120 Page Senior Design 2 Document
Fall 2017

BRAIN HELMET

Group 4

Nada Algharabawi	Computer Engineering
Stephan Morales	Computer Engineering
Ryan Mortera	Electrical Engineering
Jordan Yamson	Electrical Engineering

Table of Contents

List of Tables

1.0 Executive Summary

In today’s accelerated technological progression era, there are few ideas that have not already been dreamt up and shortly after brought forth unto the world. However, the beauty of electronics and programming lies in the ability to take past research or inventions and adapt, improve, or simplify them for specific personal use. Additionally, being able to replicate or make these changes while additionally reducing production cost is a common goal found in many work places. As a combination of future computer and electrical engineers, our team considered several different already existing technologies and chose to embrace this “improve on past design for a cheaper build” trait that would be constantly used in our future careers.

Before we hit the gas on our Senior Design Project, our team first desired to embrace Bluetooth technology and implement it into an important personal device that is used on a daily basis. Eventually we focused our attention to improving the daily ride for a motorcycle enthusiast and decided to create a Bluetooth Recreational And Integrated Navigation helmet; the BRAIN Helmet. Motorcycle helmets are typically designed without any electronic aspect but, with improving portable and rechargeable battery technologies, smart helmets are slowly on the motorcycle market’s horizon. Our team’s main goal for the BRAIN Helmet is to allow motorcyclists to appropriately utilize their cell phones without distracting or lowering the situational awareness of the rider. The helmet’s electronic group will be mounted on the back of the helmet and attached to the helmet’s integrated speakers, microphone, and heads-up display. The BRAIN Helmet will allow users to visually see navigation notifications on the minimal display while additionally being directed by the navigational audio being transmitted from their android cellular device. Additionally, users will be able to take hand free phone calls or listen to music. Overall, BRAIN Helmet users will be able to get to their destinations safely without having to pull out their phones and address incoming phone calls that may change their routes.

This document entails the research and design choices that led to the BRAIN Helmet’s construction. To begin, specifications and requirement constraints are described. Following this, the research and component selection will be detailed. Lastly, the design process for hardware and software will be explained along with their corresponding testing procedures for the BRAIN Helmet stock prototype.

2.0 Project Description

The BRAIN helmet will be a modern helmet integrated with current technologies that will utilize a Bluetooth connection to an android device. This section serves as an introduction into the BRAIN Helmet; depicting its purpose and developmental guidelines. Here you will see detailed goals, objectives, and requirement specifications laid forth by our senior design group for the BRAIN helmet.

2.1 Project Motivation

Due to the technological advantages at our disposal, traveling via GPS navigation has become a standard for nearly every form of transportation. Driving, walking, and biking to a new location has all been simplified by GPS Navigation via a cellphone. While there are many to benefit from these advancements in GPS tracking and traffic updates, motorcyclists remain limited in their utilization of mobile navigation. Our desire is to develop a Bluetooth helmet that will communicate with a motorcyclist’s cellphone navigation app to allow motorcyclists to safely travel to their locations without having to pull out their mobile devices and look down at their screens. This will not only keep the motorcyclist safe on the road but will additionally avoid causing the rider to have to pull over to check for directions. In addition to this, our helmet will address motorcyclists not safely being able to wirelessly take phone calls. In moments of emergencies or plan changes, a BRAIN Helmet user will ideally be able to save time by picking up phone calls to receive updates on whether their destination needs to change. Lastly, the recreational aspect of the BRAIN helmet entails being able to wirelessly listen to music or other forms of audio entertainment such as podcasts. Many motorcyclists create safety problems for themselves by using headphones that limit their ability to hear their surroundings. The BRAIN Helmet having external speakers with easily accessible volume control bypasses this safety concern and will allow users to further enjoy their motorcycle ride. Essentially, the overall motivation for the BRAIN Helmet is to allow riders to get to their destination safely and as quick as possible by utilizing technology, and to enjoy their ride with audio provided pleasure.

2.2 Goals & Objectives

To fully meet a motorcyclist’s needs in the above described problems, our team will implement several important factors in designing our easy to use helmet. First and foremost, few motorcyclists would not have any interest in a new motorcycle helmet that would break their bank so we are utilizing a low-cost approach by implementing only necessary and best priced components in our electronic design. Accordingly, the helmet will focus on low power consumption to allow users to have a typical day of riding without having to charge their helmet. While this new technology will differ from a casual motorcycle helmet, our design will focus on keeping the helmet’s integrity in terms of safety, comfort, and weight.

This Bluetooth Recreational and Integrated Navigation (B.R.A.I.N) helmet will function as an interface between the rider and their android phone. After Bluetooth pairing with the device, a user will be able to start a navigation route, listen to music, or even take a phone call while riding. During navigation, Riders will be able to view a small HUD displaying their current speed and upcoming turns. The HUD is not limited to just these uses and is subject to further utilization and features as the BRAIN Helmet is designed. Additionally, the Bluetooth module allows the navigational audio from the android device to relay the upcoming turns and further instructions in real time. With the user's comfort in mind, volume control will be easily accessible on the helmet's exterior via buttons along with a possible mute button.

We are intending to integrate the following objectives into the BRAIN Helmet system:

- The helmet to be completely wireless between the user's android phone and the helmet
- The phone navigation app shall send updates to the Bluetooth module to update the HUD
- The phone navigation app shall be able to update itself using google maps
- The phone shall be able to connect to the Bluetooth device
- The user shall be able to answer and decline calls as they appear
- The Bluetooth module shall send signals to the MCU to update the HUD
- The MCU shall control the HUD display
- The speakers shall produce audible enough sound over the sound of a motorcycle
- The battery of the system needs to be rechargeable
- The battery of the system needs to be lightweight
- The system shall implement a hand free environment
- The system shall incorporate low power components not capable of dissipating uncomfortable, noticeable heat during use

2.3 Requirements Specifications

The following requirement specification guidelines were decided upon before the development of the BRAIN helmet.

- The Motorcycle Helmet that we will be doing the modifications to will be a standard street bike helmet. The rough size of the helmet should be as follows:
    - ~10.6 x 13.8 x 10.4 inches
- There will be two speakers situated in the area of both ears on each side of the helmet. The wiring will be integrated into the helmet from the electronic group and will be negligible in size; not bothering the rider.
    - ~0.25" thick speaker, at least 40 mm diameter, impedance of at least 8 ohms
- The helmet will include a microphone to be wired through the helmet similar to the speakers.

- The helmet will include a Heads-Up Display (HUD) on the visor of the helmet. Movability has not been decided. The pixel resolution of the helmet shall be as follows:
    - ~ 128x64 resolutions
- The helmet will have a rechargeable battery capable of at least 6 hours life time and with the following specifications:
    - ~2000 mAh battery capacity
    - No larger than 3” x 3”
- The helmet will pair with a smartphone via Bluetooth with a relatively short connection setup time.
    - ~approximately 1-2 minutes
- The helmet will include 5 buttons for answering phone calls, track control, and toggling the display. In addition to the buttons there will also be a potentiometer dedicated to changing the volume of the audio.
- The helmet will connect to a smartphone and be able to stream GPS navigation signals to the HUD.
    - ~Using the integration of the Google maps API; the microprocessor shall toggle LEDs in patterns or symbols distinguishable by the user.
- The helmet will have a Microcontroller with a I/O at around 3.3V-5V.
- The BRAIN electronic group shall be a low power consumption device:
    - To enable 5 hours or more of life time the battery will have to be able to supply the major components drawing no more than 400 mA per hour.
- The helmet will have the PCB on the back of the helmet connecting all the parts.
    - ~The PCB should have dimensions of about 4” L * 4” W

2.4 Quality of House Analysis

The House of Quality (HOQ) depicted below in Figure 1 will be the foundation of our project and is subject to small adjustments throughout the B.R.A.I.N Helmet development. This HOQ is what the BRAIN Helmet team constructed in order to appropriately verify and keep track of correlations between marketing requirements and engineering requirements. The engineering requirements are shown on the top section of the HOQ and have corresponding values listed at the bottom of the HOQ. Alternatively, their compared marketing requirements are visible on the left-hand side of the house of quality. It is worth noting that the engineering requirement values on the bottom of Figure 1 are not set in stone for the BRAIN Helmet team. These numerical guidelines will be further detailed in the research and design section of this report.

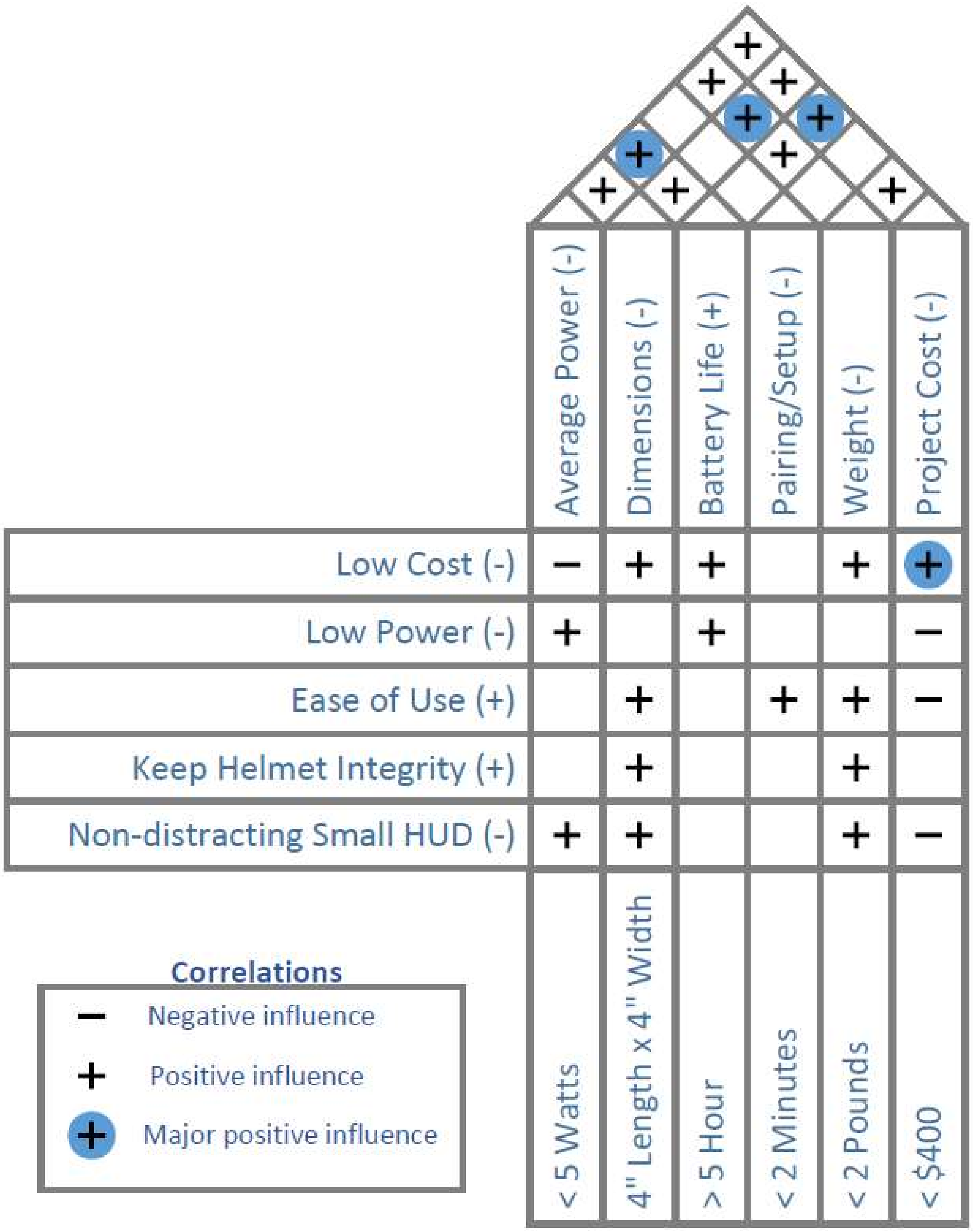


Figure 1 - House of Quality

3.0 Research Related to Product Definition

The BRAIN Helmet will be integrated with the latest technologies of its time; as the budget fits of course. The following will be a detailed explanation of previous projects similar to our idea, modern technologies such as wireless and Bluetooth. Battery technologies are also explained in this section. When it comes to modern technologies, the BRAIN helmet will be incorporated with at least 2-year-old devices. The BRAIN helmet will need all these types of technologies for the wireless system that is being integrated onto it, as well as the OLED display being attached to the screen.

3.1 Existing Similar Projects and Products

Once the general problem was addressed for the motorcycle community not having a commonly used form of smart navigation; we began researching just what efforts were made to solve the same problems the BRAIN Helmet is hoping to solve. As we researched for SMART Bluetooth helmets, our main approach involved looking around the market for similar products being sold or being developed. Many products that we considered similar to the idea of the BRAIN Helmet are shown in this section.

3.1.1 SKULLY

The SKULLY brand is quite arguably one of the most famous smart helmets on the market; until recently when they had to claim bankruptcy and pull their products off the market. The SKULLY AR – 1 was going to be one the world’s smartest motorcycle helmet. The helmet was going to include a Heads-Up Display, that would always be in focus. The helmet was to be DOT certified, fog and scratch resistant, and have a quick release chin strap. The sizes for this helmet were to vary from small to extra-large. There was to be integrated an ultra-wide-angle rear view camera, GPS navigation all powered by a powerful microprocessor. The helmet would connect wireless via WLAN and Bluetooth. The helmet would also have integrated audio and hands-free calling. However, all of these features led to an understandably large price cost. The base price of this helmet was starting at around $1500. Unfortunately, this company just filed for bankruptcy this year (2017). Our goal for the BRAIN Helmet is to meet the same marketing goals as the SKULLY helmet; but at an affordable cost. Therefore, we are not going to have nearly as stylish of a build, nor as many features as the SKULLY Helmet. The BRAIN Helmet ideally would be able to reach out to the motorcycle community as an affordable, reasonable purchase unlike the SKULLY was.

Figure 2 depicts a recent model of the SKULLY Helmet that was on the market before SKULLY went bankrupt. Several of these helmets were available to pre-order customers and were even tested and reviewed through blogs and YouTube videos. While the helmet was a great benefit to some riders, it was commonly noted that such a steep price drove the necessity and desire for the SKULLY helmet lower. With SKULLY being such a hyped helmet that failed; our BRAIN Helmet team took heavy consideration and note in the set back of building a beautiful helmet versus an effective and cheaper alternative.

Figure 2 - SKULLY Helmet

Permission to use from SKULLY

3.1.2 Reevu

The Reevu Helmet is a motorcycle helmet with a reflective polycarbonate plate, which would double as an internal safety shell for the rider. The outer shell casing is made from a mix of carbon composites for lightweight, still with full impact resistance. The most important key factor about the Reevu Helmet is that there is a rear-view mirror allowing for the motorcyclist to see directly behind them just like a rear-view mirror within a motor vehicle. Without having to move their head, a motorcyclist equipped with the Reevu can see blind spots and directly behind of rear traffic, all within the confines of a spectacular safe helmet. The Reevu mirror mechanism can be removed or adjusted for different users just in case they were too think the mirror was a distraction as they were riding. This motorcycle helmet has even astounded industry experts in motorsports without its rear-view technology, and that is due to its safety factor. The Reevu helmet is depicted along with a view of its features in Figure 3; as seen on Reevu's website.

This helmet has plenty of features such as comfort, ventilation, and low noise. The Reevu helmet has liners and cheek pads within the helmet that can be removed for washing. The ventilation for the helmet is designed using fluid dynamic properties of the venture vent. This ventilation keeps the user of the Reevu at low temperatures. Low noise for motorcyclists is a requirement due to the loudness of most motorcycle mufflers. The Reevu is design with shell liners made out of dual density EPS, that has high coefficient bump absorption. The Reevu helmet when compared to our B.R.A.I.N. Helmet lacks in its ability for hands free device communication, but it does have a technology of being able to see the rear of the motorcyclist.

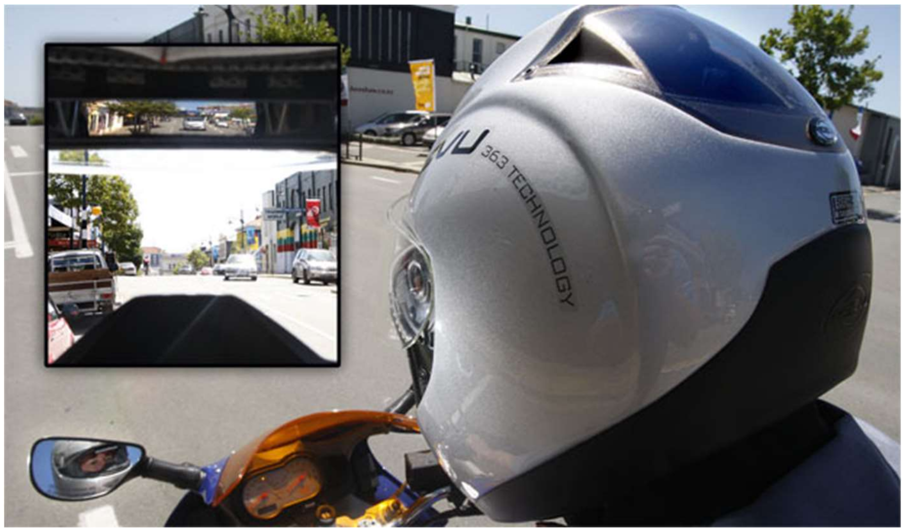


Figure 3 - Reevu Functioning Diagram

Permission to use from Reevu

3.1.3 LiveMap

The LiveMap motorcycle helmet is a new technologic helmet that has a built-in augmented reality interface, GPS, Microphone and Speakers, and action camera. This helmet is the ultimate technological helmet built and designed for everything a motorcyclist needs and can be seen in figure 4. The only drawback of this helmet is the price of it. The retail price for this helmet is going for $1500, whereas we spent around $300 for many of the same functions. The main difference when comparing our B.R.A.I.N. helmet is the display within the helmet. The LiveMap uses an augmented reality interface with a built in light sensor to portray the upcoming directions from the microcontroller. This augmented reality shows the GPS route, speed, and customizable alerts and calls. The augmented reality screen works in any time of day from extremely sunny, to the darks of night. This is a great feature not only for its stylish look but its effectiveness in portraying data to the rider without distractions. The built-in camera can record in 1080P and is built within the helmet. This built-in camera is capable of recording whatever happens while riding and can be saved to a removable MicroSD flash memory device. Another plus about the LiveMap helmet is the modular construction that it was built with, meaning you can customize exactly what you would want in the helmet. This helmet is one of the very inspirations for our project, but we wanted to make a cheaper more affordable version, which will be the B.R.A.I.N. Helmet.

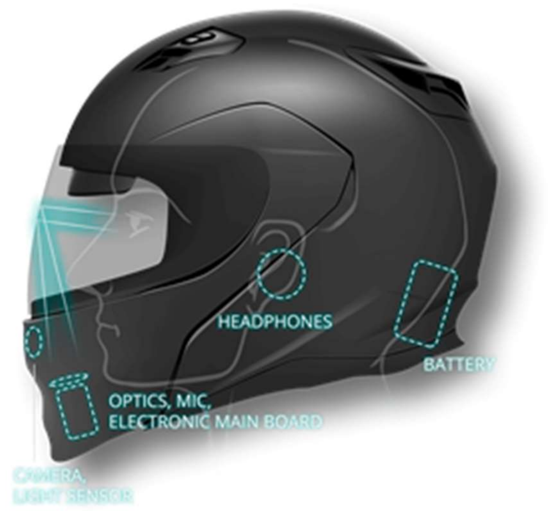


Figure 4 - LiveMap Block Diagram Helmet

Permission to use from LiveMap

3.1.4 Nuvis

The Nuviz is a helmet attachment that has a Heads-Up display designed specifically for motorcyclists. The attachable NUVIS is visible in Figure 5. It is an attachable device that can go onto most modern motorcycle helmets. It is a device that is essentially, a microphone, speakers, camera, and GPS, all with a HUD on the bottom right of your helmet glass. The Nuviz also comes with a remote control that can be attached to the throttle of your motorcycle to change the functions of the helmet. Such as, when you're receiving a phone call, you can accept the call through the other device. The HUD displays navigation as calls, as well as the music you are streaming from your phone. The device also take HD videos in a still environment, and can be updated through software upgrades in the future. The Nuviz is powered through a Qualcomm quad-core processor, that runs all the GPS, accelerometer, gyroscope, and altimeter sensors. The Nuvis has two wireless technologies, both Bluetooth and WLAN. This device has a replaceable and rechargeable 3250mAh Li-ion battery, that should power the device for up to 8 hours. The Nuviz is marketed around $700. Again, while this is a great attachment capable of being attached to any motorcycle helmet, the price range makes it a rather unsightly purchase for many motorcyclists on the market for a Bluetooth navigational smart motorcycle helmet.

Figure 5 - Nuvis Device

Permission to use from Nuvis

3.2 Relevant Technologies

The design of the BRAIN helmet includes many different modern devices within the system. Implemented into our design, modern technology is used definitively throughout. When researching which parts to use on the design, many different technologies were analyzed to see which one would fit best for the system. The following system is a list of relevant technologies that were compared and decided upon to be implemented into the BRAIN Helmet.

3.2.1 Wireless Technologies

Wireless technology provides us with an opportunity for the user wearing the SMART helmet to comfortably use the system without the use of a wire connection. The Bluetooth technology is available on most cellular phones and we are utilizing it to communicate the device to the helmet.

3.2.1.1 Why use wireless?

When it comes to our project of the B.R.A.I.N. helmet, we had to implement a system where the user of a motorcycle would be able to use their mobile device without distracting the driver. The best way possible was to make the system from the user's cell phone to the motorcyclist's helmet wireless. This would make the whole system literally able to pick and use without any assembly required from the user. When researching wireless technologies, there were many different options out there, including but not limited to: Wi-Fi, Zigbee, and Bluetooth. The following will be an explanation of each type of communication.

3.2.1.2 Wifi Wireless Network

Wi-fi is a technology for wireless local area networking. Wi-fi commonly uses a 2.4 GHz Ultra High Frequency and a 5 GHz radio bands. Many different products use this technology such as personal computers, video game consoles, smartphones, digital cameras, and even smart televisions. Almost everything is connected using

Wi Fi. There is a certain type of Wi fi that would be beneficial in our project would be ad-hoc mode. This means devices talk directly between each other without the need to first talk to an access point where all devices are connected. Wi.fi used the IEEE 802.11 standard which is a standard for media access control and physical layer specifications, for certain computer communication in the 2.4, 3.6, 5, and 60 GHz frequency bands. Wi fi can transfer data to up to around 250 mb/s. The range for a wi-fi connection can be up to about 100m. This depends mainly on the latest wi-fi version and wi-fi protocol. Also, the range for the connection of Wi-Fi can be extended using extra antennas. Multiple connections can be made under a Wi-fi network, depending on the access point. Connection complexity for a Wi-Fi network is usually very complex, since you must have to configure the wi-fi network and network security pass code. With a wi-fi network, the security of the communication is strong, with programs such as Wireless Equivalent privacy and Wi-Fi protected Access. Power usage of Wi-Fi is unfortunately very high due to all the security and longer distances of communication.

3.2.1.3 Zigbee Wireless Module

There is another wireless communication device that exists, and that is Zigbee. Zigbee is an IEEE 802.15.4 specification used for a suite of high level communication protocols to create Personal Area networks. The Zigbee module takes advantage of low power digital radios in order to connect different devices wirelessly. The Zigbee is a low-power, low data rate and low range wireless ad hoc network. Zigbee is usually implemented within light fixtures, light switches, traffic management systems, and any other industrial or consumer device that requires short range low data transfer rate. Zigbee usually worked between 10 meters. Zigbee is usually implemented low data rate applications that require long battery life. Zigbee modules could be used for the new and improving Internet of Things, due to its low power consumption, the Zigbee module can be used for many wireless devices that would require little power and slow data rates. Zigbee operates in the unlicensed 2.4 to 2.4835 GHz radio bands. There are 3 different types of Zigbee devices are they are as follows:

- Zigbee Coordinator: The coordinator forms the root of the network tree and can bridge to other networks. There is only one Coordinator in each network since it is the device that started the network. This coordinator stores information about the network, including acting as a Trust Center for the 128 security keys.

- Zigbee Router: The purpose of this device is to be exactly what its name describes a router for certain communications passing through the device. This helps when the Zigbee is connected to a bigger network and would help sort out the chaos of the network by being able to micro managing the network within itself.

- Zigbee End Device: This device contains just enough functionality to talk to the parent node, but it cannot relay data from other devices. This means that this part of the device isn't powered on most of the time, which allows the Zigbee to save on battery life. The ZED requires little memory which makes manufacturing this device the cheapest.

