also has to be a recognition of existing products and their design to make sure that the system does not directly copy any designs related to the physical system and the software design. To accomplish this, we have to perform research to make sure that our system to not infringe on any existing work.

4.2.6 Health and Safety Constraints

The health and safety of this system is the main point of the system. The system serves to aid the user in the reversal process by showing them not only what's behind them but sensing if anything will cross and alerting. In this case of altering the user we need to keep our streaming and sensor response under 100ms to give

users ample reaction time to change or stop their course of action. The safety constraints for drives on the road are to make sure the system to affixed to the car in such a way to prevent it from falling off during the users commute and created a hazard to someone behind the user on the road. To prevent this research will go into making sure the way the system is adhered to the car is overdone in such to insure the safety of others on the roadway.

The system will contain a set of batteries to power it as to keep the system as wireless and ease of use as possible but with the volatility of lithium ion batteries we will research and take into account the risks and dangers involved with having the battery exposed to the weather conditions. The system also is self-contained, and we have to ensure that any harmful material that may be produced in this product is contained to not have a harmful effect on the users.

The system will produce a varying alert as well as showing on the user's screen if something is with in the path of the vehicle. To insure the health of the passengers and users of the system we have to ensure that the alert produced is enough to get the drivers attention but not too loud to cause any hearing damage to the user nor distract the user from the task at hand.

4.2.7 Manufacturability Constraints

The manufacturing constraints for this project falls on the lack of available resources to make this system possible. For senior design two we are limited to 3D printers due to the lack of time to have our system made of injection molded plastic. Though for 3D printers there is a limited number of them available around campus and only one with a heavy priority to senior design projects. Another constraint is the material due to cost and practicality certain materials will be unable to be used for the system due to the lack of proper equipment and budget constraints. To ensure that this system can scale we have to research to see if the parts we select are available in massive quantities and if the product is not about to fall into obsolescence.

4.2.8 Sustainability Constraints

This sustainability constraints for the backup buddy system will be the stress the environment plays on the system. These environmental stresses range from the location in the world the user will be from wind, heat, rain, cold and many other combinations to make sure our system is sustainable research will be done to insure the material and components will be able to withstand these conditions. Since our system will be completely enclosed any parts that need replacement from excess vibration or use will have to be completely opened replaced and then

resealed. This causes repairs to be tedious without the end user being able to do them without breaking the seal enclosing the device hardware.

Another constraint will be how long our system will last in the marketplace with the ongoing manufacturing of new cars coming with backup cameras being standard so as older cars are replaced the need for this system falls off. Though this will take a while to come to fruition as many developing nations still have a demand for older cars as well as the trucking industry having the ability to quickly add a camera sensor system to the back of their tracker trailer and move it from trailer to trailer instead of affixing it to all trailers.

4.2.9 Time Constraints

The time frame for this project is unlike what we are used to in our scholarly career running over multiple semesters. Though this project does not get to take advantage of two full semester as this project starts in a summer semester which is about two weeks shorter than a normal semester. As well as to ensure that the demo for this project goes off without a hitch this prototype should be working midways through the fall semester. The project must be completed by November 2018 allowing roughly three months to assemble and test the prototype. One of the most important time constraints of this project will be the PCB as this part may need multiple redesigns which will hurt testing and setup time as the shipping for the PCB can take weeks thus constraining the other aspects of the project. The backup buddy must be able to stream data from the system to the user's device informing them of any obstacles that may impede their heading. To Achieve this goal in this time frame we will follow the timeline that is stated in our milestone section.

4.2.10 Testing Constraints

The testing of this system is constraint on the smart phones we can use to test on as well as the type of vehicle we can use. The reason being there a many makes and models of cars and trucks out there and to test on all of them will take many man hours that is just not available for this project as part of our time constraints. The lack of smartphones also come from the vast market of android phones that are available including the diversity of versions that range from the newest to some that are over four years old that are still in use today. There is also the constraint of the phone itself as there are many android phones that are country specific that we cannot get a hold of in a timely and costly manner. The testing for this system needs to be as real world as possible but to keep with our safety constraints we are limited to testing on closed tracks empty roads to ensure that the system cannot harm others around it if it were to malfunction during a test.

5.0 Component Research and Selection

The major components of our design make up the entire backend of our design. Each piece of this intricate puzzle has their own aspects that have to be put into consideration, such as power draw, cost, and speed. The MCU and Raspberry Pi are the heart of the hardware selection, and thus all remaining pieces have to be compatible with our choices for those components. Our biggest challenge when choosing these components was finding pieces that wouldn't draw too much power, as our design is intended to be charged through solar panels. The research done for each part gave us a better insight into our design, ultimately bettering us even after seeing so many parts we didn't end up selecting.

5.1 Microcontrollers

The universal backup camera, Backup Buddy, will make use of multiple sensor systems to accomplish many of its key requirements. With all the data being collected by the sensors on the system, a smart link between the sensors and the user is needed to listen to the data coming in and be able to respond accordingly. To assist with this task, a microcontroller acting as the central hub for these systems is needed. Before conducting any research on the many kinds of microcontrollers available to use for this project, the Backup Buddy team established a set of criteria to help evaluate each controller.

5.1.1 Microcontroller Criteria

First, the microcontroller needs to have low power consumption or have an option to switch to a lower power state when needed. This is important for multiple reasons, due to the unit being attached to the outside of the vehicle it will be exposed to the elements, one of these unavoidable elements being heat. If the board is already reaching high temperatures from basic operation due to high power consumption, high outside temperatures will only make things worse, potentially damaging the electronics. Low power consumption is also important because the system will be reliant on a battery as its main source of power, should the user opt to not wire the system into their backup lights. Due to the system lacking any way to power itself on when the vehicle does, it is necessary for it to remain in a low-power state and be ready to exit that state when the user needs it.

A secondary criterion that is needed is that the microcontroller is be capable enough to handle and process the data it will receive from the various sensor systems. With a backup camera, user safety and the safety of those around them is a key priority. The microcontroller needs to be able to capture, analyze, and act on sensor data in real-time to ensure that this priority is being met. With an established criterion that the microcontrollers need to meet for this project the team

took into consideration multiple options from two different and popular, microcontroller manufacturers, Texas Instruments and Atmel.

5.1.1.1 Texas Instruments MSP430G2553

The first microcontroller under consideration by the team is a classic among engineering students at UCF, being used in both the Engineering Computation and Analysis, and Embedded Systems classes. This would mean less time would be needed to learn how to interface with and program this microcontroller as all members of the team would already have the requisite development environment, Code Composer Studio, either downloaded on their personal computers or already be familiar enough with the software that they could make use of it immediately on machines in the engineering building. While being familiar with programming the microcontroller is not included in the criteria, in the short time that we have to build our project it is definitely an added bonus that is applicable to all microcontrollers being considered from the MSP430 family.

Moving on to the actual hardware itself, the MSP430G2553 is a very basic but useful microcontroller. It features a 16-bit processor running at a configurable clock speed that defaults to 16 MHz but can be configured to run as low as 1 MHz if needed **[11]**. It also offers 16 KB of non-volatile memory and 512 bytes of DRAM (volatile memory). The microcontroller also provides 16 I/O pins, which would be enough to support the number of sensors the system will need. Another one of it's biggest draws is that it is designed with ultra-low-power applications in mind, with a wide variety of power-saving modes that can be used to maximize the battery life of devices that rely on one. In the case of our back-up camera, which you would only want on when the car is backing up, there is a significant amount of downtime where the device is not in use. Being able to shift to a very low power mode while the device waits for an interrupt to be triggered by the user, could add a considerable amount of time to the battery life. The G2553 offers 3 different options when it comes to placing the device into a low-power state: active mode, standby mode, and off mode. When placed into the standby mode, assuming the clock is running at 1 MHz at 2.2V, the current drain by the microcontroller is a very low 0.5 µA. The supply voltage range is also very low, being between 1.8 and 3.6 volts, likely requiring us to step down whatever is supplied from the battery if we were to go with this microcontroller.

With these features in mind, the MSP430G2553 does a good job at meeting the criteria that the team needs to ensure a successful project. The only concerns come from the microcontroller's relatively low performance. Due to this project requiring real-time analysis of data coming in from its multiple sensors, there is a concern that the G2553's slow processor could result in slow response times. The project operates in an environment where milliseconds matter and can be the difference between a driver colliding with an obstruction or safely stopping in time. There is also concerns about the very low amount of RAM that the microcontroller

offers. While the non-volatile storage offers plenty of space with 16 KB, it also introduces latency in the system when data must be read from and written to it because the main memory has become full due to its very small size. However, even with these concerns in mind, the MSP430G2553 looks like a very solid option for the development of this project.

5.1.1.2 Texas Instruments MSP430FR59691

The second microcontroller under consideration by the team is also from the same line of microcontrollers that the G2553 comes from. However, unlike the G2553, the FR59691 takes advantage of a technology called Ferroelectric RAM or FRAM for short, bringing with it a variety of advantages over the standard DRAM being used on the G2553.

The biggest of these advantages is that it offers similar performance to DRAM while also offering non-volatility, retaining the contents of its memory cells even when power to the RAM is lost **[12]**. Another advantage is the speed at which the memory can be written to. FRAM can perform a write in under 50ns, giving it, in the best-case, a speed-up of 1000x when compared to other flash memory technologies **[13]**. The power needed to utilize FRAM is also very low when compared with the G2553, needing only 1.5v and very low current for both read and write operations **[14]**. While working with new technologies can sometimes result time lost due to team members having to learn how to utilize it, this is fortunately not the case with FRAM. All code written on previous MSP430 microcontrollers is compatible with other MSP430 microcontrollers that use FRAM instead of DRAM. This would make the transition for the team almost seamless.

On top of using FRAM as its main non-volatile memory, the FR59691 also offers a few other features that may be useful to the team. The amount of storage on the microcontroller increases significantly when compared to the G2553, offering 2 KB of SRAM and a whopping 64 KB of FRAM **[15]**. This would provide us with an ample amount of fast storage for code and data. The number of I/O pins also increased to 40, giving us greater capacity to attach more sensor systems to the microcontroller. The device also offers both a 128 and a 256-bit AES encryption and decryption coprocessor, giving us the option of enhanced security for the project, should it be needed. The processor runs at the same configuration options as the G2553, ranging from 1 to 16 MHz as selected by the programmer. The supply voltage range is similar to that of the G2553, requiring a minimum of 1.8V and taking a maximum of 3.6V. The FR59691 also offers 4 different power-saving modes that the device can be placed into: active mode, standby mode, real-time clock mode, and shutdown mode. When the device would not be in use, the team would likely place the microcontroller into a standby mode to wait for an interrupt signal, to indicate that it should switch back to a higher clock and power consuming state. Due to the use of lower-power FRAM, the microcontroller also offers a lower power consumption when placed into the stand-by mode, with current drain going

all the way down to 0.4 µA, resulting in increased battery-life. These features make the FR59691 something for the team to consider moving forward with this project.

5.1.1.3 Atmel ATmega328P

The third microcontroller being considered by the team is the ATmega328P, manufactured by Atmel. This microcontroller is being considered as the middle option, splitting the difference between the G2553 and FR59691. Much like the MSP430 line of microcontrollers, the 328P is also very popular among hobbyists as it is the processor used at the heart of Arduino's hobbyist line, the Arduino Uno. This is helpful as it means the documentation for the microcontroller is extensive and learning how to interface and use it would take the team very little time.

Looking at the hardware of the 328P, it offers numerous features that would be useful to this project. Once again, it's processor clock speed can be defined by the programmer, however, instead of topping out at 16 MHz, the 328P can go as high as 20 MHz if supplied enough voltage. This could be the little extra boost needed to push performance to an acceptable level if the demands of the project are too high for the other microcontrollers. The 328P also comes with the option to make modifications to the bootloader if needed, providing 1024 bytes of EEPROM. Memory also includes 2 KB of SRAM and 32 KB of DRAM, which would provide us with even more space for code if needed. It also offers 23 I/O pins for us to connect our sensors to **[16]**. Since the 328P was designed with hobbyists in mind, it also includes a wide variety of power-saving options to help extend battery-life with 6 different sleep modes: idle, ADC, noise reduction, power-save, power-down, standby, and extended standby. If being used in this project, we would likely place the device into a power-save mode when the device is not in use which would, assuming the clock is running at 1 MHz at 1.8V, bring current drain down to 0.75 µA, lowering power consumption dramatically. The 328P offers a wide variety of features that make it a decent option for this project.

5.1.2 Microcontroller Comparison

To help decide what microcontroller to use for the project we needed to fully understand what each microcontroller had to offer in comparison with the others that were researched. To do this the differences between each microcontroller were analyzed over 5 factors that pertained to performance and overall usability: clock speed, memory and storage, I/O support, power consumption, and cost. With this comparison laid out, a more educated decision on what microcontroller would be best suited for this project can be reached.

5.1.2.1 Clock Speed

The clock speed of the microcontroller is the frequency at which the CPU clock operates at, this value is measured in Hertz or cycles per second. For example, if a datasheet says that the microcontroller runs at 4 MHz, then it would mean that the CPU clock performs 4 million cycles per second. All of the code running on the microcontroller is translated into instructions for the CPU to execute, the amount of time it takes for an instruction to complete is measured in clock cycles. The value of a microcontroller running at a higher frequency is quite simple. If a specific instruction takes 100 cycles to complete then, doing some quick math, 100 divided by the clock speed of 4 million cycles per second, you would get 25 microseconds to run that instruction. If we increase the clock speed to 16 million cycles per second, the time taken to run the instruction drops to only 6.25 microseconds. Therefore, the faster the clock speed, the faster the microcontroller is able to execute instructions and process code and data. A breakdown of the clock speeds for each of the microcontrollers being considered is presented in table 4 below.

Microcontroller	Clock Speed Range	Input Voltage Range
MSP430G2553	1 to 16 MHz	1.8 to 3.6V
MSP430FR59691	1 to 16 MHz	1.8 to 3.6V
ATmega328P	1 to 20 MHz	1.8 to 5.5V

Table 4: Comparison of Microcontroller Clock Speeds and Voltages

As shown in the table above, with the MSP430 line of microcontrollers we get the exact same specs, with the only real difference being the 328P. Input voltage is included in the table as the top frequencies of these microcontrollers would not be reachable without a sufficient voltage being provided. With power consumption falling under one of our two big criteria for microcontrollers, it is important to consider this as well when comparing the clock speeds of each microcontroller. While the 328P may offer the superior performance among the microcontrollers being considered, it is important to note that it does not do this without requiring a rather significant jump of 41% to the input voltage needed to power the microcontroller. This is a rather heavy price to pay to only gain a 22% increase in clock speed. Still, increased power consumption aside, it should still be noted that the 328P is the fastest microcontroller among our options.

5.1.2.2 Memory and Storage

Another factor to consider when trying to achieve the best performance is the volatile and non-volatile memory systems in place on each of the microcontrollers. The volatile memory, also known as the RAM or random-access-memory, is where

the processor places code and data relevant to the execution of the program. It offers faster access and write back times than the non-volatile memory, allowing for programs to run faster. Volatile memory isn't much of a problem for the programmer if there is enough of it for the program being run. However, should there be too little, and the processor have to fall back to using the non-volatile memory for storing data relevant to the active program, then program performance would take a significant hit. Therefore, it is in the programmer's best interest to write code that is optimal and wastes as little volatile memory space as possible, especially when writing code for systems with small memory configurations. The non-volatile memory is where program data and code sits when the device is powered off, offering a durable, long-term solution to storing data without needing to provide power. It's where the processor will go to extract the program data and code it needs into RAM to start executing the program, and will also be used as a fallback should RAM become full.

With regards to the microcontrollers being considered for use in this project, three different memory technologies are used: SRAM, DRAM, and FRAM. SRAM or static-RAM, is a fast volatile memory that needs a constant supply of power to retain its data but does not need to be refreshed constantly. DRAM or dynamic-RAM is a fast but slightly slower than SRAM, volatile memory that requires constant refresh to maintain its data. Finally, FRAM or Ferroelectric-RAM is a newer up-and-coming type of non-volatile memory that manages to retain many of the benefits of DRAM without needing to be constantly refreshed and consuming less power as a result. With the different types of memory technology defined, table 5 displays what each microcontroller is configured with.

Microcontroller	SRAM	DRAM	FRAM	Flash Storage
MSP430G2553	None	Yes – 512 B	None	Yes – 16 KB
MSP430FR59691	Yes – 2 KB	None	Yes – 64 KB	None
ATmega328P	Yes – 2 KB	None	None	Yes – 32 KB

Table 5: Comparison of Microcontroller Memory Technologies

Based on the data presented above and the fact that we will be reliant on DRAM and SRAM to increase performance, those two categories should be weighted the most heavily when considering each microcontroller. It should also be noted that while the FR59691's 64 KB of FRAM can't be classified as volatile, it performs very close to it, with a less than 50ns access time. The G2553's very small amount of volatile memory is a concern, as noted previously, volatile memory isn't a problem until you run out of it, and with only 512 bytes of it there are concerns that this limit could be reached if the code is not completely optimized. On the other end, having too much fast memory is also a negative, as is the case with the FR59691. It is

highly unlikely we would ever be able to take advantage of the FR59691's FRAM as the program written will likely fit within the 2 KB of SRAM. This once again places the 328P at the top of the microcontrollers under consideration.

5.1.2.3 I/O Support

To be able to use and interface with the sensors and modules that we will be using in this project, programmable input and output pins are needed on the microcontroller. Each of these individual pins is capable of delivering the power needed to each of the sensors and is able to send data to and from it. The project will require a light sensor, an accelerometer, and a wireless communication module to fully function. Since each of these sensors will need connections for power, data, and ground, input and output pins are necessary. Besides considering the number of input and output pins to use, the team also needs to consider that maximum output current that can be delivered by each pin. With multiple sensors connected to the microcontroller, current will be divided amongst each of the sensors. If the sensor fails to receive a sufficient amount of current then it will be unable to function properly.

The microcontroller that we select for this project needs to be able to support at the minimum, 1 serial connection over UART (Universal Asynchronous Receiver - Transmitter) which is required for the Bluetooth module that will be integrated onto the PCB, at least 10 or more GPIO connections depending on the number of sensors that are low-power enough to be powered by the pins, and it needs to include support for both I2C and SPI serial communication standards as some of the sensors we buy may communicate over them instead of using GPIO. The breakdown of each of the microcontrollers and their capabilities in handling input and output is shown below in table 6.

Microcontroller	I/O Pin Count	Pin Max Current Output	UART Interfaces	SPI Interfaces	I2C Interfaces
MSP430G2553	16	6mA / pin (48mA total)	1	2	1
MSP430FR59691	40	6mA / pin (48mA total)	2	3	1
ATmega328P	23	40mA / pin (200mA total)	1	2	1

Table 6: Comparison of Microcontroller I/O Support

Looking at the breakdown of the different microcontrollers, purely from a numbers perspective, it is clear that the FR59691 has the most features in almost every category. Going purely off the different serial communication interfaces it has, it could support 6 separate sensors alone, not even taking into account the 40 GPIO pins, which would likely be overkill for the scope of our project. Stepping down a level, the 328P is the second best, offering 23 GPIO pins, almost half as few as the FR59691, but still enough that we could support a large number of sensors on them, regardless of whether they need 1,2, or more GPIO pins. The 328P also offers the most stable voltage drop at high currents, which would definitely be useful for sensors that source a high amount of current from the GPIO pins. Finally, we have the G2553 at the bottom, but considering it's the most barebones microcontroller of the bunch, it's still pretty good, having a matching number of serial communication interfaces as the 328P. Still only 16 GPIO pins is low, and could present an issue moving forward as the team decides on sensors. Overall, the FR59691 is the best option if the team wants the most flexibility with sensors moving forward.

5.1.2.4 Power Consumption

One of the major criteria, defined previously, was microcontroller power consumption. This factor carries more weight than others because of the nature of the project and how it must remain in a standby state, ready to go whenever the vehicle owner wants to back up. However, since backing up only happens when someone is leaving and usually lasts less than a minute, there is a significant amount of time that the system is not in use. To maximize the amount of time that the system can go without needing its battery to be replaced, power consumption during operation and in the standby low-power mode, would need to be as low as possible. The current drain in each microcontroller's respective low power modes and running in active mode at 1 MHz is given in table 7 below.

MCU	Lowest Voltage Supplied	Current Drain (Low Power)	Power Consumption (Low Power)	Current Drain (Active)	Power Consumption (Active)
MSP430 G2553	1.8V	0.50 µA	0.90 µW	0.23 mA	0.414 mW
MSP430 FR59691	1.8V	0.40 µA	0.72 µW	0.10 mA	0.180 mW
ATmega 328P	1.8V	0.75 µA	1.35 µW	0.20 mA	0.360 mW

Table 7: Comparison of Microcontroller Power Consumption

Since each of the microcontrollers being compared all have the same minimum operating voltage, the real differences between each of them in determining their

power consumption is the amount of current drained. To calculate the power consumption for each microcontroller, the voltage supplied was multiplied by the current drain in both the low power and active modes. The FR59691 was the clear winner in both the low power and active mode's power consumed during operation. It was 60% better than the 328P in low power mode and 66% better than it in active mode. Compared to the G2553, also from the MSP430 line of microcontrollers, it was 22% better in low power mode and a whopping 78% better in active mode. This makes the FR59691 a very good option to fulfill our low power criteria for this project.

5.1.2.5 Unit Cost

In the interest of making our project as widely accessible to as many people as possible, the cost of manufacturing it must be made as low as we can go without sacrificing functionality. At the heart of each of these Backup Buddy units that we produce will be whatever microcontroller we choose to use, making the cost of the microcontroller a big part in determining how much Backup Buddy will cost. In table 8, shown below, the cost for purchasing each of the microcontrollers from their respective manufacturers is compared as well as the percent difference in cost between it and the other microcontrollers under consideration.

Microcontroller	**Cost**	**Percent difference in cost (G2553, FR59691, 328P)**
MSP430G2553	$2.20	0%, - 54.7%, + 9%
MSP430FR59691	$3.86	+ 54.7%, 0%, + 63%
ATmega328P	$2.01	- 9%, - 63%, 0%

Table 8: Comparison of Microcontroller Cost

We can see from the table above that the 328P is clearly the most budget oriented microcontroller out of all of our options. It is 9% cheaper than the G2553 and a whopping 63% less than the FR59691, if we wanted to really lower costs than it would be a great microcontroller to choose. Comparatively, the G2553 is 54.7% cheaper than its counterpart from the same MSP430 family, the FR59691 and is only 9% more than the 328P, making it not a bad choice either when it comes to cost. Finally, the FR59691 features the largest amount of memory with the most features, making it obviously the most expensive choice amongst the microcontrollers. While we will need to carefully balance the pros and cons of each microcontroller before making a decision on which one to use, being able to compare the prices of each one helps in that process.

5.1.3.1 MCU Selection: MSP430FR59691

After careful deliberation and weighing all of the pros and cons of each microcontroller we ended up selecting the MSP430FR59691 for this project. While this microcontroller did come in as the most expensive option in our breakdowns of each microcontroller, it was a difference of $1.66 at most, which to the group wasn't going to break the bank. The deliberations were focused around the criteria we had established previously and what each microcontroller did to meet them.

First and foremost was the microcontroller's ability to consume as little power as possible, prolonging the battery life of the Backup Buddy system. While we had initially considered trying to power the entire system through the 12-volt line connected to the vehicle tail lights, we determined that this would be too much of an inconvenience for potentially inexperienced users who just wanted the device to work out of the box. As a result of this design decision, the only way the system was going to be powered was using a battery that the user could replace when it died. In shifting to this new design, a very big focus was placed on power consumption and how we could go about minimizing it. Due to the FR59691's impressively low power consumption, even when compared to similar low-power microcontrollers, it quickly became the front runner for our microcontroller of choice. Since the system will need to remain powered at all times, any minimizing of the overall current drain will result in more time that the user does not have to replace the battery. While the other microcontrollers were certainly impressive in their low-power modes, none were as good as the FR59691.

The second criteria that we needed our microcontroller to meet, was that it was fast enough to handle the simultaneous influx of data coming from our sensor systems and able to react to an ever-changing situation in real time. Fortunately, the demands of our system will be relatively low in comparison to the hardware that we are working with. The most demanding task will be the wireless Bluetooth transmission of data to the user's phone, requiring a constant link between the two while the system is active. However, the only data this system will be passing to the phone will be data from the ultrasonic sensors, telling the user the distance between them and whatever object is closest behind them. The sensors that we have selected operate at a refresh rate of 20 Hz maximum, and the final design will likely not include more than 3 of them. This would mean the processor would need to handle at worst-case, 120 floating point operations per second from the ultrasonic sensors. The last sensor, the accelerometer, has a configurable refresh rate and can transmit acceleration data anywhere from 2 to 800 Hz. This wide range of refresh rates is very useful as it allows us to set it to a rate appropriate for the processor and the amount of work it is doing. If it's too stressed out then we can drop the refresh rates on all of the sensor systems to bring it more under control, or if that doesn't work, increase the CPU clock frequency to compensate. While any of the microcontrollers under consideration were likely going to be able

to handle the processing that this project would demand, none offered us nearly as many options as the FR59691.

On top of meeting our desired criteria for a microcontroller, the FR59691 also comes with 40 general-purpose I/O pins, an I2C channel, 3 SPI channels, and 2 UART channels. As not all of our sensor systems will be able to take advantage of the GPIO pins for their connections, having a wide variety of serial communication options that can also be used for our various sensor systems is very helpful. Overall, the advantages of the FR59691 are worth far more than the extra $1.60 or so that we'd be saving by going with another one of our microcontroller options, pretty easily making it our microcontroller of choice for the Backup Buddy project.

5.1.3.2 Extra: Raspberry Pi Model 3 B+

Due to hardware demands needed for streaming video at high resolution at a minimum of 15 frames per second wirelessly, we needed a powerful yet compact all-in-one solution. While looking around at multiple options for camera systems, we found consistently that the cheapest and most powerful options were cameras built to be used with a Raspberry Pi. Our initial target, which exceeded our requirements, was to stream the backup feed to the phone application at 1280 by 720 pixels at 8-bit color and 30 frames per second. With quick calculations we determined we would need more than 100 Mbps of bandwidth to deliver this raw video stream. To handle the immense amount of data being transmitted we wanted to encode the video stream beforehand, which would bring bandwidth requirements down to a much more manageable rate of 10 Mbps. But finding a camera that would not only record at the desired resolution as well as handle video encoding and transmission was a challenge. However, we came across the Raspberry Pi's newest model, the 3B+, featuring a 64-bit quad-core CPU running at 1.4 GHz, built-in dual-band wireless networking, built-in Bluetooth functionality, and it could handle the encoding of our video stream to H264. All of these much needed features for only $30. It was the perfect choice for the demands of this project and maintaining a high-level of quality on our video stream.

5.1.3.3 Extra: Sleepy Pi 2 Shield

Due to the need for a Raspberry Pi in this project, as well as a need to minimize power consumption wherever possible, a means of lowering the Raspberry Pi's consumption was needed if we were to achieve any sort of useful battery life. The first thing that came to mind was to research the datasheet of the ARM CPU that powered the Raspberry Pi to see if it offered any sort of low-power modes. While the ARM Cortex-A53 does offer options for lower power consumption, the Raspberry Pi does not utilize them, opting instead for a greater focus on higher performance. The inability to enter a low power mode for the Raspberry Pi would cause significant problems for the project. After further research it was determined

that the Raspberry Pi idled at around 435 mA, which would place a massive strain on our battery system. However, to help reduce this idle power consumption, non-essential services could be turned off. For example, we wouldn't be needing the HDMI port as the Raspberry Pi wouldn't need to be hooked up to any display, cutting power to it would save 20 mA. Another feature that this project would not be using is the Raspberry Pi's 4 USB 2.0 ports, by cutting power to them we could save a whopping 200 mA. By simply closing these 2 non-essential services we could reduce power consumption by 220 mA or a 67% decrease, dropping idle consumption to 215 mA. Still even in this lowered state of power, the Raspberry Pi was consuming too much, bringing battery life down into hours rather than days. We would need to go even lower to make Backup Buddy useful, fortunately we came across a device that could do just that, the Sleepy Pi.

With the Raspberry Pi's lack of low power mode features, Sleepy Pi acts as a virtual low power mode, rather than something that unlocks those unused features in the Raspberry Pi's hardware. Sleepy Pi functions as a programmable power-switch for the Raspberry Pi, safely shutting down and powering the board back up via code or when it is awoken from an interrupt. The device takes a wide range of input voltages from 5.5 to 17 volts, offering us some flexibility in our battery selection. The processor driving the Sleepy Pi is an ATmega328p, meaning that the device can be programmed using Arduino libraries, making adoption of the device quick and easy for the team **[17]**. The device also offers a few of its spare GPIO pins that can be used to allow other devices, like our main microcontroller, to communicate with the Sleepy Pi. Most importantly, when the device shuts the Raspberry Pi down and enters a low power mode, the total power consumption from the Sleepy Pi comes out to 500 µA, a huge improvement from the already improved 215 mA that the Raspberry Pi would've consumed while sitting idle, not doing any work. Without the Sleepy Pi, our power consumption would be too high to make the device actually useful for users. But with it, the extension to the battery life makes Backup Buddy a useful addition to a user's vehicle.

Due to time and size constraints, the Sleepy Pi module was not included in our final design. During our testing phase we found some solutions that might have worked to shut down the pi and boot it back up without the use of this extra module. This, along with the time constraints of the other components of our design pushed this to a stretch goal that was not used in our final implementation.