The security of the Zigbee is based on a 128-bit key system. The trust between devices must be assumed during the installation of the key. These keys are the backbone of the security system for the Zigbee, meaning they are to be communicated through a secure channel, or someone could be using it to steal information.

3.2.1.4 Bluetooth

Bluetooth is one of a wireless technology standard for exchanging data over short distances. It used short wavelength UHF radio waves in the ISM band at 2.4 GHz. It was first invented by the telecom vendor Ericsson in 1994. It is managed by the Bluetooth Special Interest Group (SIG). The IEEE standardized Bluetooth as IEEE 802.15.1. The SIG oversees development of specification, management of qualification program, and protects the trademarks. Bluetooth uses a radio technology called frequency-hopping spread spectrum. Bluetooth divides transmitted data into packets, and transmits each packet of one of the 79 designated Bluetooth channels, each channel having a bandwidth of 1MHz. With the Bluetooth low energy, 2MHz spacing is utilized, which would in turn accommodate 40 channels. Ever since Bluetooth 2.0 +EDR which led to 4-DQPSK and 8DPSK modulation, the Gaussian frequency-shift keying was the only modulation that existed. Bluetooth is a packet-based protocol with a master-structure. One master may communicate with up to 7 slaves in a piconet. All devices share the master's clock. A piconet is an ad hoc computer network using Bluetooth technology. Sometimes, the slaves can become masters once they have connected to the device. These slaves are then lined up for commands in a round robin fashion.

With the progress of technology over the years, Bluetooth has evolved as well. When looking particularly at the different version of Bluetooth we can see how it has evolved and progressed over time. There are different types of Bluetooth that have been and these versions are as following:

- Bluetooth Versions 2.0, 2.1 with EDR
- Bluetooth Versions 3.0
- Bluetooth Versions 4.0,4.1, and 4.2
- Bluetooth Version 5

3.2.1.4.1 Bluetooth Versions 2.0,2.1 with EDR

The key features when analyzing Bluetooth version 2.0 and 2.1 +EDR is that it can increase battery life within the device it is connected to. It operates at a 2.4 GHz frequency. The basic rate that Version 2.0 without EDR supports is a bit rate of 1MBps whereas with EDR in the 2.1 version, the bit rate is double to 2 MBps. In case of pairing scenarios, there is an “eavesdropper protection” that generates a six-digit passkey that is stronger than a 16-digit alphanumeric character random PIN code. In versions 2.1, there is a system known as “man-in-the-middle” protection that would get rid of the possibility for a third party to intercept the information you are transferring. Also in version 2.1, near field communication (NFC) was introduced, this allowed communication between devices when physically touched together or brought into proximity of each other.

3.2.1.4.2 Bluetooth Version 3.0

Bluetooth Version 3.0 also has something called High Speed enabled with it. This version uses a data-substitution method which increased the throughput via the use of a secondary radio that is already present in consumer devices. Bluetooth Version 3.0, with its higher speed, allows for the transfer of music libraries between devices, downloading photos in mass, and sending video files from one another. This version is much faster than its predecessor. Key features of Bluetooth 3.0 include:

- Reduced power consumption by using the second radio only when needed, which extends battery life inside the device
- Bluetooth 3.0 makes power control faster which in turn limits drop outs, so no disconnections happen between devices
- Unicast Connectionless Data lowers latency rates, by sending small amounts of data more quickly

3.2.1.4.3 Bluetooth Version 4.0, 4.1 and 4.2

Bluetooth 4.0 are also known as Low Energy Bluetooth, which was designed with the Internet of Things in mind. The functionality of the Bluetooth 4.0 is for devices that run for long periods of time, that must only use small amounts of energy. Bluetooth 4.0 is used in many systems found in home appliances and security systems, as well as fitness monitors and proximity sensors. The key features of Bluetooth 4.0 include:

- Industry-Standard wireless protocol that allows for interoperability across platforms
- Ultra-low peak, average and idle mode power consumption
- Standardized application development architecture

3.2.1.4.4 Bluetooth Version 5.0

The newest addition to the Bluetooth family is Bluetooth 5.0, it has 2 times the speed of 4.0, 4 times the range, and 8 times the data transfer. It is still new, as in it has been released only this year (2017), but this will push forward the thought of everything being connected, also known as the Internet of Things.

3.2.1.5 Comparison of Wireless Technologies

Table 1 - Wireless Technologies

Specification	**Wi-Fi**	**Zigbee**	**Bluetooth (3.0)**
Frequency	2.4 GHz / 5 GHz ISM Bands	2.4 GHz ISM Bands	2.4 GHz ISM Bands
Maximum Range	100 meters	10 meters	30 meters
Maximum Data Rate	5-600 Mbps	250 kbps	3 Mbps
Power Consumption	50-110 mA	5 mA	30 mA

3.2.1.6 Bluetooth (Selection)

The reason why our group chose to go with the Bluetooth 3.0 technology is because the range of the module is far enough for a motorcyclist to use the helmet even from afar. The data rate is large enough for us to send signals for calls and music at the same time (the module we picked is dual channel) and the power consumption of the Bluetooth technology is perfect for our situation. We would be able to implement a system that doesn't use too much power over time, but still can send the signals we need such as the signal for the HUD we are placing on the front glass of the helmet.

3.2.2 Battery Technologies

In our current day and age, the available battery designs are constantly changing. This is especially driven by the desire for rechargeable and mobile devices such as our cellular phones. The figure below depicts the two major battery classes.

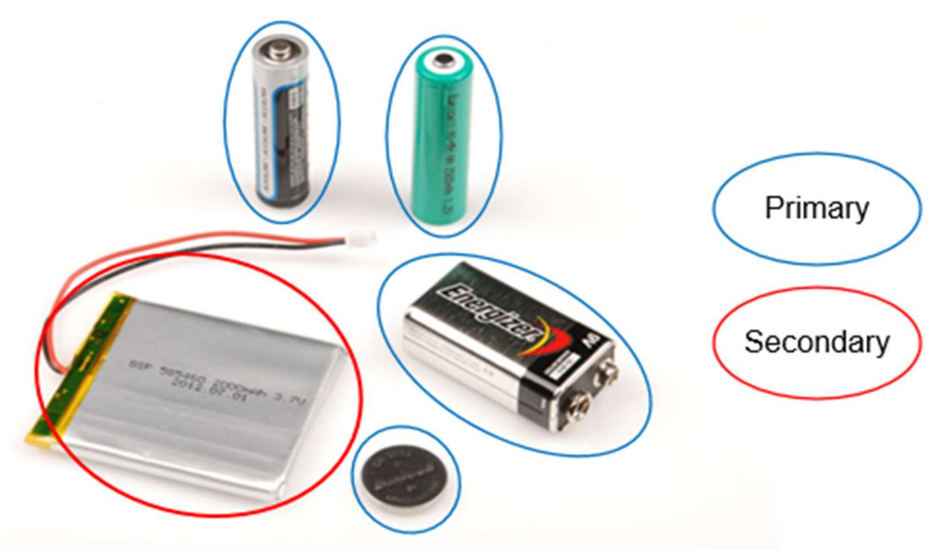


Figure 6 - Variety of Battery Types Considered

Permission from an open source

3.2.2.1 Battery Constraints

There are several factors already touched based on that we must consider in order to truly make the BRAIN Helmet a user friendly and daily instrument used by motorcyclists. Two such concerns are the size and weight of the battery. As you can see in the above figure, the varying battery types allow various shapes, weights, and versatility factors to be considered for the BRAIN Helmet. A heavy battery would cause the motorcyclist to lose comfort in his helmet and possibly strain their neck muscles over long rides. A large battery would not only likely lead to being heavy but would also obstruct the overall size of the BRAIN Helmet's group of electronics. As a result, the aerodynamics of the BRAIN Helmet will be altered. Because the BRAIN Helmet's electronic group and therefore battery is located on the helmet's rear, there will not be a noticeable change in air drag from the helmet upon forward view. However, if a motorcyclist turns his head to the left or right side, he will feel greater air drag and therefore sound if there is a larger battery. Therefore, in deciding a battery to power the BRAIN helmet, our goal is to obtain a small and light enough battery that will power the electronic components for over multiple hours.

3.2.2.2 Battery Types

For an application used daily such as the BRAIN Helmet, it is critically vital that the battery is a secondary, or commonly known as a rechargeable battery. Because of that, we avoided primary disposable batteries, such as replaceable 9V, Alkaline, cell batteries, etc. Not only do they prove to be expensive to use daily, but they would not last long enough for a daily ride as they don't have a high enough battery capacity.

3.2.2.3 Battery Capacity

Battery capacity ties in hand with the Energy Density of batteries. Battery Capacity is the measure of electric charge that can be delivered at a specific voltage, and is usually rated in milliamp hours (mAh). With that said, Primary batteries have a lower energy density than most secondary batteries. For example, disposable alkaline batteries have less energy per volume than a Lithium secondary battery.

Overall comparisons of the two battery types is shown in the table below.

Table 2 - Battery Type Comparisons

	Primary Batteries	**Secondary Batteries**
Weight	🟥⬆	🟩⬇
Energy Density	🟥⬇	🟩⬆
Battery Capacity	🟥⬇	🟩⬆
Price	🟩⬇	🟥⬆

Legend	Green = Beneficial
	Red = Disadvantageous
		Up arrow = Higher value
		Down arrow = Lower value

3.2.2.4 Charging a Lithium-Ion Polymer Battery

After reaching the conclusion that there will be a secondary battery in the BRAIN Helmet, the next issue is choosing a battery charger system that will be capable of recharging a secondary battery; specifically, a lithium-ion or lithium-ion polymer battery. Many lithium-ion/li-po battery chargers exist on the market. An important note in deciding the charger for the BRAIN Helmet involves choosing a input connector common to users, and being able to limit the amount of current drawn to charge the lithium-ion/li-po battery. In the case of the BRAIN Helmet, we have decided a micro-usb external connector is appropriate for android users. The drawn current will be discussed in further power supply design and charger selection.

3.2.2.5 Regulating Battery Voltage

The BRAIN Helmet is planning to utilize a rechargeable lithium-ion/li-po battery. , Lithium ion polymer batteries are created to quickly drop from their peak voltage to their nominal voltage, and as their usage continues, drops even further to their depleted cell minimum. This is depicted in the below figure.

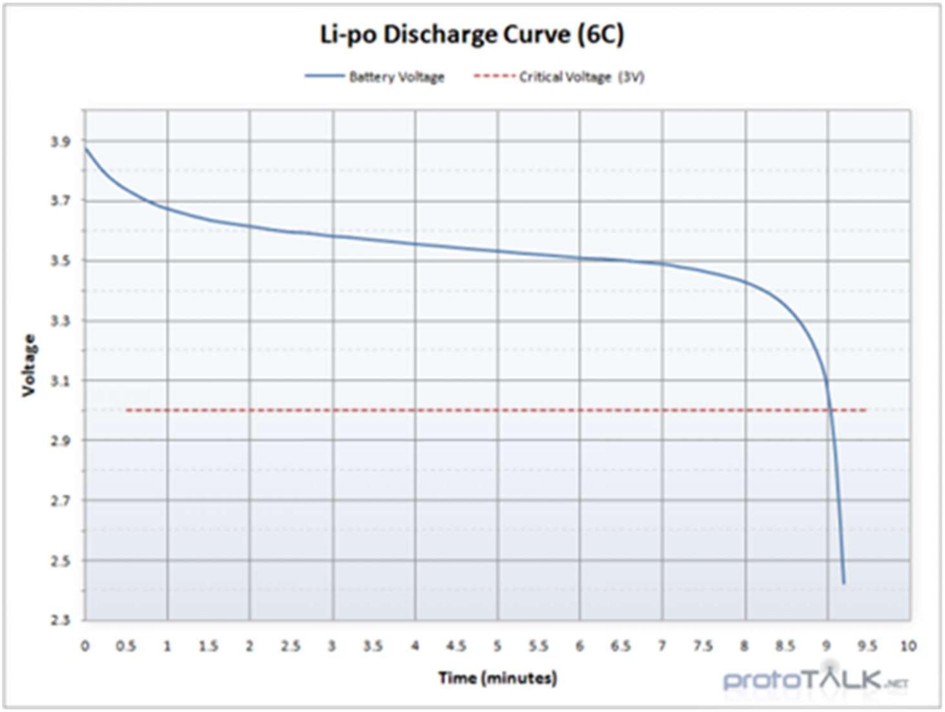


Figure 7 - Sparkfun's Lithium Polymer Battery Discharge Curve

Permission from an open source

This attribute is taken into consideration for the BRAIN Helmet's stock build. In order to keep a steady delivery of power to the BRAIN Helmet's electronic group, a switching voltage regulator will be designed and implemented with the help of Texas Instrument's Webench design tool.

3.2.3 Audio Technologies

Microphones and speakers are needed to allow the user to hear the music and the other person on the other end of the phone connection to hear. We will have a microphone that can record the voice from the user and transmit to the host device while at the same time sending signals to the speakers to create sounds that the user can hear.

3.2.3.1 Microphone

The BRAIN Helmet will require an integrated microphone in order to allow the user to communicate over Bluetooth for their phone calls or possibly voice recognition software on individual user's android devices. While there are multiple microphones available to use, a simple microphone with a preamplifier is required to amplify the voice of the user.

3.2.3.2 Speakers

To fully utilize the Bluetooth experience for the BRAIN Helmet user, two speakers are also to be integrated with the microphone into the helmet. The speakers will require sound amplification regardless of the speakers used. This will be detailed further into the document.

3.2.4 Possible Features for the BRAIN Helmet

When first coming up with the design for the BRAIN helmet, the system could have many other features added with it. The team who designed the project brainstormed a lot about this idea for the BRAIN helmet and we had lots of features in mind, but some just didn’t cut it. The ideas that are going to follow will be a list and short description of some of these features that could be implemented or would be implemented into the BRAIN helmet if time permits. Some of the ideas for the helmet also could’ve been out of the budget of our team so we didn’t want to go that route.

The first feature that could be installed would be a camera mounted on top or the side of the helmet with our system. Cameras on motorcycle helmets wouldn’t be a new innovation in the market but it would definitely come in handy due to motorcyclists usually riding dangerously and lots of road rage on the roads could lead to evidence in court. A camera that would be able to capture video at 1080P and would be able to show videos of the riders first person perspective of the ride they took on their motorcycle. The camera as a feature would be interesting to integrate into the system because we would have the signals into the companion app. This would mean we would have to implement either a higher transfer rate of bluetooth or change to wi-fi as our wireless technology. We could implement the feature of being able to transfer the videos directly from the camera onto the companion app or stream the view from the camera into the companion app. This feature was not chosen to be into the system due to the cost of this feature. The cost of this feature would add on around $300 worth of hardware. The cost would be due to the expense of a small form factor camera, that would be able to record at a decent quality. Unfortunately, due to our project being self-funded, this feature couldn't be implemented.

A rear-view camera and stream to the HUD that would be implemented into the system was also another feature that could be added. A rear-view camera would be able to stream a view of how a rear-view mirror would work in a car. This would allow a motorcyclist to see more view behind them and would allow the rider to ride more safely. This would mean safer riders on the road, and that is good for everyone. This feature wouldn’t be as difficult to implement as the forward camera previously discussed, but the reason why it isn’t going to be implemented is due to the cost again. The cost of these parts are expensive when it comes to a self-funded project.

Another feature of the BRAIN helmet that could be introduced is the use of transducer speakers. These speakers push their sound waves into a surface and that surface depending on its medium, plastic and wood work best, vibrates and amplifies the sound generated from the speaker. We could use this for a better-quality sound within the helmet due to outside noise from a highway or bust road while riding a motorcycle could be a burden on sound. This type of speaker could be implemented into our system later within the project. This would allow for better sound quality within the helmet. The reason for not being in the initial design is due to time constraints.

One feature that could be implemented is a HUD that is fully transparent and projected onto the visor of the motorcycle itself. When dealing with this idea, we had to first research the technology that would be able to project an image onto the glass, when comparing it to an OLED display, the cost was dramatically different. This idea of an augmented reality HUD on the glass was our original idea for the HUD, but then realised it is out of the scope of the project we wanted to design so we dropped the idea. The reason was due to the parts being far too expensive, and the implementation would not have worked with the time we had on this project.

With the microphone being implemented into the system, and the companion app, we could implement voice commands within the system if time permits. The hardest part of this feature would be the reading of certain voice commands within the system and then transferring those signals to the companion app and allowing it to do a function. The programming side would have more work and time would be an issue for the system as a whole. This feature was not a real importance to the system due to a motorcycle rider might not be able to adequately use these voice commands due to the high amount of noise the surrounding area would give off.

The last feature that could be implemented into the system at a later time would be removable storage for the system. This could be used with the camera that would be implemented in order to record high quality videos directly onto the system. The storage would be in the form of microSD and be able to load music or videos onto it for playback on the HUD.

Many features could've been added to the idea of the system but those were the ones that were actual viable if time permitted. This project could have always had more but the scope of the project was to just assist the motorcycle riders from distractions of their phone, and to help them with things such as navigation that wasn't always readily available for them.

3.3 Strategic Components and Part Selections

The selection of the viable devices that will be integrated into our system was an important task, due to there being many different options. When analyzing different solutions for our goals, we obtained many different angles to our problems, but the following section will explain why we selected the important components of our system. Making sure to compare between each technology, and analyzing the similarities and differences.

3.3.1 Choosing a Bluetooth Module

When comparing to other Bluetooth Modules out there on the market, there are endless numbers of them due to Bluetooth being integrated in almost everything in our daily lives. Searching high and low for a Bluetooth audio Module that would work perfectly for our system but also not be overkill or too little to our needs wasn't as difficult due to the huge demand for Bluetooth in modern technology. We reviewed many types of different Bluetooth modules, some of them include the 0VC3860, and Silicon Labs' WT32.When analyzing the similarities and differences of all 3, there are more similarities than differences. The price of each do vary, where the RN52 prices at $20, the 0VC3860 at $15, finally the WT32 at $25. All three come with programmable UART ports, which can be used to connect to a microcontroller for better control. They also have more similarities between each other such as dedicated digital signal processing modules (Analog to digital, and Digital to Analog converters). One difference that sets the 0VC3860 and the WT32 apart from the RN52, is that they come with a built-in Li-Po battery charger. This means that the system can use an on-board power when connected to an external battery with a voltage regulator, but all three systems need at least 3.3V to 3.5V to keep all applications running. Such as on the WT32 to power the Universal Serial Bus (USB) on the device, you need 3.3 V into the VDD_CHG pin. Considering the 0VC3860 has the same battery charger to use outside batteries but also needs a VDD_IO to use all applications. The following will be a breakdown of the features and key specifications of each

3.3.1.1 0VC3860 Bluetooth 2.0 +EDR stereo audio processor

Features:

- Integrated single chip Bluetooth Stereo Audio
- Low Power Consumption
- Bluetooth Version 2.0 +EDR specification compliant
- Integrated hi-fi stereo audio CODEC with -90 dB SNR DAC
- Integrated 150 mA Lithium battery chargers
- Integrated Switching Voltage Regulator
- UART and SCCB interfaces
- Low Power Mode 1.8 V
- RoHS Compliant

Figure 8 - OVC3860 Stereo Audio Processor

Permission pending

Specifications:

- Power Supply: VDD 1.7V ~1.9V
- Vio:1.7~3.3V
- Vreg: 2.2V~4.2V
- Power Requirements: active:26mA
- In Sleep Mode: 400μA
- Temperature Operating Range
- In use: -10°C to 80°C
- Storage: -45°C to 125°C
- Operation Range: up to 10 m
- Package Dimensions: 7mm x 7mm x 0.9mm

3.3.1.2 Silicon Labs WT32 Bluetooth Audio Module

Figure 9 - WT32 Bluetooth Module

Permission pending

Features:

- Bluetooth Version 2.1+EDR
- Bluetooth Class 2 Radio
- Transmit Power: +7 dBm

- Receiver sensitivity: -86 dBm
- Integrated Chip antenna
- UART host interfaces
- 802.11 co-existence interface
- 10 programmable IO pins
- Li-Ion and Li-Poly battery charger

Specifications:

- Operating Voltage: 1.8V to 3.6 V
- Temperature range: -30°C to 85°C
- Dimensions: 35.75mm x 14.5mm 2.6mm
- Audio features include I2S, PCM and SPDIF interfaces
- Integrated DSP and Stereo Audio Codec

3.3.1.3 Roving Networks RN52

Figure 10 - RN52

Permission to use from Open Source

Features:

- Fully qualified Bluetooth 3.0 Module
- UART programmable console interface
- Dedicated GPIO pins
- Dual-Channel, differential audio input and output for highest audio quality
- Supports iAP to discover iOS devices
- Integrated amplifiers for driving 16 ohm speakers
- FCC, ICS, and CE certified
- Embedded Bluetooth stack profiles: A2DP, AVRCP, HFP/HSP and SPP
- Castellated SMT pads for easy and reliable PCB mounting

Specifications:

- Maximum Data rate 3 Mbps
- Radio Frequency Impedance: 50 Ohms
- Operation range: 10 meters or 33 feet

- Sensitivity: -85 dBm
- Supply Voltage: 1.8V-3.6V
- Working Temperature: -40°C to 85°C
- Standby current: < 0.5 mA
- Package Dimensions: 26.0mm x 13.5mm x 2.7mm

3.3.2 Comparison of Bluetooth Modules

Table 3 - Comparison of Features

0VC3860 Bluetooth 2.0 +EDR stereo audio processor	**Silicon Labs WT32 Bluetooth Audio Module**	**Roving Networks RN52**
Integrated single chip Bluetooth Stereo Audio	Bluetooth Version 2.1+EDR	Fully qualified Bluetooth 3.0 Module
Low Power Consumption	Bluetooth Class 2 Radio	Integrated amplifiers for driving 16 ohm speakers
Bluetooth Version 2.0 +EDR specification compliant	Transmit Power: +7 dBm	Supports iAP to discover iOS devices
Integrated hi-fi stereo audio CODEC with -90 dB SNR DAC	Receiver sensitivity: -86 dBm	Embedded Bluetooth stack profiles: A2DP, AVRCP, HFP/HSP and SPP
Integrated 150mAH Lithium battery chargers	Integrated Chip antenna	FCC, ICS, and CE certified
Integrated Switching Voltage Regulator	UART host interfaces	UART programmable console interface
UART and SCCB interfaces	802.11 co-existence interface	Dedicated GPIO pins

0VC3860 Bluetooth 2.0 +EDR stereo audio processor	Silicon Labs WT32 Bluetooth Audio Module	Roving Networks RN52
Low Power Mode 1.8 V	10 programmable IO pins	Dual-Channel, differential audio input and output for highest audio quality
RoHS Compliant	Li-Ion and Li-Poly battery charger	Castellated SMT pads for easy and reliable PCB mounting

Table 4 - Comparison of Specifications

0VC3860 Bluetooth 2.0 +EDR stereo audio processor	Silicon Labs WT32 Bluetooth Audio Module	Roving Networks RN52
Power Supply: VDD 1.7V ~1.9V	Operating Voltage: 1.8V to 3.6 V	Maximum Data rate 3 Mbps
Vio:1.7~3.3V	Temperature range: -30°C to 85°C	Radio Frequency Impedance: 50 Ohms
Vreg: 2.2V~4.2V	Audio features include I2S, PCM and SPDIF interfaces	Operation range: 10 meters or 33 feet
Power Requirements: active:26mA	Integrated DSP and Stereo Audio Codec	Sensitivity: -85 dBm
In Sleep Mode: 400μA	Dimensions: 35.75mm x 14.5mm 2.6mm	Supply Voltage: 1.8V-3.6V
In use: -10°C to 80°C		Working Temperature: -40°C to 85°C
Storage: -45°C to 125°C		Standby current: < 0.5 mA
Package Dimensions: 7mm x 7mm x 0.9mm		Package Dimensions: 26.0mm x 13.5mm x 2.7mm

3.3.3 Audio Bluetooth Module Selection

One of the components of our B.R.A.I.N. Helmet is the RN-52 Bluetooth Module. This Bluetooth module will be the basis of how we connect wirelessly from our input to one of the outputs, such as a pair of speakers. The RN52 comes equipped with lots of features ready to be installed in a system for Bluetooth connectivity. Some of the features that exist include having built in analog to digital converters, digital to analog converters, low pass filters, and input and output amplifiers. It also has a software configurable commands over a UART console interface. The RN52 is also compatible with older version of Bluetooth such as 2.1 +EDR, 1.2, and 1.1. This module also includes dedicated GPIO pins allowing for microcontrollers to efficiently control and operate functions. The size of the module is as a big as a stamp size, built at 13.5 x 26.0 x 2.7 mm. This module is also FCC, ICS, CE, and Bluetooth SIG certified. Applications for this specific Bluetooth module include but not limited to, High quality, 2 channel audio streaming, Hands free audio, Wireless speakers, remote control for media player, and computer accessories, and finally used in our B.R.A.I.N Helmet.

RN52 Block Diagram

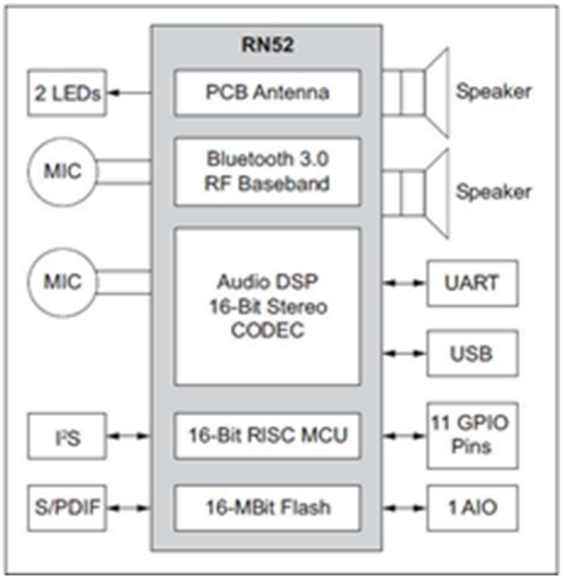


Figure 11 - RN52 Block Diagram

Permission to use from Open Source

The block diagram of the RN52 shows a better break down of the module itself. Analyzing the diagram, you can see that the RN52 comes built made for 2 microphones for input signals, Bluetooth 3.0 radio frequency baseband chip, 2 IO pins for speakers, designated channels for UART, USB, built in 16-bit RISC architecture microcontroller, and a 16-bit digital signal processor CODEC.

3.3.3.1 Features of the RN52 Bluetooth Module

- Analog to Digital Converters
- Digital to Analog Converters
- Low Pass filters
- Input and Output Amplifiers.
- LED Interface
- Microphone Input

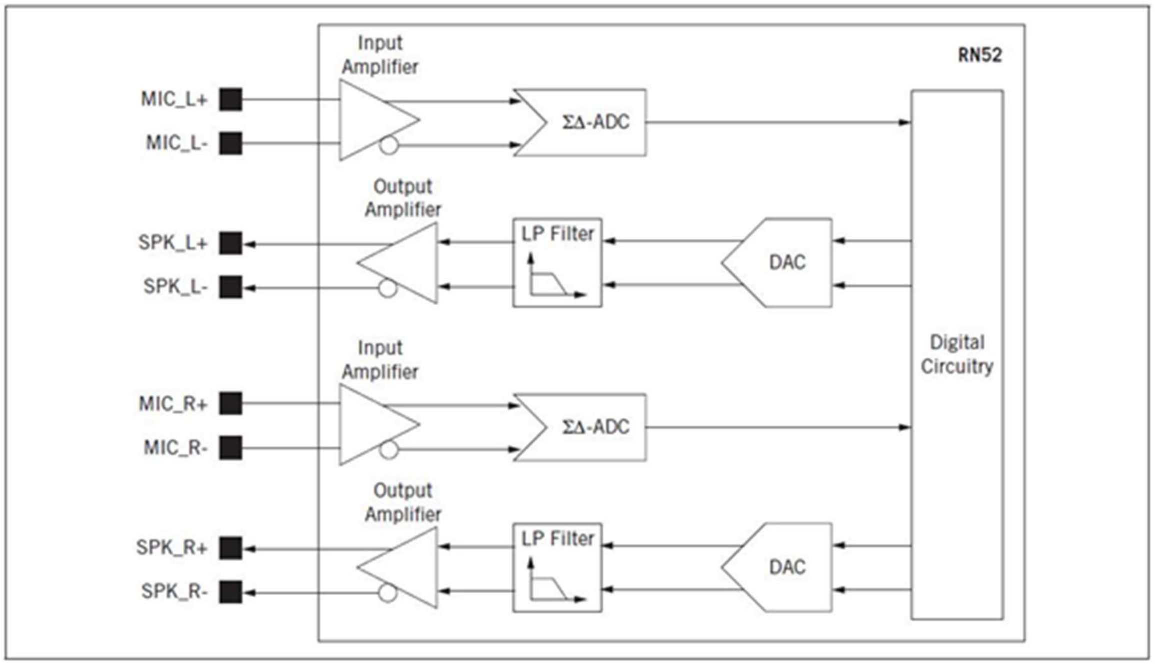


Figure 12 - Overview of Audio Interface Circuit

Permission to use from Open Source

Analog to Digital Converter (ADC)

The Analog to Digital converter that is built in the RN52 is a second-order delta sum converter. The two ADC can support certain sample rates such as but not limited to, 8 kHz, 11.025 kHz, 16 kHz, 24 kHz, and 32 kHz. These ADC analog amplifier is a two- stage amplifier, where the first stage selects the correct gain for either microphone or line input. A delta sum ADC is a modern converter to change a signal from analog to digital. It is made of two main parts, the delta sum modulator, and the digital/decimation filter. The uses of the ADC are to convert the analog signal of the microphone into a digital signal to be used within the microcontroller. The reason for a higher order ADC is to lower the modulator in band quantization noise. The following diagram is a breakdown of a second-order delta sum ADC. The second order trait can be explained by the number of integrators the signal goes through before the output.

Digital to Analog Converter (DAC)

There are two DAC located on the RN52 for separate channels with identical functions. Each DAC supports sample rates of 8 kHz, 11.025 kHz, 16 kHz, 22.05 kHz, 24 kHz, 32 kHz, and 44.1 kHz. The DAC uses the same architecture as the ADC, with the use of the sigma delta modulator, but in this case of the RN52, it has 3 integrators unlike the second order of the ADC.

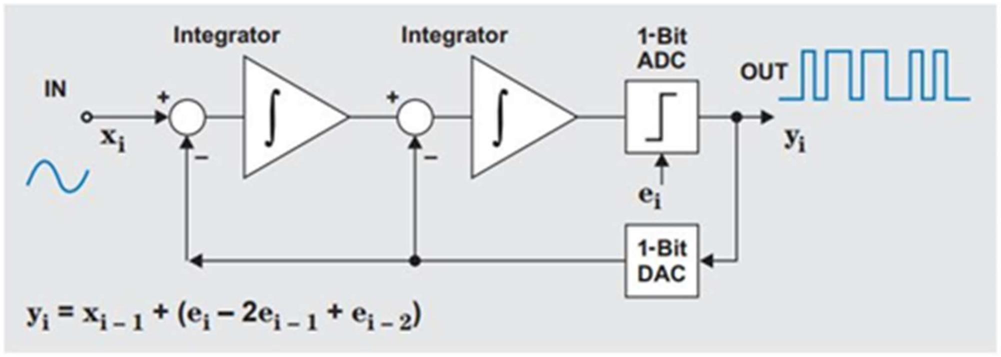


Figure 13 - Second Order ADC in Time Domain

Permission to use from Open Source

Low Pass Filter

After the DAC, the analog signal needs to be passed through a low pass filter so that the noise from the conversion doesn't output an unclear sound to the speakers. What a low pass filter does is only allows certain frequencies pass through the filter up to a certain cut off frequency depending on the filter.

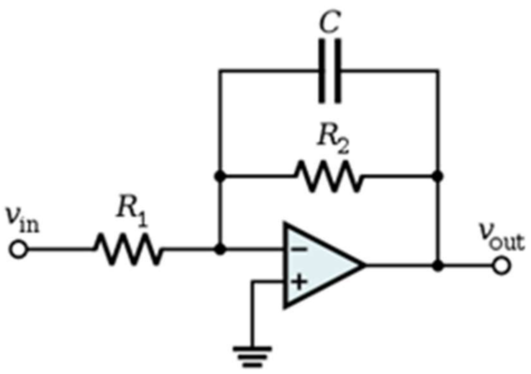


Figure 14 - Basic Low Pass Filter

Permission to use from Open Source

The use of low pass filters is almost infinite, they are used in all day to day based electronic systems. How the low pass filter operates is at low frequencies the capacitor acts like an open circuit so the current flows through resistor 2 with a gain of -R2/R1. At higher frequencies, the capacitor acts like a short to R2 so of course the current will go through the shunt and that would connect the output to ground which wouldn't allow any current to flow through. This reaction can be confirmed quantitatively by defining the transfer function of the low pass filter as H(s)= Vout(s)/Vin(s). Where we can define Vout(s)=-R2 in parallel with the capacitor 1/sC and Vin(s) is just R1. And then changing from s domain to frequency we can derive the transfer function is -K*(w/(s+w)) where K= the ratio of R2 and R1. To figure out where the filter cuts off the frequencies it lets pass through we find that it is where the maximum magnitude of the transfer function has been reduced by . The use of this kind of filter in the RN52 would be used to not allow the high frequency noise of the DAC go into the output or speakers of the system.

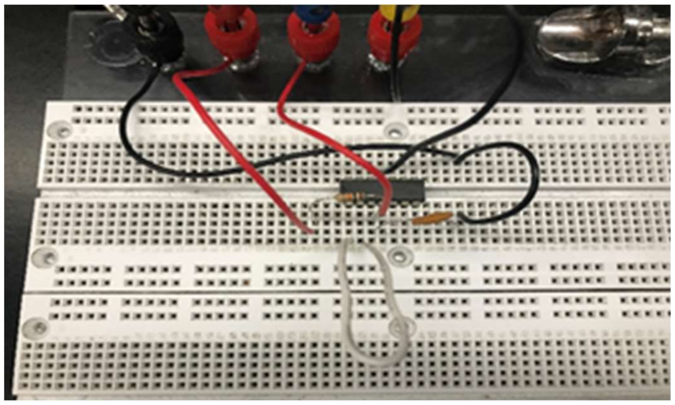


Figure 15 - Breadboard of Low Pass Filter

Microphone Input

The use of microphones is on feature that is built in with the RN52 Bluetooth Audio Module. We needed to use the microphone in order to talk on the phone that is connect wirelessly with the module. The specifics of the audio input are to be from 1 μA at 94 dB to about 10 μA at 94 dB. This means the sensitivity of the microphones must be between -40 to -60 dBV. If any microphone is used below these limits the microphone output must be pre-loaded with a large value resistor to ground.

LED Interface

The RN52 has 2 pads for driving the LED indicators on the system. The firmware that is preloaded on the RN52 can control both terminals and the battery charger that can set LED0. The terminals on the RN52 are both open drain, meaning the LED must be connect to a positive supply line, to a limiting current resistor. The

LED’s have a different combination that show different things that the Bluetooth module is currently doing.

A better explanation of the LED function, is explained when both LED’s are flashing, the Bluetooth module is discoverable through your Bluetooth device. If only LED0 is flashing, then that means the RN52 has been connected to a device already. If only LED1 is flashing, then the RN52 is connectable because it recently disconnected from its previous device. These LED’s on the RN52 system that we attained were Green and Orange respectively.

Table 5 - LED Descriptions

LED	Status	Description
LED0 and LED1	Flashing	The RN52 is discoverable to a device
LED0 only	Flashing	The module is Connected
LED1 only	Flashing	The module is Connectable

Restore Factory Defaults

Using a switch or push button, the GPIO4 pin on the RN52 can be used to restore the RN52 back to its factory default settings. This is an important application of the RN52 just in case the RN52 is misconfigured. To reset to factory defaults, the GPIO4 should be on high power on power on, then toggle from low to high, low to high, with 1 second intervals between. This would be useful in the case we accidentally get stuck in an infinite loop while trying to connect our device.

Regulations of the RN52

The Federal Communications Commission CFR47 Telecommunications, Part 15 Subpart C “intentional Radiators” in accordance with Part 15.212 Modular Transmitter shows approval of the RN52. The FCC regulates Radio Frequency devices contained in electronic-electrical products that are capable of emitting radio frequency energy by radiation, conduction, or other means. Since this energy can cause an interference of radio services in the radio frequency range of 9kHz to 3000 GHz. Almost all electronic devices emit some form of RF energy, but that doesn’t mean all devices must be tested by the FCC. Only products by design, that include circuitry that operates in the RF spectrum needs to be reviewed and complied with the FCC equipment authorization procedure. The RF device must be approved using the appropriate equipment authorization procedure before it can be marketed, imported or used in the United States. For the RN52, this RF device is under the “Intentional Radiators” subsection which is defined as a device that intentionally generates and emits RF energy by radiation or induction. Many other devices are under this subsection such as a wireless garage door opener, RF universal remote control, Bluetooth Systems, Wi-Fi, and Alarm systems. The RN52 meets all the regulations by the FCC in the RF exposure category. The compliance of the FCC RF exposure requirements, “Evaluating Compliance with FCC Guidelines for Human Exposure to Radiofrequency Electromagnetic Fields” helps to assure that the RN52 is approved by the FCC. There is a statement that must be said when using this device and it reads; “*To satisfy FCC RF Exposure requirements for mobile and base station transmission devices, a separation distance of 20 cm or more should be maintained between the antenna of this device and persons during operation. To ensure compliance, operation at closer than this distance is not recommended. The antenna(s) used for this transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.*”

3.3.3.2 Senior Design 2 Update (Bluetooth)

Throughout the course of Senior Design 2, there were some major changes to the system that were made. In the case of Bluetooth module selection, the RN52 was replaced by two different Bluetooth modules. The purpose of this split of Bluetooth Modules was due to the app development. The GPS app was developed with the Bluetooth Low Energy, which means it would not work with our previous selected Bluetooth Classic Module. The first module we turned too, was the BC127 Dual Mode Bluetooth Module. The reason behind this change was it had a dual boot mode with both a Classic Bluetooth 4.2 as well as a Bluetooth Low Energy Module for the application. The problem was solved, but unfortunately there was a flaw with how much data was sending between the BLE module on the BC127. Its data throughput wasn’t enough for the app and wasn’t sending enough data from the phone to the system, so we needed to bigger and better BLE, so we decided with the Bluefruit LE Adafruit Friend. This BLE module is very useful because it had enough throughput to send all the data from the phone to the system to update the HUD. The need to split the modules was due to Classic Bluetooth being able to have audio profiles such as SPP, but within BLE there are no audio profiles so the BC127 was for the audio system whereas the Bluefruit LE was used for communication for the system for the HUD.

3.3.4 Microcontroller

The main component of the smart helmet is through a low power microcontroller. This will allow the connection from the host device to the helmet to offer navigation instructions, music playback, answering phone calls and a heads-up display. There will be tactile buttons on the helmet available to the user to change volume of the sound from the speakers, accept or decline phone calls, and toggle the screens for out heads-up display.

3.3.4.1 Microcontroller Research

Before we can decide on what processor to use for our system, we need to determine what components that we need to attach to the system. We know that with our initial drawing schematic we can tell that we needed a lot of GPIO pins to get all the modules together. We need to anticipate the display pins to contain a lot of wires for use to connect, but then we realize that there is an adaptor that converts those ribbon wires into less wires and communicate through SPI or I2C. We did have our MSP430G2 Launchpad microcontroller we used in our Embedded Systems course the previous semester, but it does not have enough I/O pins to connect everything together. Stephan did have an Arduino 101 microcontroller, even though it had more wires, it doesn't have enough ports to use for our system. Our next step is to figure out just how many pins do we need for each component, so we done some research into the components we selected and figured out that there was a Bluetooth module that contains a ADC, DAC, and an audio DSP inside; so, it includes both stereo output and microphone input. The Bluetooth module contains 40 pins, but we are not using all of them depending on the communication medium that we will choose later when assembling all the components together. Based on the datasheet for that module, we don't need connections to do firmware updates, USB connection, and depending on the communication medium we can save approximately 15 pins to not connect. With the Bluetooth module taken care of we can confirm that we cannot use the Arduino 101 nor the MSP430G2 dev boards for our main processor because they don't have enough GPIO pins to operate that module unless we have another processor that communicates to the main processor, and that would take more work than selecting one main processor. For debugging we are going to ass LEDs onto our PCB board to troubleshoot our program running on that processor.

The other major component we need to take care of is the display. We did state that there are adaptors to those displays that minimize the I/O pins that we need to connect to the controller, it even includes an IC chip that controls the entire display and contains header files that we can use too. The display that we bought first to test is an 0.96" monochrome OLED display. This display has a 50-pin ribbon cable that is connected to a 4-pin header, VCC, Ground, SDA, and SCL. It must use I2C to communicate to the display. When we obtained the display and test it, we found a potential issue that we need to address. That issue is that the header files contains all the basic functions to add text into the display and rotate the screen but no options to draw our own objects into the display. So, now we plan to modify the header files to be able to create our own objects to visually communicate the user the time, distance to the next turn, and which direction the user need to turn while driving. Now, we need a processor with at least 40 GPIO pins to communicate to the Bluetooth display, 4 for the display, and 5 for switches and potentiometer for volume control. We want to have spare GPIO pins so that if we want to add another display or upgrade other components during our prototyping stage we would have enough room to implement it. It would not be beneficial for us to run out of room on our current processor that we have to select an alternative processor. We also need room to add debugging features into the PCB design to troubleshoot and program the processor with our code. Our electrical team will utilize the schematics that the manufacturers provide on their development boards to allow us to flash the program onto our main PCB that we will be making.

3.3.4.2 Microcontroller Options

Now that we have a great idea on what processor to have for our project we can decide on selecting the right one for our project. While researching for the correct microcontroller, we had no bias on what we prefer. We narrowed down our options down to several products that all had similar characteristics of what it can deliver for our system. All three options have a development board available for us to test on and order only the processor for our system. Our first option is Texas Instruments because they are known for their low-power microcontrollers and excellent documentation. Their low power modes on the processors that they currently provide promote efficiency on their processors by disabling the modules built onto the processor until an interrupt occurs. The software that they provide has all necessary features to run and debug the controller. The other option is Atmel because they offer a larger inventory of microcontrollers to use with an abundance of development boards to test our system. Atmel does offer microcontrollers on their store, but they are not as popular in the market than the Arduino. The Arduino microcontroller that they provide is powered by Atmel's processors. The Arduino is one of the popular and well-known microcontrollers in the market because they appeal to tinkerers at any age. The three processors that we chose for comparison are in different architectures. They are discussed on the following sections.

3.3.4.2.1 Texas Instruments MSP430

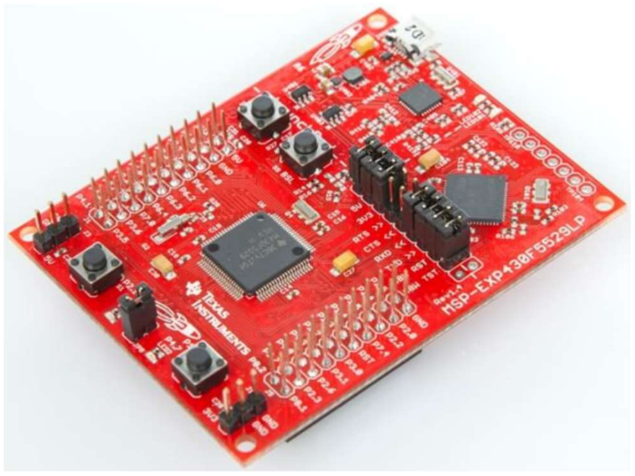


Figure 16 - MSP430 Development Board

Obtained from Ti's MSP430F5529LP Launchpad product page

The Texas Instrument’s MSP430 is our first option for our research because we are familiar with their architecture from our microcontrollers from our embedded systems course. The MSP430G2 that we used for our course is not compatible with our project because it does not have enough GPIO pins to work with the modules we want to connect to. The removable processor in that development board was the main pro because we can flash the chip on the dev board and place the processor into our PCB so we don’t have to add a debugger interface into the main PCB board. After doing some research we found a processor that we can use for our system, which is the F5529. This processor offers speeds up to 25Mhz, 128KB of flash, 8KB of RAM and an integrated 12-Bit ADC. The 16-bit microprocessor has one communication channel for UART, I2C, or SPI.

3.3.4.2.2 Atmel ATmega

The AVR architecture from Atmel is our next option in selecting the right processor to use. The popular microcontrollers that use their chips are from Arduino. We looked around the inventory that Arduino offers and there was one clear development board to use based on elimination. The dev board that we selected for comparison is the Arduino Mega 2560 (Revision 3). This board could offer us more than what we needed for the project, and it might prove beneficial in the future when we add more features during the production stage. This offers four distinct communication channels for UART, SPI, and I2C. The processor’s frequency goes up to 16Mhz, contains 256KB flash and 8KB RAM. It also contains 16 analog input pins and 11 PWM pins. This unique feature that this board provides are beneficial to us because it allows us to control the power individually to each component to save overall power of the system. These pins can allow us to fade the arrows on the display when indicating which lane the driver needs to go. If there are extra pins available to us in the production stage, it can allow us room for upgrades and/or additional features to the helmet.

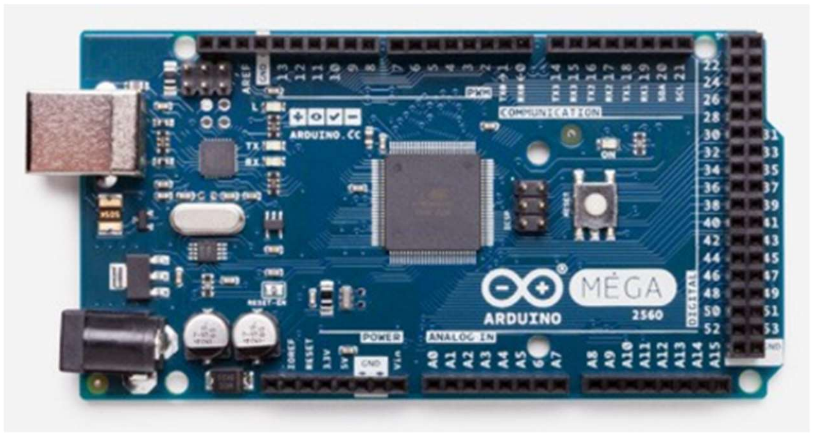


Figure 17 - Arduino Mega 2560 R3 Development Board

Obtained from Arduino's Mega 250 R3 product page

3.3.4.2.3 Texas Instruments MSP432

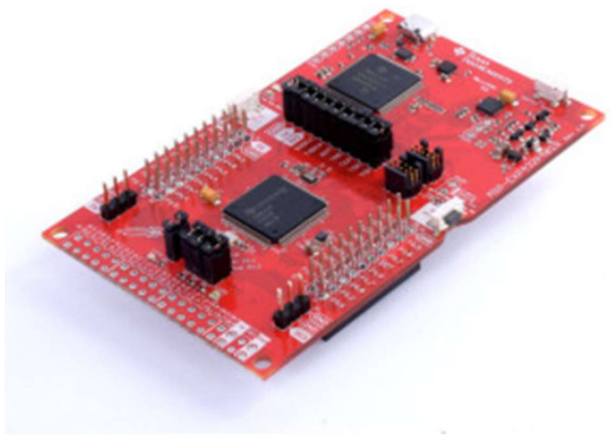


Figure 18 - TI MSP432P401R Development Board

Obtained from Ti's MSP430F5529LP Launchpad product page

The MSP432 line of processors utilize the ARM architecture. This architecture is the most popular in the market because most of the electronics like cell phones uses it. The development board we considered is the MSP432P401R. It contains a 32-bit ARM Cortex M4F running up to 48MHz, a 24 channel ADC, 256KB Flash and 64KB RAM. This is the processor that can rival the ATmega2560 than the F5529. With a generous amount of GPIO pins, this controller rivals against the Arduino Mega because of said reason. Since this is from Texas Instruments, their low power modes are another reason why we considered this for our comparisons. With this board like the Arduino, we will consider using this development board in addition to our selected controller in the production stage so we can compare the different architectures' capabilities hands-on. This processor has the capability of having an embedded operating system if we add some components, which requires us to program an application that will run independently from the companion app on the mobile devices. This would not be used for our system because it is unnecessary since it needs more power for the additional components.

3.3.4.3 Microcontroller Comparisons

Before deciding on which processor to use, we must do some comparisons to see which one we can use. The comparisons offer us to prioritize the processor's features and select the right one to use for the helmet. The table below summarize each processor's characteristics. The topics that we used for comparison is what we are considering for our helmet.

3.3.4.3.1 Power Consumption

The processor will be in active mode most of the time that the system will be connected to the host device. It will only be in low power mode if the helmet is not connected to the device. There will be a switch to turn off the electronic system if the user is not using it or without the electronic features enabled. Based on the datasheets the MSP432 offers the lowest current rate and the ATmega2560, six times higher current than the predecessor.

3.3.4.3.2 Cost

The cost of the microcontroller will be based on the development board costs and the processor die cost. We compared each development board products and determined that on the Texas Instrument's side of the dev boards, they offer almost 3.5 times cheaper than the Arduino's board. In addition to that, the Atmel processor costs an average of 1.45 times more expensive than the Ti's processors.