5.2 Camera Hardware

At the core of the user experience with Backup Buddy is the video feed from the back of the car itself. Since one of the requirements for the system is that it is able to wirelessly transmit the video feed to the user's phone, strict limitations regarding the video feed are in place. First and foremost, the resolution of the actual feed cannot be too large, or the amount of raw data produced in filming at 15 frames

per second would completely overwhelm the Raspberry Pi encoding it. It also runs the risk of completely saturating the wireless 2.4 GHz link between the Raspberry Pi and phone, resulting in dropped frames or degraded network performance. Second, the camera has to be capable of filming in at least 15 frames per second to meet the requirements of this project. Failure to meet this requirement raises safety concerns about the lack of real-time information that the driver is receiving and making decisions on. Another consideration when choosing the camera is the field-of-view of the lens. Should it be too narrow, the user will be unable to see what's potentially in the road to the left or the right of the vehicle. However, this is less of a concern as wide-angle camera lenses can be purchased separately and swapped out as needed. However, as both a time and money saving measure, it would be a plus if a camera were to come with a sufficiently wide-angle lens already installed. With these basic requirements in mind, the following list of cameras is under consideration by the team.

5.2.1 Raspberry Pi Camera Module V2

In the interest of simplifying camera module integration with our existing systems, the official Raspberry Pi camera was considered. To add it on to the Raspberry Pi was as simple as connecting an included ribbon cable with the appropriate camera serial interface port on the Raspberry Pi. This would mean little time would need to be spent setting things up and that documentation on it would be good.

Comparing this camera to the criteria previously discussed, it is safe to say that it passes all of them with flying colors. The image sensor itself is an 8-megapixel Sony Exmor IMX219 with an active pixel count of 3280 horizontal pixels and 2464 vertical pixels. The camera comes with several resolution options, based on the needs of the project, being capable of 4k, 1080p, and 720p **[18]**. However, because of limitations imposed by the hardware present on the Raspberry Pi, the best it can do while connected to it is 1080p. Still this resolution would be more than sufficient for the needs of the project.

The second criteria that the camera needed to meet was frame-rate, being able to record at least 15 frames per second. This camera is easily able to meet this requirement due to the high quality of its image sensor, which itself can do 4k at 30 frames per second, 1080p at 60 frames per second, or 720p at 180 frames per second. However, again due to the limitations on the Raspberry Pi hardware, the framerate does take a hit when connected to it. The adjusted framerates for each resolution are: 1080p at 30 frames per second, 720p at 60 frames per second, and 480p at 90 frames per second. Regardless of the limitations imposed, the camera connected to the Raspberry Pi is easily able to meet the requirements of the project.

While it's great that the Raspberry Pi camera is able to meet our criteria, it is also not without some faults. First, the field-of-view of the lens would require us to swap

it out with a better one, as 62° is not enough to fully see behind the vehicle. Another issue that could arise is that it may result in a more complicated PCB design. The official documentation highly suggests that the power supply to a Raspberry Pi using this camera be 2 A. With the rest of the PCB's relatively low power and current sinking devices, a higher current drain from the Raspberry Pi could result in added time during PCB design to ensure the camera functions properly. The last point that could be counted against this camera is that it's frankly too overkill for the requirements we are trying to meet. The most common smartphone screen resolution in the United States is 1334 by 750, just barely above 720p **[19]**. If we chose this camera and streamed it's feed in 720p we would be using less than a quarter of the pixels available to the sensor, making it seem like a waste of resources and power consumption. Still, this camera does offer a lot of great features and manages to do so at a great price point, definitely making it an option for the team to consider moving forward.

5.2.2 Omnivision 5647

The next camera under consideration uses a 5-megapixel Omnivision 5647 image sensor. This camera is also fully compatible with Raspberry Pi and connects with it over the camera serial interface on the board. This also makes this camera a great option for quickly integrating it into our existing systems. While it's lower popularity would mean less documentation to go off of, its lower specs are more appealing to the project requirements.

When compared with our criteria, this camera is also easily capable of meeting and going beyond our requirements. The sensor itself has a horizontal active pixel count of 2592 and a vertical pixel count of 1944. With such a high pixel count a variety of different resolutions are supported: 1080p, 720p, and 480p **[20]**. While the Raspberry Pi hardware still imposes some limitations on the capabilities of the camera, they are not nearly as severe and less pixels on the sensor are wasted.

The camera also easily handles the required frame rate of 15 frames per second, offering increasingly higher frame rate options as the resolution decreases: 1080p at 30 frames per second, 720p at 60 frames per second, and 480p at 90 frames per second. These are identical to the Raspberry Pi camera, but come in a smaller, more power efficient package.

Outside of the main criteria, this camera also fulfills one of our less important but nice to have needs as well. It comes equipped with a fish-eye lens, providing a 160° field-of-view, much better than the Raspberry Pi camera's 62° lens. The only real negative for this camera is that it too will still waste a ton of pixels if we record in 720p, although it will be less wasteful than the Raspberry Pi's camera sensor. This camera meets all of the needed requirements and manages to do so with lower power consumption, less wasted pixels, and it comes with a lens we were going to buy anyways if we couldn't get one included. Its price is also identical to

that of the Raspberry Pi camera, easily making it the front runner option when it comes to selecting a camera for this project.

5.2.3 Omnivision 5647 with Automated IR

One concern that arose during the search for a camera for this project was using the Backup Buddy device in a low-light or nighttime situation, a situation that the previous cameras didn't address. While visibility for the driver would be a concern in low-light situations, there is an assumption that the combination of illuminated back-up lights and tail-lights would provide enough visibility. Also safety is not a huge concern thanks to the use of ultrasonic sensors, which can "see" just fine in any light level, so even if the user has trouble seeing as they back out, the ultrasonic sensors will still provide accurate measurements to objects behind them. Still being able to include the option for an infrared lens in low-light would be another easy feature that could be tacked on to the project at no significant extra cost in terms of money.

Looking at the camera sensor itself, it is the same 5-megapixel Omnivision 5647 sensor as before with an active horizontal pixel count of 2592 and a vertical pixel count of 1944. It supports the same resolutions: 1080p, 720, and 480p, with the same amount of unused pixels should we opt for 720p recording. Each resolution comes with the same frame rate limits: 1080p at 30 frames per second, 720p at 60 frames per second, and 480p at 90 frames per second.

The biggest difference between this camera and the others is that it features not only a lens for infrared, but also includes 2 infrared lights to help the camera see in the dark. It is also capable of automatically switching between the regular daytime lens and the special infrared lens when light levels dip too low for its photosensitive resistor, with no special configuration required. The camera also comes with a 175° fish-eye lens for both regular and infrared lenses, making this camera the best option in terms of field-of-view.

The only real downside to this camera is that the addition of infrared lights and a motorized switch for the lenses means that the camera will require a larger amount of power to run. When the project is already operating on a very tight power budget as it is, the addition of new hardware that isn't even meeting a requirement, and consumes even more of our limited power, is a tough sell to the team. Regardless, this camera is a priced identically to the other 2 and packs the most features, making it a solid option, should we find more room in our power budget for new hardware features like infrared.

5.2.4 Camera Selection: Omnivision 5647

The selection of the camera to use for the Backup Buddy system is arguably one of the most important decisions that the team needed to make during the course

of this project. What we chose for the camera would have a domino-effect on further decisions regarding hardware. After carefully reviewing each of the cameras researched and discussed above and applying our mandatory criteria, the team selected the regular Omnivision 5647 camera. While the other cameras may have offered better resolution, higher frame rates, and infrared vision, for the purpose of this project, all we needed was a simple 720p camera that could achieve at least 15 frames per second.

Since we were limited already by the maximum supported resolution and frame rate by the port on the Raspberry Pi, any extra resolution from a sensor that exceeded those values was just not worth it to us, which ended up eliminating the official Raspberry Pi Camera V2 from our options.

Looking at the Omnivision 5647 with infrared, it was the exact same sensor but with the added benefit of supporting enhanced vision at night. However, this extra feature came at the cost of adding 2 infrared lights that would need to be supplied more power to take advantage of the camera's night vision. These lights would've resulted in increased power draw for the camera, which on a power budget as tight as ours on this project, would result in a decrease in the device's overall battery life, a metric that the team holds in high value. Considering the pros and cons made us realize that it was unlikely that we would even need a night vision sensor on the back of the car in pretty much any low-light situation. For example, when the car is put into reverse, two bright-white reverse lights that sit above the brake lights on the car's tail lights, turn on and illuminate the area surrounding the back of the vehicle. This would make having an infrared sensitive camera not very useful unless the driver was in a situation where all of their tail lights were out, a situation that the device is not being designed to fully handle. Even if a driver were to find themselves in a low-light situation where they needed to be more aware of their surroundings, the 3 ultra-sonic sensors could provide them with more than enough information about distances to objects around them. Since these sensors operate using high-frequency sound waves they would not be impacted by a lack of light. This would assist the driver in safely navigating the vehicle even in the darkest of environments, thus making the inclusion of an infrared camera unnecessary to meet the requirements of the project.

Eliminating those other two cameras from our options left us with the regular Omnivision 5647 camera that also came with a 160° fish-eye lens, providing us with a much better field of view than that of the official Raspberry Pi camera's 62° and only a slightly worse field of view than the infrared camera's 175°. By meeting our needs and not exceeding them unnecessarily, it easily made it our camera of choice for this project.

5.3 Sensors

To further extend the usefulness of the Backup Buddy device for users, the team needed to find ways to make it smarter by knowing more about the world around it and acting on the data that it received. After meeting and discussing what kinds of relevant data we would want to measure and analyze around the back of the vehicle, we were able to come up with a general list of sensors that we would need to enhance the functionality of the device. We came up with an accelerometer for measuring vehicle acceleration, a range finder of some kind for finding the distance to objects behind the vehicle, and a light sensor to measure ambient light and know whether the car was in a low light environment or not. With these sensors the team felt like we could help the system make smarter decisions that would make the device more useful to the end user.

While this project deals with a variety of different sensors that all measure different things, the team agreed upon a general set of criteria that all of the sensors that we choose should meet. First, much like the criteria for the microcontroller, all of the sensors that we choose to use for the project, should consume as little power as possible. To determine the impact of each sensor on the system, the power consumption of the sensor has to be considered in two different modes, an active mode in which it is actively taking measurements and a low power mode where the device is effectively asleep. Since the device is going to spend the majority of its time in the low-power mode, it is important that this mode is efficient. Also, the power consumption of the sensor in the active mode needs to be considered. While the device will spend significantly less time in this state, it is important to understand the worst-case electrical load that will be placed on the device's battery system while everything is active. The active and low-power modes of each sensor must be considered to help ensure that we get the longest battery life possible from the battery system.

The second and third criteria are closely linked, capability and affordability, the further we go in one direction the further away we get from the other. Essentially the more capable the sensor is the more expensive it gets, but the cheaper the sensor gets the less capable it becomes. The team needed options that were more focused on a balance between capability and affordability, options that didn't go too far in either direction. Considering that this system is supposed to enhance the safety of the driver and those around them as they back up, it's sensors need to be able to sufficiently handle all of the data coming in from a rapidly changing live-environment. Take for example a range finding sensor, used to give distances to objects behind the vehicle. If a child on a bike, travelling perpendicular to the vehicle backing out, were to cross into the path of the vehicle, it would be important for the sensors to quickly inform the driver of the sudden obstacle in the vehicle's path. However, if these sensors were running at too low of a refresh rate, say 5 Hertz, then there is a chance that the child on a bike would cross the entire field-

of-view of the sensor in under a fifth of a second, causing the sensor to miss seeing the sudden obstacle behind the vehicle. If the driver wasn't looking at the video feed as they were backing up and solely relying on the sensors then this could result in a dangerous situation. Hypothetical situations aside, whatever sensor is used in the system, it should be good enough that it doesn't hinder safety and affordable enough that it doesn't make the device too expensive for most users to be able to afford. With these criteria in mind, the following list of sensors for the various kinds of sensors needed was composed.

5.3.1 Accelerometers

One of the most crucial pieces of data that the team determined would be necessary for this project was figuring out if the car was in motion and in what direction it was travelling. While this usage developed into just needing to see if the car was moving at all, and not dependent on specific direction, we still needed an accelerometer for our features. Using the G acceleration data we can see if the accelerometer, and thus the device, is in motion. As part of a security feature of the Android application, knowing when the car is and is not moving is crucial.

5.3.1.1 MMA8452Q Accelerometer

Applying general sensor criteria to the MMA8452Q, it needed to be effective at taking measurements and consume little power at a low cost, which we determined it did. It possesses a triple-axis digital output accelerometer with up to 12-bits of resolution, more than enough for the needs of our project. It also offers a range of user selectable scales to use with either ±2G, ±4G, or ±8G, giving us great flexibility in what we can use it for. The data rates at which it can output also offer a wide range of options from as low as 1.56 Hz all the way up to 800 Hz **[21]**, it does so across an I2C interface, which our microcontroller of choice, the MSP430FR59691 supports. As for power consumption, this device offers incredibly low current drain in its active mode, ranging from 6µA to at most 165µA, making it a great choice for extremely low power situations like the one we face with this project. It also takes in a supply voltage between 1.95V to 3.6V, which is low enough that it can be powered with a connection to one of the microcontroller's GPIO ports. To help cut down on power-consumption even more, the device also offers motion detection, keeping itself in a low-power mode and asleep, until it is needed again. In the case of monitoring car motion, this is extremely useful as most of the time the device will sit, waiting to be awoken by user activity. If the parts that we buy were to take care of putting themselves to sleep and waking themselves up when needed, this would cut down significantly on the complicated task of programming and interfacing with all of the embedded systems that will be integrated into this device. All of these features make the MMA8452Q a very solid option for the team to consider when trying to decide on an accelerometer sensor

to use. To help summarize all of the features of this device, it's characteristics are presented in the following table 9 below:

	Values
Supply Voltage	1.95V to 3.6V
Offered Scales	±2G, ±4G, ±8G
Output Data Rates	1.56 Hz to 800 Hz
Digital Resolution	8-bits or 12-bits
Current Drain	6µA to 165µA
Programming Interface	I2C
Extra Features	Automated sleep and wake on motion
Cost	$2.95

Table 9: MMA8452Q Features

5.3.1.2 ADXL335 Accelerometer

While the previous accelerometer seemed like a pretty good choice, the team wanted to keep looking and see what else we could get for about the same price, or how much better we could go for just a little more. The second accelerometer that the team considered using for this project was the ADXL335. Much like the previously considered accelerometer, the ADXL335 is also a three-axis sensor with very low power consumption.

However, unlike the previous sensor the ADXL335 does not output its measurements digitally, instead opting to provide analog voltages that are proportional to the acceleration that it is measuring. Applying the general sensor criteria to the device it is clear to see that it passes these requirements quite handily. In terms of capability the ADXL335 offers a significantly wider range of output data rates, ranging between 0.5 Hz all the way up to 1600 Hz in the X and Y-axis **[22]**. While its output data rate can only reach up to 550 Hz for the Z-axis, most of our concerns when it comes to measuring acceleration would be focused on movements in the X and Y-axis only. It also offers only one scale of ±3G. While this gives us less flexibility than the previous accelerometer, the team also has to consider whether we even need all of the extra options that are present on the MMA8452Q. With regards to electrical characteristics, the device takes a supply voltage of 1.8V to 3.6V, also giving us the option to solely power it over the GPIO pins of the microcontroller, a big plus when it comes to how easily we can control overall power consumption through software. It also offers a very low current drain of about 350µA during operation, although this is more than twice the worst-case

consumption for the previous accelerometer. It also lacks some of the extra features noted in the previous accelerometer, like an automated sleep and wake when motion is detected. To better summarize the features of the ADXL335 they are presented in table 10 below:

	Values
Supply Voltage	1.8V to 3.6V
Offered Scales	±3G
Output Data Rates	0.5 Hz to 1600 Hz (X and Y-axis) 0.5 Hz to 550 Hz (Z-axis)
Digital Resolution	None, analog only
Current Drain	350µA
Programming Interface	None, analog only
Extra Features	No programming necessary
Cost	$2.05

Table 10: ADXL335 Features

5.3.1.3 TDK ICM-20948 Accelerometer

Considering the first two accelerometers only offered G acceleration data, we looked into a 9-axis model as well. The TDK chip we found gave us not only acceleration but a gyroscope and a compass. These extra registers and information seemed appealing, as it would allow us to implement extra features as stretch goals if it came to it, and since the price was not much more, it appealed to our needs.

We decided not to go with this model, as there wasn't as much documentation or resources available for us to use during development, and at the time of purchasing we didn't thing we would need anything for the compass or gyroscope features. Below is the data sheet used in our decision.

	Values
Supply Voltage	1.8V to 3.6V
Output Data Rate	400kHz to 7MHz
Digital Resolution	16-bit
Programming Interface	SPI or I^2C
Cost	$5.10

Table 11: TDK Accelerometer Features

5.3.2 Light Sensors

While the original intent of adding a light sensor to the device was to help the camera system decide when to switch between a regular lens and one for infrared night-vision, as we searched more for parts plans changed. Finding a camera with components already integrated into it for detecting low-light made the original purpose of the light sensor unneeded. When we decided to go with a camera that lacked light-sensing systems due to the lack of beneficial features that an infrared camera offered for the user, this put the light sensor in an awkward position. However, after further research into light sensors and how little power they consume, it was decided to continue ahead with plans to integrate one into the device and try and figure out new features for it later. Some ideas include possible notifications or reminders to the driver to be more careful and be sure the area behind them is clear before backing out in a low light environment.

5.3.2.1 OPT3001 Light Sensor

The OPT3001 caught the team's attention because of how closely it tries to measure light intensity with respect to the human eye. Meaning the measurements that it takes will effectively allow the system to understand environmental light levels in the same way as the user would. This leaves little room for misunderstanding between what might be considered a light or dark environment between the user and device. It also features strong infrared light rejection, meaning the sensor could be placed behind dark glass, in the case of our project for protection purposes from hazardous environments, with little impact on its accuracy.

Further comparing the device to our general sensor requirements, it is clear that it is more than enough for our needs. Capability-wise, the device is able to detect and measure light from as low as 0.01 lux to as high as 83,000 lux, offering a 23-bit effective dynamic range and further fine-tuning with automated gain ranging **[23]**. Considering things from a power consumption perspective, the device takes an input voltage of 1.6V to 3.6V, a range that is low enough that its power could

be supplied using a single microcontroller GPIO pin. During active mode, the current drain from the device varies between 2.5µA to 3.7µA, a value so incredibly small that it would have little impact on the overall battery life of the device. The device is also capable of taking measurements continuously or, for a more power-conscious design, in a single-burst. The output results from the device can be calculated in either 100ms or 800ms at the cost of higher power consumption. In the case of our project, we would likely opt for the slower calculation time, as light sensing is not critical to user safety. The device also communicates over the I2C interface which would make it easy to integrate into with other potential sensor systems that also take advantage of this same interface. Overall, the OPT3001 offers many useful features on-top of its already low power-profile, making it a great option for a light sensor. All the key features of the OPT3001 are summarized in table 11 below:

	Values
Supply Voltage	1.8V to 3.6V
Measurement Range	0.01 lux to 83,000 lux
Output Data Rates	100ms or 800ms
Digital Resolution	23-bits
Current Drain	2.5µA to 3.7µA
Programming Interface	I2C
Extra Features	Matches human eye light sensitivity
Cost	$2.34

Table 12: OPT3001 Features

5.3.2.2 MAX44009 Light Sensor

Like previous sensors that the team considered, we wanted to see how much better of a light sensor we could get for a little more money before deciding if the sensor was worth it. This brought us to the MAX44009 light sensor. Much like previous light sensors, the MAX44009 was optimized to be as close to the human eye level of light sensitivity as possible, a big plus in a project where user safety is considered with respect to human senses. We were pleasantly surprised to find that the MAX44009 was a considerably better sensor than the OPT3001 for only a slightly higher price.

In terms of capability, the MAX44009 is extremely sensitive to light, offering a range that goes as low as 0.045 lux and as high as 188,000 lux with 22-bits of dynamic and tune-able digital precision **[24]**. It offers both infrared and ultraviolet

blocking capabilities, which would mean it is totally safe to place behind a darker piece of glass to protect it, without hindering its ability to take accurate measurements. Looking at the power consumption of the sensor, it takes in a voltage supply between 1.7V and 3.6V, again meaning that it could be powered through a single GPIO pin on the microcontroller. In terms of current drain, the sensor offers exceptionally low current drain with a maximum operating current in active mode at 0.65µA. This is nearly 3 times less than the minimum current drain on the OPT3001. Still when comparing a few microamps of current, there will be virtually no difference in terms of battery life so while Maxim can make the claim to be the "industry's lowest-power sensor" the difference between it and other sensors is so low that the difference it would make in a project as battery sensitive as ours is insignificant. However, the MAX44009 is still a very impressive sensor and offers significant improvements to other light sensor options for the team and can't be easily ruled out, especially at the small increase in price. The key features of the MAX44009 are noted in table 12 below:

	Values
Supply Voltage	1.7V to 3.6V
Measurement Range	0.045 lux to 188,000 lux
Output Data Rates	100ms
Digital Resolution	22-bits
Current Drain	0.65µA
Programming Interface	I2C
Extra Features	Matches human eye light sensitivity
Cost	$3.57

Table 13: MAX44009 Features

5.3.3 Ultrasonic Sensors

One of the main safety features that the team wanted to implement on the Backup Buddy device was collision detection with objects that appeared behind the vehicle, outside of the user's peripheral vision as they backed up. At first, the team considered trying to use the camera system itself combined with computer vision to track moving objects behind the vehicle and warn the user if the computer thought one of them was at risk of being hit. The idea was that we could off-load the heavy lifting and computation onto the user's phone and improve safety while not hurting the power consumption of the device. However, as none of our members were proficient in computer vision and training AI using datasets, this idea was dropped in favor of using a series of low-power ultrasonic sensors

instead. Ultrasonic sensors work by firing a series of high-frequency sound waves out and listening for them again as they bounce off of objects nearby and collide with the sensor again. As the waves continue to hit the sensor a pin goes high to indicate that the high-frequency sound waves are being received and stays high until they stop. The duration that this pin is high is used to determine the distance the sound waves travelled between the sensor and whatever they collided with.

Research on ultrasonic sensors pulled up surprisingly few results that met our general sensor requirements but also introduced us to some new requirements that we had not previously thought of. For example, the maximum range of the range finding sensors and the field of view the sensor has. In the end we narrowed the list down to 3 different sensors, however 2 of them were essentially the same exact model with the only difference being that one of them was slightly newer and had an extra pin added to it to help limit the number of GPIO pins the sensor would need to operate. The last sensor ended up being considerably more expensive than the others and while it met our sensor requirements, was dropped from contention, leaving us with the same sensor with a small increase in price for the newer one, which lead us to select the HC-SR05.

5.3.3.1 HC-SR05 Ultrasonic Sensor

The main sensor that the team focused most of our research on was the HC-SR05, the updated version to the very similar HC-SR04 ultrasonic sensor. The main difference between them being that the HC-SR05 has one extra pin, allowing for a new mode where the device can be told to transmit high-frequency sound waves and listen for them coming back on the same pin. The team decided that the 50% reduction in the number of GPIO pins used would be well worth the extra cost added on from buying the newer sensor.

Looking at the capability of the HC-SR05, it runs at a maximum rate of 20 Hz, meaning we are able to see what's behind the vehicle every 0.05 seconds. If anything, or anyone were to enter the field of view of one of these sensors, they would need to be travelling very quickly to avoid being picked up in the 0.05 second downtime between each sound wave blast. The device can accurately measure distances to within one-tenth of an inch of resolution and can detect any potential hazards behind the vehicle within a range of 0.8 inches to 177 inches (or almost 15 feet) **[25]**. The HC-SR05's field of view is approximately 15°, which isn't all that great, but if we integrate multiple sensors into our device, we can safely cover blind spots on both the driver and passenger sides as well as directly behind the vehicle, creating a net of visibility around the back of the car. Since the device is more than capable of meeting the requirements of this project, the next question was, what are the electrical characteristics and how much of a hit will we take to battery using multiple HC-SR05s.

Looking at the electrical characteristics, the HC-SR05 takes between 4.5V and 5.5V as a supply voltage. Unfortunately, this falls outside the range of voltage that can be supplied from any of the GPIO pins on the microcontroller, which means we cannot directly wire it up to the board. However, using something like a transistor and a voltage source that falls within this range, we can still find ways to control the power on the HC-SR05 with the GPIO pins. While the manufacturer rates the device's current drain at 15mA, this would be a worst-case scenario, and optimizations like lowering the data output rate of the device can be exercised to further lower the power consumption and current drain to more efficient levels. Overall, the low price and great specs make the HC-SR05 a good choice for finding range on this project. A summary of the features is given in table 14 below:

	Values
Supply Voltage	4.5V to 5.5V
Measurement Range	0.8 inches to 177 inches
Output Data Rates	50ms max
Digital Resolution	0.118 inches
Current Drain	15mA
Programming Interface	Digital
Cost	$4.00

Table 14: HC-SR05 Features

5.3.3.2 HC-SR04 Ultrasonic Sensor

The HC-SR05 and the HC-SR04 were very similar in terms of their data sheets and performance, however the SR04 has one less pin on its module. The extra pin on the SR05 allows for us only needing a single GPIO pin on our microcontroller, but there was far more documentation and resources regarding the SR04 model, as it was clearly the sensor of choice for this functionality. The data sheet was the same as the SR05, being the same sensor. Looking back at our decision to use the SR05, the SR04 would have been a better choice, since the single GPIO function of the SR05 did not operates advertised, and we essentially used it as an SR04.

5.3.3.3 SU04 Ultrasonic Sensor

The main downside to the SR04 and SR05 was their size, which is why we were considering the SU04. It has only a single trigger module that acts as both the

trigger and echo for the sensor, taking up half of the size of an SR04. It featured a range similar to the SR04 but wouldn't detect anything unless it was at least 40cm away. With a much larger field of view and smaller form factor, the pros much out weighed the cons. However, since we needed 3 of these sensors price was an important consideration. The SU04 cost 5 times as much as SR04, which is why we did not end up using it. Below in table 15 shows the data sheet of this sensor.

	Values
Supply Voltage	4.5V to 5.5V
Measurement Range	40cm to 450cm
Output Data Rates	50ms max
Current Drain	15mA
Programming Interface	Digital
Cost	$14.99

Table 15: SU04 Features

5.3.4.1 Sensor Selections

Following up on all the research done into the various sensors required for this project, the team met and compared the specs of each device to the requirements originally decided upon in our general sensor criteria. It was also during this meeting that the team also decided to drop the light sensor from the design. Originally, it's purpose would've been to help the camera know when to switch from a regular lens to an infrared one to assist with backing up in low-light environments. However, once a camera was discovered that was already capable of doing this on its own, the purpose of the light sensor was greatly diminished. Still, not wanting to completely take out a sensor from the system, the team gave time for members to try and come up with another potentially useful way to utilize a light sensor on the project. This resulted in ideas like displaying specific warnings to drivers to drive more carefully in low-light, but still this was a task that could easily be accomplished by getting the system time in the app. Eventually, when the team decided to drop the idea of an infrared camera from the back of the car, the decision was also made to drop the light sensor from the project as well. In doing so the schematic and PCB design would be less complicated, and despite the researched devices very low power consumption, some battery life could be saved over a longer period of time. This left the team with 2 major decisions to make regarding the sensor systems for the project, choosing both an accelerometer, and an ultrasonic sensor.

Of the 2 different accelerometers considered, the MMA8452Q, and the ADXL335, the team chose to go with the MMA. The biggest draw being that its output was digital and very easy to integrate with our microcontroller in comparison to the ADXL and its analog output. While the MSP430 microcontroller that we are using does in fact possess an ADC (analog-to-digital converter), the team's lack of experience in using it would mean that more time would need to be dedicated to first learning and then integrating the sensor. This is in comparison to the MMA, in which we can simply wire it into the digital I2C pins on the microcontroller and be good to go. In terms of features, the ADXL had better polling rates in the only relevant axis for this project (x-axis) with 1600 Hz compared with the MMA's 800 Hz. However, for the needs of the project, the team found it unlikely that we would ever need to utilize such a high polling rate, making this feature more of an unnecessary addition. Another big concern for this project is the amount of current that the device draws as it will impact the amount of power the active system consumes and ultimately how much battery life the device will have. With the MMA, there are a wide variety of options for low-power mode's that enable the device to draw as little as 6µA of current while active. In the case of the ADXL, such features are not present with the system drawing a flat 350µA or 0.35mA while active. This is significantly higher draw then the MMA and would have a measurable impact on the device's overall battery life. For the team, this was the largest deciding factor when considering the 2 accelerometers, a nearly 200% increase in current draw was deemed too high to be acceptable for the needs of this very low-power project.

The last sensor that the team needed to pick, didn't really have much of a choice. The only choice we had with the ultrasonic sensor came down to whether or not we should use an older model or a newer one. The difference between them being the addition of a single pin, which when tied to ground, would mean that the sensor's trigger and echo pins could be handled through a single GPIO pin, instead of one pin being used for trigger and another pin being used for echo. Considering that the system was going to be utilizing 3 of these ultrasonic sensors in its final design, it made the most sense for the team to choose the newer model, the HC-SR05. While this did mean that costs would be slightly higher, the difference between the two was small enough to be deemed worth the extra cost per sensor. With all of the sensors being used in the system decided upon, it was now time to begin integrating them into the microcontroller and begin developing the schematic that would tie all of these sensors together onto one PCB.

5.4 Wi-Fi Module Consideration

In this section it will cover the various modules that are in consideration for this project these Wi-Fi modules will provide wireless communication capabilities to our microcontroller allowing it to communicate sensor information to the user's device and the system. In these sections we will take into consideration not only the power

consumption of these modules but the throughput these modules would provide to the microcontroller.