3.3.4.3.3 Memory Size

Memory size is not the biggest factor for our system since the programming code is going to be optimized to reduce space as much as possible and most of the processing will be mitigated from the Android application running on the host's device. But we still want ample space just in case for our header files that we include our system take up significant space. The MSP432 has eight times more RAM than the other two microcontrollers we considered. For Flash memory, the MSP430 has twice as less space available for program space than the 256KB space for the Arduino and the MSP432.

3.3.4.3.4 GPIO

For I/O we want a processor that has a lot of GPIO pins so that we offer functionality to the system. If there are spare I/O pins, then it also offers us room to add additional features while in the production stage or in the worst-case scenario, different components that may require more I/O pins. The Arduino and the MSP432 have similar I/O pins available with 86 for the 2560 and 84 for MSP432.

3.3.4.3.5 Clock Frequency

Clock frequency is a moderate factor in our system. The faster the clock rate allows the increase response rate of the display that we will be driving in the system. The faster clock rate also can reduce the processing latency and will only be the Bluetooth module's latency that we need to take into consideration. The MSP432 is three times faster than the 2560 and 1.92 times faster than the ATmega2560.

3.3.5 Microcontroller Choice

Based on the datasheets and comparisons that these microcontrollers offer, we decided to use the ATmega2560 processor in the Arduino development board because it contains enough GPIO pins to comfortably function our system. Even though it runs at a significantly higher power consumption rate than the others, we can utilize lower clock frequencies to decrease the power to a point where the system can run efficiently and lengthen the battery life. The unique features of the processor like the four distinct communication channels and the abundance of analog and PWM I/O pins is also why we selected this.

	TI MSP430F5529	**ATmega2560**	**TI MSP432P401R**
Processor Price	$8.06	$11.85	$8.29
Dev Board Price	$12.99	$45.95	$12.99
Power Consumption	290 uA/MHz	500 uA/MHz	80 uA/MHz
RAM	8 KB	8 KB	64 KB
FLASH	128 KB	256 KB	256 KB
GPIO	63 Pins	86 Pins	84 Pins
Clock Frequency	25 MHz	16 MHz	48 MHz

Table 6 - Microcontroller Comparisons

The above table describes the multiple options we considered for our microcontroller. While many of the options shown above and on the market currently are a possibility for the BRAIN Helmet, our computer engineering team decided to choose the ATmega2560 regardless of its higher price value as it allows the possibility for future features to be added to the helmet. The following figure is a block diagram of the Atmel ATmega 2560. This block diagram depicts multiple features that will allow the BRAIN helmet to receive and process the information from the Bluetooth Low Energy module and Android Application. From there, the ATmega2560 will decode and send characters to the Heads-up display (HUD) for the motorcycle rider.

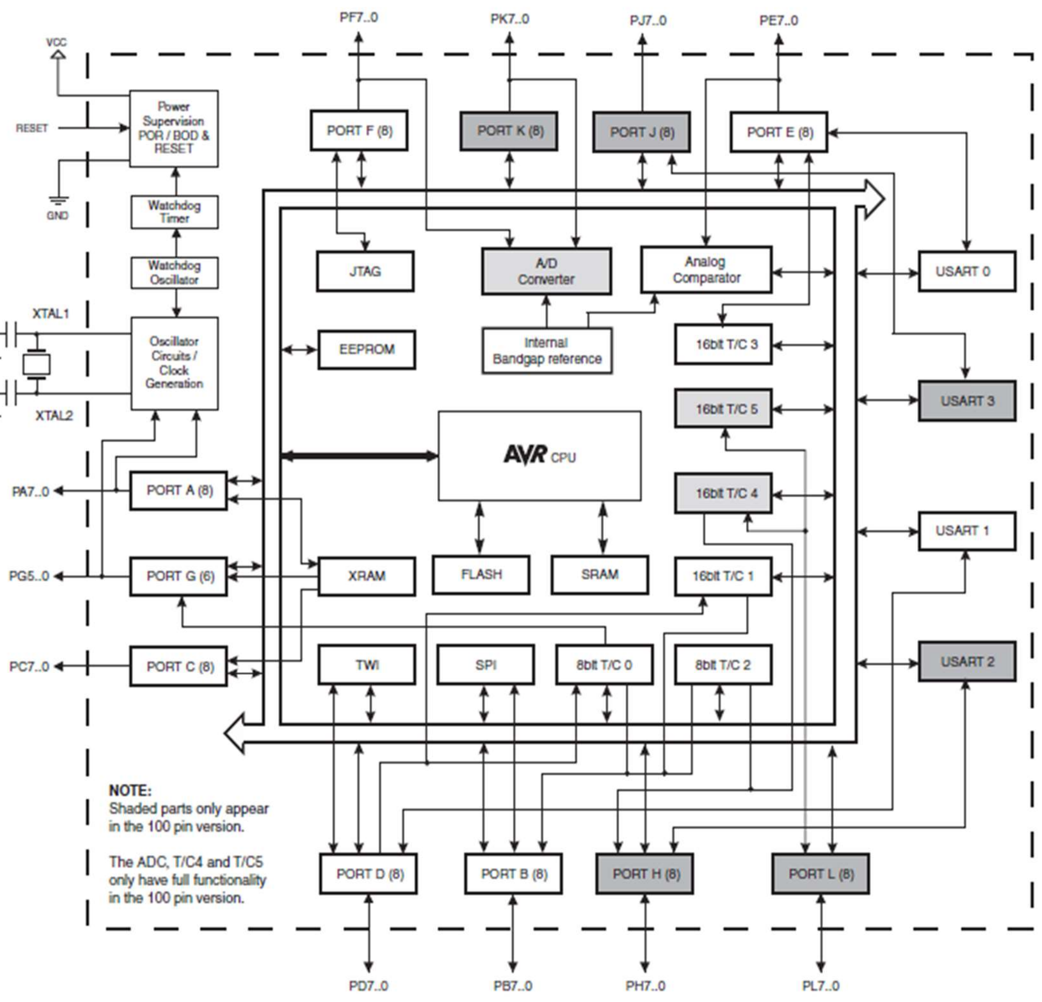


Figure 19 - Block Diagram of Atmel ATmega2560

Obtained from Arduino's Mega 250 R3 product page

3.3.6 Battery Comparisons

Once we agreed that a secondary battery would be best for the BRAIN helmet, the decision initially fell upon deciding between a lithium ion battery and a lithium polymer battery. In today's continually developing electronic world, the process of construction for lithium ion and lithium polymer batteries has become relatively cheap that even with the greater advantage over primary batteries, the lithium ion and lithium polymer battery's price remains competitively cheap. Below is a depiction of a new polymer lithium ion technology battery that we have considered to use for our project. As evident in the below figure, the size of a lithium polymer battery is barely larger than 4 quarters laid close together. While this is noticeably larger than other battery choices previously discussed, it is not drastically larger.

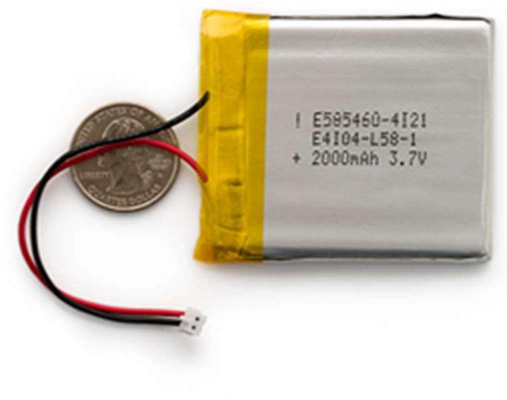


Figure 20 - Sparkfun's Polymer Lithium-Ion Battery

Permission from an open source

Table 7 - Battery Comparisons for Power Supply Unit

Battery Comparisons			
Battery	***Nominal Voltage***	***Battery Capacity***	***Price***
PRT-13813 [Lithium Ion]	3.7V	1000 mAh	$9.95
PRT-08483 [Polymer Lithium Ion]	3.7V	2000 mAh	$12.95
458292 [Lithium Polymer]	3.7V	3800 mAh	$13.59
Adafruit 328 [Polymer Lithium Ion]	3.7V	2500 mAh	$15.98
PRT-1385 [Lithium Ion]	3.7V	6000 mAh	$29.95
PRT-11855 [Lithium Ion]	7.4V	1000 mAh	$9.95
PRT-11856 [Lithium Ion]	7.4V	2200 mAh	$15.95

However, there is common knowledge that the Lithium Ion batteries can be dangerous if over charged or if the connectors become faulty they can even start sparks. This has caused the BRAIN Helmet to initially lean towards the more expensive, but safer Lithium Polymer batteries. Upon further researching and stumbling upon Sparkfun's Polymer Lithium Ion Battery, we have decided to test multiple batteries. This newer technology that allows the polymer lithium ion hybrid battery to be safer than the older lithium ion batteries has caused us to follow suit and test with the newer polymer lithium ion battery technology. We decided to also purchase Lithium Polymer batteries to compare differences. It is important to note that a proper charging circuit shall be implemented to keep the BRAIN Helmet user safety a number one priority.

Now that the battery type has been decided upon, an exact battery with a nominal voltage and battery capacity capable of powering the BRAIN Helmet must be selected. In the Table below is a list of multiple batteries that were initially considered for the BRAIN Helmet project.

Going back to the main goal at hand; Proper selection of a battery is started by summing the max active currents drawn from the components of the BRAIN Helmet and then comparing it to a battery's battery capacity to find an approximated run time between charging the polymer lithium ion battery.

$$Battery\ Run\ Time\ Between\ Charges\ (hours) = \frac{Battery\ Capacity\ (mAh)}{\sum Max\ Active\ Currents\ \ (mA)}$$

In the Battery Consumption for Potential Devices table above, both the MSP430g2453 and the OVC3860 were the least power consuming devices, however the power consumption differences are such negligible differences for our given task, that it is not a huge hit to the BRAIN Helmet's battery supply to stray from our team's decision to choose the RN-52 bluetooth module and the ATmega2560.

Now if the BRAIN Helmet utilizes the ATmega2560 or MSP430g2453 at approximately 3 volts, the ATmega2560 will pull approximately 500 μA and the OVC3860 will pull approximately 300 μA. The below calculations will show how long a battery would last if the battery was only supplying these microprocessors and these calculations are based on the following equations:

$$Battery\ life\ (Hours) = \frac{Battery\ Capacity\ (Amphours)}{\sum Current\ Drawn\ (Amps)}$$

Assuming a simple Lithium Polymer/Polymer Lithium Ion with a nominal 3.7 Voltage and a battery capacity of 1000 mAh:

ATmega2560:

$$Battery\ life\ (Hours) = \frac{1000\ mAh}{0.5\ mA} = 2000\ hours$$

MSP430g2453:

$$Battery\ life\ (Hours) = \frac{1000\ mAh}{0.3\ mA} = 3{,}333\ hours$$

Further research has shown that it is a better approximation to multiply the calculated battery life hours by 0.7~0.75 to account for extra battery drainage:

$$Battery\ life\ (Hours) = 0.7 * \frac{Battery\ Capacity\ (Amphours)}{Current\ Drawn\ (Amps)}$$

Therefore,

ATmega2560:

$$Battery\ life\ (Hours) = 0.7 * \frac{1000\ mAh}{0.5\ mA} = 1400\ hours$$

MSP430g2453:

$$Battery\ life\ (Hours) = 0.7 * \frac{1000\ mAh}{0.3\ mA} = 2{,}333\ hours$$

While the MSP430g2453 does have noticeably larger lifetime for a standard lithium ion battery, it is not necessary for the BRAIN Helmet nor does it outweigh the benefits of the extra GPIO pins the ATmega2560.

For the exact battery life necessary, these values are confirmed by the table below for our selected stock components:

- Microprocessor max active current is approximately 500 µA
- Bluetooth Module max active current is ~30 mA
- Heads-Up Display max active current ~21 mA

$$\sum Current\ Drawn\ (mA) = .5 + .15 + .15.1 + 20.9 = 51.5$$

Given the average current for the stock BRAIN Helmet build is ~52 mA, the following table below compares different battery life times for comparing batteries of different battery capacities. The Battery life values are derived from the above equations.

In the case of the BRAIN Helmet, bigger isn't always better. A larger battery capacity leads to the battery size also being larger. Multiple manufactures were considered for this approach as shown in the below table. While some batteries grew in both length and height, some manufacturers simply "flattened" the battery. A large flat battery could be implementable for the BRAIN Helmet, but would lead to more time invested into ensuring the helmet remains aerodynamic. Additionally, a larger battery is not totally necessary for the stock BRAIN Helmet features.

Table 8 - Stock Battery Consumption for Calculations

Stock Battery Consumption for Calculations			
Device	***Operational Voltage***	***Active current***	***Sleep current***
Microprocessors			
ATmega 2560	1.8 – 5V *Not using above 3.6V	~500 μA	0.1 μA
Bluetooth Modules			
BC127	3 ~ 3.6V	~15 mA	<1 mA
UART Friend	3 - 16V	Peak: 15.1mA	<1 mA
Heads-Up Displays			
LCD-13003 Breakout [48x64 pixels]	3.3V *generates V_{LCD} = 7.0-7.5V	10-20.9mA	<300 μA

Table 9 - Battery Comparisons of Life Time

Battery Life Time Comparisons	
Battery Capacity (mAh)	**Battery Life Time (Hours)**
1000 mAh	13.59
2000 mAh	27.18
2500 mAh	33.98
6000 mAh	81.55

3.3.7 Battery Selection

Due to the minor “disadvantage” in price, size, and weight of implementing a 2000-3000 mAh battery, the BRAIN Helmet will use a 2500 mAh, 3.7 battery; the Adafruit 328. During the research and minor testing phase, the Ofeely 458292 Lithium-Ion Polymer was also purchased due to its low price and high battery capacity.

However, the lack of a JST plug, less amount of public reviews, and the overall grander size cause the BRAIN Helmet team to stray away from the 458292 and back towards the Adafruit 328 for being superior in these three mentioned critiques of the 458292 Lithium Ion Polymer battery.The dimensions of the Adafruit 328 are approximately 2" x 2.55" x 0.30" (50.5mm x 60.2 mm x 7.9 mm) while the size of the Ofeely 458292 Li-Po battery is 5.5” x 4.7” x 0.5”. The Ofeely 4582992 nearly doubles the size of the Adafruit 328 for a battery capacity that does not seem to be necessary for the BRAIN Helmet given the previous calculations. The schematic drawing below depicts the basic rectangular shape of the Adafruit 328. The relatively small Adafruit 328 will have no problem fitting into a designed waterproof, protective, and aerodynamic 3-D printed casing to fit on the back of the BRAIN Helmet.

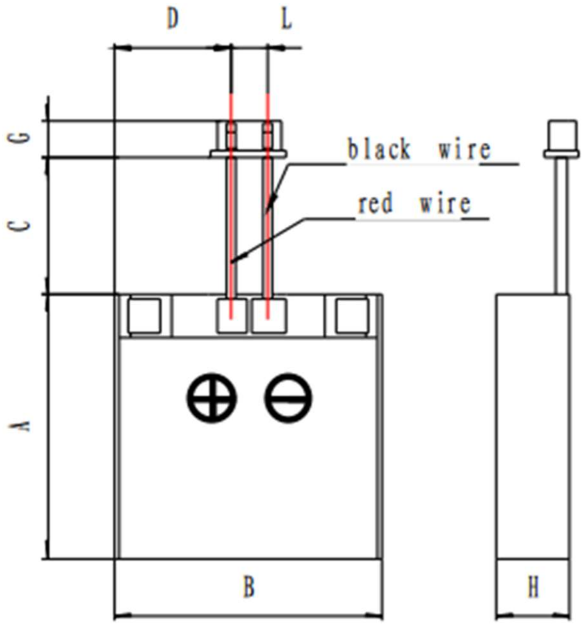


Figure 21 - Adafruit 328 Drawing

Permission from an open source

The baseline for the BRAIN Helmet’s power system was originally to have a battery life of full run time near 5 hours. The Adafruit 328 simply far shoots this lifetime by over 6 times the run time as seen in the above calculations. This battery option was chosen not only for its moderate size and price but additionally because the spare battery lifetime may accommodate future features that may be included in the BRAIN Helmet. This could include extra LEDs, possibly an accelerometer to lessen the Bluetooth data transfer from the phone, or even a higher resolution HUD. This also helps to avoid running the battery too low, as a fully drained Lithium Ion/Lithium Polymer battery can be potentially damaged. Due to this, the batteries considered for the BRAIN Helmet all include a protection circuitry. This protection circuity is put in place to not only cut out the battery when completely dead at

~2.8V, but also to prevent over-charging when the voltage of the battery is going too high. In terms of the Adafruit 328, the max charge of the battery is ~4.2V.

3.3.8 Charger Selection

In order to appropriately choose a battery charger for the BRAIN Helmet, a few standard guidelines for the Adafruit 328 had to be addressed. The following protocols are listed as critical guidelines for charging the Adafruit 328 on the Adafruit website:

1. Do not overcharge the battery past the maximum safe voltage (~4.2V)
   - Issue addressed by the on-cell protection circuit
2. Do not discharge the battery below the minimum safe voltage (2.75V)
   - Issue addressed by the on-cell protection circuit
3. Do not charge the battery with more current than 1C
   - For the Adafruit 328 this means do not charge with 2.5A
   - Issue addressed by the on-cell protection circuit
   - Also addressed by the Micro-usb to JST battery charger (See figure below)
   - Standard charge current is 0.2C = 500 mA
4. Do not draw more current than the battery can provide (0.5C)
   - Issue addressed by the on-cell protection
5. Avoid charging the Adafruit 328 below or above specific temperatures (0~50 degrees Celsius)
   - This problem is typically controlled by the charger and its charge rate being set properly.

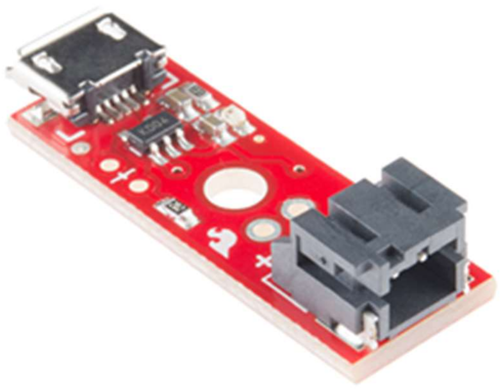


Figure 22 - Sparkfun Lithium Ion Battery Charger

Permission from an open source

The BRAIN Helmet shall utilize the SparkFun LiPo Charger Basic depicted in the above figure. This MCP73831 charger utilizes:

- Charges 3.7V batteries at a rate of 500mA
    - This is the standard recommended charge rate for the Adafruit 328
- Can Charge Li-Ion and Li-Polymer batteries.
- Standard Micro-USB connection to charge the battery
- This is a fitting connection for the BRAIN Helmet as the helmet will be utilizing Android phones that for the majority uses a Micro-USB charger port.
- JST Jack to charge the battery
- $7.95
- 29.4 x 10.88 mm

The typical application circuitry for this MCP73831 Li-Polymer battery charger is depicted below. This circuit is integrated onto the Sparkfun lithium-ion battery charger. The input voltage originates from the micro-USB connection while the single Li-Ion (Polymer) Cell Battery connects via a JST connector.

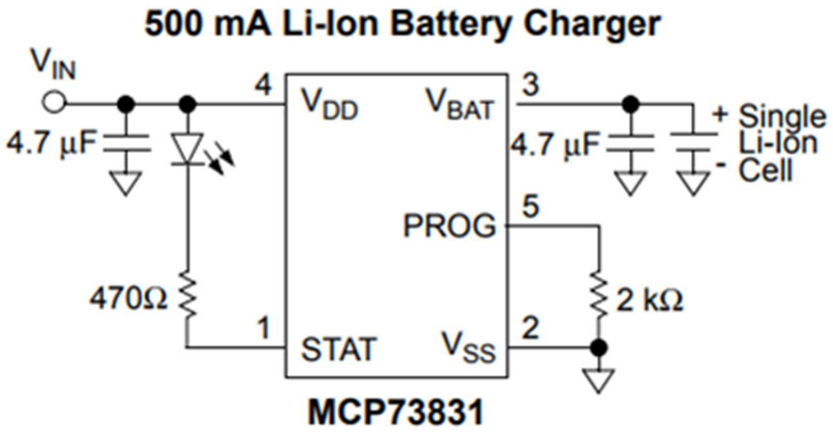


Figure 23 - Standard MCP73831 Li-Po battery charger circuit

Permission from an open source

3.3.9 Voltage Regulation Selection

Originally a Linear Voltage Regulator, or LDO, was planned to be used to regulate the BRAIN Helmet's power system. Upon the second semester of Senior Design, changes were made to utilize a switching voltage regulator; the buck convertor. This will be described in future detail.

3.3.10 Helmet comparison & Selection

Multiple helmets were considered for the BRAIN Helmet. Our primary marketplace for shopping took place on amazon for this matter. They ranged from all sorts of prices from forty dollars to three hundred dollars or more. As a first prototype of the BRAIN Helmet, our immediate goal is to ensure that the electronics components of our smart helmet work more than anything else. Therefore, we already decided we'd go with an economically understandable pricing helmet. The decision came down to deciding between a modular helmet, that is capable of rotating the mask and visor up and over the head, or a full face helmet with only a rotating visor. Because we did not want wires being stretched, or loose to allow extra movement, we decided to stick with the SH-FF0016 full face helmet.

3.3.11 HUD comparison & Selection

As the front runner of the BRAIN Helmet, the Heads-Up Display (HUD) is a crucial component for this product. The Heads-Up Display will be quite literally the only visibly appealing feature of the BRAIN Helmet and therefore several aspects of choosing the right display were taken into consideration. These include:

- A relatively lightweight module that will not add noticeable weight to the BRAIN Helmet
- Non-distracting minimal display to avoid eye strain or blocking the road
    - ~(0.7”-1.5”) diagonal size
- Low Input Voltage to keep power consumption low
- Low cost in case of replacement screens being required by User

Table 10 - HUD Comparisons

Considered Heads-Up Displays			
Display Details	***Operational Voltage***	***Active current***	***Sleep current***
Nokia 5110 LCD [48x84 pixels]	2.7 ~ 3.3V *Internally generates V_{LCD} = 6.0-9.0V	240-300 μA	42 μA
ACROBOTIC LED [128 x 64 pixels]	3.3~5VDC	Unavailable	Unavailable
Diymall IIC OLED [128 x 64 pixels]	3.3~5VDC	Unavailable	Unavailable
LCD-13003 Breakout [48x64 pixels]	3.3V *Internally generates V_{LCD} = 7.0-7.5V	10-20.9mA	<300 μA
uLCD-144-G2 GFX 1.44” [128x128 pixels] RGB	4.0 - 5.5V	40-70mA	500 μA
μOLED-128-G2-GFX 1.5” [128x128 pixels]	4.0 - 5.5V	60-200mA	500 μA

These features for the HUD has led to multiple Heads-Up Displays to be considered. Comparisons of these displays are shown below.

The BRIAN Helmet was originally drafted as a user-friendly helmet with a minimal screen. Therefore, it is a necessity that the HUD has an appealing, and visible HUD. Below in figure 24 is the uLCD-144-G2 GFX 1.44” diagonal length display. This would be one of the fancier and larger screens available for the BRAIN Helmet in terms of size and display elegance. In order to use such a display however would lean towards using a second battery in series in order to maintain a decent daily run time while supplying the higher 4~5.5V input this display requires. Alternatively, an easier approach is using a higher voltage battery such as the 7.4V as shown in the figure 24 below, however the size of this battery would be rather inconvenient for the BRAIN Helmet; creating aerodynamic problems and therefore discomforting the rider.

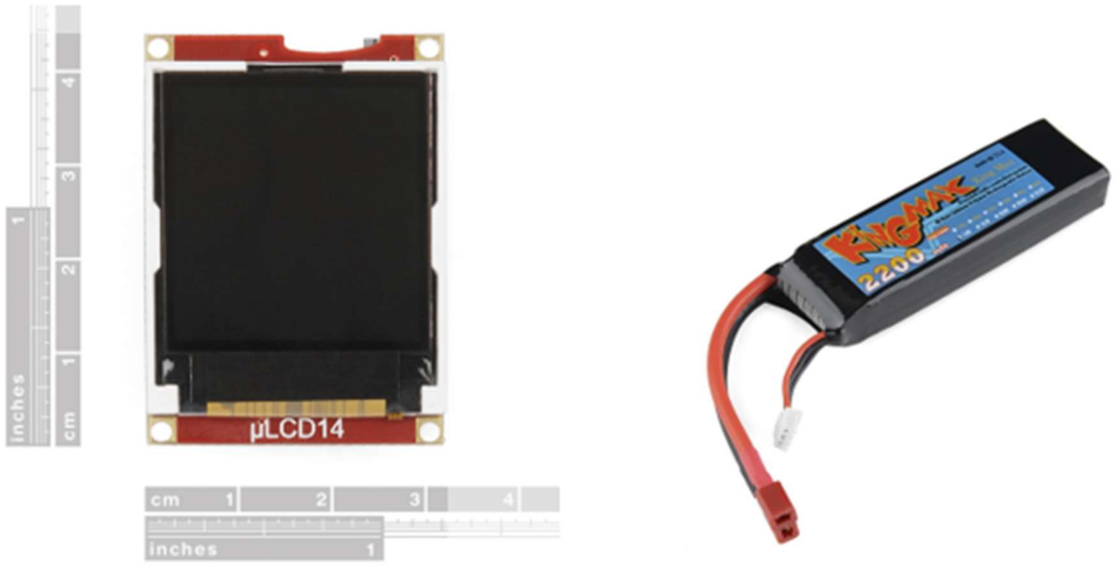


Figure 24 - uLCD-44-G2 GFX and required 7.4V lithium-ion battery

Permission from an open source

The use of the above LCD display or others with a common size and resolution arises not only a higher power cost problem, but as a domino effect causes the PCB and eventual 3D printed casing for the BRAIN Helmet’s electronic group to increase in size and weight. While they appear to be small changes during the research phase of the BRAIN Helmet, such a domino effect can lead to a less comfortable motorcycle ride for the user; and therefore, defeating the purpose of the BRAIN Helmet.The simple solution therefore is keeping a single 3.7V battery that would require using a less powerful display such as the Diymall I^{2}C HTDS-WS96 OLED display with a 0.96” diagonal length. This I^{2}C OLED HUD is not only a decent 0.96” size display,but is additionally cost effective and easy to replace and test.

As a light emitting diode (LED) display, the HTDS-WS96 maintains a lower power consumption than a LCD counterpart; consuming only 0.08W when the entire screen is illuminated, and 0.06W when the screen fully displays ASCII characters. This is due to the OLED display not needing back light; and simply illuminating itself. The main advantage of the HTDS-WS96 however is the variety of voltages it can function at. According to the Diymall datasheet, the HTDS-WS96 can operate between 2.7~6 volts, and therefore is a great candidate for the BRAIN Helmet. With a voltage regulator of 3.3V, the Diymall HTDS-WS96 will operate without the need of a larger battery voltage and can be powered by a 3.7V Lithium Polymer battery.

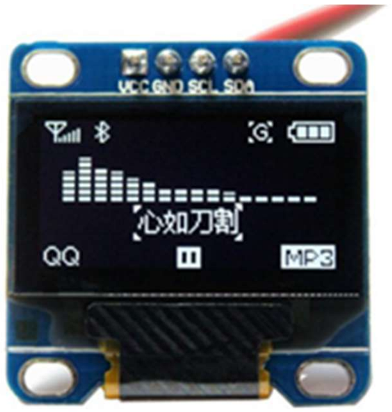


Figure 25 - HTDS-WS96 HUD

Permission pending

3.3.12 Speaker Selection

The B.R.A.I.N helmet is going to have a feature of being able to stream music and calls straight from the helmet to the wireless device, or the user's phone via Bluetooth. This means there must be speakers and microphone integrated within the helmet in order to achieve this goal. This means the speakers and microphone system must be able to:

- Fit within the confines of small space between the cushion of the user and the shells of the helmet
- Make sure not to tamper with the safe integrity of the helmet
- Have the wires not show when connected to the Bluetooth Module as for aesthetic reasons
- Work even if embedded behind a cushion
- Shall be able to hear the sound clearly from the speaker set, even over the loud motorcycle
- A clear and not static input into the microphone for clear calls and commands

When selecting different speakers and microphones we had to consider those conditions. There weren't many choices to go with since we needed a thin speaker for the inside of the helmet, and we needed to have not that sensitive of a microphone or it would pick up too much outside noise, i.e. the motorcycle itself.

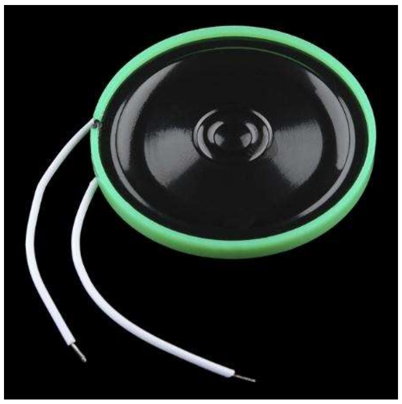


Figure 26 - Cover Industrial Co. Dynamic Speaker

Permission from an open source

This is a dynamic speaker that has an external diameter of 40 mm. The rated power input is 0.25 watts. The thinness of this speaker is really what attracted us to it. The speaker's thickness is at 4mm, which is astounding. The rated input power is 0.25 watts. The impedance is 8 ohms. This speaker is one of the choices we would go with due it its thin nature and its cheap when comparing to other speakers. This speaker is $0.95.

Another option if we wanted to go the more expensive route, is to buy a set of specifically built wired motorcycle helmet speakers and be able to just splice the input cable (since it will be a 3.5mm jack) from the motorcycle speakers. We would have to apply an external amplifier if it comes to this case. In Senior Design II we will prototype if it is better to have the motorcycle helmet speakers made ergonomically for the helmet or not. The amplifier would have to achieve a higher gain to allow more clear sound to be sent to the bigger speakers. When comparing this to the Cover industrial dynamic speakers, the system might be able to be heard better with the motorcycle ergonomic speakers.

3.3.13 Microphone Selection

The microphone selection now isn't set in stone but we have an idea of what we are using to prototype. The microphone we are going to use initially is the Sparkfun Electret Microphone for testing and design purposes. This electret microphone breakout has a built-in amplifier for the gain. However, during testing phase it is likely that the microphone will be soldered off and connected directly to the RN52 Bluetooth audio module. If during testing a separate microphone is instead selected, it will be very similar in size to this electret microphone due to its relatively cheap price; all while remaining effective for the BRAIN Helmet.

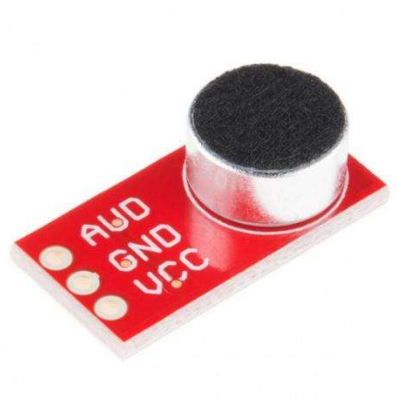


Figure 27 - Electret Microphone Breakout

Permission from an open source

This is a breakout of the actual mic that would be implemented into the PCB of our system. This microphone uses the Texas Instrument oPA344 rail-to-rail precision amplifier in order to achieve the gain it needs. This microphone has a built in 60 gain amplifiers before being set up to Microcontroller, in our case it needs to be biased due to this amplifier. This will be done by using 47 nF capacitors and will be discussed in the Design of the Bluetooth subsystem. This microphone is powered at 3.3V to 5V which means this microphone can be set up on the same voltage bus as the Bluetooth Audio Module. This would help when designing the PCB.

3.4 Possible Architectures and Related Diagrams

The design of the BRAIN helmet includes possible architectures of how to integrate the PCB onto the actual helmet itself. This section will include how we will being dealing with the implementation of the system directly onto the helmet as well as how the User interface will look within the HUD. The HUD will also have time and navigational assistance projected onto it.

3.4.1 Potential HUD GUI

Once the BRAIN Helmet's HUD and electronic group is installed, customization and personalization options to the HUD GUI are available. However, a stock HUD GUI is visible in the below figure and would include a navigation screen and a punctual time screen. These screens can be toggled at the switch of a button that will be located on the side of the helmet near future implemented volume keys. These stock HUD GUIs are subject to change in styles, however the BRAIN Helmet aims to still incorporate the same navigational turn and distance with accurate time updates.

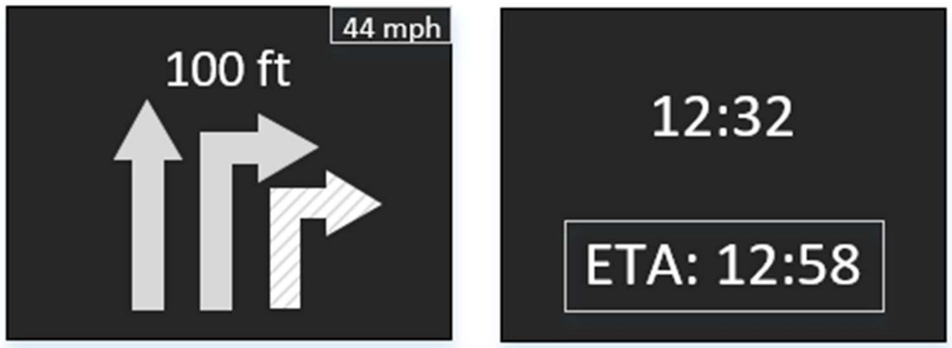


Figure 28 - Stock Navigation and Time HUD GUIs

<u>*Navigation*</u>

- Depict the motorcycle speed on the top right corner of the GUI
- Centrally display the distance before next turn
- Display turn signals of approaching turns by periodically blinking the lane that needs to be turned. In the case of the figure above, it is the right most arrow.

<u>*Time*</u>

- Centrally display the current time in military or standard time
- During navigation display the google maps given Estimated time of arrival (ETA)
- Outside of navigation, the display would not have ETA or a value displayed on the bottom of the display. Alternatively, a song title could be listed instead of the ETA.

3.5 Parts Selection Summary

This section of the report will depict and detail the parts selected for the BRAIN Helmet. Many of these parts will be visible within Figure 29. There labels and descriptions will be described in the following sections corresponding to the letters next to their part in Figure 29.

3.5.A RN52 Bluetooth Module and Breakout Board

The RN52 is a 3.0 Bluetooth Module that has high transfer rates and easy connect ability to android and iOS devices. This bluetooth module is perfect for our system size wise, when thinking in a broad-spectrum due to it going on a PCB that will be installed on the exterior of a motorcycle helmet. The breakout board RN52 was used for testing the whole system before building it into a PCB. This was very helpful due to troubleshooting and testing if the system or sub system worked.

3.5.B Speakers and Microphone

The choice for the speakers and microphone didn't impact the design as much as the other parts due to just having to be able to fit within a motorcycle helmet. The thin speakers with a impedance of 8 ohms, and the electret microphone that needs the same voltage as the other parts to operate. The microphone and the speakers will be integrated directly into the helmet. Each speaker will be implemented on both sides of the helmet within each ear cavity. A single microphone will be lined within the BRAIN Helmet and mounted near the mouth hole of the helmet in a place enabling little to none wind sound.

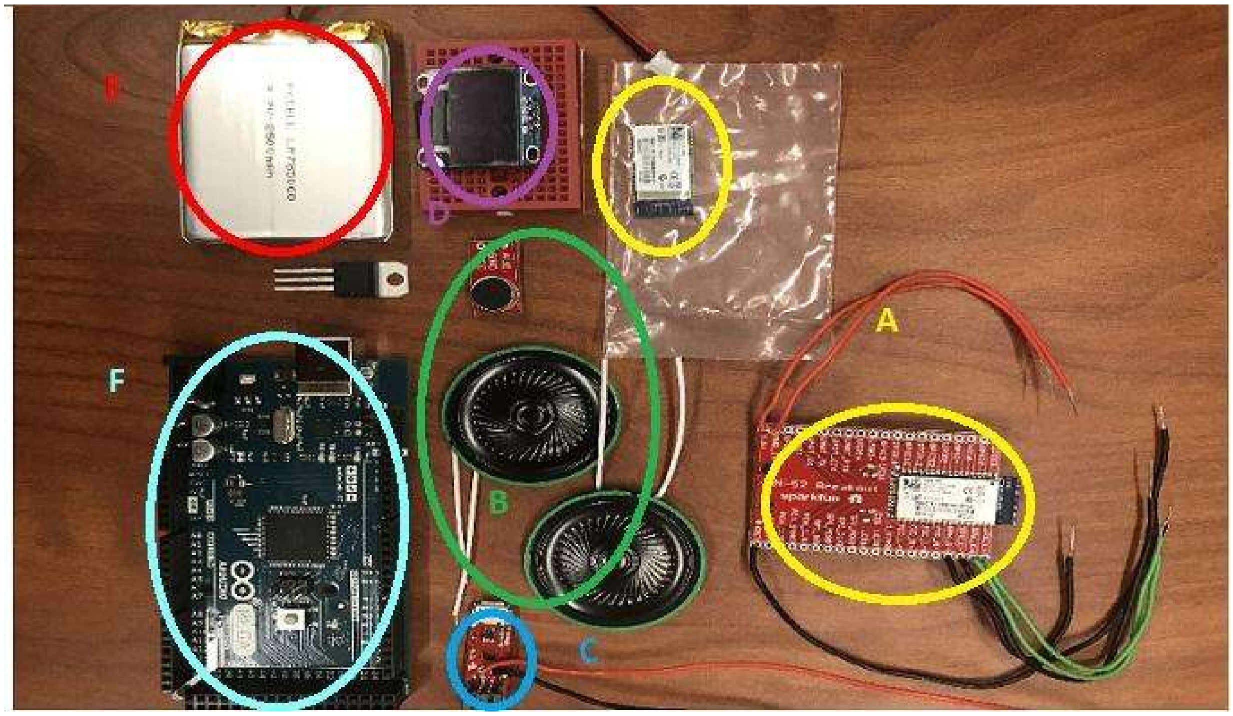


Figure 29 - Purchased Parts Laid Out

3.5.C Li-Po battery Charger

The battery charger we used was a standard breakout charger that allowed easy pull off wires from the positive and negative terminal from the lithium polymer battery. Additionally, it is easy to mount onto the BRAIN Helmet's housing and allows a micro-USB to charge the battery. The JST jack as well allows the possibility to switch our Lithium Ion batteries that standardly utilize JST connectors. This could be to replace a faulty battery, reduce, or increase the battery capacity for the BRAIN Helmet system. If the BRAIN Helmet is deemed to not need a 2500 mAh battery capacity battery, swapping the Adafruit 328 to a smaller battery capacity battery will allow the BRAIN helmet's electronic housing to be even smaller and more comfortable for the user.

3.5.D OLED Display

The OLED display will allow us to minimize power consumption than the typical LED display. With the selection of OLED displays small enough to attach to the helmet and minimize the field of view to the user made this a viable option to consider. We have decided to go with the Diymall Oled display available on amazon. With an economical price, replacing a cracked screen is also a viable option.

3.5.E Lithium-Ion Polymer Battery

The Adafruit 328 Lithium Polymer battery is a decent sized battery with enough battery capacity, 2500 mAh, to keep the BRAIN Helmet operating for an entire day. It's minimal size as seen in the previous figure will easily fit into the electronic group housing along with the PCB and be replaceable if needed to; thanks to the universal JST connector.

3.5.F ATmega2560

The reason why we pick the ATmega 2560 is to communicate to all of the modules that utilize different communications. There are plenty of GPIO pins available on this microcontroller just in case we want to upgrade the modules. This could include a better display, or any additional modules needed for future extra features on the BRAIN Helmet. It is also an advantage economically as the ATmega2560 development board is already in our possession.

4.0 Related Standards and Realistic Design Constraints

When dealing with modern technologies, the BRAIN helmet will need to be built to certain standards. This means the helmet will be using the following sections standards to the highest degree. The system will be implemented with all modern standards in mind in order to keep the BRAIN helmet safe and otherwise functional in modern day. Further standard changes in the future will induce an adapted design for the BRAIN Helmet if necessary.

4.1 Android Standards:

Android official programming language is Java. There are some additional rules and guidelines that Android demonstrated in order to ensure that Java coding criteria is met and considered. These guidelines are designed to give Android developers' good understanding of the rules that has to be followed to get a proper Java coding environment. There are some certain rules and guidelines for writing Java code that developers should follow and contribute to their code. Also, there are some bad coding practices that Android Java developer have to avoid.

Android Company has described these standards and guidelines that Android Java developer must understand and follow. These guidelines demonstrate good coding skills that developers must understand and follow. Moreover, bad coding skills are also demonstrated in order to be understood and avoided by all Android java developers. All the required steps and suggestions that programming developers need while writing their code is described with all required details. These details include also all information needed to solve some bad coding practices that developers must avoid.

1. Do Not Ignore Exceptions

Ignoring exceptions is one of the important things that needs to be handled while writing your code. Many programmers may think they can avoid this bad skill, but it is one of the most common things that developer must handle. When checking errors and report it, error can be fixed and user can know that something wrong has happened. For example, when parsing a string into an integer, the developer must check if the string is actually a number then if an overflow can accrue or not. Some acceptable alternatives are available to contributors such as: throwing the exception to the caller of your method, throwing a new exception based on your level of abstraction, and handling exception then substitute appropriate value in the catch block. Avoid throwing a runtime exception unless this error cause the program to crash.

2. Do not Catch Generic Exception

Avoid catching generic exceptions and throwable exceptions because it will be caught in application- level error handling. Including generic exception in the code obscure the failure handling properties of the code. This can cause compiler to miss notifying developer with any types of exception that need to be handled differently. Thus, it is a good coding practice to specify the type of exception might accrue in your code instead of just throwing generic exception to the compiler. The compiler need the type of exception in order to figure out how to solve the problem and look for available solutions. Some alternatives that developer should consider are: caching each exception individually, having more fine-grained error handling, and rethrowing the exception.

3. Do not Use Finalizers

Finalizers are beneficial when an object is garbage collected and chunk of code need to be executed. Android avoid using finalizers since no guarantees as when a finalizer will be called and executed. Although, Android does not use Finalizers but it is still can be implemented by defining a close() method.

4. Fully Qualify Inputs

There are two possible ways when a class bar is used from package for. These two possible ways include:

- The number of import statements should be reduced.
- Trying to make the classes that are actually used more obvious to maintainers so they can simply follow and analyze the code.

4.1.1 Android Activity Lifecycle

The new activities in an Android project are maintained as a stack. Each new activity is placed on the top of the stack and waits his turn to run. There are four important states for each activity on Android Studio as mentioned on Android Official Website. There are the following:

- Activity at the beginning of the stack which means it is an active Activity or a runing Activity.
- Paused Activity is alive until it killed because of the low memory problem of the system.
- A stopped activity because of uncertainty by another activity. A stopped acitivity can retains all state but it is no longer visible by user.
- The system can remove an activity from memory if it is paused or stopped.

There are three important loops that must be understood in each activity wich are: The entire lifetime, visible lifetime, and foreground lifetime. There are serveral figures demonstrating the lifecycle of an activity in Android Studio

Project from Android Official’s Website in order to give a clear picture of how to deal with activities and the important states of each activity.

4.1.2 Turn by Turn Notifications

One of the important features in our mobile Application is the navigation option that facilitate to the driver getting sound notifications by the attached speakers and direction notifications that will be displayed as signals on the LCD displayer that will be attached on the front of the helmet. The software team is planning to send these notifications to the rider when they become close enough to the turn or the exit that he or she must go through. The Bluetooth connection between Mobile Application and the main module should be good enough to handle all of these sound and string streams that will be sent to the driver. Also, Microcontroller should be set to receive these notifications and transform it to signals that will be displayed on the LCD displayer. Driver should enter his location and the address for the heading location and the application should calculate the direction and send the notification to the driver.

4.2 Java Style Rules

1. Javadoc Standard Comments

Describe the purpose of the class or Interface in the Javadoc comments. The copyright statement should be at the top of every file followed by package and import statements then class or interface description.

2. Write short Methods

There is no certain limit for methods length, but it is preferred to be short and consist. Programmer need to think about breaking up a method if it exceeds 40 lines without affecting program structure.

3. Define Field in standard places

The field can be defined at the top of the file or before the methods that will use them.

4. Limit Variable Scope

Reducing the local variable scope to the minimum is a good coding practice. Keeping local variable scope to the minimum increases programmers’ ability to maintain and read code. Local variables should be declared in the innermost block at the point where they are first used.

5. Order Import Statements

There are certain orders for import statement which are

- Android imports
- Imports from third parties
- Java and javax

6. Use Spaces for Indentation

Do not use taps for indentation instead use four space indents for blocks and 8 space indents for line wraps, function calls, and assignments.

7. Follow Field Naming Conventions

There are certain rules for Field Naming that have to be considered:

- Start with m for non-public and non-static field names
- Start with S for static field names
- Start with a lowercase letter for another field
- Use only caps letters for all public static final fields.

8. Use Standard Brace Style

Braces should be started from the line before, they should not be on the same line where programmer decided to use them. For conditional statement braces has to be on the same line around the statements.

9. Limit Line Length

100 characters should be the maximum length for each line of text in your code. There are two exceptions for having the maximum length to be 100 characters which are: if you have import lines or if the comment includes literal URL then it can go beyond 100 characters.

10. Use Standard Java Annotations

Annotation should be listed before other modifiers for the same language element. Simple annotations can be listed on the same line while if there are multiple ones they should be listed one per line in alphabetical order.

11. Treat Acronyms as Words

Treat acronyms as word and try to make variable, classes, and methods names easier to read.

12. Use TODO Comments

TODO comments is short term solution for temporary code. It is a recommended solution but still it can be used. In order to use it programmer should include TODO in all caps followed by a colon.

13. Log Sparingly

Try to keep logging consistent in order to keep it useful. Long logging has bad effects on performance. There are five different levels of logging which are ERROR, WARNING, INFORMATIVE, DEBUG, and VERBOSE.

14. Be Consistent

These global style rules are presented in order to have a common vocabulary of coding so people can focus on the main ideas of the code rather than how the code is presented as described in Android Official Website.

4.3 FCC Standards

Using Bluetooth devices, and other devices that emit radio frequencies, there must be some regulation due to the concern of health hazards. The Federal Communications Commission regulates the standards for Radio Frequency devices. A radio frequency device is defined as an electronic-electrical product that are capable of emitting radio frequency, energy by radiation, conduction or by other means. Such products have the ability to interfere in radio services operating in the radio frequency of 9kHz to 3000 GHz. Per the FCC, all electronic-electrical devices are capable of emitting radio frequency energy. Most, but not all, must be tested to demonstrate compliance to the FCC rules for each type of electrical function that is contained in the product. The rule for products, by design, that contain circuitry that operate in the radio frequency spectrum need to demonstrate compliance through the applicable FCC equipment authorization equipment. For example, the device must go through verification, and also Declaration of Conformity, in order to comply with the FCC regulations. There are certain types of devices that are regulated by the FCC and others that can be under multiple categories. This is difficult for the FCC sometimes, due to the multiple uses of one device. The following are the categories the FCC see as Radio Frequency devices that need to be regulated:

4.3.1 Incidental Radiators

This is an electrical device that is not designed to intentionally use, or intentionally generate or emit radio frequency over 9 kHz. This means that the electronic device may not be for a radio frequency purpose, but in the circuitry the device emits radio frequency energy of over 9 kHz. This type of product does not have to obtain an equipment authorization. It is regulated under the general operation conditions of the FCC, and if there is harmful interference, the user must stop operation and resolve the interference from the system. Examples of these types of electrical devices are AC and DC motors, and basic electrical power tools (tools that do not contain digital logic within).

4.3.2 Unintentional Radiators

This device by design uses digital logic, electrical signals operating at radio frequencies for use within the device, or sends radio frequency signals by conduction to associated equipment via connecting wiring, but is not intended to emit radio frequency energy wirelessly by radiation or induction. Some of the devices for example are coffee pots, cash registers, garage door receivers, and other common electronic-electrical equipment that rely on digital technology.

4.3.3 Intentional Radiators

This device intentionally generates and emits radio frequency energy by radiation or induction. This will include Bluetooth devices and Wi-Fi systems. These devices are the main reason these regulations were put in place to make sure consumers don't get hurt and also designers don't interfere with radio frequencies to the point it could hurt others.

4.3.4 Industrial, Scientific and Medical Equipment

The devices that produce radio frequency energy other than telecommunications such as production of physical, biological, chemical effects, heating, ionization of gases, mechanical vibrations, acceleration of charged particles. Medical devices aren't under this category unless specifically designed to use generated radio frequency energy for medicinal use. Examples of this are fluorescent lighting, arc welders, microwave ovens, and medical diathermy machines.

4.4 Radio Frequency Radiation Standards

Whenever a circuit has any voltage or current flowing through it creates an electromagnetic field around the material, whether it be a piece of wire or some other form of hardware like a capacitor or inductor. This radiation is moving at the speed of light around the piece of equipment. The different forms of electromagnetic energy are referred to as the electromagnetic spectrum. A radio frequency field both include an electric and a magnetic field, the intensity of a field is defined by units of measure. These units of measure are volts per meter and ampere for meter (volts for electric field and amperes for magnetic field). A radio frequency wave both have a wavelength and a frequency. The wavelength of a wave is the distance that is covered over one period to finish one cycle (due to waves being in the time domain). The frequency of a radio frequency wave is the number of waves passing at a given point in time. Frequency is measured by hertz (Hz). One hertz is equal to 1/s, which means once cycle per second. Where the radio frequency's frequency lands on the electromagnetic spectrum is in the range from 3 kHz to 300 GHz.