5.4.1 ESP32-D0WDQ6 Wi-Fi Module

The ESP32-D0WDQ6 Wi-Fi chip manufactured by Espresso is a 2.4 GHz Wi-Fi module. This chip is a small form factor chip that can be used to add Wi-Fi capabilities to this project. The ESP32 is compatible with I2C, I2S, SDIO, CAN, and most importantly SPI and UART. This chip retails for $3 which makes it quite an affordable chip and has a transfer rate of 150 Mbps. In Table 16 there is a list of this chips specifications which will be used in comparison with the other Wi-Fi chip that is being considered.

Specification	**Value**
Sensitivity	-97dBm
Frequency	2.4GHz
Supply Voltage	2.3 to 3.6V
Supply Current	95ma - 100ma
Transfer rate	150 Mb/s
Program memory size	448kB
Power output	20 dBm
Interfaces supported	I2C, I2S, SDIO, CAN, UART,SPI
Mounting style	SMD/SMT

Table 16: ESP32-D0WDQ6 Specs ***[26]***

In the table 14 above, it can be seen that the Expressif ESP32 Wi-Fi chip is a full featured Wi-Fi chip that can provide communication capabilities to a microcontroller. Though this chip is commonly used for the Arduino line of devices it can work and function with others as it features a full range of communication protocols that any microcontroller can use to communicate with the device.

5.4.2 ESP8266 Wi-Fi Module

The ESP8266 Wi-Fi module is another Wi-Fi chipped manufactured by Espressif systems. This is another small form factor module that will give our microcontroller the ability to connect to a Wi-Fi network. This module comes is being widely used in the industry at large with the huge demand of the IoT market. This module is built with power consumption in mind with varying modes which include active

sleep and deep sleep greatly reducing the consumption of power for our device which is important as this device will be completely wireless, so we need to achieve a long-lasting power life. This module has the capabilities of handling the elements as the operating temperature of this chip can range from -40 C to 125 C.

Specification	Value
Sensitivity	-72dBm
Frequency	2.4 - 2483.5GHz
Supply Voltage	2.5 to 3.6V
Supply Current	80 mA (average)
Transfer rate	72 Mb/s
Program memory size	1M B
Power output	20 dBm
Interfaces supported	I2C, I2S, ADC, UART, SPI
Mounting style	SMD/SMT

*Table 17: ESP8266 Specs **[27]***

As seen in the table above the supply current averages at 80mA but during its three modes tis consumption varies drastically. In deep sleep mode it needs about 20uA of power to run and less than 1mA to stay connected to an access point. Though when in active mode and delivering 54Mbps with a power output of 15dBm it draws about 140mA at 3V. And at 11Mbps with a 17dBm power output it draws 170mA. This is a significant power consumption spikes of about 0.42 watts that we have to keep in mind when positioning the device and having a powerful enough microcontroller to power the unit.

5.4.3 Wi-Fi Selection

Comparing both the ESP8266 and ESP32-D0WDQ6 Wi-Fi modules they are similarly specked with the ESP8266 offering the ability to put the device in low power mode a better consideration over the ESP32. But the ESP32 offers a higher amount of throughput and with a limited amount of compute power that is offered by micro-computers transcoding the video live is not an option so this high throughput would allow the system to transfer HD video to the users device with little to no transcoding this not only keeps power down but reduces the amount of performance the microcomputer needs to provide in the system.

This Wi-Fi module fit everything that we needed, however when it came time to choose a microcomputer, the selection that we made had built in Wi-Fi, negating our need for an external component. This module could still be of use to us in terms of long term development of our design, as the Raspberry Pi has more features than we needed, and if we were to try and combine the Pi and our PCB, we would need to use the

5.5 Bluetooth Module Consideration

In this section it will cover the various Bluetooth modules that are up for consideration of this project. This section goes into detail of the various modules that could be used with our microcontroller to provide Bluetooth capabilities to our system. These modules will vary from the various Bluetooth versions and their power consumption and which would fit best in our system.

5.5.1 HC-06

The HC-06 Bluetooth module is manufactured by Sunfounder. This compact Bluetooth module can give the microcontroller the ability to interact with Bluetooth devices. This module works well with the Arduino microcontroller. The HC-06 can only act as a slave in the Bluetooth relationship meaning that another device in the network must be an master. This Bluetooth module supports the SPP which is the simple pairing protocol which means the master device must be able to support the protocol. Table 16 below goes into the specifications of this module.

Specification	**Value**
Sensitivity	-84dBm
Frequency	2.4GHz
Supply Voltage	3.3V
Supply Current	50mA
Transfer rate	1 Mb/s
Bluetooth version	2.0
Power output	4 dBm

*Table 18: HC-06 Specs **[28]***

From the table we can see that this Bluetooth module supports a high bandwidth transfer rate of 1Mb/s this is because it is only a slave mode. Since the slave mode it takes in the data and is treated as serial input and is not modified by any code for the device. This device has an operating range of 30ft though by going up to this 30ft range the latency of the device drops off significantly.

5.5.2 HM-10

The HM-10 Bluetooth module is designed and manufactured by Sunfounder. This module was made for the Arduino Uno. This module is useful for connecting and transmitting commands to any device wirelessly though it was designed for the Arduino Uno it can work for any device that can interface with a UART interface. Being that this module is a commonly used one gives a vast amount of support and documentation of the module. Table 17 below shows the specifications. From the table we can see the HM-10 module is a 4.0 Bluetooth capable module. This module features the simple pairing protocol as its vital to this project allowing the system to quickly pair with the user's device. With quick pairing the module will allow the system to boot from low power mode to be ready for the user. This module also features a verbose data sheet which contains a schematic of the device. This module is built on top of the CC2540 offered by TI because of this chip being part of the TI line it adds to the credibility of the chip and also the module. This chip is rated for 2 - 3.7 volts and consumes a maximum of 50 mA allowing it to be a very low power module to be as effective as possible on battery time.

Specification	**Value**
Sensitivity	-84dBm
Frequency	2.4GHz
Supply Voltage	2-3.7V
Supply Current	50ma
Transfer rate	6 kbp/s
Bluetooth version	4.0
Power output	4 dBm
Interfaces supported	UART

*Table 19: HM-10 Specs **[29]***

5.5.3 RN-42

The RN-42 Bluetooth module is a module made to work with many microcontrollers by default this module comes with an external antenna this allows for a simple setup to get Bluetooth functionality added to the system. The module allows for connecting by UART or SPI. For under 20 dollars it comes with a ready to use Bluetooth module capable of delivering 3Mbps for a distance of 20 meters. From table 18 this module has a respectable transfer rate of 1.5 Mbps using the older Bluetooth 2.1 technology which is good as it allows for the simple pairing protocol. This protocol will allow the users device to connect to the system in a timely manner which would allow the system to boot up from low power mode and be ready for the user to use. The power draw on the module is at the active mode as the module has the ability to be programed to be in low power mode. This chip features encryption abilities which allows the systems to securely communicate with the system and the user's device not allowing any form of eavesdropping. The RN-42 chip is capable of acting in a low power mode enabling the system to save as much power as possible.

Specification	**Value**
Sensitivity	-80dBm
Frequency	2.4GHz
Supply Voltage	3.3V
Supply Current	30ma
Transfer rate	1.5Mbps burst to 3Mbps
Power output	4 dBm
Interfaces supported	UART, SPI
Specification	Value

*Table 20: RN-42 Specs **[30]***

5.5.4 HC-05

The HC-05 module is Bluetooth module that is made to work with the Arduino lines of microcontrollers but will work with others that can connect via UART or I2C, or

USB. This module uses the frequency hopping spread spectrum which allows it to uniquely connect to other devices. The module has two operating modes which are data mode and AT command mode. The data mode is of course for sending data and the AT command mode is for sending commands to module to execute. From table 19 we can see that this device has a higher transfer rate than the other modules and has the benefit of being both the slave and the master device. This device has a range of about 100 meters but like the other chips as you go further from the module the latency becomes more and more apparent. Though from the table you can see this device is a Bluetooth 2.0 device which is before the simple pairing protocol came to be, so the device would be a lot slower to pair with than a 2.1 version. Since our system is using the Bluetooth capabilities to power on the device fully as the user comes into range to be as power efficient as possible the system needs to be paired with the user device as quickly as possible.

Specification	**Value**
Sensitivity	-97dBm
Frequency	2.4GHz
Supply Voltage	4-6V
Supply Current	30ma
Transfer rate	3 Mb/s
Bluetooth version	2.0
Power output	4 dBm
Interfaces supported	USART, I2C,USB

Table 21: HC-05 Specs ***[31]***

5.5.5 Bluetooth Module Selection

From these four Bluetooth modules the HM-10 is the one that was initially used for the system. This has come from comparing the four modules together they allow feature capable interfaces that work with the microcontroller chosen for the system. Though the HC-05 and the RN-52 use less power than the HM-10 and the HC-06 the HC-05 uses Bluetooth 2.0 which means that module lack quick paring which is needed for the system to have the microcomputer awaken from sleep. Though the RN-52 is a Bluetooth 2.1 device and allows for quicker pairing it is still an older

technology that compared to that which is on the market currently and the module is also almost double the price of the HM-10 which is a constraint of this project to keep the cost low not only for group members but for the possible users of the system. The HC-05 uses the same amount of power as the HM-10 but is only Bluetooth 2.0 which means it lacks the ability to quick pair and uses more power than the other devices. Communication throughput is not necessary to be considered as after more research the system cannot transcode the video to the low enough bitrate that can be transferred over the Bluetooth protocol, so the system will use Bluetooth to detect the user's device and boot up the system as well as communicate sensor data between the user's device and the system. From all of this the module choice of the HM-10 beats out the others and makes a good fit for the system as it is full featured and not the lowest power consumption out of the lot but still low. As well as the module being the cost appropriate compared to the others.

During our testing, we found that the HM-10 didn't hold up in terms of range and data transfer. Due to it using Bluetooth Low Energy, the range was not the same as that in the data sheet, and we were constantly missing data transmissions. Because of this we chose to purchase the HC-06. As a slave device only, it allowed us to simply send things to the UART interfaces on the microcontroller and read the buffer on the Android app side. The lack of BLE was a downside, but the savings in energy were deemed mute if we were losing as much data as we were.

5.6 Wireless selections

For this system with all of the wireless options listed for Wi-Fi, Bluetooth and Zigbee. The microcontroller will be using Bluetooth this is due to its lower power consumption. Though it is comparable with Zigbee Bluetooth eliminates the need for a receiver station that will convert the signal to something that the user device can interact with. Though the microcontroller will not be using Wi-Fi to transmit data the raspberry pi will as it is the only wireless option that can transmit data at a fast-enough rate to ensure quality. From the options of Bluetooth devices, we have decided on the HC-06 as it does not have BLE and has a more consistent range and little to no data loss.

5.7 Solar Panel Consideration

In this system there will be no wires to connect to the car meaning that any power sourced would have to come from within the device's batteries. These batteries would eventually lose charge and at that point the user may need to detach the device from their car and plug it in to charge it. With the use of solar panels in the right environments we would eliminate the need for the user to detach the system and charge it and instead the system will run off of renewable energy. For these

reasons, we decided solar charging would be a stretch goal, and wanted to decide on the best solar panels to use with our system. While we did not implement solar panels with the system, the system does allow for the addition of solar panels if they are wired to the appropriate DC jack that is already implemented on the PCB.

To select panels for this project the size and shape have a major impact on which panels could be selected as the stretch goal for this project. The weather may impact the solar panels ability to generate power. For this solar panels they will be used to charge up the lithium ion batteries to allow an even more extended battery life. These batteries cannot be trickle charged so in the consideration of the panels they would need enough to power the charge controller. There are three current solar technologies we have in mind for this project being monocrystalline, polycrystalline, Thin-Film panels.

5.7.1 Monocrystalline solar panels

For this project Monocrystalline panels are the most efficient meaning we would need less to get more power than competitors which is useful as this system needs to be within a certain size with only a limited surface area for panels. This solar technology has a good low light rating which could be useful for climates that are not as sunny as Florida **[32].** This technology is set to last longer than others. Though with these benefits there are draws to the monocrystalline panels one being cost which has to be greatly taken into consideration as the system needs to fall with in the market bounds to be effective. A more major downside is the durability of this panel technology being that if not properly protected these panels can easily be broken from extreme windows or debris that the car might throw up on the back end. If we were to use this panel technology, we would have to ensure that they don't flex and are sheltered from the elements. As well as to take into consideration how this panel technology handles shading which is something a car may experience often which has been found to greatly hinder the panels lifespan.

5.7.2 Polycrystalline solar panels

Polycrystalline panel technology is another solar technology we considered for this project. With polycrystalline cells they are cheaper monocrystalline which in this project will be useful to keep the system marketable. As with age they follow that of the monocrystalline panels lasting for a good amount of time. Though their drawback is efficiency as they are less efficient than the monocrystalline panels which will probably lead to us not using these panels in our system as you save some money, but the power gain needs to be as efficient as possible to make use of the small amount of surface area available to us. These will have to be heavily considered over the monocrystalline panels if we can get enough power out of

them from the small amount of space to justify the cost savings that the polycrystalline would provide.

5.7.3 Thin-Film panels

The last solar technology in consideration are Thin film panels feature something the last two panels don't the flexibility of these panels add durability which is definitely more suitable for our system [12]. As having these panels in the back of the vehicle it will definitely succumb to some sort of flex and bending from the high speed and the rough terrain. Another feature of this panel is the ability to handle shade without damage to the cells this is a definite benefit for our system as the vehicle could be parked in a parking garage and could be completely shaded from the sun or just the angle the sun may be at when the car is parked cannot hurt the solar panel in this system which would make this a definite pro over the other two technology choices. These panels are of lower cost like the polycrystalline which is a benefit, but they also share the same fate with the drop-in efficiency **[32]**. Though these panels are cheaper, can handle shading, flexible their life is not as long as the other two technologies which may be a trade of we have to make.

From table 20 below, we see that though monocrystalline is more efficient but the tradeoff is cost. As for the two lowest cost options polycrystalline and thin film solar the choice would be thin film. Thin film features a flexible design which allows it to be more durable to the polycrystalline panel. Thin film also works in shaded environments without harming the panel. Though there is a trade off in efficiency it adds up over time to have a panel that will last longer. As functioning in the shade is one of the most important aspects of a solar panel technology in this design being that a vehicle can spend quite some time in shaded environments. Though from researching and locating thin film panels for this system it appears that the amount of small sized thin film panels are small as it appears that one manufacturer has the hold on smaller sized panels and one other off brand panel. To have some diversity in options from different manufactures the option for polycrystalline panels will be included as they are another low-cost alternative for panels with the slight trade off in efficiency but still good as it will fit the small system.

Specifications	Thin-Film	Polycrystalline	Monocrystalline
Cost	Low	Low	High
Durability	Highest	Medium	Medium
Function in shade	Yes	No	No
Efficiency	Low	Medium	High

Table 22: Solar Comparison

5.7.4 Flexible Thin-Film Solar Module MP3-25

In this section we will be discussing the flexible thin-film solar module made by Powerfilm. Powerfilm is an American based company that specializes in manufacturing flexible thin film solar panels and has a reputation of making good products. As adding solar to this project adds complexity to the PCB and weight and an increase in size to the overall project it is important to choose a small form factor panel that can charge the system. As Powerfilm provides numerous small form factor solar modules it is important to select the one that meets the needs for the system.

As seen in table 23 the MP3-25 solar module is a smaller form factor module that is meant to recharge 6 and 12-volt batteries. At 25mA at 3V this panel by itself will not be enough to charge our system we will need at least another panel to overcome the voltage needed to charge the lithium ion batteries and even then, may need at least one other to make up for inefficiencies needed from the charge controller and the varying strength of the sun throughout the day.

Specification	Value
Operating Voltage	3 (volts)
Operating Current	25 (mA)
Weather Resistance	None
Size L x W x T (inches)	4.5 x 1.0 x .01

*Table 23: Solar MP3-25 Specs **[33]***

5.7.5 Thin-Film Solar Module MP3-37

In this section we will be discussing another solar panel manufactured by Powerfilm. As this panel is another Powerfilm panel there is not much of a difference in their thin film line other than the slight change in either voltage or current and a change in size. This panel as the last is part of their smaller line of panels that would fit out design goals. As their other panels that offer weather resistance features are much larger than what is needed in of requirement specifications.

From the table 24 below, it is seen that this panel has a 3V operating voltage and a 50 mA operating current which means we would still need about two of these panels to charge the device.

Specification	**Value**
Operating Voltage	3 (volts)
Operating Current	50 (mA)
Weather Resistance	None
Size L x W x T (inches)	4.5 x 1.5 x .01

*Table 24: Solar MP3-37 Specs **[33]***

5.7.6 Thin Film Flexible Solar Cell

This thin film solar panel comes from an eBay seller with no reference to the brand. This being an off brand solar panel does run some risks as the only reviews for the device are those available from the eBay's seller page. They do offer a generous return policy and he panel is a stated to be a waterproof panel that can handle being exposed to the elements. Table 25 shows the specifications of the cell.

Specification	**Value**
Operating Voltage	1.5 (volts)
Operating Current	330 (mA)
Weather Resistance	Yes
Size L x W (mm)	195 x 50

*Table 25: eBay thin Film Specs **[34]***

Though this is an unbranded solar panel it does have some useful features and would still be a contender if the specifications match the data sheet. The panel having a 1.5-volt operating voltage means we would need about 4 panels to overcome the voltage of the batteries to charge them but comes with the benefit of 330 ma which could almost supply enough power to cover the system completely while operating. The panel is of a smaller design which fits our systems needs it is just to justify if this panel is worth the risk as the shipping time to receive one is almost 2 months.

5.7.7 Panasonic BSG AM-8801CAR

This solar panel offered by Panasonic which is a well-known company. This panel offers two pre-soldered on leads that then can be used to hook up to the charge controller. This panel is not weather resistant but can be remedied by being put under an acrylic screen to protect the device and allow it to still get sunlight as it is needed.

Specification	**Value**
Operating Voltage	4.5 (volts)
Operating Current	41.9 (mA)
Weather Resistance	No
Size L x W (mm)	57.7 x 55.1

Table 26: Panasonic BSG AM-8801CAR ***[35]***

This panel from table 24 features 4.5 V and 41.9 mA for this panel to work with our system we would need multiple to get over the input voltage needed to use many of the charge controllers for the system. This panel is significantly smaller than other panels and because of this the system could easily fit two to hit the voltage requirement the only downside is if the system needed more current with this panel it would need at least four panels to up the current as with the sun we would not get this current all the time and would only be around peak times so we would definitely need more to make a significant charge gain to the system.

5.7.8 MikroElektronika MIKROE-651

This panel is from the manufacturer MikroElektronika this panel offers two pin points on the underside of the panel for the system to connect to and has a very limited data sheet. Though this solar panel is offered through a reputable

manufacturer who specializes in embedded systems, where the specifications can be seen in table 27 below.

Specification	Value
Operating Voltage	4.0 (volts)
Operating Current	100 (mA)
Weather Resistance	No
Size L x W (mm)	70 x 65

Table 27: MikroElektronika 651 Specs ***[36]***

Though this panel has a lacking data sheet in the and lacks some useful information on other devices this product is commonly used in it is still a fit for our system based on its specifications. This panel like many of the other would need more than one to work in our system to overcome the voltage needed to be used with the charge controller. This panel is still low on the amount of current meaning it will take longer to charge the batteries and will probably lead to a deficit in charge unless we have multiple panels to up the current which would require more space to be taken up by the system.

5.7.9 Solar Panel Selection

From table 26 below, we can see the options are of the Powerfilm brand solar panels this is not due to favoritism but due the amount of market shares this brand holds on the smaller thin film solar panels. The eBay off brand panels could be of great use to this project if they meet their specifications but with time being an important constraint in this project, they may not be worth the risk as well as needed at least 4 of the off-brand panels to match the voltage needed with charging would be more expensive than the other panels. There is also the Panasonic BSG panel and the MikroElektronika panel which would be good contenders though their lack of ability to handle vibrations may hold them back in this system. These panels do offer higher currents and voltages than the thin film panels as these are the more efficient type of solar panels which means our system would still be able to generate power in low light environments as it would be enough voltage to overcome the charge controller. Between the MP3-37 and the MP3-25 there specifications are relatively the same with the only trade off being the extra 0.5 inches in width to accommodate the MP3-37 which would give another 25ma of charging power. Thought that trade off adds up to making the design at least 3 inches wide. If we were to have achieved our stretch goal of

adding solar, or if the end user was to desire to add this feature on their own, it seems best to choose the MP3-37 as the added current will allow the system to take advantage of the sun time it receives to charge the device. Also allowing us to quickly receive the device with a proper support channels and documentation.

Specification	Thin Film (eBay)	MP3-37	MP3-25	BSG AM-8801CAR	MIKROE-651
Operating Voltage	1.5 (volts)	3 (volts)	3 (volts)	4.5 (volts)	4.0 (volts)
Operating Current	330 (mA)	50 (mA)	25 (ma)	41.9 (ma)	100.0 (ma)
Weather Resistance	Yes	No	No	No	No
Size L x W	195 x 50 (mm)	4.5 x 1.5 x .01 (in)	4.5 x 1.0 x .01 (in)	57.7 x 55.1 (mm)	70 x 65 (mm)
Shipping time	30-40 Days	5 days	5 days	5 days	5 days

Table 28: Comparing Solar Options

5.8 Charge controller

As a part of this system being a completely wireless backup camera system the need for batteries draws the need for those batteries to be topped up with new power. In this section we will cover the charge controller options best suited for this system. These charge controllers will be a range of ones all featuring the battery selection we made of lithium ion and some having the capabilities to handle solar charging.

5.8.1 Texas Instruments bq24650 Charge Controller

This charge controller manufactured by Texas instruments and is meant for solar to handle solar power. This chip is widely used in solar applications and remote monitoring stations. It accommodates lithium ion batteries LiFeP04 and lead acid batteries. Based on these key elements this chip is well suited for our project as

we are hoping to meet a stretch goal to include solar capabilities which this one supports.

As seen from table 27 below this is a well featured chip that can benefit the system. Though it comes with a limit of 8 amps on the supply current which means the system would be able to handle enough power that the solar device or through a wall adapter it can give off a significant charge to the batteries. This chip does benefit our system as it features support for the batteries, we plan on using as well as lead acid batteries so if the design doesn't go to plan we can easily swap the batteries. The range for input voltage is somewhat overkill for the system current as the solar selection features panels that don't go far over eight volts. This chip features an extensive range in temperatures allow it to be used in an environment where we cannot control the temperature like the outdoors which is a great fit for this system. The chip features many extra capabilities allow not only basic charging above but is capable of detecting bad batteries which would be a great benefit to not only the user of the system but to us as we design the system this detection can be used in debugging to ensure the batteries are working and where the fault may be in the design or products. This chip also features cell temperature monitoring which is useful as to allow us to protect the device and the batteries without having to design this system to check on the batteries. As with the plan being to use lithium ion batteries it is useful to have the ability to monitor the temperature of the cells as if these cells operate in unsafe temperatures or reach unsafe temperatures they could be hazardous to the users or to the environment. This chip can output 5-28V and 0-8A which a lot of power and will allow a great deal of both expansion in selecting solar cells but also the amount of batteries we choose to set up. This chip comes with a verbose reference design that shows is a great start for novices.

Specification	**Value**
Solar Support	Yes
Solar Input Voltage	5-28V
Battery Support	Lithium ion / Lead acid
Dimensions	3.5 x 3.5 mm2
Sleep Mode	Yes
Power Consumption	15uA
Temperature Range	-55 to 155 Celsius
Maximum Supply current	8A

Table 29: Charge controller b124650 Specs ***[37]***

5.8.2 Microchip MCP73831 Charge management controller

The Microchip MCP73831 chip set is a small form factor charge management controller meant to be used in a small form factor devices and designs. This charge management controller is meant and designed for lithium ion or lithium polymer-based batteries. This chipset is commonly used in personal electronic devices, digital cameras, Bluetooth headsets and cellphones.

As seen in table 28 this chip is well featured for our system if we don't go for our stretch goal of adding a solar panel. This chip features a input voltage of 7V which means we will still need two chips for our system as the batteries will be roughly 9V so we need to overcome that voltage to charge the system. This chip like the last one is a compact chip which works for the small form factor of our system keeping the PCB design simple and clean. This chip doesn't give the system to use batteries other than lithium ion batteries. Though it does allow for a higher supply current so if the user plugs in the device it will charge fast which is a benefit for the user. This chip also features charge stats output which can drive LEDs this feature can be used to give the user an indication that the device is actually charging. This chip features thermal regulation allowing the chip to stop handle if it is overheating on in an environment where the temperature is too high. These chips allow multiple charging options that can be used not only for batteries of today but of the batteries that may come out in the future allowing this chip to be future tolerant.

Specification	**Value**
Solar Support	No
Input Voltage	7V
Battery Support	Lithium ion / Lithium Polymer
Dimensions	2 x 3 mm2
Sleep Mode	Yes
Power Consumption	15uA
Temperature Range	-40 to 85 Celsius
Supply current	15 - 500ma

Table 30: Microchip MCP73831 Specs ***[38]***

5.8.3 Microchip MCP73844 Charge controller

The Microchip MCP73844 is another charge controller that is available from microchip this is another small form factor design meant to charge batteries. This charge controller does not have solar charging available out of the box either and would still need another chip to supplement this controller.

This charge controller also only supports 1 - 2 lithium ion cells which would be around 1000mah for our system which is not an ideal amount of operating time without a charge. Though we can have multiple of these chips to support the amount of batteries needed for the system. The 30ma max on the supply current is something that is not beneficial to the system as it greatly increases charge time. This chip also features the status indicator so we could send the user the status of the batteries and how much charge is currently held and charging. This chip also features and automatic shutdown capabilities saving power but shutting off the chip when there is no longer power being delivered to charge the device. This chip also features a fast charging mode allowing the system to significantly gain charge but there is a limit to how fast it can charge. Table 31 below shows the data for this charge controller.

Specification	**Value**
Solar Support	No
Input Voltage	12 V
Battery Support	Lithium ion
Dimensions	0.118. x 3.0 mm2
Temperature Range	-40 to 85 Celsius
Supply current	30 ma
Specification	Value
Solar Support	No

Table 31: Microchip MCP73844 Specs ***[38]***

5.8.4 Linear Technology Charge controller

The Linear Technology LT3652EMSE is a charge controller that is available from linear technology this is another small form factor design meant to charge batteries from solar power. This chip is meant for more power consuming device than our platform but is still cost effective and within our spec range.

This charge controller supported input voltages from 4.95V - 32V and a current up to 2A. This is more than capable for being able to handle the power that would be generated from solar in our system. This charge controller does come with a caveat of requiring the battery pack voltage to be 14.4V which is more than our system is going to require and have as we would have to step that down to 6V for the rest of the system and all of the components to safely use it. Table 32 below shows the data.

Specification	**Value**
Solar Support	Yes
Input Voltage	32 V
Battery Support	Multi-Chemistry
Dimensions	0.118. x 3.0 mm2
Temperature Range	-40 to 125 Celsius
Supply current	2 A

*Table 32: Linear Technology Charge Controller Specs [39**]***

5.8.5 STMicroelectronics Charge controller

The STMicroelectronics L6924D is another charge controller that is available from STmicroelectronics this is another small form factor design meant to charge batteries from solar power. This is another small form factor chip mean for consumer electronics and small appliances.

The chip is meant to charge lithium ion batteries via solar. This microchip also features a reference design of the L6924D which is very useful for our design as we can remove what is not needed or get a better understanding of its impact on the system. This is another chip that features fast charge capabilities and has the

status outputs to allow us to interface and see if the device is charging and to use it or not. Table 33 below shows the information for the controller.

Specification	Value
Solar Support	Yes
Input Voltage	5.5 V
Battery Support	Lithium ion
Dimensions	3.0 x 3.0 mm
Temperature Range	-40 to 125 Celsius
Supply current	1 A

*Table 33: STMicroelectronics Charge Controller Specs **[40]***

5.8.6 Microchip MCP73871

The Microchip MCP73871 is another charge controller that is available from Microchip. this is another small form factor design meant to charge lithium batteries from either a 5V USB input or a solar input. This is another small form factor chip mean for consumer electronics and small appliances.

The chip is meant to charge lithium ion batteries via solar. The most impactful feature of this charge controller is that it includes load-sharing. This means that when there is sufficient power coming from either the 5V USB input, or from any solar panels connected, the whole system will be powered from these inputs rather than the batteries. In fact, this charge controller can even use a partial amount of the power from the solar or USB input, and supplement the rest of the power needed with the batteries. Also, this chip only requires 9 external components. Table 34 below shows the information for the controller.

Specification	Value
Solar Support	Yes
Input Voltage	4.4 V - 6 V
Battery Support	Lithium ion
Dimensions	4.0 x 4.0 mm
Temperature Range	-40 to 125 Celsius
Supply current	1 A

*Table 34: STMicroelectronics Charge Controller Specs **[40]***

5.8.7 Charge Controller Selection

From the tables above discussing the charge controller it can be seen that the choice is to take a hit if we use a solar panel on plug in charge time as the complexity of having a chip that does solar slows down the amount of current the TI bq24650 can accept. Thought he TI bq24650 can accept less current it comes with the added benefit of supporting other battery types if plans deviate from the current design and research with a higher tolerance for input voltage than the microchip charge controller. The Microchip MCP73831 does have an advantage over the TI chip as it can take in a higher supply current allow the battery to charge faster. While the TI chip does have a higher heat rating than the MCP73831 both are rated for pretty extreme temperatures that most climates won't get to but have to keep in mind how much heat they create while charging the batteries. In our system the better choice would be the Microchip MCP73831 as it will allow the user to charge the device at a faster rate, but this is if we go with only one chip. Keeping space and size of the PCB in mind there should be enough space on the PCB to accommodate both of these chips as they are for such a small form factor so the best method for this project would be to have both of these chips on the PCB to allow the solar panel to charge the battery at a lesser rate but if the user needed to charge the batteries themselves as they don't meet a climate suitable for solar it is available to them.