The reason why this is such a huge deal in safety, is that radio frequencies are mainly used in telecommunication devices. This includes your cell phone, television broadcasting, radio communication and satellite communication. Of course, these waves are passing through your body to get to their destination, but how much is too much? The quantity used to measure how much radio frequency energy is absorbed by a body is defined as specific absorption rate (SAR). It is expressed using unit of watts per kg or milliwatts per gram. With respect to a whole-body human exposure, an average human adult can absorb up to 80 to 100 MHz of radio frequency energy. Due to the body being able to only handle up to this point of radiation, safety standards must be put in place on order for the protection against bodily harm.

Effects of radio frequency radiation

The heating of tissue by radio frequency energy is called “thermal” effects. Excessive exposure to high energy levels of radio frequency energy could lead to the rapid heating of biological tissues. When there are low levels of radio frequency radiation, there is no evidence that there are any negative biological effects due to the energy. In most cases, such as living your day to day life of walking around and radio frequency waves passing through you from your cellphone for example, has no real effect on the temperature of the body. Although, there are situations where high-powered radio frequency sources in a manufacturing facility may cause harm to person.

In 1996, the World Health Organization established a program called the International EMF Project, which was designed to review the science concerning biological effects of electromagnetic fields. Other various countries have created their own exposure to radio frequency energy standards. For the Unites states, the Federal Communications Commission handles these affairs, ever since they were created in 1985. Other agencies such as Environmental Protection Agency, and National Institute for Occupational Safety and Health all have been involved with monitoring the effects of Radio Frequency energy exposure to the public.

4.4.1 IEEE 802.15.1 Standard for Wireless medium access control

4.4.1.1 What is IEEE?

The Institute of Electrical and Electronics Engineers, is an association dedicated to the innovation and technology. It is the world’s largest technical professional society. It is designed to serve professionals involved in all aspects of the electrical, electronic, and computing fields. This association helps create standards for these fields.

4.4.1.2 IEEE 802.15.1 Standard Security

This standard was made for wireless personal area networks (WPAN), which are used to convey information over short distances among a private, group of participant devices. The reason this standard must deal with our B.R.A.I.N. Helmet, is because of the use of Bluetooth technology, which is a form of WPAN. This standard defines the physical layer and medium access control specification for the wireless connectivity, for fixed, portable and moving devices. To protect the peer-to-peer communications between the Bluetooth module master and slave, there must be some way to secure the information. The standard for Bluetooth security is embedded into the application and link layer of the system. Due to this being used for a peer environment, each device must have an authentication and encryption routines that are the same for all. There are 4 different procedures for security when it comes to IEEE 802.15.1 devices, and is described in the following table:

Table 11 - Procedures for Security

Procedure	Size
BD_ADDR	48 bits
Private user key, authentication	128 bits
Private user key encryption length	8-128 bits
Random Number	128 bits

When analyzing the secret keys both for authentication and encryption, the keys are derived during initialization. The size for the authentication key is always 128 bits, but for the encryption it varies due to the country you are living in. Increasing the key size, will increase the security of the key. The encryption key is completely different from the authentication key, but one is used to make the other. The keys are usually static values, unless the devices connected wants to change it, whereas when comparing to the random number. The random number generated is dynamic is continually changes for security purposes. Usually, the random number is nonrepeating, which means in the lifetime of the authentication key, the value of the random number will never repeat. Randomly generated number means that is impossible to predict the value of the number. Key management within the system is usually set by the factory design presets and not by the user. There are two types of keys, semi-permanent and temporary. The semi-permanent key could be stored within the system and be used after the current session is done or power to the system is gone. This means once the system is done using that key for a session, it could be used again after, such as remembering a device that was connected to the system before. A temporary key is limited to the session it is created within. Once the system is done with that key, it will not be saved into nonvolatile memory.

4.5 Motorcycle Helmet Standards

The standards regarding to the design standards of the helmet we are restricted with regards to the safety of the helmet is detailed in this section. In one of the standards we cannot modify the helmet or the integrity of the helmet will risk the safety of the user.

4.5.1 Department of Transportation

From the department of National Highway Traffic Safety Administration, FMVSS part 218, which establishes a standard minimum performance requirement for helmets designed for motorcyclists. The safety integrity of the helmet is a big concern when dealing with our project, due to us making changes to the outside and inside. The DoT and NHTSA have established safety requirement standards when it comes to motorcycle helmets. This standard is applied to all helmets designed for motorcyclists. In order for a helmet manufacturer in the US to get the DOT sticker, it must pass these certain standards. When testing a helmet, the impact attenuation must be in accordance to the following: Peak accelerations shall not exceed 400g, accelerations more than 200g shall not exceed a cumulative duration of 2 ms, and acceleration more than 150g shall not exceed a cumulative duration of 4 ms. When a penetration test is conducted, the striker shall not contact the surface of the test head form. Configuration of the helmet, must have a protective surface of continuous contour at all points on or above the test line. The helmet shall provide peripheral vision clearance of at least 105 degrees. These standards are put in place to keep the manufacturers of such helmets on check, so that there is a standard of safety for those riding motorcycles.

4.5.2 Snell Memorial Foundation

The Snell Memorial Foundation is an organization that sets higher standards for motorcycle helmets than the DOT does, which means if a manufacturer were to get a Snell certification on their helmet, it would have a greater safety rating. The Snell standards are said to the toughest standards in the world. Both standards (DOT and Snell) both test their standards the same exact way, just the passing specifications are much harder when it comes to the Snell’s requirements. The helmet is tested by dropping the top of the helmet (also known as its head form) onto a fixed steel anvil. The test is repeated on at least 4 different areas on the helmet against either a flat or curved anvil. The amount of damage the helmet takes against these impacts is measured in mechanical energy. This mechanical energy or impact energy is measured in joules and is one of the standard specifications that the helmet must not exceed. The following is a table of how much Impact standards are when comparing a DOT and Snell Standards.

As you can see from the table, the standards for the Snell Foundation are more rigorous than the DOT. Which means helmets with the Snell certified sticker have a higher safety rating when it comes to standards of crashing. When choosing a motorcycle helmet, using these standards helped get a grasp of how the standards affect the safety integrity of the helmet.

Table 12 - Comparing Standards of Motorcycle Helmets

Item	DOT FMVSS 218	Snell M-95/M2000
Impact Severity		
Flat Anvil	Small-63 Joules Medium-90 Joules Large – 110 Joules Fall of about 1.83 Meters	All sizes 150 Joules Fall of about 3.06 Meters
Curved Anvil	Small- 47.3 Joules Medium- 67.6 Joules Large- 82.5 Joules Fall of 1.38 Meters	All sizes 150 Joules Fall of about 3.06 Meters
Impact Criteria		
Allowed Peak Acceleration	400 G	300 G
Allowed Duration requirement	2 ms over 200 G 4 ms over 150 G	N/A

4.6 C++ Standards

C++ was not initially standardized when it was first created by Bjarne Stoustrup at Bell Labs in 1979. This extended language provides high-level features for program organization. C++ standardization is controlled by the International Organization for Standardization (ISO) starting from 1990 for the ISO/IEC 9899:1990. This borrows the C programming language and adds onto the language more features like additional data types, templates, exceptions, operator overloading, references, and additional library features. The standard introduced in 1990 will be briefly explained the most important features that is important to our system.

4.6.1 ISO/IEC 9899:1990

4.6.1.1 Implementation compliance

To comply with the programming language, it has to follow the rules established by the International Standard. For example, if a program contains a violation while compiling it will output one diagnostic message, except if it contains a violation for a rule that does not need to display a message, the International Standard does not make a requirement on how to implement with respect to the program. If a program contains no violations of the rules from the International Standard, then an implementation should, depending on the hardware capabilities, should accept the program and will execute the program correctly. Their definition of correctly executing the program can include undefined behavior based on the data being processed. For classes and their templates, they can specify partial definitions. Each implementation of the libraries should be complete based on the definitions on the library clause. Implementations for functions, objects and its values it should supply definitions consistent with the library clause definitions. The scope of the variables is based on the permissions the C++ includes in the program. There are two types of implementations, hosted and freestanding. Hosted implementation is defined by the International Standard of the available libraries it has access to. For freestanding implementation, it does not need to execute the program with dependence to the operating system and has its own set of libraries needed to run the program. An implementation may have extensions if they do not alter the behavior of a complete program. They are required to diagnose programs using these extensions based on the International Standard.

4.6.1.2 C++ Object Model

C++ can be able to create, destroy, access and manipulate objects. It is created by a definition, new-expression, or by implementation if needed. The properties of the object must be defined when a new object is created. When an object is created, it can have a name, storage duration based on its lifetime, type, and polymorphic. The implementation will generate information associated with the objects in focus to make it possible to determine the object's type during the program execution. Each object can be associated with another object within its class, from another class, or an array element.

4.6.1.3 ISO/IEC 14882:2003

This standardization was replaced in 2003 for the C++03 version named ISO/IEC 14882:2003. This version was updated to ensure greater consistency and portability for implementers. It addressed language defect reports, library defect reports, and introduced value initialization.

4.6.1.4 ISO/IEC 14882:2011

In August of 2011, C++03 standard is replaced by the C++11 version based on the ISO/IEC 14882:2011. The core language improved by including multithreading support, generic programming, uniform initialization, and increased performance.

4.6.1.5 ISO/IEC 2014 and 2017

The most recent standard in 2014 supersedes the C++ 2011 version by expanding the standard library. This version was not significant enough to be officially standardized in 2014 but in the 2017 version it will officially be standardized when it finishes later this year. In the 2014 version, it extended the language to provide high-level features for program organization.

4.6.2 Design impact of relevant standards

The impact of the standards that are listed above affect our design of the B.R.A.I.N. helmet are the standards for RF devices and the coding standards. The design of the helmet has a bluetooth module within its system, so when analyzing the standards, we must make sure that the individual subsystems followed them. It didn’t affect the design as much in a negative way, because the bluetooth module we purchased (RN52) already met all the FFC and IEEE standards.

As for the coding standards, the C++ standard that we are using to program doesn’t affect our system negatively unless we want to use a feature on a non-standardized library. We are strictly using the most recent approved standard for compatibility with the compilers and the IDE.

4.7 Soldering Standards

When connecting different electronics together either on a PCB or in general, the connection between them must be conductive, which comes the term soldering. NASA, has made a technical standard document “Soldered Electrical Connections” which defines a certain standard when it comes to these connections between electronic parts. This standard is needed when dealing with PCB’s or any electronic system in order to have no faults within the system as well as a basis for systems that will last.

When dealing with a printed circuit board, there are through holes in which your chipsets such as in our project the RN52 Bluetooth Module would fit. Then there would be connections from the RN52 onto the PCB itself. There should be no stress relief within the leads from all components of a system. This means there should be no constraints and when soldered to the board there is no possibility of movement. When soldering onto a PCB the joints must not be subject to stress. When positioning parts onto a PCB, it must be taken into account that no other parts are obscured with others. Other components that have a conductive casing should be never be mounted in close proximity to other conductors due to the chance of conductivity. The visibility of markings wherever possible such as a part type, value of part, etc. is a must when dealing with PCB, due to replicability. Polarity is the most needed marking, then Traceability Code, and then value and type of piece, when it comes to which type of marking should have precedent over others. Glass encased parts such as diodes, and resistors, will be covered with transparent resilient sleeving, such as conformal coating, or encapsulating where damage from other sources is possible. When the connection between a PCB and an exterior part such as a microphone, a hookup wire should not exceed the length of 1 inch unless being soldered to the board.

Mounting of terminals in this standard must be made sure to check the terminals of the PCB before soldering the parts together. The radial split of the terminal should only have 3 cracks and must be separated by at least 90 degrees, if not this will lead to faulty connections within the system. The parts of a system will be mounted parallel to, and in contact with the mounting surface. This mean when connecting to the PCB it should be flush horizontally. The lead lengths of both sides of a part (such as a diode) should be equal on both sides before making the soldered connection. There is a difference in mounting onto a PCB when the through hole is plated or not. Most likely when dealing with our system, we will go with the plated through hole for ease of use purposes. This means when soldering we do not have to have a stress relief bend in order to make the connection non-faulty.

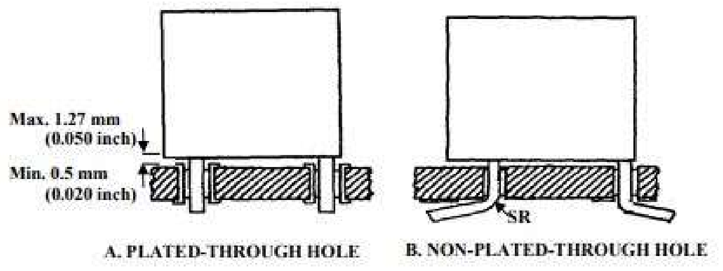


Figure 30 - Plated through Hole vs Non-Plated permission submitted to reproduce]

Permission to use from Open Source

Designing and implementing parts onto a PCB, only certain parts go in one through hole. Never shall there be two parts into more than one hole, this can lead to a shunt, and lose connection or worse, loss of usability of a device. Other non-acceptable behavior when it comes to mounting onto some PCB from NASA's standards are never to use pressurized air onto a freshly soldered board because this can lead to fragile solder joints or improper connection between the parts and board. In order to allow for a good connection to be made the solder should be left to cool at room temperature.

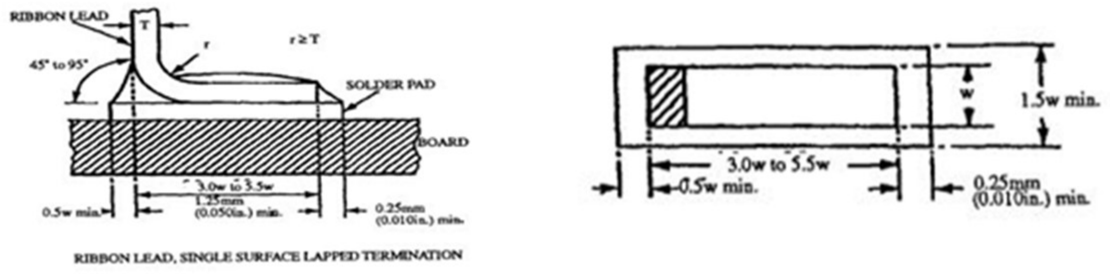


Figure 31 - Ribbon Lead Measurements permission submitted to reproduce

Permission to use from Open Source

In the final part of the PCB mounting, is the single surface lapped termination section of the standard, where it talks about how to connect a ribbon lead to the solder pad on a PCB. The solder pad should be a minimum 3 lead widths to a max of 5.5 lead widths. The only portion that should beyond the pad is if it will be connected to another part of the system if not it should not exceed the pads length. The cut off of the lead shall be a minimum of 0.25 mm from the end of the soldering pad. The solder fillet should be able to fit around at least three lead edges of the pad, that's why there must be extra space on the pad. If not the connection between the part and the solder pad may not be good enough to be conductive. The above figures show the measurements of what is allowed when it comes to soldering pads and a ribbon lead terminal.

This standard explains how to mount onto a PCB with proper soldering techniques. This standard will be used when creating our PCB in the near future. In order to have viable connections between the electronics parts and the board, this standard will be followed. The need for this standard in our system is one of the biggest standards to be followed because if not followed correctly the system could not work, all due to a faulty connection between two parts.

4.7.1 Lead Solder Safety

When it comes to PCB implementation, to connect different parts onto the board, solder must be used. In our case, Sn40/Pb60 was used, which means 40% tin and 60% lead. There must be safety precautions when dealing with this material due to health hazards in can cause due to the high temperature it is heated to. Care and safety must be taken into consideration when dealing with lead and tin in this situation.

Lead is known to be a neurotoxin and can lead to reproductive, digestive, memory and concentration problems. Lead could also lead to muscle and joint pain. There for when dealing with solder, such as ours that is 60% lead is considered to be toxic for your health. When dealing with solder precautions must be taken or else the handler can be inadvertently exposed to this "poison." The risk of health issues greatly reduces when the handling of solder is treated appropriately.
The biggest risk of exposure to solder is the ingestion of lead due to surface contamination. In order to fight the risk of lead getting into your digestive system, gloves can be worn or washing your hands before eating or touching of your face, etc. This must be taken into consideration due to how often engineers don't see lead soldering as a big issue due to most of our group not worrying about the residue that could be left on your body without you knowing. Chewing fingernails is the number know overlooked habit that causes lead poisoning after dealing with solder. The reason for lead poisoning to be so dangerous is due to the body's ability to excrete it from the system is almost nearly impossible on its own. Remember to always be careful when it comes to lead soldering because health complications could arise.

Another risk when soldering is the actual physical heat that is being dispersed into the solder to melt it. The solder that we are using will be heated to around 400 degrees Celsius, which as you know extremely hot. When the electrical engineer majors from the team deal with soldering onto the PCB, they must take into account how hot the soldering needle is and make sure to always return it to its stand before soldering another piece onto the PCB. If not, someone could get burnt or a faulty soldered connection could be made. The soldering of the prototype of the breakout boards to each other will be the biggest part of soldering in our project, we must be very cautious to not burn ourselves or others when handling the solder equipment.

Including the above risks, another is soldering with lead or any other metals, may produce fumes from the heating of the metal. These fumes can be very hazardous to one's health. If inhaled it can result in occupational asthma or create worse asthma conditions. The fumes from the solder can also cause eye and respiratory tract irritation. When dealing with the fumes a mask and goggles could be used in order to safely handle the fumes and make sure not to inhale whilst soldering the parts onto the board. An exhaust or good ventilation while soldering can make a huge difference when it comes to the fumes of the system.
The following will be a list of general safety precautions that should be taken to dampen risk of injury (Carnegie Mellon University):

- Do not touch the tip of the soldering iron. It will be heated to around 400 degrees Celsius.
- Wires that are to be soldered, to be held with tweezers or pliers to avoid burns of objects being heated
- When dealing with the soldering irons itself, make sure the sponge that it will be resting on is wet during use
- Use a level surface when soldering, to avoid from imbalanced solder connections
- Never put solder iron down on workbench, always return it to its stand
- When not in use, make sure to unplug soldering iron
- Wear protective eye gear in order to protect from splatter of heated solder
- Use low lead percentage or lead free solder when possible
- Always wash hands after dealing with soap and water and solder
- Work in well ventilated areas to avoid inhalation of toxic fumes
- Solder on fire proof surface that is not easily ignitable.
- If a situation arises where you were burned, immediately cool the affected area under cold water for 15 minutes. Do not cover with cream or ointment.

- In order to manage the waste of lead solder, you must discard it in a container with a lid. This container must be metal and labeled hazardous material.

In order to safely solder parts together we as a team use these safety guidelines in order to not injure ourselves in the process. We take safety seriously and must respect these rules in order to have a good product in the end, whilst not hurting ourselves or each other.

4.7.2 RoHS Compliant

The RoHS stands for Restriction of Hazardous substances and this compliancy impacts the entire electronics industry as well as many electronic devices. The original RoHS was a directive that originated in the European Union in 2002, that restricts the use of 6 hazardous materials found in electrical and electronic products. All products in the EU market must pass this compliance, so if we were to ever sell our product overseas our device would have to be RoHS compliant. Electronic products from July 1, 2006 and onward must be RoHS to be sold in the EU. Only a certain amount of work areas is affected by this compliance, basically places that sell electrical or electronic products, equipment, cables or components. The specificity of the RoHS is so precise is that it doesn’t allow max level of certain hazardous materials, and they include but not limited to Lead, Mercury, Cadmium, Hexavalent Chromium, Polybrominated Biphenyls and so on. The main reason for bringing this compliance up into this paper is the fact that the compliance must have less than a 1000 ppm. This means that there must be less than a thousand lead parts per million. This means we couldn’t be able to use the 40% 60% Lead solder previously spoken about in the above section. The use of a non-lead solder would lead to faulty solder points in the system and would degrade our PCB system heavily. The only concern about RoHS compliant is if we were to try and sell our product across the Atlantic, the product would have to be soldered entirely different.

4.8 Realistic Design Constraints

The initial cost of the project could be a setback when looking at the project. The lack of experience in designing and building a PCB could be a potential roadblock in the future. Time, is a constraint for everyone as we must finish this project in the allotted time. A constraint that could come up would be how to mount the PCB onto the helmet as well as integrate the microphone and speakers into the helmet without compromising the safety of the helmet. The following will be a list of what may come or has come to be a constraint when designing the B.R.A.I.N. Helmet. These constraints are mainly the issues that came up whilst the designing phase of this project. Through the knowledge, we learned to battle these constraints, made the project have an overall better quality.

The following few pages will discuss the different kinds of constraints that can have important effects on our project. Constrains are some conditions that a design needs or would like to have. These constraints will place some limitations on the requirements and operation conditions under which our project can be functioned. These limitations include but not limited to: size of the final device, features that must be included, and environment of operation. Some of these constraints can change the shape of the design based on the budget limitations, or the technology of the design might change based on the environment of operation. Constraints can make certain things available in the design to guide the user to perform some certain interactions. There are many types of constraints that must be considered such as:

- Economic
- Time
- Health
- Safety
- Ethical
- Social
- Political
- Environmental
- Sustainability

4.8.1 Economic and Time constraints

When first designing, and coming up with the parts for the B.R.A.I.N helmet, the economic constraints came and go. Due to our group wanting to keep our project as our own, we had to fund our project out of our own personal funds. This made choosing hardware for our project a little difficult as we didn't want to increase the cost of the project too much. We all equally paid for the parts so we all own an equal percentage of the project. Due to the limitations that we have in our budget, we have reduced the quality of parts that we have to elaborate into our project.

As we all know, everyone works and go to school full time so meeting up and working as a group is a constraint. The way to work around this is to always keep communication with the team and always update as soon as possible if anything bad came up. When coming up with times to meet up, the progress of our design started smoothly. As you know, time and finishing the project in a shorter semester could have a huge effect on the final product, but as long as we kept the work up and no procrastination, nothing is impossible.At the end of senior design two the design, prototyping, implementation must be completed and the final device must be completely functional in order to meet engineering requirements. To ensure that everything works as supposed to be, team members should follow the plan and the timelines demonstrated on the milestones section. These tables that are shown on the millstones section list all the important tasks that have to be completed and the competition dates for these tasks.

4.8.2 Environmental and Social constraints

Environmental and social constraints are extremely important for our device. Our device will be used outdoor for motorcycle's drivers, so it has to meet some requirements. Our device consumes an average power that is very small and should not have bad effects on the environment. Also, it should increase the safety for motorcycle drivers which will decrease accidents and save our environment. For social constrains our project is designed for motorcycles' drivers. Motorcycles have the same design all over the world so our project does not favor one group against the other. Overall our project will cover all environmental and social to give our users better experience from other products and options.

4.8.3 Health, and Safety constraints

There is always a need for concern of health and safety when it comes to handling electronic devices. When building a circuit, we must consider some safety guidelines to follow. Never to disobey lab rules when working on a circuit in the lab, and always to follow directions in case of an emergency. There come standards when working in the lab at the University of Central Florida, we must take note of the Emergency Disconnects located in the room. We must make sure to report any broken equipment to the lab instructor that is present. Taken into consideration we were using the school's equipment, we must make sure to not destroy or break any of the lab equipment.

The main purpose of our project is the safety of motorcycle drivers which will also increase the safety of the surrounded people and vehicles. Thus, one of our project priorities is the safety which is taken into account by using high quality materials as much as possible based on the project budget. Driver safety increases by using our product since navigation notifications is sent to the driver and displayed on the LCD displayer so the driver does not need to be distracted by checking mobile device, Moreover, sound notifications will be sent to the drivers to notify them for the close destination directions.

Moreover, all the parts is designed and placed on the most practical positions that maximize the safety and usability of the device. The size of the LCD displayer has been researched and studied to best fit the size of the helmet and the distance between the driver eyes and where it should be placed so driver can catch the signals without being distracted by the size of the displayer.

Also, all the electrical parts of the product have been checked and placed carefully. All the wires and battery attachments are carefully placed and attached together in a good position to maintain the driver safety.

4.8.4 Manufacturability Constraints

One of the important constraints that we have considered is the availability of chosen materials. Our team has considered the duration of availability of the required material and we also have taken into account the expected time for those materials to be available on the market. In order for our project to be continued as we have designed it, those materials have to be available for users on the market. Otherwise, users have to do some design changes on order to integrate new parts and technologies to our main idea.

Another constraint to take into account for our project is the availability of the required and planned parts when start implementing the product. Sometimes some of the parts are discontinued, no longer available on the market, or need a long time to arrive. Thus, as our advisor suggested we have purchaes all the required parts for our project, Also, we have purchased more that what we need for example we have purchased three microcontrollers in addition to some extra parts from each picea we have in order to be on the safe side.