5.9 Selected Components and Testing

Below is a picture of all of the major hardware components purchased for our project. Some of the components are placeholders, used for testing purposes. The

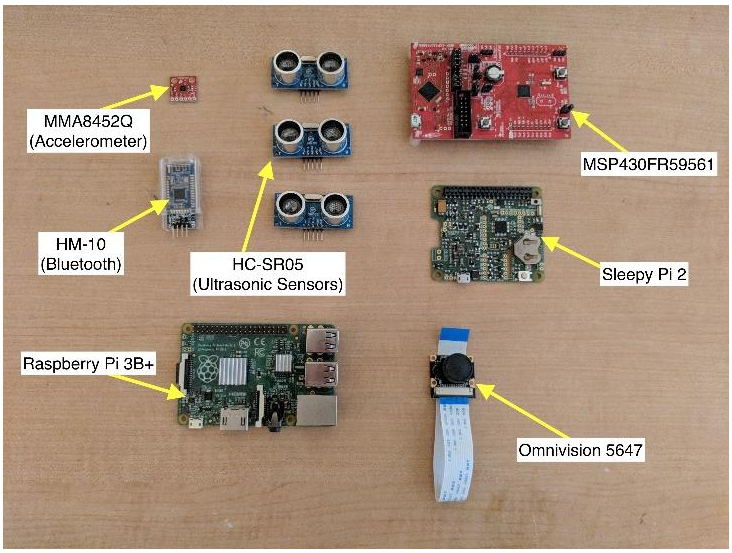
raspberry pi in this image is a 2B, where as we intend to use a 3B for our final design. The main differences between the two are power consumption and built in Wi-Fi, 2 aspects that were not a concern when it came to initial testing. The microcontroller is also the debugging version, that is built onto a testing board. This will be used not only for debugging, but for programming the MCU that is to be placed on the PCB.

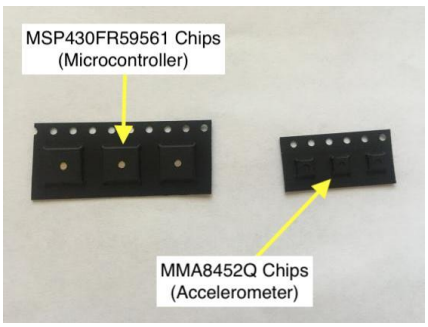
Initial hardware testing yielded positive results from all of the major components. The ultrasonic sensors gave us feedback at varying distances, which we were able to change. As we updated the distance at which we wanted the sensors to detect an obstruction, the closer we got to the sensors we found the accuracy greatly decreased. We are able to counteract this by not only having multiple sensors, but to be testing for obstructions that are quite large given what a car would consider to be an obstruction. We also took advantage of using the SR-05 models rather than the SR-04, allowing us to use less GPIO pins for the same results.

The accelerometer gave us a lot of information, both in terms of acceleration in the X, Y, and Z directions, but the orientation of the device itself. This gives us more than enough information to determine, to a good fine degree of accuracy, what direction the car is going and how fast it is going in that direction. Through our testing we also found that, unlike some of the documentation we found, some resistors are needed from the I2C pins on the accelerometer, which is a good piece of information as we will avoid damaging the breakout board moving forward.

For the HM-10 we were able to successfully connect a cell phone to the Bluetooth connection, which was the easiest method to make sure the connection could be made. The Bluetooth module will be doing a very limited amount of work, simply looking for ping signals from the phone, so a successful connection gives us a green light with the rest of its implementation.

The raspberry pi and camera worked very well with one another. The resolution and framerate that we got during testing is more than we need, and we saw a possibility of reducing the framerate in order to gain faster connection speeds or even longer battery times. It should be noted that we were using a different model raspberry pi than we intend to use in the final design, but this didn't make a difference for testing purposes, as we were still able to use the camera and the same firmware will be used in the model we use later on.





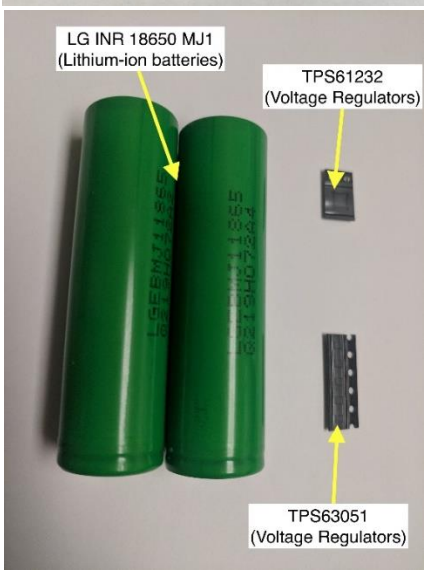


Figure 9: Batteries and Voltage Regulators
Figure 10: Major Components
Figure11: MCU

5.10 Major Component Block Diagram

Below is a block diagram of how the major competent will work together to make up both the hardware and software designs of this system. Hardware and software go into more detail, specifically regarding the PCB and electronics, and how the app will work, in the following sections. This figure serves to combine the two sections, as we must treat them as separate entities during the development phase, but they each make up a single part of a bigger picture. One note about the block diagram is that the work is broken up per group member, where each group members specialty was considered when assigning roles. All members work together in the end, as this serves to represent the diversity of our group's special traits.

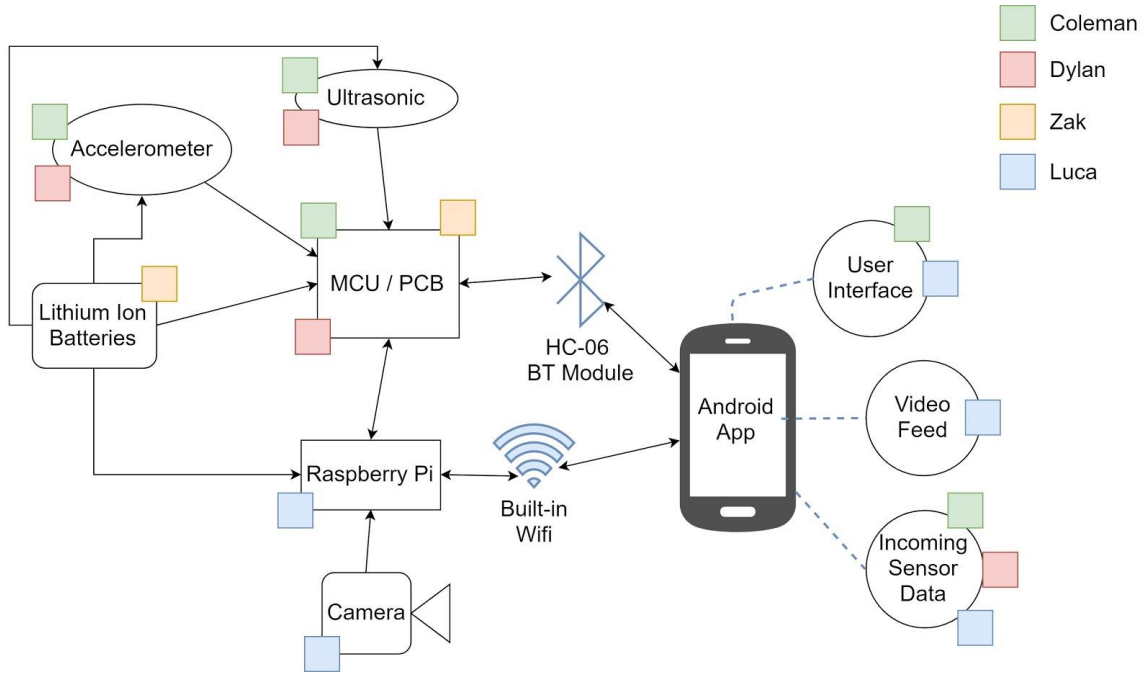


Figure 12: Overall Block Diagram

6.0 Hardware Design

The hardware is the first of the two main halves of the entire system. The information from the sensors and camera needs to be sent correctly and power efficiently to the app component for the design to operate properly. One of the biggest concerns with the hardware is the power efficiency. Using lithium ion rechargeable batteries means we have to be a lot more conscious of the power draw from the sensors, Raspberry Pi, and microcontroller. This section covers all of that, as well as electronic components that will be used to connect all the major components listed in the previous section.

6.1 Power Solutions

In order to power the PCB holding the backup camera, microcontroller, and multiple sensors, we needed a reliable source that is either internal or that can be easily accessed from outside of the rear end of a vehicle. We contemplated three possible solutions that could work to supply our device with power. The three options including attaching our device to the backup light circuit that already exists within our car, using strictly battery power, and using solar panels. In order to choose which power supply would be best for our specific application, we contemplated each one.

6.1.1 Brake Lights

The first and most obvious solution to power was to hook the PCB to the car's internal electrical system as well as the reverse lights. The positive terminal of the reverse light wiring outputs a 12V DC signal. This is more than enough energy needed to power the components in our PCB. This choice was obvious because of the close proximity of the reverse lights to where the backup camera would be placed - it is the closest possible source of energy. Also, since the reverse lights only activate when the car is actually in reverse - it would also act as the way of knowing when the backup camera needs to be activated. When the 12V source is on, that means the car is in reverse, and when it outputs 0V, then the backup camera system is not needed. The disadvantage of using the car's circuitry to supply power to our device is that our device will only have power once the car is already turned on, which leaves little time for both our hardware and software to initialize and begin functioning. The typical time of use for a backup camera comes within a few seconds of turning the car on - when a driver needs to back out of a parking spot. While our microcontroller can work quickly, the Raspberry Pi device used to transmit the video wirelessly will need 15-30 seconds to initialize, which is way too long for a driver to realistically wait after turning on their car. The alternative - connecting our system to the user's car battery could run the risk of draining the user's battery if the vehicle is not powered on in a certain lapse of

time. We decided not to utilize the car's internal electrical system to power our device, although we will keep the idea open as a possible stretch goal if we meet all of our other goals.

6.1.2 Solar Panels

Our second option for the consumer that we considered was to offer a set of small solar panels. These solar panels charge a battery that is able to power the PCB and its components. The reason for including this option to the consumer was because of one of the goals of our design - to create an easy to install backup camera system. We want people with no technical knowledge to able to simply stick the camera on the back of the car, download the mobile application, and have it working. While connecting the backup camera to the reverse light is not extremely difficult, there still may be consumers without any technical knowledge who are afraid to take on that task. Those consumers would instead be able to power their device by using the solar panels as an easy to install option. The major disadvantage of the solar panel is due to the location of our device on a vehicle. The device is meant to be placed on the rear end of a vehicle, which is not an optimal location to receive sunlight. This means that our solar panels would run the risk of not absorbing enough sunlight if placed near where the device is mounted or would require long wires to hang if solar panels were placed on top of the vehicle. The latter option would not be very aesthetic, and possible dangerous with loose electrical wires hanging over a fairly long distance. However, if proper solar panels were found that could minimize these issues, this would be a viable addition to the project.

6.1.3 Battery

The final option in order to power the backup camera device is to use battery power. This power option will be the easiest to install - as all necessary power will already be included in the system. The user would simply have to put the device in place, and it would work as intended. Also, since the battery is always on, the device would be able to either be always-on or be programed to initialize before the vehicle is turned on. This is an improvement over using a vehicles pre-existing power, because the device would be initialized and ready for use at the instant, and even before the instant, that the driver turns the car on. The tradeoff with this option is that battery life is finite - the user will either have to replace or recharge the battery in order to continue getting functionality. This may be an inconvenience to the user - but the installation of the other powering techniques as well as the system's initialization time may be an inconvenience as well. We decided it would be best for the system to run on strictly battery power. As long as we maximize the battery life to need a recharge or replacement at reasonable intervals, the

conveniences that a battery powered system will supply are big advantages that we wanted in our easy to use system.

6.2 Battery Selection

The two main battery types used in portable devices are lithium ion (Li-Ion) and nickel metal hydride (NiMH). In order to choose the best option for our device, we had to look at the advantages and disadvantages of each one.

Lithium Ion: Lithium ion batteries are typically used in cell phones and notebook computers due to being lightweight yet holding lots of energy within a compact package. Another advantage is their low self-discharge when the battery is not in use. Since lithium ion batteries commonly operate at 3.7 V, the voltage is easily and efficiently able to be stepped down without much power loss since the MCU can operate at a voltage of 3.6 V, only a 0.1 V difference. The biggest advantage of lithium ion batteries is their self-discharge rate, which can typically be from 0.5% - 3% of the batteries capacity per day **[41]**. This is a major upgrade over the NiMH batteries, and requires less changing or charging of the battery on the user's end. The major disadvantage of lithium ion batteries is their fragility - they are easily damaged and need to be protectively encased within a circuit for safety purposes. They also tend to be slightly more expensive than any NiMH counterparts with the same energy capacity. The lithium ion batteries also change their output voltage as they discharge - which creates an additional requirement for our voltage regulator and may cause variable power efficiency in the circuit as the voltage changes.

There are many types of Lithium Ion batteries as well. Each type has a different chemical makeup and have different properties that are beneficial to the particular needs of one's design. The most common types for small electronics are Lithium Ion, and Lithium Polymer. Lithium Ion batteries tend to have a higher energy density and a significantly lower cost. The negative aspect of them is their instability. If the casing of a Lithium Ion battery is breached, it is possible to start a fire or explosion. When choosing a Lithium Ion, one must pay attention to the safety ratings of that particular model - if a manufacturer cuts corners it could be potentially dangerous. Lithium Polymer on the other hand, tends to be significantly more expensive. It also tends to have a shorter lifespan than Lithium Ion batteries, and a smaller energy density. The positive aspects of Lithium Polymer batteries are that they are robust and even flexible, lightweight, and much safer.

Nickel Metal Hydride: Nickel metal hydride batteries are usually the more typical AA, AAA, and D batteries that users are accustomed to in many home appliances. The main advantage of this battery type is the cost - they tend to be cheaper for the same amount of energy capacity of any lithium ion counterpart. Another

advantage is that they operate at a constant voltage regardless of their charge level - which creates a reliable output and an efficiency that does not vary. This constant output also means that the NiMH batteries are able to use all the charge within them, which is a positive. On the downside, each NiMH battery is only 1.2 V, meaning we would either need to step up the voltage or use two or three combined batteries in order to power our device. In essence, this means we must either add complexity or space and weight to our device. Another big disadvantage is that NiMH batteries typically discharge when not in use much faster than lithium ion batteries, up to between 1% and 5% per day **[41]** - meaning they will have to be recharged or replaced more often - as consumers typically do with household electronics that use such batteries. This self-discharge rate becomes even larger when the batteries are operating at a high temperature - which may be the case if the outside temperature happens to be high. For a device that is left on the back of the car and counted on to be used in a moment's notice, this may be a big inconvenience. In order to help us choose which battery type to use in our design, we created House of Quality Table 35 below, with the corresponding key.

		Battery characteristics		
		Self-discharge per month	**mAh/weight**	**Cost/mAh**
Battery Types	**Lithium ion (Li-Ion)**	▲▲	▲▲	▲
	Nickel metal hydride (NiMH)	▼▼	▼	▲▲

▲	Positive Correlation
▲▲	Strong Positive Correlation
▼	Negative Correlation
▼▼	Strong Negative Correlation

Table 35: House of Quality with Legend

After considering both types, we decided we would use a lithium ion battery for our device. Since this device is meant to be used on-demand, we figured it would be very inconvenient for the batteries to have to be replaced or recharged more often. The fact that NiMH batteries discharge rate is less resistant to temperature was a major factor as well - since our device will be outside in potentially hot environments, the possibility of a major drop in the efficiency of the battery is a huge disadvantage. Within the Lithium family, we decided to use Lithium Ion instead of Lithium Polymer. This is due to the fact that we strongly desired a higher

energy density. The negative aspects of Lithium Ion batteries could be minimized by choosing a good manufacturer and creating a stable holder for the battery to stay in while in use.

6.2.1 Battery Life

In order to choose an appropriate battery for our system, we had to look at the current that all of our components would draw. Since our device will work in both a low power and an active state, we had to see what the current draw would be in each scenario to get an idea for the total battery life that our system would have before needing to be recharged or replaced. Below is a table comparing the power consumptions of various components.

Component	**Active mode current draw**	**Low-power mode current draw**
MCU (MSP430FR59691)	100 µA	0.40 µA
Raspberry Pi module	450 mA	150 µA
Ultrasonic sensor * 3 (HC-SR05)	45 mA	0 mA (off)
Accelerometer sensor (MMA8452Q)	165 µA	6 µA
Bluetooth module (HM-10)	8.5 mA	1 mA
Camera module (Omnivision 5647)	96 mA	20 µA
All components together	599.765 mA	1.158 mA

*Table 36: Component Power Consumption**
*All values are approximate and when tested may result in different values.

In order to calculate our approximate battery life, we can use the following equation:

(Battery capacity)/(Total current draw)=Battery life

For estimation purposes, a 3000 mAh battery capacity will be assumed in the following calculations. After the calculations are complete, it will be easier to analyze whether the capacity should be higher or if a 3000 mAh battery is sufficient.

Lower power mode: (3000 mAh)/(1.158 mA)=2590 hours =107 days
Active mode: (3000 mAh)/(599.765 mA)=5 hours
Due to our system having two modes that operate at different levels of power consumption, it was necessary to calculate the battery life of each power mode separately. We found that our battery life in low power mode with a 3000 mAh battery would give us around 107 days of battery life. The active power mode could give up to 5 hours of battery life - however the actual time spent using this power mode can vary greatly depending on the consumer's use of the device. Also, the power consumption amounts in Table 36 are only rough estimates according to datasheets and can vary greatly depending upon our design and ability to optimize each component. While these estimates are decent, the only disadvantage to getting a battery with more stored energy is the cost - there is nothing trade-off that is integral to the design.

With the above estimates in mind, we decided it was necessary to have a battery that could handle at least 2 amperes of maximum current to be safe. It was also beneficial to choose a battery that could be rechargeable, mountable, and safe. We decided to use a standard 18650 battery type, but as a lithium-ion that is at least 3000 mAh. Since these batteries are replaceable the exact model did not matter, we just had to find the best deal in terms of price per energy that supplies anywhere between 2.5 and 5 volts, due to our selection of voltage regulators. After some shopping we found a LG INR18650 MJ1 battery with 3500mAh and a 3.7 voltage as our stock battery, although this component selection retains flexibility to change later on in the project if we determine we need more energy or find a better component at a better price. Any replacement must supply a similar voltage range, between 2.5 volts and 4.2 volts. In order to fit in the same enclosure any replacement battery would have to be an 18650-size battery, although for prototyping purposes outside of the enclosure this is not a requirement.

6.3 Voltage Regulators

In our design, all of the components on the PCB are designed to have low input voltages - typically between 1.8V - 3.6V. However, with the battery connected, we will have a 3.7V - 5V DC input, depending on the battery we choose. In order to operate our microcontroller and attached components, including the camera, we will need to convert the battery into the appropriate lower voltage. There are four possible design options that were researched that can achieve this, each with their own pros and cons, discussed in the paragraphs below.

A simple voltage divider would theoretically do the job and it is very cheap - only needing a single resistor. However, since the resistor value never changes it only supplies the voltage intended with a constant load impedance - which may not hold

true in a real-world environment. While that is a negative aspect of the reliability - there is a positive aspect in that it is impossible for there to be a short circuit - which means the components are safe. This circuit will waste most of the power supplied into the resistor, which means very low efficiency, less than 50%.

Linear voltage regulators are also extremely cheap, with the associated integrated circuits typically less than one dollar. This circuit would be able to supply a constant voltage regardless of the load impedance, which is an important improvement on reliability. However, this circuit is not short-circuit proof, but it will limit the amount of current in a short circuit scenario. Also, this circuit will dissipate tons of power into the voltage regulator - which may require a heatsink and airflow in order to not overheat.

A switching voltage regulator circuit is noticeably more expensive than the others, it is typically between one and fifteen dollars. It will also be able to regulate the output to be practically constant. It is like the voltage regulator in that it is not short-circuit proof, but it can limit the amount of current in a short-circuit scenario, which may protect other components. In terms of efficiency, switching regulators will be the most efficient by a large margin due to the ability to switch off the connection to the source and use the internal inductor's energy to step the voltage down while the power is disconnected. In this case, there will be no heatsink needed - when the inductor is fully charged the switch will disconnect so power is not wasted. Efficiency of these devices can typically be between 70%-95% **[42]**. An additional trade off to consider when using this circuit is the added complexity and electromagnetic interference due to the inductor.

The cost of an emitter follower step down circuit is still extremely cheap - the transistor, diode, and resistor all together can be bought for less than two dollars. This circuit will be able to supply a constant voltage to the load regardless of any slight changes in the source voltage or load impedance, but it is not short-circuit proof. In terms of efficiency - it actually wastes more power than the other options due to the extra current needed to run through the diode. The transistor will become very hot as well and may require a heatsink and ventilation.

With four different possible ways to step down the voltage, we created the House of Quality Table 37 below in order to compare the circuits in terms of cost, efficiency, and reliability.

		Circuit characteristics		
		Efficiency	**Reliability**	**Cost**
Voltage Regulator Type	**Voltage Divider**	▼	▼▼	▲▲
	Linear Voltage Regulator	▼	▲▲	▲▲
	Switching Voltage Regulator	▲▲	▲▲	▼▼
	Emitter Follower	▼▼	▲	▲

▲	Positive Correlation
▲▲	Strong Positive Correlation
▼	Negative Correlation
▼▼	Strong Negative Correlation

Table 37: House of Quality

With a quick glance at the House of Quality Table 37, we can quickly eliminate the voltage divider due to it having a negative correlation in two categories. Between the linear voltage regulator and the emitter follower, we can see that the linear voltage regulator is both cheaper, more reliable, and slightly more efficient - an obvious choice. The ultimate decision was between the linear voltage regulator and the switching voltage regulator. We ended up choosing to use the switching voltage regulator. The major factor in this decision was the efficiency - the idea of needing a heatsink and good airflow to accommodate the inefficient power dissipation in the linear voltage regulator was unappealing. The fact that our device is meant to be small and used in outdoor environments, having to deal with that extra heat would be inconvenient and a direct obstacle to our goals. Also, although switching voltage regulators were relatively more expensive in comparison to the other circuits - in the grand scheme of our project the price of a switching voltage regulator was not overwhelming. The major negative aspects of the switching voltage regulator, cost and electromagnetic interference, were not extremely important in our specific application.

6.3.1 Switching Frequency

The switching frequency within a buck, or even a boost converter for that matter, is how often the circuit's switch is turned off and on. While this does not impact the voltages or currents outputted from the converter, it does impact the efficiency, noise, size, and cost. We researched the advantages and disadvantages of having a higher or lower switching frequency in order to help us decide the optimal switching frequency for our application.

High Frequency Switching: High frequency switching buck/boost converters simply switch off and on faster, creating more cycles of charging and discharging for the inductor than a lower frequency converter. This creates a few benefits - one of them being that the inductor and any capacitors in the buck/boost converter circuit can be smaller. This impacts the size and cost of our voltage regulating circuit. Another advantage is using high frequency switching can help improve noise by avoiding many busy and noise-sensitive lower frequency bands, such as AM radio. The negative aspects of higher frequency converters are the extra power loss, every time the switch is flipped, there is a small amount of power loss at that instant. Due to higher frequency converters executing the switch operation at higher intervals, this means more opportunities for that power loss. This can contribute to more heat dissipation - although modern high frequency converters are fairly efficient at minimizing this power loss. This is especially true for devices with smaller currents - which tends to be the main market for converters in the 3MHz-4MHz range.

Low Frequency Switching: Low frequency switching buck/boost converters require a larger inductor and capacitor in the voltage regulator circuit. This is due to the circuit switching less often, requiring an inductor that stores more energy to be able to discharge over longer time intervals when the switch disconnects the battery. Lower frequencies also tend to be the ones already in use for many commercial applications, which could cause unnecessary noise. However, low frequency switching converters also create less electromagnetic interference, which can be crucial to designs that will be nearby other sensitive components. Low frequency buck/boost converters also are slightly more efficient with power, due to the power lost every time the switch is executed. This power efficiency, however, is more noticeable with loads that draw larger currents.

In terms of our design, we decided that higher frequency voltage converters would be more beneficial. This is mainly due to the ability to have a smaller inductor and capacitor in the circuit, as well as the fact that the power loss was minimal when currents are as small as they will be in our device. However, cost is still a mitigating factor when deciding which converter to use. As long as the voltage, current, and temperature specifications all matched our requirements, we decided to look for converters by looking at the price versus the switching frequency.

Pulse Width Modulation/Pulse Frequency Modulation: Pulse width modulation (PWM) and Pulse frequency modulation (PFM), although not integral to component selection, were important to research before attempting to design the voltage regulator modules of our project. These two modes of operation were included in nearly every converter that we researched. Due to the fact that PWM and PFM impact the efficiency of the converter, it was necessary to familiarize ourselves with the positives and negatives of each one to truly understand all the information in the datasheets. We learned that the main purpose of these two modes of operation were to choose between the tradeoff of minimizing switching losses and minimizing noise. PWM helps reduce the noise by having a constant frequency, but the switching loss of efficiency is maximized when there are low loads because the converter will continue switching at a rate that is faster than necessary. On the other hand, PFM will change the frequency of the switching as the load requires more power - which can increase efficiency by switching less when it is able to do so. On the other hand, changing between different frequencies creates more unpredictable noise that could make the output less stable than desired. This information helped us to decipher the information in the datasheet and gain a better understanding of the actual efficiency we will be able to achieve with our design.

6.3.2 Maximum Output Current Requirements

In order to decide on the maximum output current necessary for each of the two voltage regulators that will be used in our system, it was necessary to determine how much current each component under that specific voltage regulator could possible consume. Since we are only specifying the maximum values, it was only required that we look at maximum current that each component will draw in its active mode. The exact amounts of current consumption for each component were already specified in Table 33 while determining the battery life. In Table 38 below, these component current consumption values are aligned according to the voltage input they require so we can determine the maximum output current when choosing a voltage regulator.

Component	Current Consumption	
	3.3 V Required	5 V Required
MCU (MSP430FR59561)	100 µA	
Raspberry Pi module		450 mA
Ultrasonic sensors (HC-SR05)		45 mA
Accelerometer sensor (MMA8452Q)	165 µA	
Bluetooth module (HM-10)	8.5 mA	
Camera module (OV5647)		96 mA
Total	9 mA	591 mA

Table 38: Component Current Consumption

It is important to keep in mind that these values are estimates and taken from either the component datasheet or other online research, and not actual testing. It is important to safeguard our design by obtaining components that can handle much more current than these estimates require. Based upon this information, it was decided that the 5 V voltage regulator should be able to handle 1.5 amperes, and the 3.3 V voltage regulator should be able to handle 100 mA. For the 3.3 V voltage regulator it will also be important to make sure the IC has a very small minimum current for good efficiency, so it can operate as expected even with extremely low currents that it operates at in the low power mode. Based upon Table 36 in the Battery Life section, the low power load current connected to the 3.3 V voltage regulator could be as small as 1.0064 mA or lower. In order to make sure our circuit keeps operating, we decided a good threshold for a minimum current where the efficiency is still good would be 1 µA.

6.3.3 Model Comparison

While searching for specific voltage regulator models, we had to list the exact requirements that we desire in order to narrow down our search. We decided to look for voltage regulators for the 3.3 V portion of the power supply with the following characteristics, which were chosen based on the surrounding sources and components in the system.

- Voltage input capable between 2.5 V - 4.5 V
- Voltage output capable of 3.3 V
- Operating temperature between -30°C - 65°C

- Minimum output current capability of 1 A
- Efficiency above 85% when load current is 1 mA

MCP1700: The MCP1700 is a linear regulator. Due to this, the efficiency depends entirely on how large the voltage difference is between the input and output. In our application, this means the efficiency could be anywhere between 79% and 91% depending on the charge status of the battery. Below are some of the characteristics:

- Voltage input range: 1.2V - 5 V
- Voltage output range: 3.3 V
- Maximum current: 250mA
- Operating temperature: -40°C - 125°C
- Efficiency above 79%-91% depending on battery charge

In this case, the maximum current does not match what we desire from our system. Although it technically would be enough, since this portion of the system would only be using a portion of that current draw, we wanted to safeguard the system and allow much more current. Also, the efficiency runs a bit low when the battery is discharged. For these reasons, we decided against using this system.

TPS64200 Family: The TPS64200 Family of voltage regulators are an excellent choice for our application, being able to achieve up to 95% efficiency under certain conditions. They all share these common characteristics **[43]**:

- Voltage input range: 1.8V - 6.5 V
- Voltage output range: 1.2V - 6.5 V
- Maximum current: 3A
- Operating temperature: -40°C - 85°C
- Efficiency above 85% with a 1 mA load current

All of these characteristics match our requirements above. Within the TPS64200 family, there is the TPS64200, TPS64201, TPS64202, and TPS64203. The only difference between each of these devices is the switching frequency. The switching frequencies offered are between 350 kHz to 800kHz. With the price difference nearly negligible between the different switching frequencies, our choice from this family would be the TPS64203 which operates at 800kHz. This set of voltage regulators meet all of the specifications, but are somewhat less efficient due to being meant for higher power systems and having a lower switching frequency.

TPS63051: The TPS63051 acts as both a buck and boost converter. This switching voltage regulator is excellent choice for our application, being able to

achieve up to 95% efficiency under certain conditions. Below are some of the characteristics **[44]**:

- Voltage input range: 2.5V - 5.5 V
- Voltage output range: 3.3 V
- Maximum current: 1A
- Operating temperature: -40°C - 85°C
- Switching frequency: 2.5 MHz
- Efficiency above 87% with a 1 mA load current

All of these characteristics match our requirements above. The 2.5 MHz switching frequency is ideal for our application, it will allow us to have smaller impedances in the accompanying voltage regulator circuit. However, with the output only working at 3.3 V, it would require us to purchase a separate model for our second voltage regulator that must operate at 5 V.

Based upon these voltage regulators, we decided to use the TPS63051 for our 3.3 V applications. The biggest factor in deciding was due to it having a significantly higher switching frequency than the TPS64200 family of devices. The difference between the other characteristics - such as efficiency and the voltage input range were negligible. The major advantage the TPS64200 family had was the maximum current being 3 amperes, but for our design this much current would not be necessary, in fact we only estimated needing 9 mA. The maximum currents of 1 ampere in the TPS63051 will be more than enough to supply power to all of the components that require a 3.3-volt VCC.

6.4 Logic Voltage Level Shifting

Our system has components that operate at two different voltage levels - 3.3 volts and 5 volts. In order for any components that operate at different voltages to communicate with each other, the voltage level of each data pin must be shifted along the connection path in order to not cause damage to components or give unreliable results. In terms of our design, the specific components that operate at different voltage levels but were required to communicate to each other were the HC-SR05 ultrasonic sensors and the MSP430FR59561 microcontroller. We researched two ways to achieve the voltage shifting that our design required.

6.4.1 Voltage Divider Shifting

The most basic way to shift the voltage level of a data connection was through using a simple voltage divider. This solution is simple and cheap. However, it would create excess heat in our system. Another negative aspect is that it would not give

a stable voltage - the resistances could vary with excess heat and cause unreliable voltages. Also, the data signals when using a voltage divider to shift the logic level could potentially give off rounder waveforms which could potentially trick our microcontroller into reading data at the wrong time. It also involves having two resistors for each connection, and since we have three ultrasonic sensors within our system this would call for six new components with corresponding traces on the PCB. However, since this approach is simple it will suffice for any prototyping and testing that needs to be done.

6.4.2 Voltage Level Translation IC

The other approach was to purchase a manufactured voltage level translator. The positive aspects to this approach are that we could find this in an extremely small package in order to save space in our system, and that it would be much more efficient and waste less power while undergoing the voltage shifting process. The negative aspect of this is that the IC we choose would most likely not be breadboard testable due to size constraints and we would simply have to hope that it works as intended while designing our PCB. In order to choose a voltage level shifter, we looked at a few different models. The basic requirements for any level shifters we looked at are listed below.

- Auto bi-directional
- Capable of 3.3 volt to 5 volt conversion and vice versa
- Low power consumption

LFS0204: This device is small and designed for low power consumption. It is a 4-bit device that can handle voltage levels between 1 volt and 4.5 volts on one end, and 1.8 volts and 5.5 volts on the other end. It has an extremely small propagation delay - 1.5 nanoseconds maximum, which is good for ultrasonic sensors which require measuring extremely small lapses in time to estimate distance. The maximum current output is 64 mA which is sufficient for our design but does not leave a lot of leeway for extra unanticipated current **[45]**.

TXB0104: This device is designed for extremely low power consumption. It is a 4-bit device and can handle voltage levels between 1.2 volts and 3.6 volts on one end, and between 1.65 volts and 5.5 volts on the other end. There is a 1 to 4 nanosecond propagation delay, which is very small relative to our application but not as small as the LFS0204. However, the TXB0104 can handle 100 mA of output current which gives us a little more safety in terms of extra unanticipated current **[46]**. In order to safeguard our design, we decided to use the TXB0104 for our system.

6.5 Raspberry Pi Power

The Raspberry Pi is the component that will draw the largest amount of power in our system. In order to power the Raspberry Pi, we need a voltage source supplying 5V. Since our lithium ion battery will not supply this voltage, we will need a DC step up converter, also known as a boost converter. Due to the Raspberry Pi's large power consumption relative to all the other components in the system, we also must research techniques in order to minimize the power consumption.

6.5.1 Boost Converter Model Comparison

The boost converter will be necessary to create the 5V we need for the Raspberry Pi from the smaller output the lithium ion battery supplies. When the switch is closed it creates a short circuit, essentially creating current in the left loop of the circuit with no current going through the diode or the load. This causes the inductor to charge with energy. When the switch is opened, the sudden drop in current causes the inductor to create a counter-electromotive force, which is essentially a new voltage across the inductor. This added voltage, combined with the battery's supplied voltage, is what generates a larger overall voltage in the load. Typical manufactured boost converters will add other components for output stability as well as circuit protection. In order to power our Raspberry Pi, these are the requirements that we will be looking for when shopping for a boost converter:

- Input voltage from 2.5V - 4.5V (Li-Ion charge/discharge voltages)
- Output voltage 5V
- Minimum current capabilities of 1.5A
- Operating temperature between -30°C - 65°C

MIC29301: The MIC29301 chip is a linear regulator, so the efficiency is entirely dependent on the difference between the input and output voltage. In this application, that means we would have fairly low efficiency, between 60% and 84%. It has the following characteristics, which meet our requirements:

- Voltage input range: -20 V - 60 V
- Voltage output range: 5 V
- Maximum current output: 5A
- Operating temperature: -40°C - 125°C

This meets all of our requirements and has a very high max current output, which would be good for safeguarding our system against the high current draw of the Raspberry Pi. Despite that, the efficiency only being between 60%-84% is quite low, and we did not want to lose that much power, otherwise our battery life would

not be as good as we desired. Due to these reasons, we decided against using the MIC29301 chip.

TPS61253A: The TPS61253A boost converter is an excellent choice for our application, being able to achieve around 95% efficiency under certain conditions. It has the following characteristics, which meet our requirements **[47]**:

- Voltage input range: 2.3 V - 5.5 V
- Voltage output range: 5 V
- Maximum current output: 1.5A
- Operating temperature: -40°C - 85°C
- Switching frequency: 3.5 MHz

This meets all of our requirements and has a very high switching frequency which will give us great efficiency when the components are turned off and the load current is small.

TPS61232: The TPS61232 is a boost converter. It is able to achieve up to 94% efficiency under certain conditions that our application would fulfill. Below are some of the characteristics **[48]**:

- Voltage input range: 2.3 V - 5.5 V
- Voltage output range: 5 V
- Maximum current output: 2.1 A
- Operating temperature: -40°C - 85°C
- Switching frequency: 2 MHz

The 2 MHz switching frequency is slightly smaller than the previous chip. However, this chip is able to handle more current which could be beneficial in case we end up needing to use more current then anticipated when implementing our design.

The trade-off when deciding between these components was the switching frequency and the maximum current output of the switching regulators. Since this portion of the power supply would be used to power the Raspberry Pi module, which recommends for a 2.5 ampere power supply, we decided to use the TPS61232. This component, despite having a lower switching frequency, still held efficiency above 90% at the low current ranges that our system would operate in low power mode. It also was a safer choice due to being able to handle more load current, in case of any unforeseen current drains in our implementation.

6.5.2 Raspberry Pi Power Management

Due to the large amount of power that is necessary to operate the Raspberry Pi, which can be from 200mA-600mA depending on the peripherals used, we wanted to figure out how to manage the power of the device. If the Pi was to operate in an always-on state, our device would not last very long due to being powered by a finite source such as a battery. To make sure our battery life is sufficient, we wanted to able to control the power of the Raspberry Pi and minimize the time that it is powered on, so only when absolutely necessary. This requirement creates another obstacle, because the Raspberry Pi can take anywhere from 20-30 seconds to boot up from being completely off. We wanted our system to be ready-to-use upon the user putting the car in reverse - meaning that the Raspberry Pi would have to be turned on at least 30 seconds before the user entered the vehicle. We desired to attack these obstacles with three approaches - minimizing the Raspberry Pi's boot up time, minimizing the power consumption while the Raspberry Pi is active, and creating a power control module that can turn the Raspberry Pi off and on via Bluetooth command.

Power Consumption: In order to minimize the power consumption of the Raspberry Pi, we researched how to turn off all unnecessary components and peripherals. Through benchmarking research, we found that this can make a noticeable difference in power consumption. Although it is hard to say the exact power saved by turning off each individual component without testing these products in person, we found through research that each HDMI and USB port turned that is not in use and turned off can save around 30mA of current drawn **[49]**. By turning off multiple inputs and outputs not in use, as well as any other components not in use we should be able to improve our battery life by a considerable amount.

Startup Time: Minimizing the startup time of our system helps us improve the power consumption of our system as well. Our system is being used for a driver to be able to back their car out of a parking spot, which in many scenarios could be fairly fast. In simple parking scenarios, backing out of a parking spot could even take five seconds or less. The Raspberry Pi, if unaltered, can take up to thirty seconds to start up - which is six times as long as the action that the device is being used for in the aforementioned five second scenario. Due to the extremely high current draw of the Raspberry Pi, every second that it operates is a sharp decline in our system's total efficiency. One way to improve the startup time of the Raspberry Pi is to purchase a Micro SD card that operates faster than the stock memory. This allows the Raspberry Pi's internal microcontroller to start up all boot processes faster. The other method that can significantly shorten the startup time of the Raspberry Pi is to disable all the software services that are unnecessary for our design. By implementing both of these methods, we should be able to reduce the startup time by a significant number of seconds.

Bluetooth Switch: In order to optimize our battery life, it is essential that the Raspberry Pi can be turned off while not in use. The high current drain when the Raspberry Pi is in use makes it unrealistic to remain on since our system is powered through a finite source. However, the Raspberry Pi does not offer any way to remotely turn the system off and on. This means that we had to either find or design a way to power the Raspberry Pi off and on through our separate microcontroller. Due to the long startup time of the Raspberry Pi, we had to decide what action would initialize the startup of the Raspberry Pi. We decided that using the startup of the car as the initialization action would not allow for enough time for the Raspberry Pi to startup before the user would be ready to back up their car, since the typical use of a backup camera is as soon as the car is started. We decided that the Bluetooth module attached to our microcontroller could work to turn the Raspberry Pi off and on whenever it receives a signal from the user's smartphone. This requires the user to open our application before entering the car, the application to send a Bluetooth signal to the Bluetooth sensor attached to our microcontroller, and our microcontroller to switch on the power for the Raspberry Pi. The power management aspect of this idea is to be handled by the Sleepy Pi 2 device, which can put the Raspberry Pi into low power mode when not in use, where it will then consume less than 200 μA **[50]**. That will be a major factor in significantly minimizing our power consumption.

It is important to note that these are simply optimizations, and even without completing these tasks to improve the efficiency of using the Raspberry Pi, the system could still achieve decent battery life. These optimization techniques could be used in future iterations of the design since they were not completed within the timeframe of this project.

6.6 Initial Design

The hardware design section will explain how the project will be designed in terms of the physical components. This will include the schematics and reasoning for the power and control aspects of our project. To begin designing the hardware, there had to be an overall design that encompassed all hardware components and their interactions with each other. This design served as the building block of the entire project, from research to testing. In Figure 13 below, our initial idea of how the hardware would be connected and interact is shown.

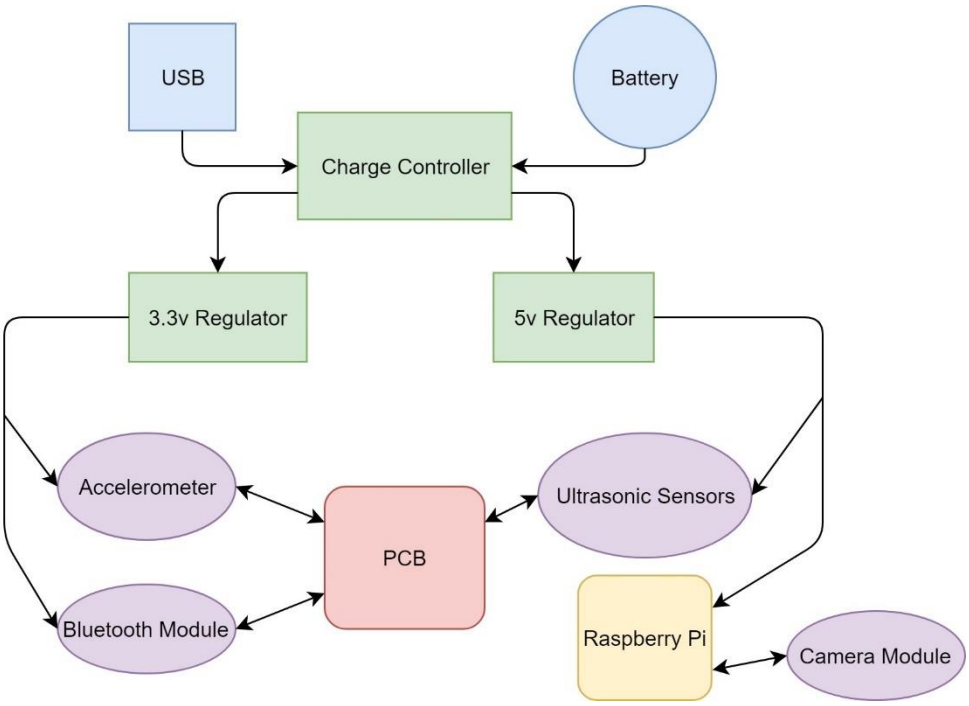


Figure 13: PCB Block Diagram

This chart served as the general basis from which started the hardware design of the project. This general layout of the hardware was necessary in order to figure out any design constraints. Although this was the initial idea, it was essential to retain flexibility in case of any unforeseen obstacles or improvements that would develop as more research or testing was done.

The general idea of our initial flow chart was for the microcontroller on the PCB to execute all necessary functions in terms of the sensors. This is ideal due to the fact that our microcontroller consumes very little power and would maximize the battery life. The function of the Raspberry Pi Module is to receive the video feed and transmit it from the camera module to the user's cell phone wirelessly, which was something a low power microcontroller running on a battery would not be able to achieve. The Raspberry Pi Module includes smart power management techniques in order to reduce its power consumption.

6.7 Power Design

The beginning aspect of designing the hardware was to make sure that it was sufficiently powered. This portion of the design is shown in the green blocks in the flowchart in Figure 13 above. The battery connects with two separate voltage regulators to provide two separate constant voltages to various components.

6.7.1 Battery Design

In terms of the battery, the design only needed to be able to mount the battery and have a footprint for it in order to put it on the PCB, in order to trace it to the various voltage regulators. Since the battery was the 18650 type, which is fairly large in comparison to the rest of our components, it was decided to mount the battery to the device's enclosure and simply wire it on to the PCB. For this purpose, on our PCB we simply had to create a pin for input voltage and ground and wire each side to a side of the battery. The battery would simply be attached to ground on one end and attached to a single terminal on the other end. The battery could then be wired a short distance off of the PCB, at the opposite end of the device enclosure. This will help minimize the size of the PCB and project in general. It will also help minimize the effects of heat - when the battery discharges it may become slightly hotter and it would be optimal if this heat did not transfer to our PCB to affect the efficiency of the rest of our design.

6.7.2 Voltage Regulator Design

The use of a lithium-ion battery, which has a varying DC voltage, requires the use of a voltage regulator in order to provide a constant DC voltage output to all the components. Based upon the battery's voltage range, which is between 2.5 volts and 4.2 volts depending on the charge level of the battery, a voltage regulator that could accept such a range of inputs was necessary for our design. Our design included the need to power up the MSP430FR59691 microcontroller, which operates at 3.3 volts, the Raspberry Pi Model 3B, which operates at 5 volts, as well as the various sensors that operate at one of these two voltages. Due to the components needing to operate at different voltages, our design required two voltage regulators - one to step up and down to the required input voltage of 3.3 volts depending on the charge level of the battery, and the other to step up to the required input voltage of 5 volts.

Battery to 3.3V: In order to power the microcontroller as well as multiple sensors, the design of the voltage regulator had to output a constant 3.3 volt signal. Since the input could vary between 2.5 volts and 4.2 volts, we needed to be able to step down and step up the voltage as needed, seamlessly switching between both operations. After researching multiple integrated circuits that could assist in this design, we chose to use the TPS63051 buck converter. According to the datasheet, the TPS63051 would be able to provide between 90% and 95% efficiency when operating with an output current of at least 1 mA, which would be the case in our design at all times.

Since this circuit design supplied to us in the datasheet supplies a constant 3.3 volt output it would be sufficient for our supplying power to our microcontroller and

various sensors that operate at 3.3 volts. Although we could simply use the values of the impedances that were given in the datasheet, we did have to be careful in choosing the type of capacitors and inductors according to the datasheet to achieve the expected results. The datasheet specified that for best operation, the capacitors should be of the X5R or X7R type **[44]**. For the inductor, the datasheet recommends a low DC resistance to minimize conduction loss. Although we plan on using these recommendations for PCB implementation, it is possible that we may prototype the system using different types of capacitors, but still at the recommended values.

Battery to 5V: In order to power the Raspberry Pi Model 3B, camera module, and ultrasonic sensors we required a power system that could output a constant 5-volt signal. These 5 volts would need to be achieved from the same battery as the previous design, meaning that the input voltage to the power system could be between 2.5 and 4.2 volts. For this application, we chose to use the TPS61232 step up converter, which was specifically made for Lithium-ion battery applications according to the datasheet.

This schematic that is supplied with the datasheet is built for a standard 5 volt output. Since our design also requires a 5 volt output, the values of the impedances in the circuit will be sufficient for our design, although we will be able to customize them later in the process if required. It is important to note that the datasheet specifies that the inductor should have a quality factor above 25 at the operational switching frequency to maximize the efficiency of the circuit. This datasheet also specifies that the capacitors should be either the X5R or X7R type, because other types of capacitors tend to become resistive at high frequencies **[48]**.

Power Supply Schematic: The complete schematic for the power supply system, including the battery and both voltage regulators is shown in Figure 14 below, created with Eagle PCB software. This schematic shows two outputs, one at 3.3 V and one at 5 V, that can supply power to the rest of the components in the system.

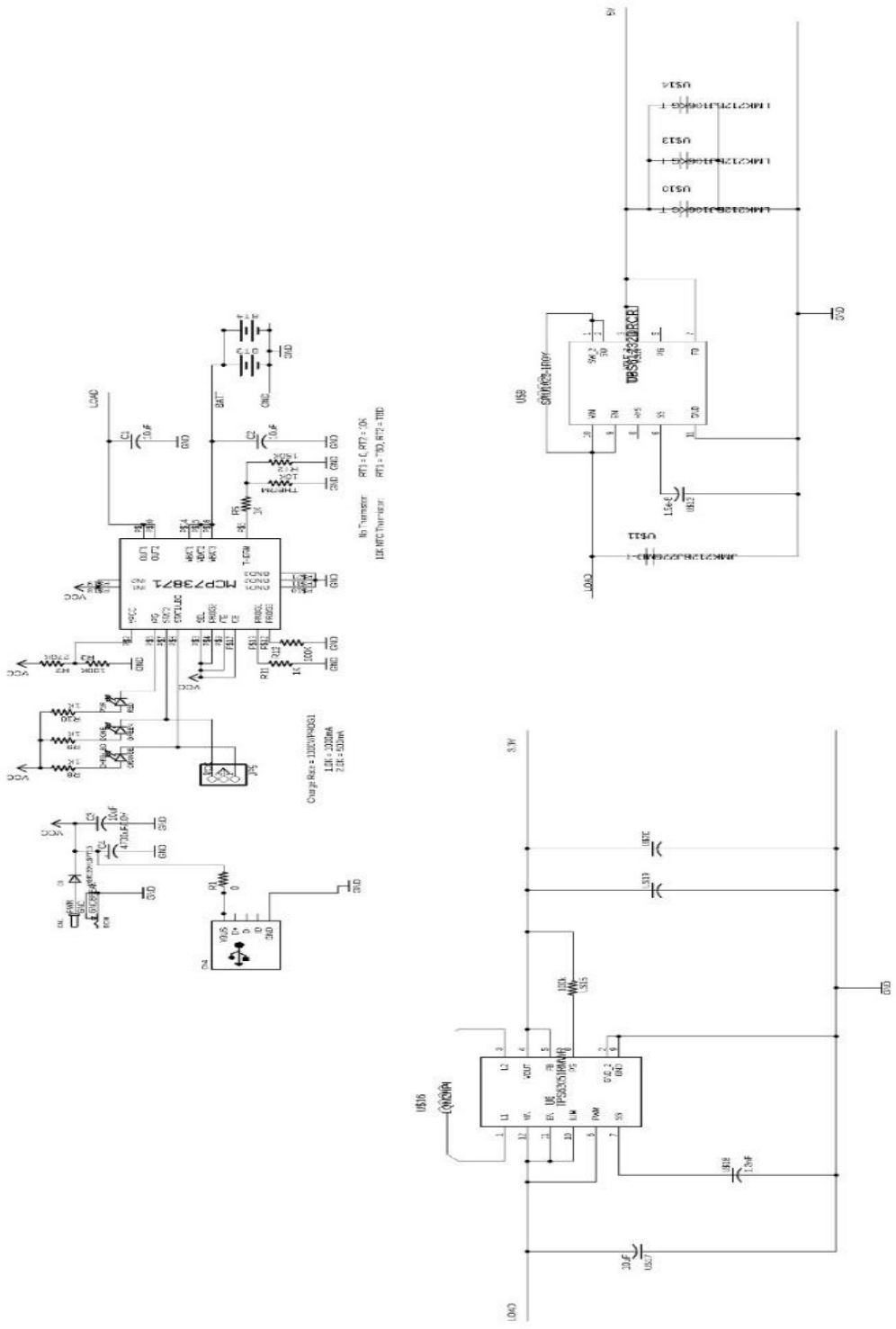


Figure 14: Power Supply Schematic

Listed below is the bill of materials for this power supply portion of our system, in Table 39.

Part	Manufacturer	Model	Quantity
INR 18650 3500mAh Battery	LG	MJ1	2
TPS61232 Voltage Regulator	Texas Instruments	TPS61232YFFR	1
TPS63051 Voltage Regulator	Texas Instruments	TPS63051YFFR	1
MCP 73871	Microchip	MCP73871-2CCI/ML	1
4700 uF Capacitor	Wurth	860010278024	1
10µF X5R Capacitor	Murata	GRM188R60J106ME84D	6
1.5µH Inductor	Murata	1269AS-H-1R5M	1
1.0µH Inductor	Coilcraft	XFL4020-102MEB	1
2µF X5R Capacitor	Murata	GRM21BR60J226ME39	1
22µF X5R Capacitor	Taiyo Yuden	LMK212BBJ226MG	3
10nF X7R Capacitor	Murata	GRM31BR72J103KW01L	1
1k Resistor	Panasonic	ERJ-6ENF1001V	4
100k Resistor	Rohm Semiconductor	TRR03EZPF1003	2
150k Resistor	KOA Speer	660-RN732ATTD1503B25	1
270k Resistor	Susumu	RR1220P-274-D	1
10k Thermistor	Amphenol	NK103C1R5	1

Table 39: Bill of Materials for Microcontroller schematic

The battery to 5-volt system is shown on the left, while the battery to 3.3 volt system is shown on the right. The general configurations as well as impedance values were created with recommendations from the datasheet for our specific applications. If able, our stretch goal of adding solar panels will be implemented in this schematic by connecting to the battery.

It is important to note that this is the schematic, and the implementation on the PCB may be arranged in a different way. For the PCB implementation the battery will not be directly on the board, but two wires running from the battery's holder will be soldered on to the board. The purpose of this is to keep our PCB, enclosure, and device as a whole as small as possible. This will also help minimize the heat from the battery dissipating on to our PCB and possibly affecting the efficiency of the rest of our components.

Testing: Unfortunately, we are unable to test the two voltage regulators that will be implemented in our design. Although it was in our best interests to test every component before proceeding with implementation, we are not able to do so with the voltage regulators due to their size. The TPS61232 has the dimensions of 3mm x 3mm, and the TPS63051 is even smaller with dimensions of 1.6mm x 1.2mm **[44] [48]**. With components this small, it is impossible to test them on a breadboard, or at all without soldering. Due to the economic constraints of this project, soldering to test these components was not feasible. When testing other components, we will operate under the assumption that the voltage regulator portion of the design will work as intended and output the 5 volt and 3.3 volt signals.

6.8 Microcontroller Design

The microcontroller is the fundamental block of our device, it will control and interact with all the sensors as well the user's cellphone. It is connected to the ultrasonic sensors, the accelerometer, and the Bluetooth module and must be programmed to perform the necessary functions of each one. It will have a VCC of 3.3 V from the previously designed voltage regulator module. We have designed it so that the accelerometer and ultrasonic sensors attach to the GPIO pins of our MSP430FR59691 microcontroller, while the Bluetooth module will connect to the microcontroller via UART. The microcontroller also has various impedances attached to certain pins as recommended by the datasheet. The full schematic for the microcontroller and attached sensors is shown in Figure 15 below, following by the Bill of Materials for this schematic in Table 40.

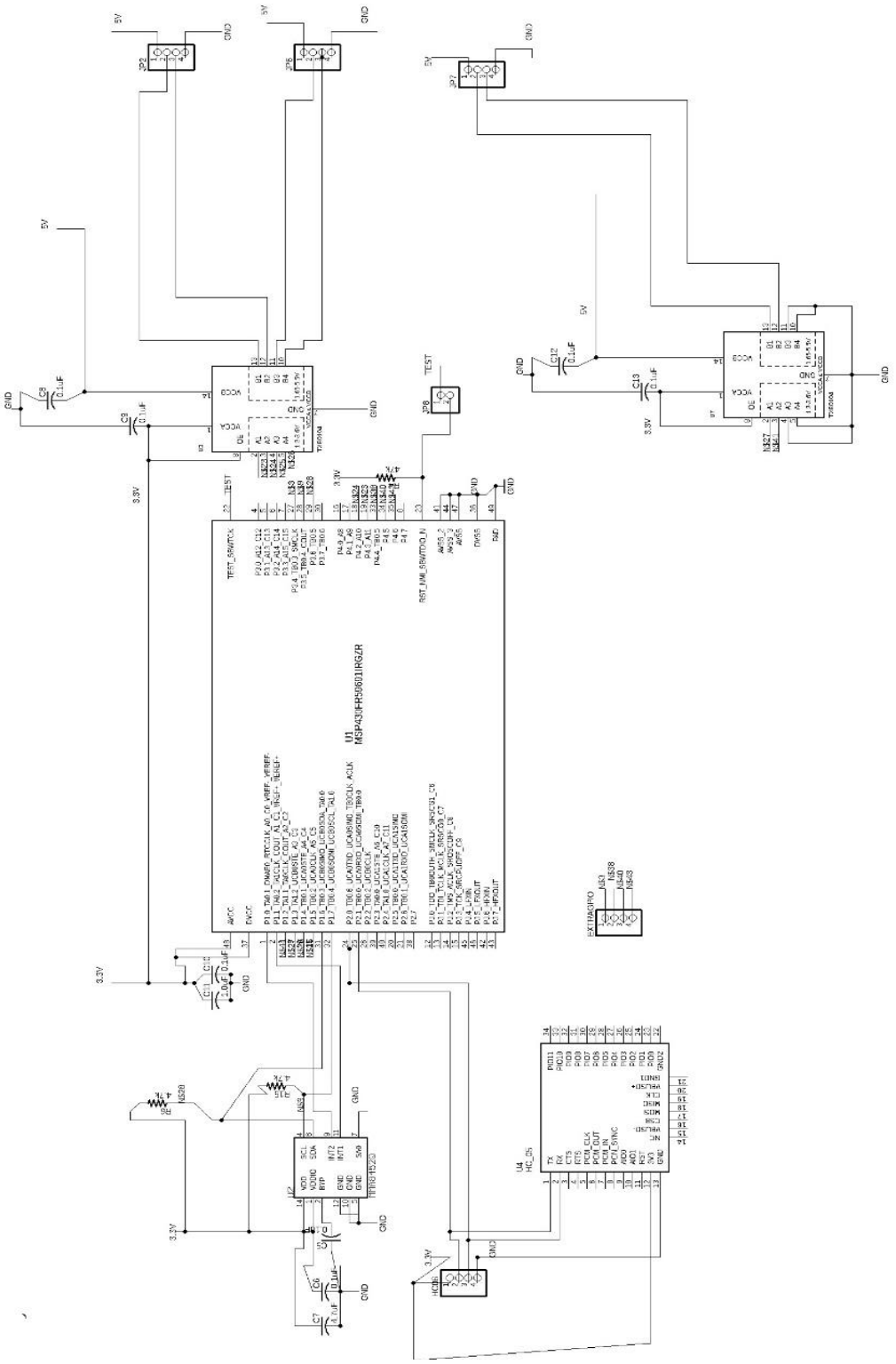


Figure 15: PCB Design

Part	Manufacturer	Model	Quantity
HC-06 Bluetooth Module	Olimex Ltd.	HC-06	1
MMA8452Q Accelerometer	Xtrinsic	SEN-12756	1
HC-SR05 Ultrasonic Sensor	Iduino	HCSR0501	3
TXB0104 Level Shifter	Texas Instruments	TXB0104DR	2
4.7 µF X5R Capacitors	Taiyo Yuden	EMK212BBJ475MK-T	1
0.1 µF X5R Capacitors	Taiyo Yuden	JMK042BJ104MC-W	8
4.7k Resistor	ROHM Semiconductors	ESR01MZPJ472	3
50k Resistor	Vishay	CRCW040250K0FKED	1

Table 40: Bill of Materials for Microcontroller schematic

The HC-06 Bluetooth module communicates with UART via pins P3_6 for transmitting data and P3_7 for receiving data **[51]**. These will connect with the MSP430FR59691 on its respective UART pins 2.5 for receiving data and 2.6 for transmitting data **[52]**. The other most notable pin on the HC-06 is pin 12, which is where the 3.3-volt power source will be supplied. For implementation, we planned on soldering the Bluetooth module directly on to the PCB, however due to shipping issues we were never able to accomplish this.