Since the B.R.A.I.N Helmet is supposed to be help motorcyclists while they are riding, the helmet must also be functional as a helmet still. We had to make sure we didn't drill any holes in the helmet our tamper the physically integrity of helmet to make sure that the helmets original purpose was still intact. When considering where to position the PCB and speakers and microphone system on the helmet without breaking any of the regulations of the safety of the helmet was a constraint.

4.8.5 Political Constraints

Political constrains covers those issues related to:

- Products that are designed to outline the downsides of a specific race or gender
- Projects that are supported by public fundings
- Products that are designed to be used in contradiction of the United States of America
- Products that are designed to be used against the homeland security of the United states of America
- Products that are mentally or physically destructive for users.

These are some of the issues that are considered by political constraints. After doing an extensive research we found that political constraints are not applicable to our project. Thus, any political constraint should not be relevant to our final product.

4.8.6 Sustainability Constraints

The sustainability for products lie on some factors that include but not limited to environmental factors, business survival, reliability and robustness of the product’s supposed function. Our B.R.A.I.N Helmet’s sustainability goal is to be able to function under the assumed normal operational conditions for a life span of at least about 6 hours. Our system should function more than six hours since it does not consume too much power. The environmental conditions such as rain, wind, temperature, and hail need to be taken into account for this constraint. The quality of material for the integrated parts need be good enough for the system to function as proposed under different environmental conditions. However, waterproof feature will a future designed improvement that we are planning to add. For us as computer and electrical engineers the main design is the important for us now and waterproof feature can be added easily later in future improvements.

To be effectively functioning on the motorcycle, system should be successfully on the Helmet. The weight of the device should not be an issue so the driver should be comfortable while driving and is able to use all the provided features easily and safely. The LCD displayer should also be placed in a position where is easy for the driver to see the signals without being distracted from driving. All parts of the system will be placed inside a little box which will be located in a position that is best for the driver safety and usability.

Moreover, the demand of the product based on how long it will be valuable in the market. The lifetime of the product bas on the manufacturability material is important but also the demand for the product in the market is an important constraint that must be considered too. In the market safety is one of the most important things not only for consumers but also as one of the most important business values. B.R.A.I.N Helmet’s product is designed to make every person and vehicle on the road safer in addition to the main purpose of our project which is keeping motorcycles drivers’ safer. Thus, our product should be one of the most demanded and suggested products in the market since it increase safety not only for motorcyclist but also for all drivers on the road in general which will attract all drivers and most of motorcycle’s companies.

4.8.7 Ethical Constraints

One of the most important priorities of this project is to increase the safety of the motorcycle drivers in addition to all people and vehicles on the road. All the integrated parts on the system have been chosen carefully after doing an extensive research. Our team has decided to avoid using any material that are potentially toxic, radioactive materials, or affect the lifetime of the product. When it comes to choosing our project material, we have considered quality as we considered the price. We have not decided to choose any part that has a bad quality of bad materials in order to decrease the amount of our project budget.

The safety of our team during developing and implementing the design has been taken into consideration seriously. Our team always try to follow all the safety instructions that are demonstrated in University of Central Florida's labs. Also, when testing our final product on the road, all driving safety rules will be considered.

Moreover, our team avoids using or inferring any existing patents. This has been accomplished by doing an extensive research to make sure of all existing project that are similar, and to give credit for any work that is done by some other people.

4.8.8 Software Prototyping Constraints

Software Prototyping is basically designing a prototype for the software application. This should give us a basic idea of how the final design should look like. There are many constraints that should be considered when designing a software prototype. Developers should avoid focusing on the prototype more than the actual design. Prototyping sometimes distract the contributors from doing the actual work that they have to do. Developers should also avoid user confusion between the final design and the prototype. Sometimes users misunderstood by thinking that prototype is the actual system that will be implemented. Also, sometimes the developer misunderstood user needs by proposing that all users share the same needs and objectives. Developers should schedule meetings with consumer to evaluate the effectiveness of the user interface and discuss potential alternatives to the proposed system to consider all their needs and expectations for the final product. Moreover, developer should avoid spending too much time on the prototype and start working on the actual system. Also, designers should avoid implementing expensive prototypes since prototypes are designed to give users a very brief idea of how the final system should look like. For our project, Software team has designed simple prototype including the main features that must be offered for our project users. We have designed the prototype in the early stages of our project after we have decided what is the final product for our project. Our team did not spend too much time on it because it is just a basic picture of the final design. Also, it was free it did not cost us money since it was designed on Android Studio which is available for free for all Mobile Application programmers on Android Official Website. Finally, after the prototype was designed and all team members agreed on it, Software team has started working on the actual code for the Software Application.

4.8.9 Testing Constraints

Our project is designed for people who drive motorcycles, so they can drive more safely by avoiding using their mobile devices while driving. To test our design we need a motorcycle, no one of our group members has a motorcycle. Our team has already discussed this issue and one of the group member has suggested borrowing his father motorcycle. Also, our advisor suggested testing our device on a bicycle if needed. The team will test the device on bicycle as many time as necessary since the availability time of the motorcycle is limited. Thus, our team can avoid this issue by doing the necessary test on the bicycle and schedule every three weeks one appointment with our friend father so we can test our design on a motorcycle. One of our group members has a good experience for driving motorcycles so he will drive it and do a real-world test for the design. The real-world testing is very important and needs to be done as many times as possible a decent enough time before the final demo so we have enough time to fix any problems. Any new project and design will have some problem that will be discovered during testing time, but nothing is impossible if we figured out these problems before the final demo we can work as a group and fix these mistakes to submit a well-finished design at the end.

5.0 Project Hardware and Software Design Details

As the hardware and software research stage of the BRAIN Helmet project drew to conclusions for methods and components to create the BRAIN Helmet, parts were ordered and the System Design stage of the project immediately began. The hardware team concentrated on finalizing a regulated, mobile, and rechargeable power system that would be able to properly power all components of the BRAIN Helmet's electronic group. Additionally, audio amplification and signal clarity post Bluetooth module communication was focused on in an attempt to create a speaker and microphone system that would satisfy the BRAIN Helmet user. On the other side of the design implementation, the software team focused on developing an android specific application for interfacing the BRAIN Helmet's electronic group via Bluetooth to communicate to a user in the form of audio and a heads-up display for safe navigation and recreational uses. Implementation of the B.R.A.I.N. helmet will roughly follow the block diagram shown in Figure 32 below. The work incorporated into these components per person will be detailed in the Administrative Content section of this report.

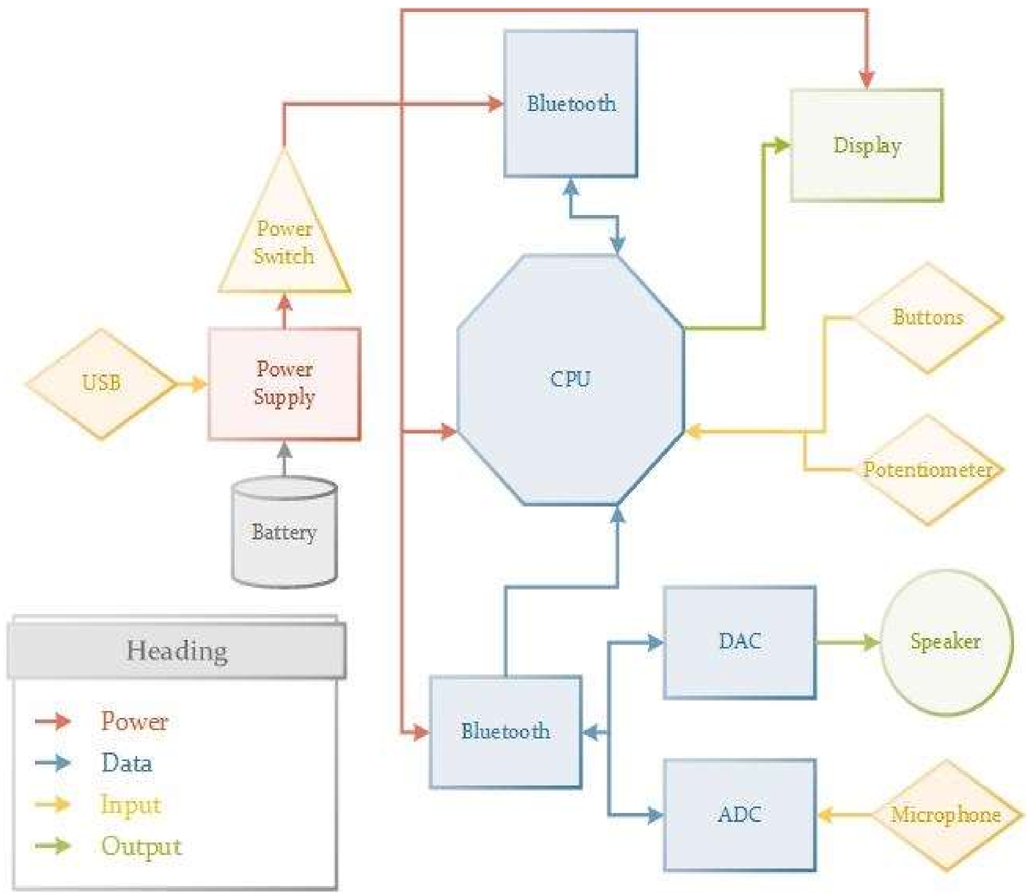


Figure 32 - BRAIN Helmet Component Block Diagram

The BRAIN Helmet Electronic group consists of a battery, a micro-USB to JST battery recharger, Bluetooth modules, the ATmega2560 microcontroller, a microphone, two speakers, and an OLED heads-up display. The BRAIN Helmet's PCB will hold the Bluetooth module, ATmega2560, battery, and the micro-USB Lithium-Ion Polymer battery recharger. Pending decisions will decide whether to use Bluetooth modules with built in DAC, ADC, and filtering. The Bluetooth functioning PCB will be located on the back of the helmet and connect to 2 speakers and a mic as depicted in Figure 29.

5.1 Initial Design Architectures and Related Diagrams

The following two tables are the stock pins initially used in the design of the BRAIN Helmet. These are the pins that will be interconnected as according to the final schematic (figure 42). We chose a microcontroller and Bluetooth Module capable of adding features at a later period due to extra GPIO pins, so this table is subject to update in the final design of the BRAIN Helmet.

Table 13 - ATmega2560 Pins

ATmega2560	
Pin Number	**Signal Characteristic**
10, 31, 61, 80	VCC
11, 32, 62, 81, 99	Ground
20	Bluetooth SCK
21	Bluetooth MOSI
22	Bluetooth MISO
30	Power Switch
51, 52	Hardware to Flash the Program to Memory
53	Display Analog Port

Table 14 - Used RN52 Pins

Bluetooth Module RN52	
Pin Number	**Signal Characteristic**
1, 18, 27, 44	Ground
32, 33	Two Debugging LEDs
40, 41, 42, 43	Speakers
34, 35, 36, 37, 38	Microphone
21	Power Switch
22	3.3V Power
31	MCU MOSI
30	MCU CLK
29	MCU MISO
20, 2, 6, 7, 8, 9, 10	MCU GPIO
4	MCU Analog GPIO

5.1.1 Helmet Electronic Group Housing

The BRAIN Helmet shall contain a single electronic group mounted onto the back side of the helmet in an aerodynamic approach by using a 3-D printed case. This case will hold the primary components of the BRAIN Helmet’s Electronic group; allowing the micro-USB jack to be exposed to the user for recharging the BRAIN Helmet’s battery. The housing will additionally allow buttons to be exposed for the user. Other components such as the speakers and microphone will be embodied into the helmet. In order to keep an aerodynamic approach, the electronic group housing will breakout directly into the helmet and the speakers and microphone wires will be lined through the helmet’s padding into their appropriate locations for the users. Similarly, the wires to the Heads-up Display will be lined through the helmet padding to allow the HUD to be at an appropriate position for the user.

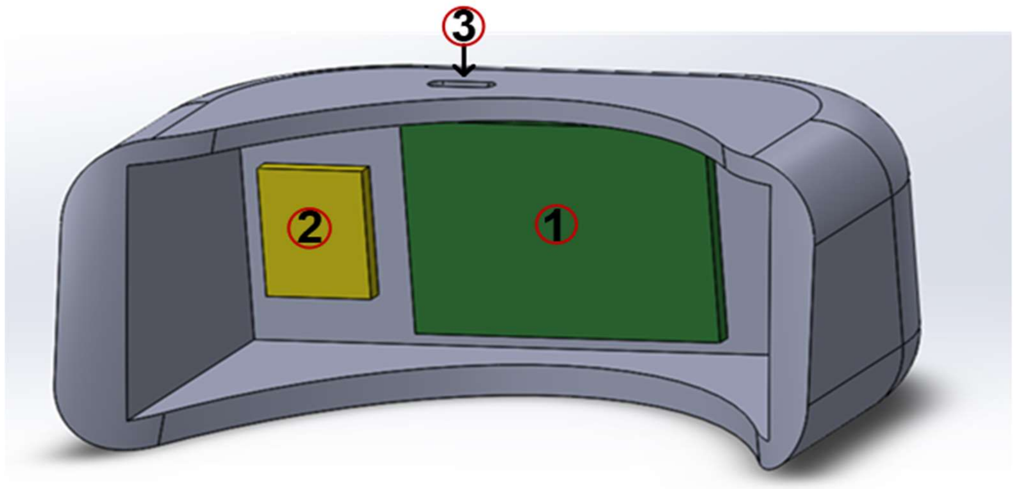


Figure 33 - Prototype housing to attach to helmet

The above figure depicts an unofficial rendering of the electronic group housing we would have 3-D printed for the BRAIN Helmet. The PCB (noted 1 in Figure above) and the battery (noted 2 in figure above) will be the primary components contained in the housing. Noted as a number 3 in the above figure is the earlier described micro-usb input port to charge the battery. Testing will decide whether or not the 3-D printed plastic housing can be fully enclosed or whether the bluetooth module antenna will need to be exposed.

5.1.2 Power Supply Layout

The following figure is the layout for the Power supply unit powering the BRAIN Helmet.

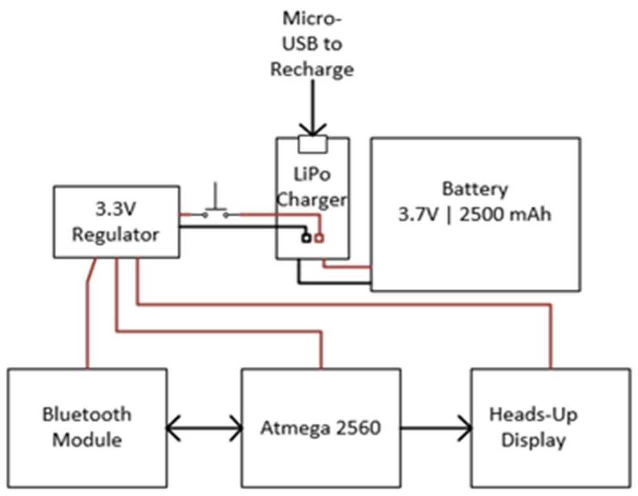


Figure 34 - Power Supply Unit Layout Block Diagram

5.2 First Subsystem - Power Supply Unit

5.2.1 Power Supply Overview

The Power supply of the BRAIN Helmet consists of a 3.7V 2500 mAh Lithium-Ion Polymer battery that is recharged at a current rate of 500 mA by a Micro-USB to JST charger. The ground and ~3.7 positive voltage is then picked off the Lithium-Ion Polymer battery recharger via breakout terminals as seen in Figure. These breakout terminals allow the battery to not only supply the components of the BRAIN Helmet electronic group, but also be recharged. However, in order to avoid swaying voltages, a voltage regulator is implemented into the BRAIN Helmet's power system to regulate a 3.3V power supply to the rest of the helmet's electronic group. The voltage regulator maintains a 3.3 output voltage with a max output current of 800 mA.

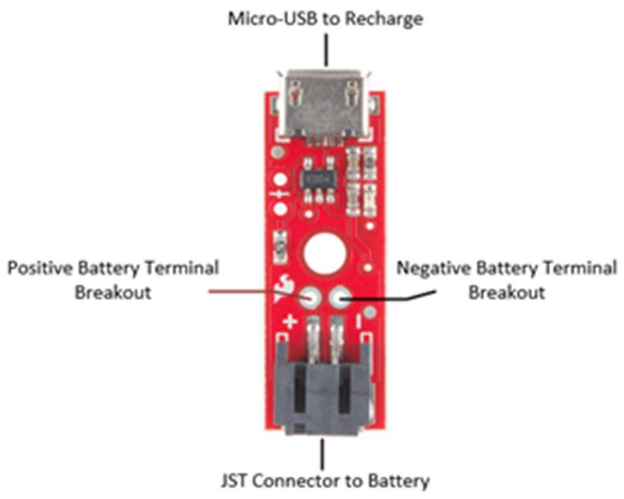

Micro-USB to Recharge
Positive Battery Terminal Breakout
Negative Battery Terminal Breakout
JST Connector to Battery

Figure 35 - Battery Charger Connections

Permission from open source

Currently the BRAIN Helmet electronic group pulls less than 100mA, so this was deemed to be rather overkill, but a necessary precaution for future builds of the BRAIN Helmet. This extra flexibility in current drawn from the battery will ensure no accidental power drop outs and therefore avoid system failure during use.The responsibilities of the entire power supply is depicted in Figure. The regulated 3.3V will be used to power on the Bluetooth module, the ATmega2560, and the Diymall 128x64 pixel OLED Display.

5.2.2 Battery Recharging Design

Particular concern was aroused when first testing the Lithium-Ion Polymer Battery for the BRAIN Helmet. Recharging Lithium-Ion batteries without precautions such as an overcharge protection circuit can lead to harmful results for not just the

BRAIN Helmet electronic system but worst-case scenario to the motorcycle user. The first step to avoid this problem was to purchase a Lithium-Ion Polymer battery that is known to be safer than a pure Lithium-Ion battery. Additionally, the Adafruit 328 used has a built-in circuit protection that prevents current flow as the battery reaches full charge. For the Adafruit 328, full charge is reached at ~4.2V. The following figures show this circuit protection taking place. On the leftmost figure, as the Adafruit 328 is charge a red Led is illuminated indicating the current flowing and therefore the battery charging. The voltage coming out of the battery that would be tied to the voltage regulator in the actual power supply build is being measured at 4.16 in the left picture. However, in the right picture the battery is near full charge (~4.2V) and the Adafruit 328 protection circuit cuts off further recharging of the battery. This is evident on the Lithium-Ion battery recharger board as the red led is no longer illuminated.

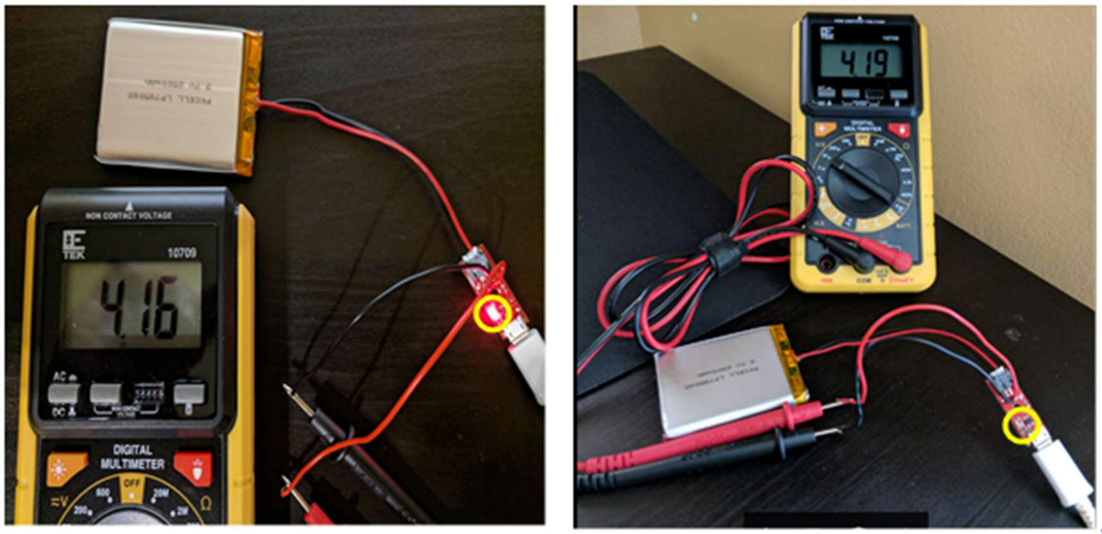


Figure 36 - Charging the battery with circuit protection

5.2.3 Regulating Voltage

A linear voltage regulator, or LDO, was the initial choice of voltage regulation on the BRAIN Helmet. Typically, LDO regulators dissipate a lot of power, but because the 3.7-4.2 varying voltage input was not much higher than the regulated 3.3 volts, this was not still a concern. However, a higher efficiency is always attainable by using a switching voltage regulator and for better experience this was our final decision. The Voltage regulation circuit was designed with Texas Instrument’s Webench design tool and produced the schematic shown in the figure below. Therefore, in the final design of the BRAIN Helmet, every component’s supply voltage, VCC, will be tied directly to the output node of the voltage regulation circuit.

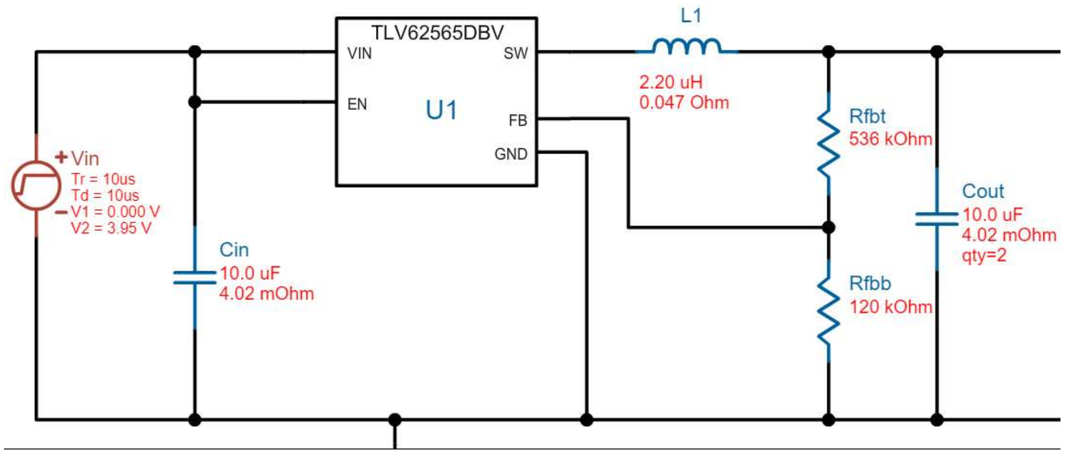


Figure 37 - Regulated Power Supply

5.3 Second Subsystem - Bluetooth & Speakers

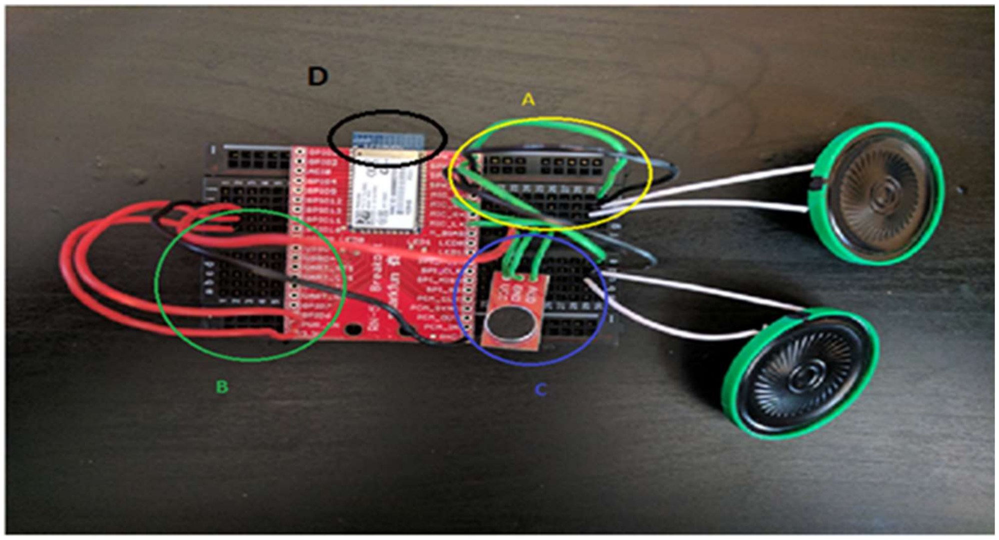


Figure 38 - Bluetooth Audio Subsystem

The Bluetooth Module is one of the most critical parts of the project, due to the wireless we need when connecting between the phone and the system on the helmet. The Bluetooth system should do the following:

- Receive and Send data to and from the User's Device
- Control those signals and output them either to the MCU or speakers
- Maintain compatibility with any User Device, using Bluetooth 3.0.