The MMA8452Q accelerometer connects using both I2C and GPIO connections **[53]**. The SCL and SDA pins each have a pull up resistor configuration in order to avoid floating values and are connected to the SCL and SDA pins on the MSP430FR59561. The interrupt pins are connected to generic GPIO pins on the microcontroller, and the inputs have decoupling capacitors in order reduce high frequency noise in the power supply - as recommended by the datasheet.

The three HC-SR05 ultrasonic sensors are designed to function as a three-pin accelerometer in order to reduce the amount of GPIO connections needed for our

design. Since the accelerometers operate on a different voltage than the microcontroller, it was ideal to reduce the amount of connections between the accelerometers and the microcontroller. In order to accomplish this, it was necessary to tie the OUT pin to ground as to be in a low state. The low state, according to the datasheet, activates the 3-pin functionality of the HC-SR05. This enables the trigger pin to act as both the input and the output, disregarding the need of the echo pin.

In order to have the HC-SR05 ultrasonic sensors, which operate at 5 volts, interact with the microcontroller, which operates at 3 volts, we needed to shift the voltage level of the input and output pins. Our first consideration was a simple voltage divider - but this solution could give us unstable voltages and some extra heat that would be dissipated through the resistor. Since our design required three separate connections between the ultrasonic sensors and the microcontroller, it was determined that a dedicated level shifting integrated circuit would be the best route in order to shift the voltage levels. This logic level shifter, the TXB0104 connects each I/O port of the microcontroller with the corresponding I/O port of the ultrasonic sensor. It also connects to both power supplies along with a pull-down capacitor to reduce noise. The OE pin on the level shifter helps protect the circuit - it tells the circuit to remain in a high impedance safety state until it detects that the power supply has been turned on.

This schematic incorporates our microcontroller as well as all the accompanying sensors, and the connections between them. It is important to note that the PCB design will incorporate other important design techniques that may change the placement of devices within the system. For example, capacitors are recommended to be as close as possible to the respective devices they are filtering noise from - but in this schematic the general functionality is shown but not the exact placement as required by the PCB functionality. In terms of prototyping however, this schematic is functional.

6.9 Raspberry Pi Module

The Raspberry Pi Module is necessary for all of the video recording and transmission of our system. Ideally our module was going to consist of three parts - the camera module, the Raspberry Pi, and the Sleepy Pi. The camera module is for recording the video and transmitting it to the Raspberry Pi with the built-in connections. The Raspberry Pi’s purpose is to transmit the video wirelessly to the user’s cellphone. The Sleepy Pi’s purpose is to manage the power consumption of the Raspberry Pi to optimize our battery life. This portion of our system consumes the most power, so it is important that we optimize this design for the lowest power consumption possible through both software and hardware techniques. The block

flowchart in Figure 16 below shows the layout of how the Raspberry Pi module will be connected, including the camera

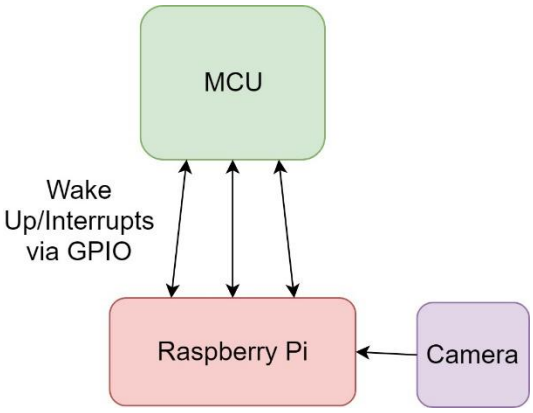


Figure 16: Raspberry Pi Block Diagram

In Table 41 below, we listed the Bill of Materials for the original Raspberry Pi module portion of the design. The Sleepy Pi and some GPIO hookups were all that was connected to the pi aside from its power. These GPIO pins are what allowed us to communicate with the Pi, starting and stopping the video stream on command.

Part	Manufacturer	Model	Quantity
Raspberry Pi Model 3B+	Raspberry Pi	MS.004.00000024	1
Sleepy Pi 2	Spell Foundry	SFY-10011-S	1
OV5647	Omnivision	B0033	1

Table 41: Bill of Materials for Raspberry Pi Module

In order to supply this module with power our 5-volt source would plug directly into the Sleepy Pi 2 device. The Sleepy Pi 2 device receives power either via a USB connection or the I/P header. The datasheet of the Sleepy Pi 2 specifies that if powered with the I/P header the low power mode will only be able to slow power consumption down to 10 mA - 20 mA, which is not as low as we desired **[50]**. The USB connection on the other hand will give us the low power mode consumption results of less than 200 µA as expected. In order to implement the connection with a USB cable, we must solder the voltage and ground wires on one end of a USB cable on to our 5-volt source that is able to be plugged in to the Sleepy Pi 2 to power this entire module.

The Sleepy Pi 2 device connects to the Raspberry Pi Model 3B on the GPIO pins. In order to have complete control over the Raspberry Pi, it connects to all 40 GPIO pins simultaneously. With the Sleepy Pi 2 plugged in to the Raspberry Pi, there is no need to supply the Raspberry Pi directly with power. The Sleepy Pi has an internal switch that will be able to power the Raspberry Pi off and on as necessary through the GPIO pins. The Raspberry Pi datasheet specifies that the GPIO pins that can power the device are pin #2 for 5 volts, and pin #6 for ground. While this

information is not especially vital to the implementation, it is good to know which pins are responsible for supplying power in case any power related issues were to show up during implementation.

The camera connects directly to the Raspberry Pi via the AMP connector, which uses a MIPI Camera Serial Interface 2 (CSI-2) to power and receive video feed from the camera. The camera receives power through this interface and has no need for an external voltage source. Due to the Raspberry Pi's GPU being closed source, only official camera modules such as the OV5647 are able to seamlessly interface with the Raspberry Pi. Once the camera module's pad connects to the Raspberry Pi's AMP connector the Raspberry Pi should recognize the camera immediately and be ready for software implementation, as long as everything is powered on.

Unfortunately, due to time constraints, the Sleepy Pi was never implemented in this portion of the design. Instead, the Raspberry Pi was simply sent a shutoff command upon pressing the close button in the app. While this meant our system did not have the peak battery life and startup time that we had hoped for, we still ended up with a fairly efficient system and the Sleepy Pi could be easily added in the future generations of the design if they were to be pursued.

6.10 PCB Fabrication

In order to realize our project, it was necessary to create a custom PCB that fulfilled the needs of our design. There are many options to choose from in terms of a vendor to manufacture our design, as well as PCB design software to help us create the perfect design. We evaluated many options and made our choices based upon the various constraints of our project. This was an integral part of the senior design project - separating it from the work of a hobbyist to that of an engineer.

6.10.1 PCB Vendor

Choosing a vendor for our custom PCB is a crucial step in constructing our design. We require a custom PCB that is sturdy, low-cost, and able to be shipped in a short amount of time. In order to choose which vendor would be our best option given these restraints, we created comparison Table 42 below with a few options.

	JLPCB	PCBWay	OSH Park	Elecrow
Cost 2 layer PCB	$2/10 pieces	$5/10 pieces	$155/10 pieces or $77/3 pieces	$4.90/10 pieces
Cost 4 layer PCB	$15/10 pieces	$49/10 pieces	$372/10 pieces or $155/3 pieces	$49/10 pieces
Shipping options	$18 - 5-7 days $17 - 17-22 days	$21 - 5-8 days $12 - 10-15 days	$20 - 1-2 days $5 - 2-3 days $0 - 1-5 days	$20 - 6-10 days $18 - 7-14 days $11 - 11-17 days
Reviews and Remarks	Great quality Inconsistent customer service	Great quality Good customer service Fast build time	Great quality Great customer service	Good quality Responsive service

*Table 42: PCB Vendor Comparison**
**Based on 100mm x 100m, 1.6mm thick, 1 oz. copper boards*

With any of these choices, compatibility wouldn't be an issue as they all take gerber files that can be created with any modern PCB design software. While OSH Park's shipping times are unbeatable, their prices are simply too expensive for our budget. When comparing the other three PCB fabrication companies - the price is nearly negligible, and the biggest factor becomes quality, delivery times, and reliability. Based upon our research, we decided that PCBWay offered the best combination of these three factors and decided to use them for our custom PCB. PCBWay has many options for customizing the PCB, however many of the options are not within our budget. Below are some listed specifications of the characteristics and options within our budget.

Specifications:

- FR4 boards with a TG130-TG140 temperature rating
- 1.66mm thickness with 1 oz. finished copper on both sides
- HASL surface finish with and without lead is affordable. Immersion gold may be affordable for the final design
- Minimum hole size is 0.3mm
- Minimum track spacing of 6/6mil

Assembly of the board will be attempted by our team personally soldering the components on to the PCB. It is important to note that none of our team has soldering experience. Due to some of the components being extremely small, less than 4mm2, the difficulty may be higher, and it is possible that we seek external assistance to solder these specific components.

6.10.2 PCB Design Software

In order to create a custom PCB, it is required to use a software that will be able to create a virtual version of our design. There are many professional software options, however we chose to focus on two due to monetary constraints. The two that we compared before choosing for our projects were Eagle, which is free for students, and KiCAD, which is completely free and open source. With a quick internet search, we were able to see that both software's had an adequate amount of learning resources in order to familiarize ourselves with the software in a timely fashion. While researching the comparison between these software's, we learned that placing the PCB components in Eagle was more intuitive than in KiCAD. On the other hand, KiCAD has a built-in 3D viewer - which could be helpful in visually checking the PCB design for errors before sending it out to be manufactured. Eagle's biggest advantage was compatibility with the Texas Instrument WeBench software, which could load schematics and PCB layouts directly into our software. This could be very helpful, especially since many of the components being considered for our project were from Texas Instruments. With this knowledge in hand, we decided to use Eagle to design our PCB. With this software, we could create a gerber file to submit to the manufacturer.

Although none of our team has created a custom PCB before, various online resources exist that helped familiarize us with the software and workflow. In particular, Sparkfun served as an extremely valuable website filled with tutorials and the general rules of thumb to create a satisfactory PCB. For the purpose of learning, we decided against using the autorouter within the PCB software. We decided it would be beneficial to our understanding of PCBs if we had to learn the rules and frustrations associated with manually routing the traces on the PCB.

In order to create a custom PCB with various components in the design software, it's necessary to have both the schematic symbol as well as the physical footprint

of the component. This is necessary in order to the manufacturing company to execute drilling of the circuit board that is accurate for our specific components. With all Texas Instrument devices, both the schematic and physical footprint are available to download on the accompanying product website. For more obscure components it was necessary to either download these schematics and footprints from a third party or design them ourselves. When downloading from a third party, it was necessary to verify that the schematic and footprint are accurate according to the datasheet of the component, since these files weren't created by the manufacturer of the actual components. When using files from a manufacturer website it was important to use the footprint that matches the type of package that we ordered our device in - since one integrated circuit could be sold in many different packages. The risk of not selecting the correct package or not verifying a third-party designer's footprint is that we may order a PCB that does not fit the components we have in hand - requiring us to either switch components or reorder a PCB. This is an outcome that is best avoided due to our monetary constraints, so attention to detail while designing the PCB and choosing or creating the footprint of our components is highly important to avoid these obstacles.

7.0 Software Design

Our software can be broken up into two main areas: The android application and the hardware programming. The android application is the highlight of our design. It is where the user will spend the majority of their time and shows them all of the information that the mounted hardware is providing. It is critical that every component of a system appeals to the demographic it is intended for, and for our design it starts with the application. The microcontroller is not meant to be seen or accessed by the user, however it plays a critical role in that data that is being sent to the phone. The same can be said for the Raspberry Pi, which handles not only the video encoding but the transmission of the signals over Wi-Fi.

While there are very different functionalities behind these two areas, the methods that we will use for the development and testing remain similar. This section goes over how we will collaborate, what we will use for programming and debugging, and how the hardware fits in with the various software written for it. Our final decisions for how we go about our software design are a combination of research from the different technologies at our disposal, as well as a combination of the experiences from the developers on this team.

7.1 Development Environment

Taking advantage of the best tools we have available to us will ensure this software's timely completion and that the collaborative effort will not become clustered. Our development environment consists of the Android application and the microcontroller. While the microcontroller does not have a UI to be developed, there is a programming to be had for it which we will use the same process to keep track of everything. The aspects of our development environment include the IDEs, keeping track of version history, and the methodology we will follow for overall development.

7.1.1 IDE and Development Board

When it comes to the IDEs used for our programming, we had a few choices for both the MCU and the app. For the app, we considered both Android studio and the Netbeans IDE. Netbeans has the advantage of being taught in academia for our developers, so everyone has had experience with it. Netbeans, as well as Android studio, does tend to take up a lot of processing power and ram, which is something we have to deal with regardless of the IDE. Android studio however, is developed by and updated by Google, who also makes the Android operating system. One of our developers has had extensive use with this IDE, specifically with developing an Android application.

In the end, we decided on developing the Android app using Android studio. This IDE offers a lot of features that are useful for app testing and development, such as an emulator of an Android phone. Section 7.0 goes into more detail of the testing of the application, which involves loading new builds onto the test phones, but being able to emulate our app on the computer will save us a lot of time by being able to debug quickly after building. This IDE is also the most popular choice for android development, and thus there are plenty of resources online for us to use, which was a deciding factor.

For programming the microcontroller, we have a development board that will allow us to program the MCU even after it has been soldered onto the PCB. This was done in contrast of getting a development board where the MCU much be plugged into it, so that even if we finished our design and everything was soldered on, we could still change the code. The development board is simply plugged into a Windows computer using USB 2.0 and can be programmed one of two ways, both of which we will utilize.

Energia is TI's own IDE for their microcontrollers and offers a wide range of features that give us quick and easy access to utilizing the microcontrollers. Code Composer Studio, or CCS for short, is a much more complex IDE also made by TI, that can be used to program their microcontrollers as well. The main difference between the two is that CCS has more tools for debugging, such as being able to see variable changes within the program, while Energia has a much more limited tool set. The main use of Energia for us will be for basic initial hardware testing, where we only need basic code to ensure the functionality of the sensors. CCS will then be used for bringing all of the hardware together, due to our need for more serious debugging in that stage of development. Both IDEs use the C language, however Energia's code is much more simplistic.

7.1.2 Version History

Keeping track of the changes to your developing code base is essential, and a platform that makes this process as easy as possible is just as important. The use of Git was unanimously voiced by all members of the team, as all members working on the application have used it in the past. **[54]** Git, according to the Git website, is, "a free and open source distributed version control system," which makes it the most accessible and resourceful addition to our development environment.

Our team will be utilizing GitHub to keep track of our software progress and revisions. The version control system of Git is great in general for keeping track of everything, but since we have three programmers, having one central repository that each one can access is crucial. GitHub also has different ways to access their platform, utilizing both a command line and a GUI platform. The most useful feature of GitHub that we are taking advantage of however is the website format itself.

While each member has to push and pull from the repository on their own, the GitHub website gives a great overview of all the folders and files we are pushing to the repo, and even the push comments we send with them.

7.1.3 Agile Development - Scrum

Our team had the choice between the two main software development methods, those being agile and the waterfall method. The waterfall method follows a single, unobstructed path from conception to the finalization of the project. There is very little room for change, and any optimizations that come up during the development process have to be approved by the customers and the managers before they can be applied to the design. This sort of software development works great for large companies, with large amounts of people working on the project, as changes should be approved so communication can remain fluid.

In our case, considering we only have 3 developers in our team, the agile method makes the most sense. It is centered around the understanding that there is an element of unpredictability in software development, and that parts of the design will change as development continues. Agile encourages communication between developers and allows us to make changes and improve on our code as we make it, giving us more creative freedom. It allows our small group of developers to update the requirements and design of the app as we go along, and to have working builds of the app at regular intervals, keeping us accountable for one another and give us useful feedback more often. It is for these reasons that our group will be utilizing the agile method for software development. **[55]** Below is a figure, figure 17 that shows the main differences between the two. As you can see, waterfall is very static, as opposed to agile which is much more dynamic.

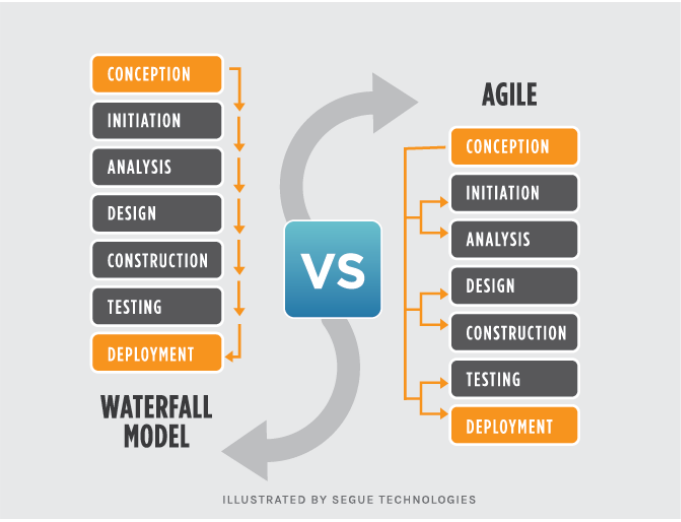


Figure 17: Waterfall vs Agile Method (Courtesy of Segue Technologies)

The way our team will take advantage of agile is by the use of scrums. The scrum method has us break up the overall software development into stages, called sprints. Each sprint is treated like its own smaller software development project, with its own due date and a final, runnable build at the end of that sprint. With our development period running from the end of July to roughly the end of November, this gives us about 4 months to complete the Android app and the programming for the hardware. Of course, it is not smart to plan according to the presumed date of the senior showcase, so our set completion date is set at **November 16th, 2018,** the Friday before Thanksgiving. This date was set as a goal for completion of major components, however we continued to work until the day before our presentation to ensure proper functionality of all hardware.

Though agile and scrum do refer to software development, the hardware must be considered since the two have to communicate with one another. Below in table 43, our sprint schedule is broken down. We are going to consider the microcontroller, raspberry pi, and Android app developing programming in this scrum development lifecycle. This allows us to look at the hardware from purely a functional standpoint and give us a chance for all 3 components to be scheduled around one another. Of course, hardware cannot be ignored in these scrums, and the progress of certain sprints might be compromised due to hardware failures or other related issues, so this schedule is under a best-case scenario situation.

Features to implement	**Start Date**	**End Date**
Set up all development environments	8/1/2018	8/3/2018
Build framework for the Android app Set up code skeleton for all hardware	8/3/2018	8/8/2018
Create the GUI menus for app Program timing and communication between hardware components	8/8/2018	8/22/2018
Create transitions between GUI menus for app Get Bluetooth communication setup between hardware and app Get Wi-Fi communication setup between hardware and app	8/22/2018	9/5/2018
Camera feed can be sent to the phone with no interruption Flags are raised from accelerometer and ultrasonic sensor information	9/5/2018	9/26/2018

App is able to boot up the hardware using Bluetooth Hardware wakes up and starts to transmit the camera feed to the app	9/26/2018	11/26/2018
Finalize design of the UI Finalize backend workings of the app with the hardware	10/17/18	11/27/18

Table 43: Scrum Breakdown

7.2 Features

The question to be answered about our Android app is quite simply, “What can this app do?” Our hardware is the backbone of the design, but is effectively useless without the application bringing all of that data together. Of course, the highlight of our entire design is having a rear facing camera that displays on an Android phone, but our app has a lot more depth than that, giving the user an experience they will enjoy and come back to. There is also the alert system using in unison with the ultrasonic sensors. The ability to switch to other apps in the user’s phone, and customizability, to tweak certain visual components of the app to the users liking. Smaller features are to be implemented, but these 4 big picture items are what we consider the selling points of the app

7.2.1 Camera Feed

The transmission of the camera feed over Wi-Fi is arguably the main feature of our design, and the highlight of the android application. When the application first connects to the hardware, and is opened by the user, the camera feed will begin to be transmitted immediately. The user will have the choice of having their phone in the horizontal or vertical positioning, with the result being the video feed being in a smaller resolution in the latter. Input from the sensors goes into more detail in the following section, but the same screen that is housing the video feed will have visual cues. The camera feed coming in will also have an overlaid distance gauge on it, giving the user an estimated view of how far away a curb or other obstruction would be when they might be trying to reverse.

7.2.2 Input from the Sensors

The second main feature of the application is taking the input from the ultrasonic sensors and relaying that back to the user in a meaningful way. There are two components to this, the first being the microcontroller, which is where the distance is to be set that the sensors will detect obstructions behind the driver. The second component is how this information is displayed on the application. While the user

is using the backup camera, there is an icon in the corner that is grayed out by default. When an obstruction comes into place that icon then changes the red and an audible alert tone can be heard by the user, giving both a visual and auditory aid to let them know to stop the car and check for the obstruction. This concept was adapted to 3 bars on the bottom of the screen, one for each of the ultrasonic sensors. The color by default will be green, with it turning to yellow when an obstructions comes within the first range of distances, and red if it comes within the second programmed range.

7.2.3 Transition to Other Applications

As stated previously, we are looking for a minimalistic approach when it comes to the content on screen while the user is driving. Once the user has actually begun driving, and is no longer using the Backup Buddy system, they need to have a default screen to look at. There is no reason for our users to continue using the app once they have begun driving, unless they wanted to see what was behind them while they continue to drive. For this reason, we wanted to implement a feature that allows the user to switch to different applications that are already on their Android device, right from our own app. The apps we would include for the user to switch to were picked based on common apps that our development team tends to use while driving, which include Google Maps and Google Play Music. These are apps we consider to be common apps that a driver might use while driving. This was one of the main features that was scrapped early on, as we adapted a simpler approach to the app, only allowing the user to interact with our hardware and not the rest of the phone.

7.2.4 Customizability

Section 7.3.4 goes into more depth of the settings we are allowing the user to change, where we mention not wanting the user to change the functionality of the app, for fear of performance issues. We do want the user to have the ability to customize some aspects of their experience however, such as some of the visual cues, opacity for some of the buttons, and how loud the alerts will be as well as their actual sound. We found from personal experience that some back up cameras today either give very quiet alerts when detecting something in the way, or the sound is not appealing to the ears, and all is needed is a visual queue. Everyone has different preferences, and giving the user the option to change some of these settings is very important to the design of the app. The final build of the app only allows the user to discover and change some Bluetooth connectivity settings, as the visual cues were implemented in a way that hard coded settings were not necessary for the user.

7.3 Software Design Overview

The conception of our vision for the app and hardware programming starts with what we consider to be the essential features. Creating user stories and developing requirements based on those stories lead into our class structure, use case diagrams, and eventually how the app will look in the user's hands. The diagrams in this section are intended as initial design concepts and serve as the building blocks for our apps final design.

7.3.1 User Stories

When coming up with the features of the application, and how the app will perform, we had to put ourselves in the consumers place. The following user stories were developed as a baseline for how the app would handle in the hands of our targeted user. By taking a step back and looking at what finished features the consumer would want to have, we can start to visualize the app from a developer's standpoint, thus coming up with more features. Table 44 below shows the various user stories.

ID	As a...	I want to be able to...	So that I can...
1	User	See the camera feed from the rear facing camera	see what is behind me as i backup
2	User	Get visual cues when an obstruction is behind the car	act accordingly so that a collision does not occur
3	User	Get audio ques when an obstruction is behind the car	act accordingly so that a collision does not occur
4	User	Connect to the camera assembly via Bluetooth	have it turn on before I get to the car
5	User	Connect to the camera assembly via Wi-Fi	get the various sensor information sent to my phone
6	User	Switch to other apps on my phone	Continue using my phone after I'm done using the camera
7	User	Adjust different visual and audible queues depending on my needs	Turn off sounds or visual cues if I feel I need to

Table 44: User Stories

From these user stories we were then able to form the requirement specifications for the Android app. The requirements are different from the requirement specifications of the system as a whole, as they go into more than the features of

the app, and into the inner workings and some of the more specific functions that don't necessarily reflect the whole system. These requirements also look to give us a view of the app both as a part of the entire system, and as a standalone entity. So instead of us saying that the app will give an audible tone when something gets in the way of the car, we say when the sensors detect something in range. Having this terminology when referring to the development of the app gives us a frame of mind to develop with the data in mind first, and then the system as a whole. Below in table 45 are the requirement specifications for the Android application that were implemented in the final design.

1	The app shall show the user a feed from the rear facing camera when the user selects the option to view it
2	The app shall notify the user when the ultrasonic sensors detect something in range with a visual queue
3	The app shall notify the user when the ultrasonic sensors detect something in range with an audio queue
4	The app shall be able to connect to the hardware assembly via Bluetooth 4.0
5	The app shall be able to connect to the hardware assembly via a 2.4GHz Wi-Fi connection
6	The app shall enter a security mode when the user selects the option to do so
7	The app shall keep track of previously recorded videos during the security mode

Table 45: Requirement Specifications of the Android Application

7.3.2 Class Diagram

The classes of our design are broken up by their broad functionality. Android studio allows us to design the UI with a bit of “drag and drop”, similar to other UI tools such as JavaFX. This gives us a bit of automation when it comes to the classes associated with the user interface. The functionality of various buttons and transitions between the UI panes that we create will have its own class, and anything related to the application itself without the data being gathered from the sensors will be used in that class as well.

The input that we are gathering via the camera, accelerometer, and various sensors will be included in a class called Hardware. The information gathered from the camera only has one function, but we are using the speed of the car to

determine certain app functionality, and this has many branching uses. The user will also have information specific to their use of the app, which more or less has to do with any settings they will change. These will be kept in a class called UserData. We can also use this class for keeping track of app usage and statistics for the user to access if they are interested. The final class will be called Main, which brings all of the other classes together. The boot sequence of the application will happen here, as well as managing how all of the data brought in from the hardware is to be used. Below is a class diagram that shows the various classes described, as well as how they interact with one another.

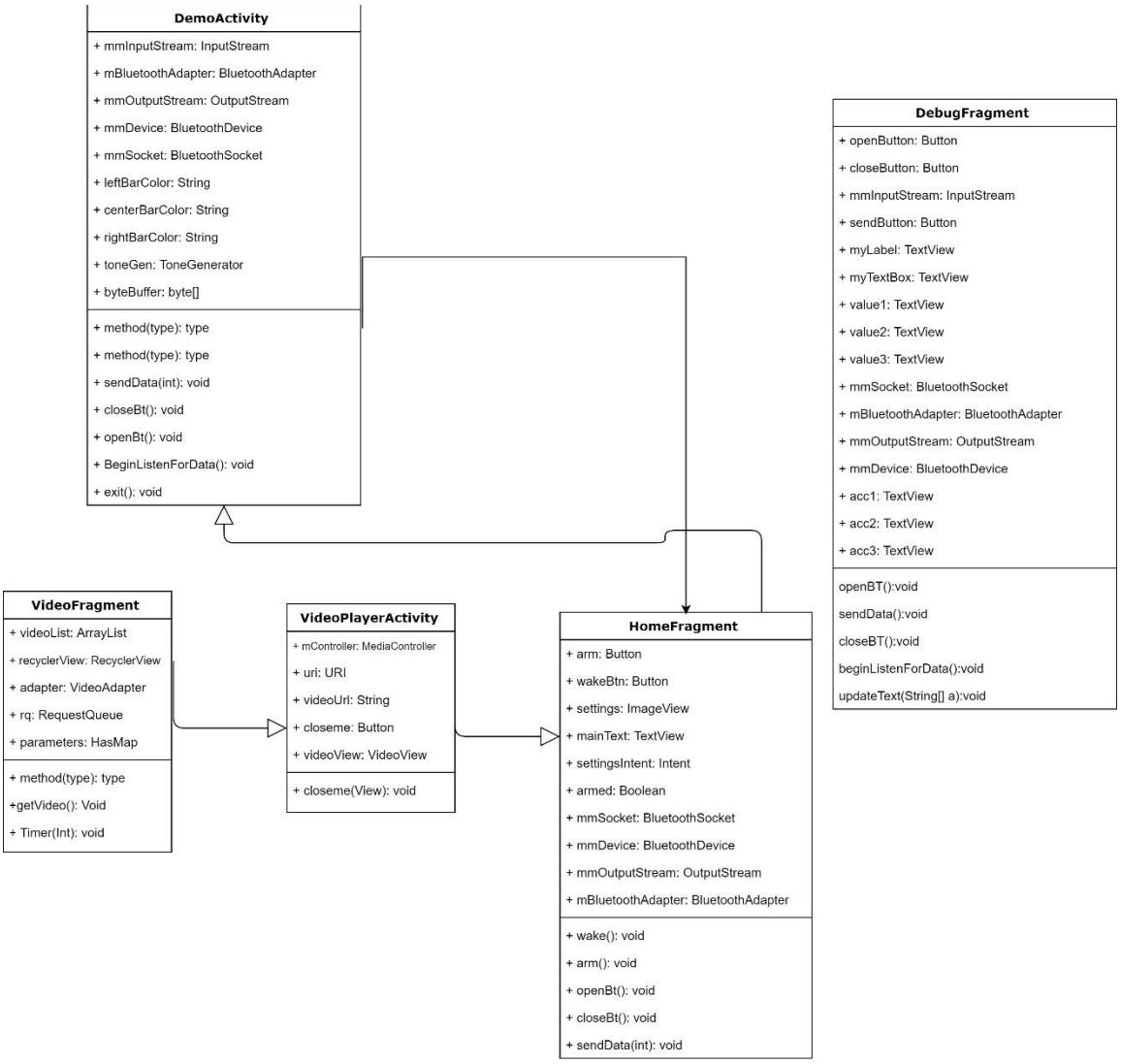


Figure 18: Class Diagram

7.3.3 Use Case Diagrams

The experience the user will have with our application is not intended to vary much between users. The application will boot into a state where the camera will be available initially, and then once the user is driving, they can choose to change some settings, or simply access another app on their phone. Branching paths are scarce in our design, and this was done intentionally. Again, considering that this is an application to be used in the car, the simplistic design continues into the user experience and the choices they are given. The less options they have to make when using the app, the less of a distraction their phone will be when they are backing up their vehicle.

Below is a use case diagram of the general operation of the application. The user will input info to the app, as well as receive information. The various sensors and the camera relay information through the microcontroller and raspberry pi (not picture for simplicity), and that information is then converted for use in either audible or visual cues for the user

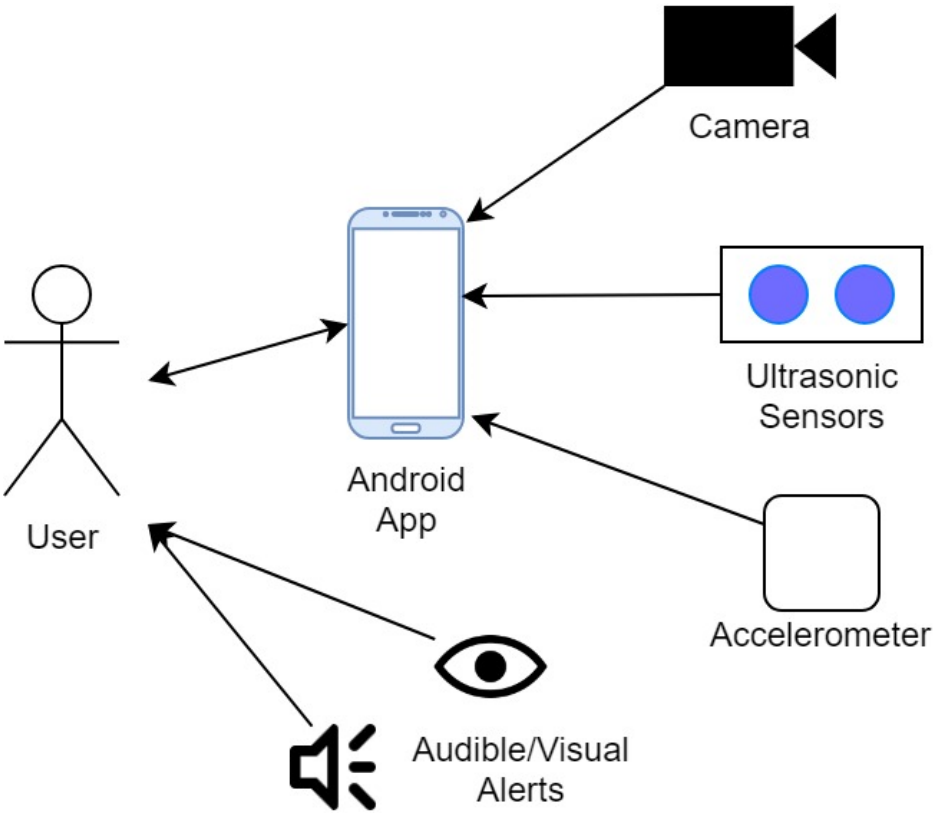


Figure 19: General Use Case Diagram

7.3.4 App Design

As stated above, we believe a simplistic design is not only good for an application to be used in a motor vehicle, but for a mobile application in general. The app starts with showing the camera feed, with if the car is going to be going in reverse. The pane that shows this video will take up the full screen, either horizontally or vertically, at the expense of a smaller view. The buttons on this pane will be overlaid with some lowered opacity, eliminating the need for borders on this pane but still giving the user the chance to exit or access the other buttons. Below is a schematic of the basic design of the camera pane. We are going for a simplistic design, once again, where the user is presented with only the information that they need to see

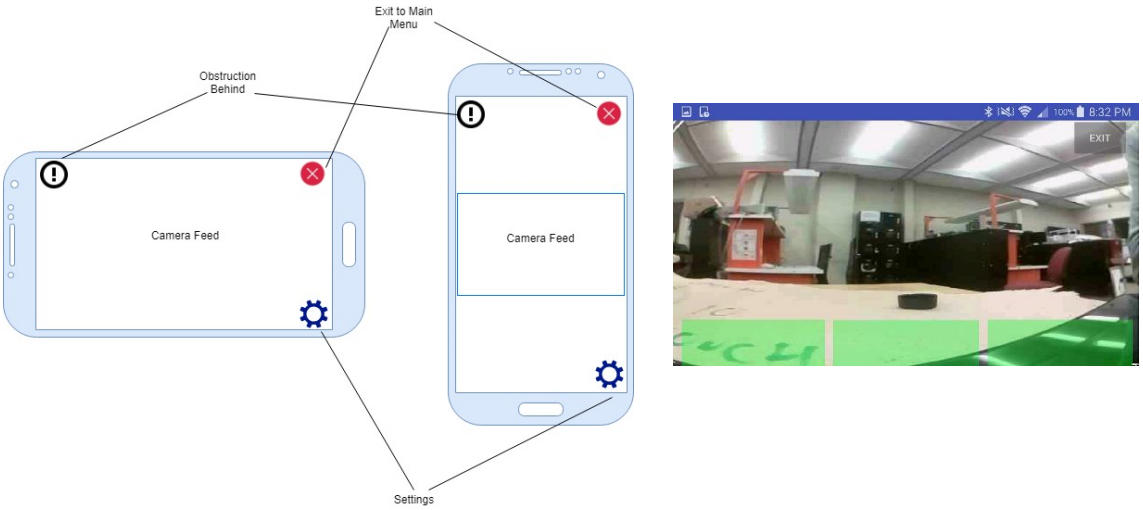


Figure 20: Camera View Horizontal and Vertical, concept and final

The main menu of the application is set as the default pane to go to when the car is driving, or in a state where the Backup Buddy camera is not being used. This main menu is where the minimalist principle comes into play, as this menu will be the first thing the driver will see by default once they begin driving. We are implementing large, rectangular buttons with large icons on them that can easily identify the function of pushing that button. Section 6.2.3 goes into more detail of the feature regarding being able to open other apps on the user's phone, and these apps with be the buttons and icons on the main menu. Settings and camera view will also be options on this menu. And while it is noted that the battery on the device is not designed for extended use when not in reverse, if the user wants to them can open the camera back up manually using this option. Below in figure 21 is a schematic of the main menu layout, which will only be available in vertical orientation.

Figure 21: Diagram of the Main Menu, concept and final design

The settings for the application will be aimed more at the user experience, rather altering the performance of the system itself. This was done so as not to compromise the architecture of our design, possibly altering how the systems performs. When the user opens the settings pane of the app, they will be greeted with a layout identical to the main menu, with large rectangular buttons that has icons on them making their purpose clear to the user. We do not want to give the user access to the backend workings of the system, which could potentially lead to performance issues and ultimately a worse experience, so the settings options are limited to preferences for each user. The settings that we are allowing the user to alter are the Wi-Fi name for the device, the default pane to open the device with, when to shut off the hardware, adjusting the opacity of overlay contents, the sound of alerts, and disabling alerts. Below in figure 22 is a schematic showing the layout and look of the settings pane

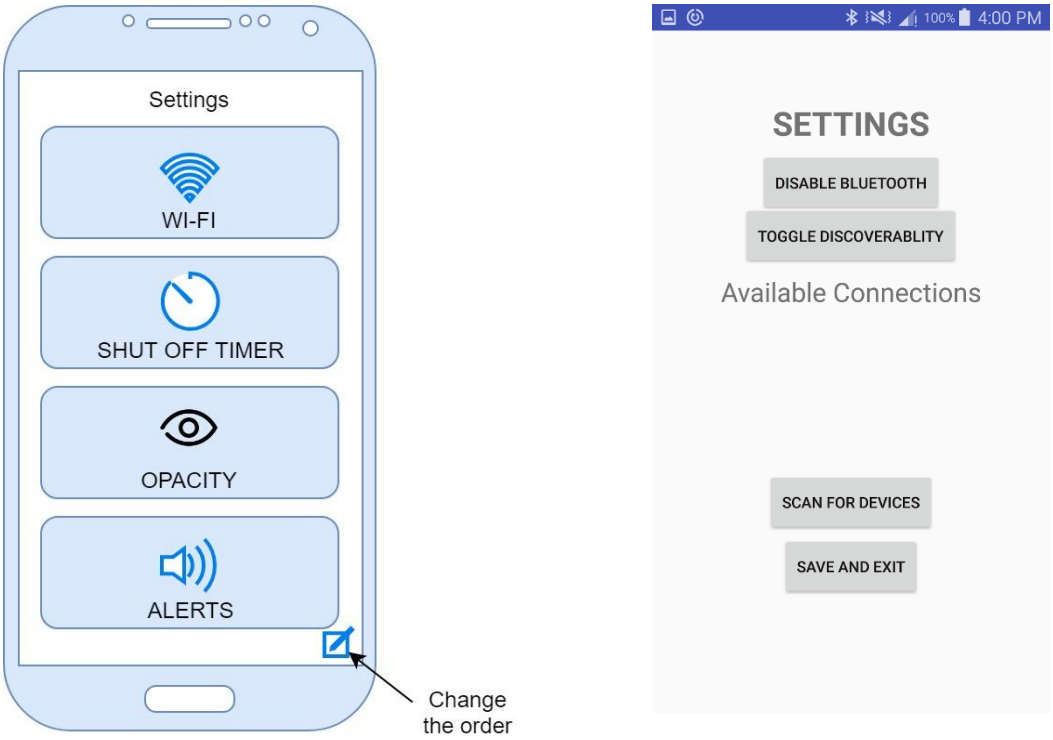


Figure 22: Diagram of the Settings Pane

The final part of the application is the security feature. During development we removed the feature that allows the app to shut down the camera feed automatically based on the cars speed, and replaced it with the user being able to set a security recording to start when the vehicle moves unexpectedly. When set from the main menu, the accelerometer is monitored and when there is sufficient change in motion, the camera feed will start with no indication to anyone looking at the hardware, and save it to a web server on the Raspberry Pi. From there the user can then look at the previously recorded footage. Below in figure 23 is both the list of videos as well as the player for recorded videos.

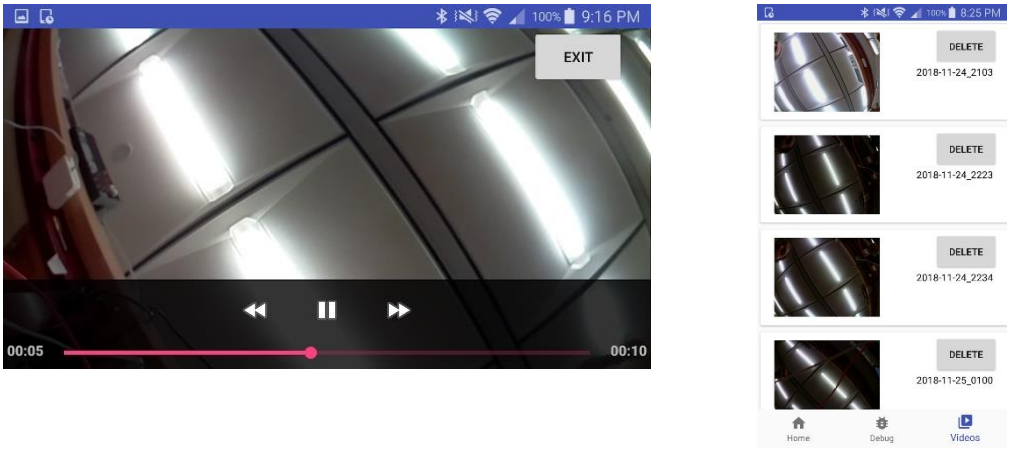


Figure 23: Security View, player and video list

7.4 Wi-Fi Network and Bluetooth Communication

Our design utilizes both Bluetooth 4.0 and 2.4 GHz Wi-Fi for transmission of data. The Bluetooth module, the HM-10, is a part of the microcontroller assembly, and is used in the booting of the raspberry pi. The user will boot up the app as they are going to their vehicle, and when the Bluetooth connection is made, the signal from the microcontroller is sent to the sleepy pi to take the raspberry pi out of its sleep state and turn on. The Wi-Fi communication is used to transmit the camera feed to the android app. Below in figure 24 is a diagram of the communication between these platforms and our application.

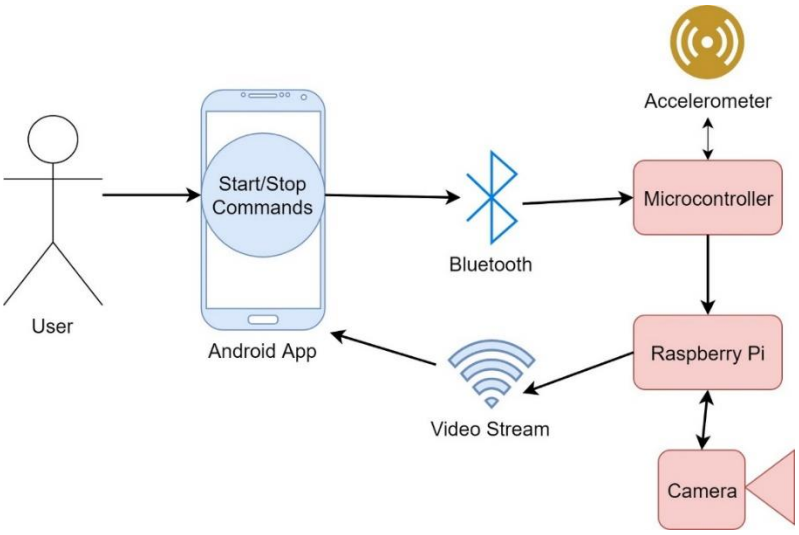


Figure 24: Wi-Fi & Bluetooth communication between app and hardware

8.0 System Testing

When it comes to testing, we want to make sure we have a set of predetermined procedures that we can always come back to and retest during the various development stages of the project. Testing can be broken up into 3 main ideas: Ensuring that all scenarios that the system is expected to be able to handle are considered, creating tests that can be redone and reverified at multiple stages of the design process, and creating tests that are as automated as possible to minimize wasted time.

The two main parts of our design are the hardware, which includes the camera, sensors, and housing that is mounted on the back of the car, and the phone app that will run on the Android operating system. These two systems stand on their own in some aspect, but complete testing cannot be done for one of those components without including the other one. This allows us to break our testing into 3 parts, one for each of the two parts of our design and then one part that accounts for the two parts working with one another. The testing process will give us the data that we need to improve on aspects that we know are necessary, but also allow us to make the decision to cut features or aspects of the design that might not be necessary.

Our end goal is to make a product that can fit universally on any motor vehicle, however this simply isn't possible for us to test and verify that our design could fit on every vehicle. We will however have a small sample that includes cars from some of the members of our group. This includes a 2014 Ford C-Max Hybrid, a 2008 Honda Civic, and a 2005 Toyota Corolla. These vehicles vary slightly on their rear design, giving us enough variance that the feedback from testing will be useful.

The application is made for the Android operating system, and the phone used for testing and demoing is a Galaxy S6 running Android version 6.0. While this application should, in theory, run on any version of Android, we chose a version that is a few years old to ensure compatibility and that we had a phone we could use exclusively for testing.

8.1 Hardware Testing

The hardware component is limited to the camera assembly and its power source. As mentioned above, while the system is of course designed to communicate with the android app, the testing for the hardware will only include features or aspects that do not include the app. The aspects of the hardware we need to run test on are proper power consumption and efficiency, adequate response from the various

peripherals, and ensuring that the hardware will stay intact when the car is in motion.

8.1.1 Power Consumption and Efficiency

Since this device is intended to not be tampered with on a regular basis by the user, having quality power consumption will ensure that the device only draws the power that it needs and doesn't leave any to waste. This is attributed to not only using minimal power when in use, but also effectively utilizing microcontrollers with low power mode to power down the device when certain features are not being used.

Our design was done with the intention that most of the activity of the device will be done during the vehicle's initial startup and going into reverse the first time. After this there is no camera feed being sent to the phone, and the device should theoretically be using less power. This will be the source of our tests to ensure that the correct power consumption is being used. Activating the devices low power mode during times that the camera feed is not being used, or the sensors data is not needed to be sent is the first step in this process.

8.1.2 Adequate Response from Peripherals

Our design encompasses many different inputs from the surroundings, which includes video feed, ultrasonic sensors, and light sensors. Making sure that these various peripherals are operational and are supplying not only consistent but accurate information to the microcontroller is of critical importance. Testing for these features needs to not only be done at a hardware level, but at a final design level when they are all acting alongside one another.

The video camera is expected to be transmitting a video feed to the application whenever the car is in reverse. On its own, ensuring that the camera feed to the phone is not interrupted for any unintentional reason is the first concern. This can be verified by simply powering the camera and connecting the phone to the devices own Wi-Fi network, to ensure that the feed is coming through as intended.

Checking the surrounding area behind the car for obstructions is done with the ultrasonic sensors. Testing for this feature can be done in a lab setting, with the sensors placed unobstructed facing an open direction, and are expected to send an alert to the application, or raise a flag, when an obstruction does enter. The vehicle will be in motion in the real-world setting. An unofficial test that we can do on the spot would be to simply walk with the device in a general direction to ensure that any obstructions would set off the flag, and as we move from a test bench to

a real-world environment using batteries, we would get better results and more usable data.

The accelerometer will be used to determine if there is motion on the car when it is not supposed to. Due to the sensitivity of the registers on the device, configuring the data to only mark that a change has happened when enough motion has occurred is important to the effectiveness of the device. For these testing purposes, we wrote the code so that the raw data coming in, which would range from -220 to +220, would activate the security recording when a change of 10 or more was made in less than 100ms.

Test Procedure	Expected Results
Shake PCB so motion would be captured from accelerometer data to turn on security feature	Security recording starts when the PCB is shaken or moved
Test that the rear facing sensors detect obstructions at various distances set by us	A flag will be raised from microcontroller when an obstruction comes into range
Test that the camera sends video feed to the android app at proper resolution and frame rate	Camera feed in test environment on computer has expected frame rate and resolution

Table 46: Overview of tests to be run on sensors and peripherals

During our testing, all of our main features worked, and we were able to move forward from testing to finalizing our design. The testing environment was done in a lab prior to everything being mounted In the enclosure, and repeated once everything was in said enclosure.

8.1.4 Hardware Mounting

Since motor vehicles are designed such that some can go over 120mph, our design needs to account for the vibrations the car would experience at high speeds. The testing for the system, to ensure the camera functions properly when backing up, does not need to be tested for this kind of environment. It is when the camera is no longer transmitting video and the assembly is simply in idle waiting for the car to go in reverse again that it will experience these high velocities. As a team we determined a good threshold for speed that would give us a good percentage of the average vibrations a driver would experience, while also not risking too high of speeds that could potentially damage our design. The velocity

we came up with is 45 mph, which we deem the max that we will expect our design to withstand on our test vehicles.

Our testing procedure for the hardware mount is very simple, at various stages of our design we will mount the assembly to the back of one of the cars, and drive the car to see if the assembly stays on. As we continue our design, the assembly will without a doubt change shape and weight, allowing us to record any change in behavior of it as it is mounted. When testing is being done, the car will be driven on the road for a total of 10 minutes, to simulate a driver's typical commute. Within these drives we will make sure the car comes to complete stops, accelerates, brakes hard and slow, giving us a complete assessment of how the various driving conditions affect the hardware. Further detail into what we will be testing for is found below.

Test Procedure	Expected Results
Fast Brake	The device should remain attached to the vehicle and not turn off
Accelerating from stop	Same as above
Accelerating to 45mph	Same as above
Coming to a complete stop	Same as above

Table 47: Overview of tests to be run for hardware mounting

Due to time constraints we did not get to mounting our device on the back of a car. This was due to the final design not being 3D printed, and the wooden enclosure we made would not have withstood the forces acted upon it during driving.

8.2 Software Testing

The software for our design is in the form of an Android application. The app has its own user interface and controls, as well as displaying information that is sent from the hardware component of our design. When it comes to application functionality on its own, testing needs to be done to ensure there are no app crashes, and that the Android version that we are designing for can run the application without any issues.

Unit tests will be created based on both backend functionality and the user interface for the app. Automation of the tests is not completely necessary in terms of design but makes our job as programmers easier to test new functionality. These

tests will also be kept track of in a spreadsheet, allowing us to track their progress as we update both the tests and code. The user interface will have tests as well, but since we need an actual person to use the test phone, automation is not an option. A spreadsheet with more diligent notes will be used in place of said automated tests, to keep track of the feedback these tests give us.

The detail of the software tests does depend on the code itself. Certain tests are obvious no matter the software, such as crash testing, but specifics regarding test scenarios will be up to how our implementation of requirements is. The final tests used for our app revolved around testing code as it was written, since the major functionality was UI design and getting data from the hardware. As new features were added to the app, the hardware would send its data and we would check the functionality of everything before moving forward.

8.3 Software and Hardware Testing

The Android application and device hardware are designed to work in unison, thus there are tests that need to be done regarding both components. These tests are centered around the functionality and performance that the user is expected to come in contact with. This includes the data that is sent from the device to the Android app, flags and triggers that come with certain events while driving, and alerts given to the driver through visual and audible tones.

The testing procedure for what is essentially the final design involves a lot of the scenarios that we would expect the consumer to go through. These tests will not only have to be done in a lab setting, where we can quickly debug and fix any issues that we come across, but also in a motor vehicle under real world conditions. The aspects that we would be testing at this stage in development would be the following:

1. Camera feed goes to the phone when the app sends the wake signal
2. Alerts are sent to the phone, both audible and visual, when an obstruction comes into range of the rear facing sensors
3. Application reverts to a separate page once the camera feed is cut off
4. Hardware goes into a low power state once the car is moving forward and no longer transmitting the camera feed
5. Hardware will come into full operational mode upon the app being started and a successful Wi-Fi connection is made

8.4 Testing Environment

With the various components to our overall design, there are some definitions that we made as to what aspects of testing will remain static to ensure a consistent

testing environment. 3 motor vehicles that are owned by 3 different members of this team will be used as test vehicles, for both the mounting of device and test drives to ensure the devices functionality. The end goal of this project is to make a universally mountable backup camera; however we cannot state this unless we test multiple different cars to ensure that it can be mounted this way. Having a consistent set of vehicles that we can test on will allow us to make some assumptions as to the mounting capabilities of our design. The motor vehicles that we are using are a 2008 Honda Civic, a 2005 Toyota Corolla, and a 2014 Ford C-Max Hybrid. The images below show the rears of the vehicles where the hardware would be mounted.



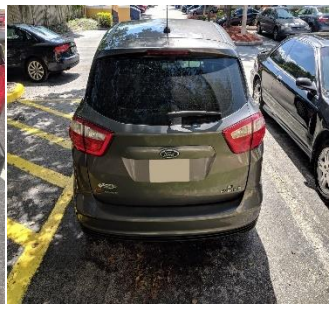


Figure 25: Honda Civic
Figure 26: Toyota Corolla
Figure 27: Ford C-Max Hybrid

The testing environment for the vehicles is the not static, having 3 different cars can eventually lead to design issues and multiple revisions. But this is something we need, as our intention is to fit this on as many cars as possible. The variance in the three vehicles listed above gives us varying tailgate designs, brake light placements, and vehicle heights. This much needed diversity was intended to be a great testing environment, unfortunately only a single car was used for testing and development in the end.

The Android application of course is not being written for a specific device, but for any device that is running the version of Android we write the app for. We decided that we wanted to have a dedicated Android device that we can use to upload new builds of the app to ensure its functionality. We have on reserve 3 different phones with 3 different specs. Our test phones are an LG MS500 Optimus F6, a Galaxy S5 and a Galaxy S6. The LG phone is a bit more outdated than the Galaxy phones and has considerably slower speed. However, we wanted to include this in our testing environment to see the real-world consequences of the limited speed of this phone with our app, giving us more feedback on its performance. In the end we just used the Galaxy S6, as it was the most current device, with the best hardware, and was the phone we did our demo with. Below are the images showing which Android OS is running on each of the test phones.

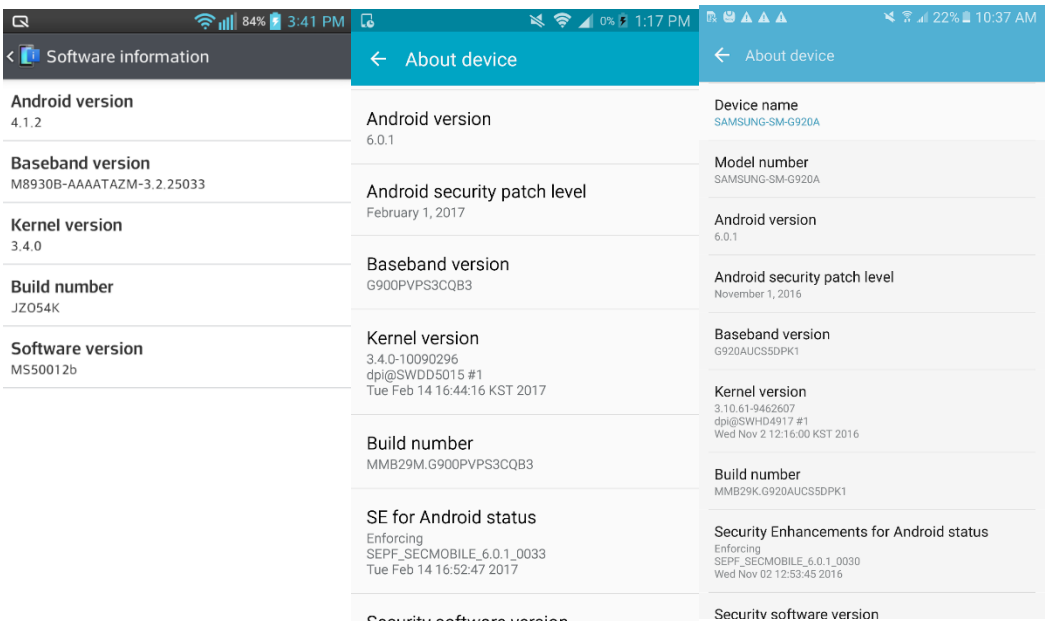


Figure 28: LG MS500 Optimus F6
Figure 29: Galaxy S5
Figure 30: Galaxy S6

Our team is utilizing git and the website GitHub for our code revisions and product builds. This is also how we will be keeping track of the unit tests, product backlogs, and the various builds of our application. This is gone more in depth in section 6.2.

8.5 Test Schedule

Testing isn't simply to ensure that the design is working as expected, it is also a way to learn from mistakes and improve that design. We want to test as much as possible, however we can't test our hardware as frequently as the software, considering software is a lot easier to automate and can be done with a few mouse clicks. The schedule is set around frequent yet efficient testing, as the data and feedback from testing is just as beneficial as the code and components these tests are designed to run for.

For our software testing, we will create and run tests for our code as we implement that functionality. Tests that can be run in the IDE will be automated, and once the user interface is implemented, major builds will be re loaded onto the test phone to be re tested. Waiting until major functionality has been updated or changed to re test the app on the phone will eliminate wasted time, since we would have to have someone test the app in person, a process which can't be automated. When it comes to the hardware component, there is no automation that can be done. We would have to test periodically as more sensors and components come together, to ensure adding a component did not compromise the circuit and everything

remains operational. For testing the app and hardware together, anytime either of those aspects receives a major change, that has past testing on its own end, testing between the two aspects will occur. Our ideology for testing is to test frequently enough that we can correct any mistakes right away before they get embedded and difficult to reverse, but not too frequently that we waste time running tests that are going to be repeated soon anyways. Below is a table giving an overview into testing categories and automation.

Design Aspect	Test to be Run	Automation?
Software	Tests based on backend functionality	Yes
Software	Tests based on UI on a device running Android OS	No
Software	Tests based on UI run on emulator	Yes
Hardware	Tests based on sensor data	No
Hardware	Tests based on device housing integrity	No
Hardware/Software	Tests based on application and hardware	No

Table 48: Testing Schedule Breakdown

9.0 Administrative

As stated in previous sections, the budget for this project was based around us as students, since this is a self-financed project. This is merely a constraint to us, as we have the resources on our college campus to not only prototype and get our system to its final stage, but some of the parts were free to use as college students.

9.1 Budget

The budget consists of the parts that are in our design. Shipping has not been included in this table, as this meant to represent the cost of the hardware that is in the final design. The cost of the components was fairly reasonable, with any components being bought in small bulks, reducing the overall cost for that component. The most expensive component, the Sleepy Pi module, did seem unnecessary, especially for the price. However, it is a very critical component, giving us a better use of the Raspberry Pi and being able to take advantage of its processing power. Below in table 49 is our cost for each of the components

Component	Model Number	QTY	Source	Dev Cost	Unit Cost
Ultrasonic Sensors	HC-SR05	4	Amazon	$15.98	$11.97
Accelerometer	MMA8452Q	4	Karlson Robotics	$25.76	$2.95
Microcontroller	MSP430FR5969	4	TI	$34.56	$3.86
BT Module	HC-06	1	Amazon	$8.99	$8.99
Sleepy Pi	Sleepy Pi	1	Spell Foundry	$55.95	$0.00
Camera	OV5467	2	Amazon	$53.98	$26.39
Raspberry Pi	Pi 3B+	1	Amazon	$35.99	$35.99
3.3v Regulator	TPS63051YFFR	5	TI	$0.00	$0.00
5v Regulator	TPS61253YFFR	5	TI	$0.00	$0.00
Lithium Ion Batteries	LG INR 18650 MJ1	2	Amazon	$14.98	$14.98
Charge Controller	MCP73871	3	DigiKey	$5.52	$1.84

PCB	Custom	20	PCBWay	$105.00	$5.25
Enclosure	Wood	1	Home Depot	$22.70	$5.67
Development	-	-	-	$17.19	-
TOTAL				**$396.90**	**$117.89**
EXPECTED BUDGET				**$500.00**	**$200.00**

Table 49: Budget

From our initial estimate of how much this would cost us, we were within the bounds of those projections. Making this something that could be used by the everyday car driver was important to us, and the price does reflect that. In terms of bringing something like this to market, from our testing and hardware research we estimate that we could bring this price down by at least $15, but a definitive statement cannot be made until the final product is released.

The costliest of the hardware components was the Sleepy Pi module. The raspberry pi is a very powerful machine for its size, however this accessory for it is necessary since it is known to have a long boot time. Through much debate and discussion, we deemed it necessary to be able to wake the Raspberry Pi up using a ping from the user's phone, but adapting this type of technology into something cheaper, and especially smaller, would drastically affect the price of our components. The raspberry pi itself will be configured in a way that is unnecessary features will be turned off, making its speed and Wi-Fi the only thing we are after, another room for budget improvements.

The ultrasonic sensors were another area of research that may yield a place for budget improvement. There were so many different ultrasonic sensors, and of course the SR-05 module we chose works best for our needs. But future iterations of our design, with other hardware changes considering, its possible the sensors could be changed out for something cheaper. We estimate that with all of these proposed budget revisions in the future and considering the updates that happen during the propitiating phase, that our total cost could actually be reduced by over $15. This is with the consideration of the product going to market, and rather us using consumer grade hardware like the pi, using a computer intended for mass producing inside of another enclosure.

At the end of Senior Design 2, we came in under budget both for the development cost as well as the unit cost of everything we were able to demo. Since we went through 3 iterations of the PCB we had to spend over $100 on the newest prints. This was the most expensive piece of our design, but was the most necessary as

each print gave us more information into what we needed to bring all of the hardware together.

9.2 Project Milestones

Post Senior Design 1 we had features and events that were to be completed at various points in the semester. Our professors told us to set November 16th as the date to aim for to complete everything, and while we aimed for this as well, we continued to work on our design until the last day to ensure everything was working. Many unexpected events caused us to have setbacks, both financially and timely. Below is the timeline of events that occurred over the past 2 semesters.

Task	**Start Date**	**Completion Date**
	Senior Design 1	
Brainstorm Ideas	5/14/2018	5/31/2018
SD Bootcamp	5/31/2018	5/31/2018
Project Selection	5/31/2018	6/8/2018
Divide and Conquer Document	5/31/2018	6/8/2018
Research on Component and Design	6/13/2018	8/3/2018
Order and Test Components	6/13/2018	7/27/2018
60 Page Documentation Draft	6/13/2018	7/6/2018
100 Page Documentation Draft	7/8/2018	7/20/2018
Final Documentation	7/20/2018	7/30/2018
	Senior Design 2	
Build Prototype (Breadboard)	8/20/2018	8/30/18
PCB Prototyping and Manufacturing	8/20/2018	9/15/18

Testing and Redesign	9/1/18	10/1/18
Finalize Design	10/1/18	11/26/18
Midterm Demo	10/31/18	-
Final Report	12/3/18	-
Final Presentation	11/28/18	-

Table 50: Project Milestones

The milestones also show that senior design 1 was more or less turning in the reports on time. This gave us a lot of flexibility when it comes to how we divided the work and split up our time. We found during our research that even though we found out a lot of the technical information of the components from the datasheets, the hardware testing gave us an insight into how they would act and perform when we start prototyping. We are using this information moving forward to spend more time in the lab, getting as much prototyping done as possibly, so we can run into the issues early on, correcting them before they build up and its too late or too expensive to change our design.

9.3 Division of Labor

As shown in the overall block diagram, the different aspects of our design were passed off to the different members of the group. A more accurate table showing how many people will be working on each design is shown below. We understand each person has their strengths, but we also acknowledge the need for each member to work alongside one another rather on their own and bring their results to the table

Member	Embedded Development	App	Circuit Design	PCB Design	Raspberry Pi Video	3D Enclosure
Dylan	P		S	S		P
Coleman	P	P	S			
Luca		P	S		P	
Zak	S		P	P		

Table 51: Division of Labor

Everyone helped at some point with every part of the project. The embedded development was directly related to the hardware design since this code would change how the hardware was being used. In turn whenever, new discoveries were made regarding the ultrasonic sensors or accelerometer, the PCB design needed to be changed, and we made sure to make as many changes as we

needed before ordering, since new changes would require another order, causing setbacks.

9.4 Issues and Challenges

The setbacks that we did have during our project timeline only furthered our understanding of the task at hand. The first major setback was our ultrasonic sensors. The 5 pin model we bought didn't function as was advertised, so we had to make a change to our PCB because of this, which was inevitable, but because of the form factor we needed it caused a lot of wires to be very close to one another. The accelerometer was also the cause of some challenges. Since the data we got from it was not what we had anticipated when doing our initial research, and testing proved that this data wasn't viable for our uses, we had to update the use for it. This caused a major function to be cut from our final design, replacing it with another feature, as well as another PCB modification.

10.0 Conclusion

Before the team began creating this document and conducting heavy research on the topic, there were admittedly some slight hesitations regarding the feasibility of the project. Looking at major retail sites like Amazon yielded many devices similar to what we wanted to build but all were wired solutions, and we were looking to build a device that was wireless. There was concern that these companies, with hundreds of thousands of dollars to spend on constructing and manufacturing these backup cameras, didn't go with wireless solutions because they were too difficult to make or not practical. Now, after spending the last month and a half of diving into every single aspect of the proposed device that we are making, there is a much stronger belief now that what we are setting out to make is totally possible and with enough optimization, also able to be made practical for the user as well.

While there were many challenges that arose as we began to actually build the proposed device in this document, all of our serious concerns had been eased in the beginning of development. Originally, we needed a chip capable of not only recording video, but also able to process, encode, and stream it out to the user's phone. Designing this on our own would have been a huge technical challenge in and of itself probably needing a second PCB. It would require a video sensor module, a Wi-Fi module, and a very powerful microcontroller driving them to be able to keep up with the rather extreme demands of capturing and streaming video. Fortunately, we were able to side-step this huge hurdle with the decision to utilize a Raspberry Pi and camera that is compatible with it. Not only was this the most practical decision that would allow us to focus primarily on the main PCB, but it

was also the most economical as well. The cost of manufacturing a second PCB on top of buying more chips that would've definitely been more expensive since they were higher performance parts outweighed the cheap cost of a Raspberry Pi and camera module. This process has truly demonstrated what it often means to be an engineer, searching for a solution to some kind of technical problem and trying to get it done on a certain budget.

In some of our original documents that we turned in for this project, there was also a belief that we would transmit all of our video to the user's phone using the Bluetooth wireless protocol. At the time the team thought that since it was used in wireless products like headphones, that we could easily use this protocol to also communicate with the device and transmit the backup camera video to the phone. This was our first dose of reality, when we realized that the maximum bandwidth provided by the Bluetooth protocol would only give use around one third of the bandwidth that we would need to stream a 720p video. Early on in research this was truly one of the biggest hurdles, finding a means to stream the video to the user's phone, and doing it without the internet, as most users would likely not have access to it, sitting in their car in the middle of a random parking lot. We initially did research into how drones managed to stream video to user's devices from great distances and learned a bit about radio frequency transmission. However, this would've required that the user's phone have a large and bulky receiver attached to it somehow to receive all of the data coming in, something that we found would hurt practicality. Eventually our research lead us to using 2.4 GHz wireless, but instead of connecting and transmitting our data over two devices using the internet, setting up an ad-hoc network and transmitting data directly between the backup camera and phone, a much better solution to the problem.

During the entirety of Senior Design 2, we spent all of our time implementing each feature, and then either working through every problem we found, or changing our approach and thus modifying our original design. This showed us a better understanding of the prototyping process, and as new problems arose we felt more comfortable with how to approach them. The biggest setback for us was the accelerometer, but due to our amazing time management, we were able to create a new feature for our app that used the data we were getting from the accelerometer. This project showed us not just what we were capable of, but how we could learn and build off of our skillset. The final design was not what we envisioned in Senior Design 2, but was none the less a success in our eyes, capturing the overall requirements we set out to achieve.

Appendix A: References

Bustamante, Elizabeth. “The Top 10 DC/DC Converters.” *SnapEDA Blog*, 16 Aug. 2017, www.blog.snapeda.com/2017/08/16/the-top-10-dcdc-converters.

Coates, Eric. “Switched Mode Power Supplies.” *Learn about Electronics*, 3 Sept. 2017, www.learnabout-electronics.org/PSU/psu30.php.

“Control-Mode Quick Reference Guide Step-Down Non-Isolated DC/DC.” *Texas Instruments*, 2017, www.ti.com/lit/sg/slyt710a/slyt710a.pdf.

Dahl, Oyvind Nydal. “KiCad vs Eagle - Which One Is Best? [2018 Comparison].” *Build Electronic Circuits*, 12 Jan. 2018, www.build-electronic-circuits.com/kicad-vs-eagle-2018-comparison/.

“Eagle.” 9.1.0, Autodesk, 2018.

Hauke, Brigitte. “Basic Calculation of a Buck Converter's Power Stage.” *Texas Instruments*, Aug. 2015, www.ti.com/lit/an/slva477b/slva477b.pdf.

Joseph. “Tech Article: How to Have a Very Fast Boot Time with Raspberry Pi.” *SamplerBox - Tech Article: How to Have a Very Fast Boot Time with Raspberry Pi*, 20 May 2015, www.samplerbox.org/article/fastbootrpi.

“A Price Comparison Site for Printed Circuit Boards.” *PCBShopper*, pcbshopper.com/.
“Raspberry Pi FAQs - Frequently Asked Questions.” *Raspberry Pi*, www.raspberrypi.org/help/faqs.

Vancourt, Christophe. “Choose the Best Inductor, Capacitor for DC/DC Converters in Portables.” *EETimes*, 24 May 2006, www.eetimes.com/document.asp?doc_id=1333480.

Vancourt, Christophe. “Choosing Inductors and Capacitors for DC/DC Converters .” *Texas Instruments*, Feb. 2004, www.ti.com/lit/an/slva157/slva157.pdf.

Appendix B: Citations

[1] United States, Congress, Cong., Committee on Transportation and Infrastructure. “NHTSA.” *NHTSA*, Nov. 2008. 110th Congress, 2nd session, report DOT HS 811 44 , crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/811144

[2]“ZUS Connected Car System.” *Nonda*, 2018, www.nonda.co/products/smart-rear-view-camera.

[3]“Pearl RearVision Wireless Rear-View Camera and Alert System.” *Crutchfield*, 2018, www.crutchfield.com/S-9faocQ60Cjv/p_994PEARLCM/Pearl-RearVision-Wireless-Rear-view-Camera-and-Alert-System.html.

[4]Shaw, Keith. “802.11: Wi-Fi Standards and Speeds Explained.” *Network World*, Network World, 30 Jan. 2018, www.networkworld.com/article/3238664/wi-fi/80211-wi-fi-standards-and-speeds-explained.html.

[5]“Topology Options | Bluetooth Technology Website.” *Bluetooth*, 2018, www.bluetooth.com/bluetooth-technology/topology-options.

[6]“ZigBee® Open Source Stack.” *ZigBee Open Source Stack - What Is ZBOSS - ZigBee Open Source Stack*, 2018, zboss.dsr-wireless.com/projects/zboss/wiki/What_is_ZBOSS.

[7] “Java Code Conventions” *Sun Microsystems, 1997* *http://web.archive.org/web/20090915035719/http://java.sun.com/docs/codeconv/CodeConventions.pdf*

[8] “ISO/IEC 9899:TC2 Programming Language - C” *ISO/IEC,* 2005, http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf

[9] “App Permissions Best Practices” *Android,* 2018, https://developer.android.com/training/permissions/usage-notes

[10] Hesham Alaqail and Shakeel Ahmed “Overview of Software Testing Standard ISO/IEC/IEEE 29119” *IJCSNS International Journal of Computer Science and Network Security,* VOL.18 No.2, February 2018*,* https://www.researchgate.net/publication/323759544_Overview_of_Software_Testing_Standard_ISOIECIEEE_29119

[11] “MSP430G2x53 Mixed Signal Microcontroller.” *Texas Instruments*, 2013, www.ti.com/lit/ds/slas735j/slas735j.pdf

[12] “Ferroelectric RAM.” *Wikipedia*, Wikimedia Foundation, 27 July 2018, en.wikipedia.org/wiki/Ferroelectric_RAM

[13] Thornton, Scott. “Memory Technology from Floating Gates to FRAM.” *Microcontroller Tips*, 25 Aug. 2017, www.microcontrollertips.com/memory-technology-from-floating-gates-to-fram/.

[14] “FRAM FAQs.” *Texas Instruments*, 2014, www.ti.com/lit/ml/slat151/slat151.pdf.

[15] “MSP430FR59xx Mixed-Signal Microcontrollers.” *Texas Instruments*, 2017, www.ti.com/lit/ds/symlink/msp430fr59691.pdf.

[16] “8-Bit AVR Microcontrollers ATmega328/P.” *Atmel Corporation*, 2016, ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf.

[17] Watkins, Jon. “Sleepy Pi 2.” *Spell Foundry*, 2016, spellfoundry.com/product/sleepy-pi-2/.

[18] “Sony IMX219 Sensor.” *Electronics Datasheets*, www.electronicsdatasheets.com/download/5721ed8ce34e24fd697a913a.pdf?format=pdf.

[19] jkielty. “Most Used Smartphone Screen Resolutions in 2018.” *DeviceAtlas*, 13 July 2018, deviceatlas.com/blog/most-used-smartphone-screen-resolutions.

[20] “CMOS QSXGA (5 Megapixel) Image Sensor OV5647.” *Sparkfun*, 2009, cdn.sparkfun.com/datasheets/Dev/RaspberryPi/ov5647_full.pdf.

[21] “MMA8452Q, 3-Axis, 12-Bit/8-Bit Digital Accelerometer.” *NXP Semiconductors*, 2016, www.nxp.com/docs/en/data-sheet/MMA8452Q.pdf.

[22] “Small, Low Power, 3-Axis ±3 g Accelerometer ADXL335.” *Analog*, 2010, www.analog.com/media/en/technical-documentation/data-sheets/ADXL335.pdf.

[23] “OPT3001 Ambient Light Sensor (ALS).” *Texas Instruments*, 2017, www.ti.com/lit/ds/symlink/opt3001.pdf.

[24] “MAX44009 Industry’s Lowest-Power Ambient Light Sensor with ADC.” *Maxim Integrated*, 2011, datasheets.maximintegrated.com/en/ds/MAX44009.pdf.

[25] “HC-SR05 Precision Ultrasonic Sensor.” *Behnam Robotic*, dl.behnamrobotic.com/shop/datasheet/module/module-srf05-ultrasonic.pdf

[26]“ESP32-D0WDQS.” *Mouser Electronics - Electronic Components Distributor*, 2018, www.mouser.com/ProductDetail/Espressif-Systems/ESP32-D0WDQ6?qs=chTDxNqvsykWgzfXx0gR%2BQ.

[27]“ESP8266.” *ESP8266 Overview | Espressif Systems*, 2018, www.espressif.com/en/products/hardware/esp8266ex/overview.

[28]“HC-06 Product Data Sheet.” *HC-06*, 2018, www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf.

[29]*HM-10 Bluetooth Datasheet*. 2018, fab.cba.mit.edu/classes/863.15/doc/tutorials/programming/bluetooth/bluetooth40_en.pdf.

[30] #747001, Member, and Member #34415. “Bluetooth SMD Module - RN-42 (v6.15).” *SEN-13266 - SparkFun Electronics*, 2018, www.sparkfun.com/products/12574

[31] “HC-05 DataSheet.” *DataSheet*, 2018, www.electronicaestudio.com/docs/istd016A.pdf.

[32]“Blog.” *Solar Reviews*, Solar Reviews, 2018, www.solarreviews.com/blog/pros-and-cons-of-monocrystalline-vs-polycrystalline-solar-panels.

[33] “PowerFilm Solar.” *PowerFilm Thinfilm Data Sheet*, 2018, www.jameco.com/Jameco/Products/ProdDS/227993.pdf.

[35]“Amorphous SIlicon Solar Cells.” *DataSheet*, 2018, media.digikey.com/pdf/Data%20Sheets/Sanyo%20Energy/Amorphous_Br.pdf.

[36]“Solar Panel Specification.” *DataSheet*, 2018, www.mouser.com/datasheet/2/272/solar-panel-datasheet-1291089.pdf.

[37]“bq24650 Synchrounous Switch Mode Datasheet.” *DataSheet*, 2018, www.ti.com/lit/ds/symlink/bq24650.pdf.

[38]“MicroChip MCP7383 DataSheet.” *DataSheet*, 2018, cdn.sparkfun.com/assets/learn_tutorials/6/9/5/MCP738312.pdf.

[39]“Linear Technology DataSheet.” *Linear Technology*, 2018, www.analog.com/media/en/technical-documentation/data-sheets/3652fe.pdf.

[40] *L6924D*. 2018, www.st.com/resource/en/datasheet/l6924d.pdf.

[41] “What's the Best Battery?” *Battery University*, 21 Mar. 2017, www.batteryuniversity.com/learn/archive/whats_the_best_battery

[42] Hussein, Faisal, and Brittany Eckmann. “Guide to Choosing the Best DCDC Converter for Your Application.” *Design And Reuse*, www.design-reuse.com/articles/39498/choosing-the-best-dcdc-converter.html.

[43] SOT23 Step-Down Controller.” *Texas Instruments*, Aug. 2003, www.ti.com/lit/ds/symlink/tps64200.pdf.

[44] “TPS6305x Single Inductor Buck-Boost With 1-A Switches and Adjustable Soft Start.” *Texas Instruments*, July 2013, www.ti.com/lit/ds/symlink/tps63051.pdf.

[45] “LSF0204x 4-Bits Bidirectional Multi-Voltage Level Translator for Open-Drain and PushPull Application.” *Texas Instruments*, July 2014, www.ti.com/lit/ds/symlink/lsf0204.pdf.

[46] “TXB0104 4-Bit Bidirectional Voltage-Level Translator With Automatic Direction Sensing and ±15-KV ESD Protection.” *Texas Instruments*, Apr. 2006, www.ti.com/lit/ds/symlink/txb0104.pdf.

[47] “TPS61253A 3.8-MHz, 5-V / 4-A Boost Converter in 1.2-Mm x 1.3-Mm WCSP.” *Texas Instruments*, Mar. 2017, www.ti.com/lit/ds/symlink/tps61253a.pdf.

[48] “TPS6123x High Efficiency Synchronous Step Up Converters with 5-A Switches.” *Texas Instruments*, Jan. 2014, www.ti.com/lit/ds/symlink/tps61232.pdf.

[49] Geerling, Jeff. “Power Consumption Benchmarks.” *Raspberry Pi Dramble*, www.pidramble.com/wiki/benchmarks/power-consumption.

[50] Watkins, Jon. “Sleepy Pi 2 FAQ.” *Spell Foundry*, 2018, www.spellfoundry.com/sleepy-pi/sleepy-pi-2-faq/.

[51] “Bluetooth 4.0 BLE Module Datasheet.” *JNHuaMao Technology Company*, 8 Mar. 2014,

fab.cba.mit.edu/classes/863.15/doc/tutorials/programming/bluetooth/bluetooth40_en.pdf.

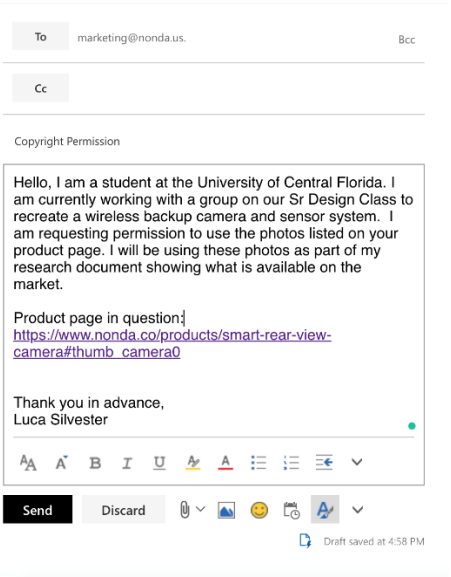
[52] “MSP430FR59xx Mixed-Signal Microcontrollers.” *Texas Instruments*, Oct. 2012, www.ti.com/lit/ds/symlink/msp430fr59691.pdf.

[53] “MMA8452Q, 3-Axis, 12-Bit/8-Bit Digital Accelerometer.” *NXP Semiconductors*, Apr. 2016, www.nxp.com/docs/en/data-sheet/MMA8452Q.pdf.

[54] “Git” *Git, 2018,* https://git-scm.com/

[55] “Waterfall vs. Agile: Which is the Right Development Methodology for Your Project?” *Segue Technologies, 2013* https://www.seguetech.com/waterfall-vs-agile-methodology

Appendix C: Permissions for Various Images



To marketing@nonda.us. Bcc

Cc

Copyright Permission

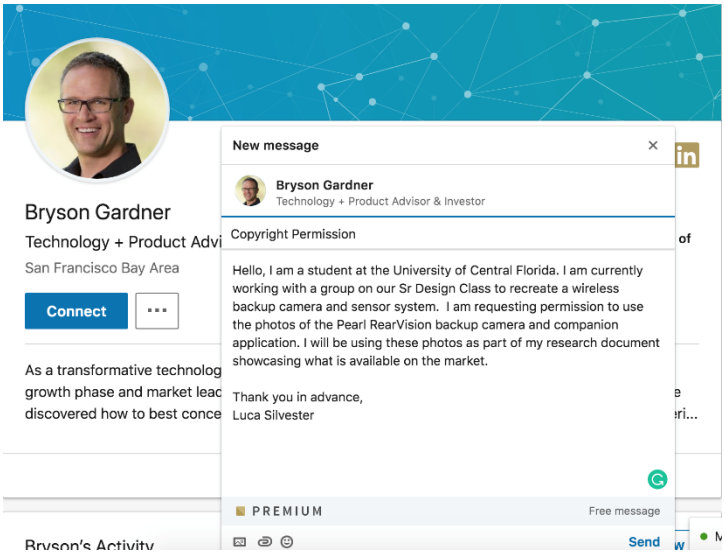
Hello, I am a student at the University of Central Florida. I am currently working with a group on our Sr Design Class to recreate a wireless backup camera and sensor system. I am requesting permission to use the photos listed on your product page. I will be using these photos as part of my research document showing what is available on the market.

Product page in question:
https://www.nonda.co/products/smart-rear-view-camera#thumb_camera0

Thank you in advance,
Luca Silvester

Send Discard

Draft saved at 4:58 PM



Bryson Gardner
Technology + Product Advi
San Francisco Bay Area
Connect ...

As a transformative technolog
growth phase and market leac
discovered how to best conce

Bryson's Activity

New message

Bryson Gardner
Technology + Product Advisor & Investor

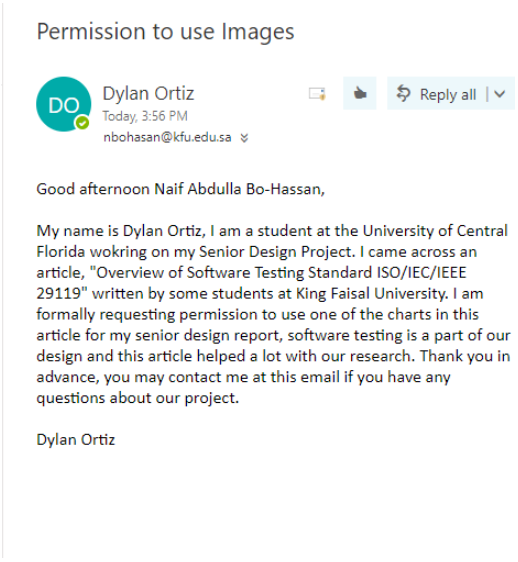
Copyright Permission

Hello, I am a student at the University of Central Florida. I am currently working with a group on our Sr Design Class to recreate a wireless backup camera and sensor system. I am requesting permission to use the photos of the Pearl RearVision backup camera and companion application. I will be using these photos as part of my research document showcasing what is available on the market.

Thank you in advance,
Luca Silvester

PREMIUM Free message

Send



Permission to use Images

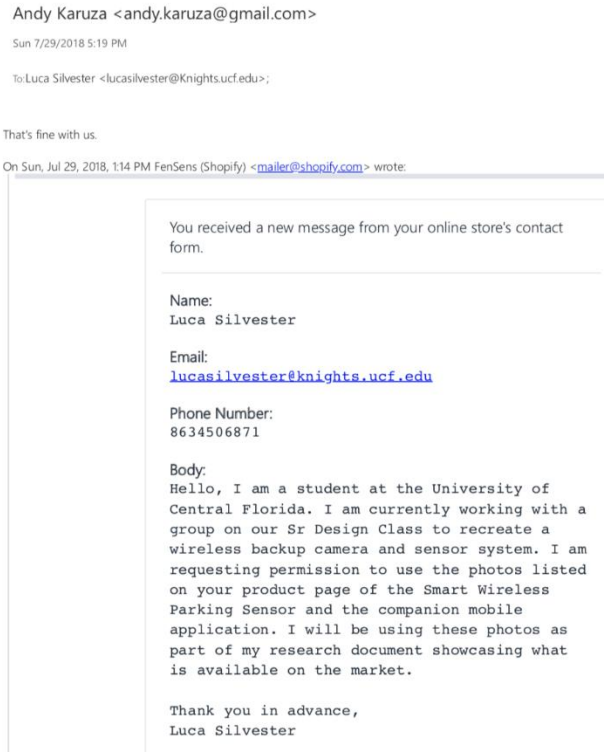
DO Dylan Ortiz
Today, 3:56 PM
nbohasan@kfu.edu.sa

Reply all

Good afternoon Naif Abdulla Bo-Hassan,

My name is Dylan Ortiz, I am a student at the University of Central Florida wokring on my Senior Design Project. I came across an article, "Overview of Software Testing Standard ISO/IEC/IEEE 29119" written by some students at King Faisal University. I am formally requesting permission to use one of the charts in this article for my senior design report, software testing is a part of our design and this article helped a lot with our research. Thank you in advance, you may contact me at this email if you have any questions about our project.

Dylan Ortiz



Andy Karuza <andy.karuza@gmail.com>
Sun 7/29/2018 5:19 PM
To:Luca Silvester <lucasilvester@Knights.ucf.edu>;

That's fine with us.

On Sun, Jul 29, 2018, 1:14 PM FenSens (Shopify) <mailer@shopify.com> wrote:

You received a new message from your online store's contact form.

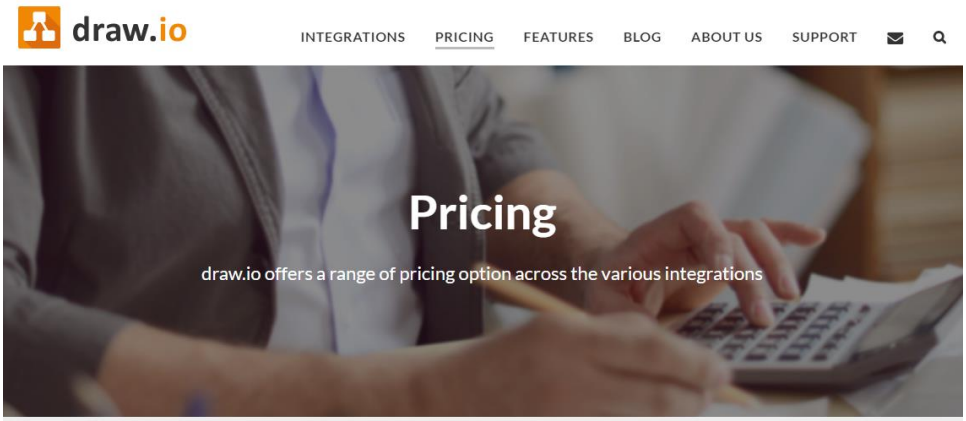
Name:
Luca Silvester

Email:
lucasilvester@knights.ucf.edu

Phone Number:
8634506871

Body:
Hello, I am a student at the University of Central Florida. I am currently working with a group on our Sr Design Class to recreate a wireless backup camera and sensor system. I am requesting permission to use the photos listed on your product page of the Smart Wireless Parking Sensor and the companion mobile application. I will be using these photos as part of my research document showcasing what is available on the market.

Thank you in advance,
Luca Silvester



draw.io pricing

draw.io online is a free-to-license web application for everyone. It is completely free to use for any purpose, there is no premium pay-for functionality, watermarking, or other limitations. You own the content you produce with draw.io and may use it for any purpose, including commercially. We don't sell your personal information or data. We don't store your data. You own your data and the application is open source.

The exceptions to draw.io being free are listed below, these are how draw.io is funded.