The Bluetooth module we initially chose was the Roving Networks RN-52. The following is a parts description:

Part	Description	Cost
RN-52 Bluetooth Module	Roving Network RN-52 Bluetooth Audio Module	$25.00

When designing the Bluetooth subsystem, we first had to figure out how the rn52 pins were laid out. At first, analyzing the datasheet for the Rn52 audio Bluetooth module wasn't too difficult, due to its plug and play usability. The profiles on the RN52 Bluetooth device automatically came with factory settings that would allow connection of the two most major phone types, iOS and android devices. The first thing we did was see where to implement the speakers and if there were any need for external amplification outside of the internal amplification of the RN52 Bluetooth device. The RN52 Bluetooth module sub system is broken into 3 different parts, the speakers, the microphone and the Power enable to the Bluetooth. The connection of the Microcontroller to the Bluetooth module will be using the GPIO pins along the left side of the Bluetooth module. The following will be a breakdown of the internal and externals within each part of the Bluetooth subsystem.

5.3.1 Speaker Breakdown (A)

When looking at the picture at the top, you can see that the speakers are connected to the speaker pins on the top right of the RN52 Bluetooth module breakout board. It is denoted by using a yellow circle. The speakers are 0.25 watt and have an 8-ohm resistance. The speakers are in 40 mm in diameter and just a little over 4 mm thick. The reason for these choices of speakers were for the actual thinness of the speaker. We are integrating these speakers into the shell of a motorcycle helmet, so the thinner we can get a speaker the better due to comfortability of the speakers and the safety of the user. Externally the speakers are wired to the positive and negative of the speaker output from the RN52 Bluetooth Module. This is due to the need for both a left and right side of audio. When first designing the speakers, we had to make sure we could hear the speakers without any noise or static. Due to the low wattage of these specific speakers, the Digital to Analog converter that is built inside of the RN52 Bluetooth module can supply enough wattage that there is no static when listening to the speaker. When building our system, if we do end up changing our speakers, we would have to add in an external audio amplifier. This would increase the gain to the speakers therefore making them sound clearer. This is only if we use higher wattage speakers later down in the road. Even if we use different speakers, we would have to find a set of thin or just as thin speakers. We will also implement a potentiometer or a volume up or down with this system to increase the volume on the speakers.

5.3.2 Power Breakdown (B)

The bottom left pins of the Bluetooth module are the PWR_EN, and VDD. The Power enable pin can be connected via a button to turn on and turn off the system. In our case, we would connect the Power system to the Bluetooth module via VDD. The following is a table is the specifications of the RN52 Bluetooth Module:
Table X: Specifications of RN52 Bluetooth Module

Table 15 - RN52 Properties

Specification	Description
Frequency	2.4 GHz
Max Data Rate	3 Mbps
Operation Range	10 Meters
Supply Voltage	3.3 V

The Rn52 has a supply voltage of 3.3 V, which luckily is the same as the supply voltage for the microcontroller we are using, so we can have ease of voltage regulation when it comes to the power system. This means that the microcontroller and the RN52 can be on the same supply voltage bus and not cost us anything.

5.3.3 Microphone Breakdown (C)

The microphone system will be connected via the MIC_L + and MIC_R- pins. There is a built-in analog to digital converter within the RN52 as explained before, and will convert the incoming audio into a digital signal. This mic is powered at 3.3V which means that the mic and the Bluetooth module itself both can be powered on the same voltage bus without having to connect another node all together. The mic must have capacitors at least of value 47 nF due to the gain of the internal amplifier from the Bluetooth module. The mic will need some biasing due to the amplification within the microphone breakout itself.

5.3.4 Antenna (D)

This part of the Bluetooth design has to do with the antenna of the actual system. The following is a diagram is of the dimensions of the RN52 Bluetooth Module:

The problem with the antenna is that it cannot be covered with any metal or it will act like a faraday cage and block any outgoing or incoming signals. So, when designing and integrating our Bluetooth module onto the PCB and putting it on the helmet, we need to make sure that we don't block any of the signals going in or out of the Bluetooth module or it would stop the whole system from working. We must make sure that the antenna of the system is not blocked with metal in any direction.

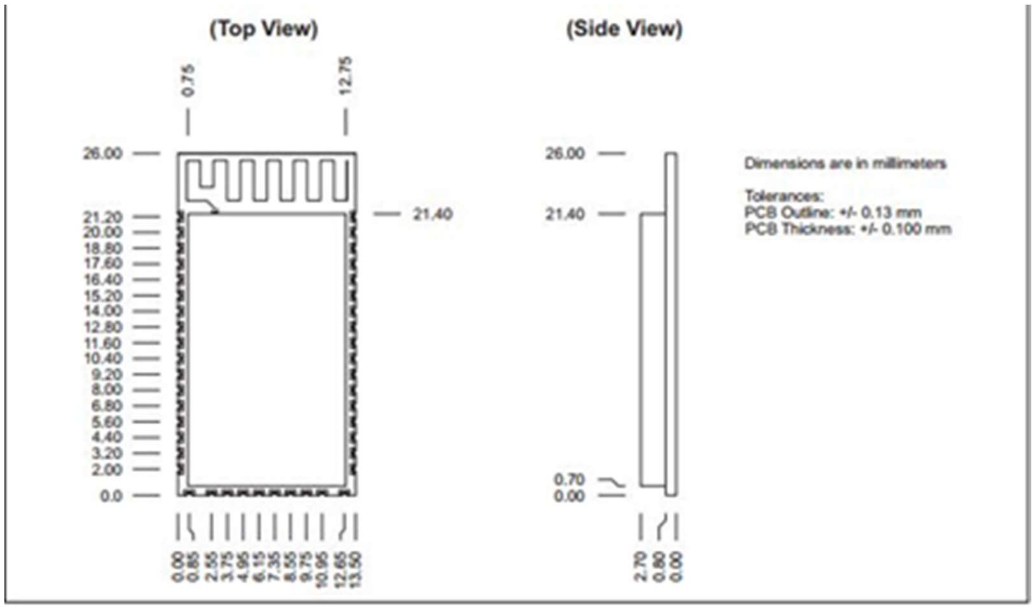


Figure 39 - RN52 Antenna Blow Up Schematic

Permission from open source

5.4 Third Subsystem - Microcontroller & HUD

Our software team will oversee creating two programs, one companion application on the mobile device and the other programming the processor and components on the helmet. The communication medium that the two devices that communicates between them is through Bluetooth. We will program the application onto a phone to test its capabilities to use the cellular data network from Google Maps for directions.

5.4.1 Software Testing Methodology

When the microcontroller powers on or experiences a POR or a brown out, it will initialize the components and check if everything is done correctly. When the system is initialized, the system will start the Bluetooth connection and will search for a device to connect. It can either search for a specific device saved into memory or wait for a ping from a device to connect. If the helmet is not connected to a device we will put it into low power mode and after a certain threshold we could automatically turn off the system to save power. When a device is connected to the system it will communicate to the host device to transmit audio. For the display, we will initialize by showing the “SMART Helmet” text and the current version of the software once it boots up. After a few seconds of the system being initialized, the screen will refresh and tell the user of the current state of the system. It will either display “Searching”, “Not connected”, or “Connecting”. With the companion application, it will also inform the user about the current state in case of the user is not looking at the screen. While the system is powered on it will display the current battery state and can change color if it is low on battery. We are planning to have the battery gauge in green for sufficient charge. When it reaches a certain threshold, say 20%, then the system will change the color of the gauge to yellow to inform the user that it is approaching to low charge. After that state, the system will change the battery gauge to red when it approaches 10%. The system will continue to function properly apart from the display only showing the battery state and the current time. When the battery status changes to 5% and lower, the battery gauge will flash red and disable the audio streaming and only display the flashing battery gauge until it turns off or runs out of charge.

We want to be able to implement an error code system so that we can see if there is something wrong with the code and prevent undetermined behavior. If at any time the system encounters an error while turned on, the screen will display the error code in hex format. Depending on the error code we can either display it on the entire screen or show it in the corner of the screen. Depending on how we implement our system, it may stop the execution of the system or will inform the user that the system needs to be restarted to acknowledge the user that it did see the error and troubleshoot properly.

Registers will be utilized in the processor. One of the registers will contain information sent from the host device. With the companion application it can send information on the current state of the application. This string of numbers will have information that will be used to display the right information on the screen. The first byte of the string will have the distance left before the next turn and the direction of the next turn. The next byte of the string will have information on the current time. The third byte will have the current song playing to show on the display. And the fourth byte can be the status of the application. We may utilize this method but if the information needs more space, then we will dedicate more registers for it. The other registers will be information received from the modules. For the Bluetooth module, we will have a register on the information received from the device for the status of the module.

The outputs are the buttons and the potentiometer that is going to be attached to the helmet at positions where the user can easily determine the functions of them. The potentiometer will be used to adjust the volume of the audio driven to the speakers. This will allow precise control on the volume the user wants to hear the audio than having a button to step down the audio’s amplitude. The dial will be on the right side of the helmet next to the other buttons. The two buttons near the potentiometer will be used to change the tracks on the current song that is playing, one for the previous and the other for the next track. A third button will be added as a function button. These buttons will function as a confirmation to accept the phone call when the user receives a phone call. On the left side of the helmet, there will be a button near the display to toggle the screens displaying the different information.

5.4.1.1 Connection via Bluetooth

One of the main concerns of the Bluetooth module is the options to establish the connection to the host device. We could assume that the device would be visible and can easily connect, but there are uncertainties of confirming that it works. We know that every Bluetooth module in the market has a unique MAC address as a fingerprint. What we would like to implement into our Android application is to automatically detect that one device that we program to search. This would allow us to help us out during our troubleshooting when the helmet would not connect to the host. If we were to put this system into the market we would then consider reprogramming the application to detect the device based on the name programmed to the helmet.

5.4.1.2 Heads-up Display

For the Heads-up display we want to be able to easily inform the user information that is short and intuitive. The display will have a button near the display to toggle the different screens. We should not add too much information to the user at the same time. At the moment we will have four screens in no specific order; It is current time, distance to next turn and direction, time remaining of travel to destination, and time and music. The colors that we will use will have a high contrast to discern the objects from the background. With the header files the display includes, there are some modifications that we need to implement to add our own objects like the arrows to turn and center the text on the display.

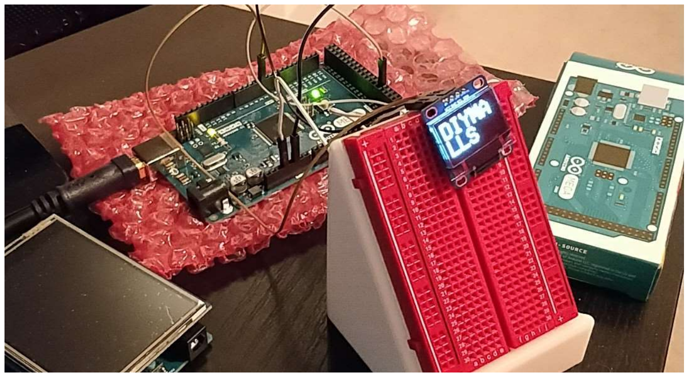


Figure 40 - Display operating on breadboard with MCU

5.4.2 Development Environment

Programming the microcontroller requires us to use an IDE for flashing our code to it. We could avoid using an IDE to program, but we would not be able to take advantage of the debugging features that comes standard for them. We did have experience on several IDE's that have the capability to program our microcontroller, but it is not compatible with our architecture. For example, we would like to use Code Composer Studio but it is exclusive for the Texas Instruments processors. The software is utilizing the Eclipse IDE so we tried to get that to work with our Arduino development board and it proved to be too much of a hassle to send the file to the memory. We decided that we are going to use the Atmel Studio program that the company offers to us. This is 100% compatible with our processor and we can use this for debugging and troubleshooting. And as a backup of our flashing to the memory we will also use the Arduino IDE that they provide to easily upload the program into the board. Both programs use the C++ programming language.

5.4.2.1 Arduino IDE

The IDE that Arduino provides is a user-friendly interface that appeals to every age of programmers. It is simple enough that anyone can use it, but with flaws for the experienced programmer. It does not have a debugging option on the current version and only allows us to compile and upload the code.

5.4.2.2 Atmel Studio IDE

Atmel provides their own version of the IDE for use with their architecture. It does provide more advanced features than the Arduino software. One of the main advanced features is the capability of writing AVR assembly codes. There is an addon available to the software that has the capability to create and modify Arduino files.

5.5 Fourth Subsystem - Android App

This section basically introduces the main methodology that our software team has followed to develop and design Mobile Application. Also, we give a brief description on connection between Microcontroller and Mobile Application. Then, we described the importance of the developing tools that are available for developer in order to create an efficient application with the minimum amount of code. Finally, we will introduce the main functionality of our application and we have provided a prototype created on Android Studio.

5.5.1 Mobile Application design Methodology

Software team has decided to follow Agile Methodology while developing Software Application. Agile methodology is one of the most flexible methodologies because any changes can be implemented on the design as any new information is collected. While developing the project and implementing the design, new changes and improvements can be added to the design. This is the first time for use to build a project completely from the scratch, so there is a high chance that some of the design criteria, parts, requirements, and anything related to the project might change. Also software application might change based on that, some functions may be added or deleted. Moreover, software libraries might change too. Based on Agile methodology, any adjustments can be implemented on the application at any time of implementation easily. The system be delivered in the early stages with flexibility to accept any adjustments.

Our team has divided responsibilities and the tasks that have to be completed based on our strengths and skills. These tasks have to be completed by certain dated that are demonstrated on the milestones table. Each week or two weeks the team will meet to confirm any progress or modifications that must be implemented on the project. Which will also be confirmed with the software application. Thus, the design will be consistent and contributed to the reports and discussion of these weekly meetings. If there is any design modification all the group members can work together and change it easily. This is the huge benefit of Agile Methodology which is extremely better than any traditional methods that was previously implemented. These methods are based on a strict plan that does accepts any changes. There are order stages that must be followed and completed one by one before moving to the next stage. Once one stage is complete, contributors can move to the next stage without going back to the previous one.

5.5.2 The Connection between Microcontroller and Mobile Application

The Microcontroller is a small device that has multiple functions which controls the devices that are connected to it. Our project microcontroller is connected to multiple devices which are speakers, microphone, power supply, PCB and Mobile Application. Microcontroller and Mobile application have wireless communication via Bluetooth. This connection is mainly established to transfer back and forth streams of audio and string data.

For example, our application will call google maps functions to get navigation direction will be transferred via Bluetooth to the microcontroller. This data has two parts audio and string streams. The audio streams will be transferred to the driver via speakers while the string streams will be converted to signals that should be displayed on the LCD displayer. This transferred data is considered an output because it is transferred to the user. There is also an input data that can be displayed to the application which is the data received from the driver when he or she receives a phone call and answer via the attached microphone. The driver listen to data (an output) via speakers, replay and answer to phone calls (an input) via microphone.

5.5.3 Development Tools

There are many useful tools that software writer must take advantage of in order to develop an efficient application with the minimum amount of code, time, and effort in addition to the best experience for users. To build our Brain Helmet's application software team will use Android operating system. This operation system has an official integrated development environment called Android Studio. Android Studio programming depends on Java language and the function and packages available by Java. There are certain standards for Android and Java that are demonstrated on the standards section. Also, there are detailed discussion of android and Android Studio on the Software Design section. In order to make it easily on software team to collaborate and keep updated with all changes and the coding progress our software team has decided to use GitHub. This is a version control system where the programmers can continue working on their code from any place they want. There is also a detailed dissection about that in the last section of this document.

5.5.4 Mobile Application Main Functionality

The main purpose of Mobile Application is to send and receive data. The most important function that needs to be implemented is the Bluetooth connection. The user should be able to click the device connection button on the main screen and browse the available device which should include the Brain Helmet's device. After establishing a successful connection, user can use other feature such as navigation.

The other important function that should be implemented in our application is the navigation function, This function is based on calling Google's maps function to transfer data to our function which also be transferred to the microcontroller that will transferred it to signals displayed on the LCD displayer. The remaining functions is related to receiving phone calls and receiving and transforming data from the phone to the microcontroller and vice versa.

5.5.5 Software Application Prototype

The initial prototype for the application is shown on the attached figure. This prototype demonstrates the main features for our B.R.A.I.N Helmet's application which are: Bluetooth connection, Navigation, and phone calls. Once the Bluetooth is connected to the main module, the application should function as proposed and be ready to use by motorcyclist. This prototype is designed by using the official building environment for Android which is Android Studio.

This prototype consist of a main activity which is the main screen for the application. This main screen has three button which assigned to three different tasks. The first task is the Bluetooth connection. After the Bluetooth connection is established the driver can use the other two tasks. The navigation task is basically connected to Google's maps which has sound notifications and text directions that will be sent to our application. Then these information will be manipulated and sent to the microcontroller.

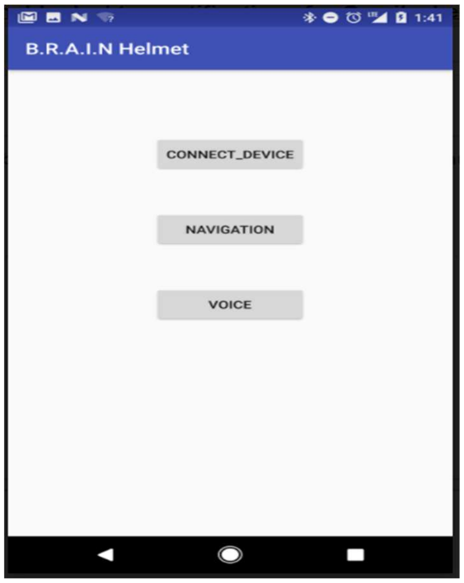


Figure 41 - BRAIN Helmet Application Prototype

5.6 Software Design

Before we start programming the interface and system, we need to plan ahead on how are we going to implement it. Designing a system will be describe abstracted since we don't have a system built yet.

5.6.1 User Interface Design

Our goal from designing a user interface is to make the user experience as simple as efficient by providing necessary features in a design that facilitate user's usability.

5.6.1.1 Design Consideration

Good user interface designs should allow users easily to perform main tasks without wasting time on the application itself. Each interface design should be designed based on the project features that user should be able to use and users' needs. A good understanding of users' needs can help software designer to design an application that efficiently fits all users' requirements. The layout of the design should be as simple as possible to help users with different experiences accomplishing tasks quickly and more effectively. There are primary elements that should be available in each interface design. All users are used to see those elements in most of the applications they have. These elements include but not limited to Checkboxes, Radio Buttons, Dropdown Lists, Buttons, Toggles, Text Fields, Search Fields, Sliders, Tags, Icons, Notifications, Message Boxes and

Tooltips. Once interface designers tend to design an interface some important information needs to be known about users. Designers should know what their users' needs and what could prevent them from achieving their tasks. Moreover, they should be aware of their user's skills, experiences, tendencies, and goals. Also, the designer should know how users will use the application by setting with them and watching them using the product. Users can interact with applications in two ways directly and indirectly. Direct use of software interfaces includes Tapping a button, Swiping a card and Dropping an item with a fingertip. While clicking with a mouse, tapping into a form field, and drawing on a Tablet are examples of indirect interactions. Another important aspect that designers should consider when building Software Applications is consistency. User interface should be consistent so that users can remember how to use it again. Also, that can know how their basic tasks can be accomplished. Consistent designs will increase efficiency for users' experiences.

5.5.1.2 Design Layout

Keep the design simple, clean, and consistent. Light colors should be used in the Background of the user interface. In particular, white color is preferred when designing the background of an application. Dark colors should be avoided on backgrounds because it is very distracting and irritating for users. Also, avoid using light text on light backgrounds which will decrease users' readability. Contrast is one of the essential elements that are needed to improve readability. The maximum value of contrast can be reached by using a black text on a white background.

5.6.2 Android

We have chosen Android which is one of the most popular mobile operating systems designed by Google to build our Mobile Application. This mobile operating system is designed particularly for touchscreen devices such as smartphones and tablets. User Interfaces based on Android require direct interaction which involves tapping and swiping. For Text inputs, a virtual keyboard will be provided. The response will be immediate to user actions. The home screen can be made of multiple pages that user can go back and forth. The default Android's user interface has a status bar on the top of the screen which includes information about the device and its connectivity. Also, a Notification screen is included to show important updates and information. Most Applications are written using Java programming Languages and Android Software Development Kit (SDK). This kit involves Software Libraries, Debugger, and an Emulator based on QEMU. Android Development Tools (ADT) was used by Eclipse to support Google's Integrated Development Environment (IDE). Android Studio is an Integrated Development Environment which was developed by Google. This Integrated Development Environment is based on IntelliJ IDEA which is a Java Integrated Development Environment for developing computer Software designed by JetBrains which is a software development company whose tools are targeted towards software developers and project managers. Android is supported by different kinds of Hardware such as x86, x86 - 64 MIPS64, and MIPS. ARM is the most popular hardware that supported Android's user interfaces. Non-compulsory hardware components may be combined with Android devices such as thermometers, GPS, accelerometers, pressure sensors, and gaming controls.

5.6.2.1 Android Studio

Android Studio is the official Integrated Development Environment for developing Applications using Android. Android Studio provides more features that facilitate user experience while building Android Applications in addition to on IntelliJ's code editor and developer tools. These features include but not limited to the powerful and fast emulator, unified environment, extensive testing tools, NDK support, C++ support, and a flexible Gradle-based build system. There are three basic elements in each new Android Project. These three elements are resource files, source code, and modules. Android can use one or modules such as Application, Library, and Google App Engine modules. All Files can be founded on the Gradel Scripts at the top level. Different kind of folders such as manifests, Java, and res can be included in Application Module. The view of the project files can be customized so that it focuses on certain aspects of designer's Application.

5.6.2.2 Building Software Application Using Android

In order to create an Android, Project developer has to install Android Studio which is the official Integrated Development Environment for developing Applications using Android. Create a new project then enter the name and domain in the specified fields. Keep the other options in the default settings then click next. Also, maintain the default values in the Target Android Devices screen then click next. Select an empty activity in the Add an Activity to the Mobile screen then click next. Next, keep the default values in the Customize the Activity screen and click finish. Android will open Integrated Development Environment (IDE) after some processing. After reviewing the necessary files and making sure everything is correct the Android project now is ready to be used. The next step is to run the Application on a device or an emulator. The steps for running Android Application on a device or an emulator after creating Android project are very different. To execute the application on a real device, the developer needs to connect the device to the development machine using a USB cable. Then, USB debugging on the developer device has to be enabled by going to Settings then click on Developer options. Then run the App from Android Studio by clicking on the App Modula on the Project window the select Run. After that, in the Select Deployment Target, select the attached device and click OK. Running an Android project on an emulator involves more steps. The developer has to create Android Virtual Device (AVD) in order to run his or her Application on an emulator. Then, close the Android Virtual Device Manager window and return to project screen when the emulator begins in order to run the Application. Then follow the same steps of running an Application on a device to run the Application on an emulator.

5.6.3 Our Application

The main features that we are focusing on in our Application are calls, navigation, Bluetooth. From the settings tab, the user can check connectivity to the device and manipulate the remaining setting if he or she wants to change or reset something. Calls and navigation will be displayed on LCD display so the driver can continue driving focusing on the road without touching his or her phone. Also, the driver can see notifications on the LCD display so he or she can continue driving safely.

5.6.3.1 Navigation

In present navigation is one of the most important features that are provided in any useful application which is designed to help drivers. This technology has eased drivers' lives and kept many people safe. The idea behind our project is trying to help Moto sickle's stay safe as much as possible. By providing LCD display that has notifications for the desired direction, the driver can focus on his or her way without messing up with his or her phone. For now, we are thinking of integrated this feature by calling google's maps Application on the same device that has application in order to send data for our application which in turns will display direction's notification on the LCD display.

5.6.3.2 Google Maps API

Google offers developers access to their Maps API free of charge for testing, and if the user wants to commercialize their product then they can get a paid service for licensing. The API that they offer comes in several languages, but the one that we are going to use is Java. With the Android Studio IDE software available to us and a template of the Maps software they have all we need is an API key to obtain permission to access their data. What we are planning to use is the directions API within the overall API, which provides us the navigation that we need to help guide the user to its destination. The companion app that we are developing will offer the user a streamlined way to get navigation instructions while our helmet displays that information. We are obtaining the navigation data so we push data to the display like distance to the next turn, which lane to enter to turn, the estimated time of arrival, and voice navigation. The API also allows us to get directions based on the modes of transportation that they offer to broaden the application use of this helmet for cyclists. Google's documentation on how to use their API provides an abundance of examples and explanations to help us develop the application.

For now, we are thinking of integrated this feature by calling Google's Maps application on the same device that has application in order to send data for our application which in turns will display direction's notification on the LCD display. We did consider purchasing a GPS module and accelerometer and create our own application to assist the user on the current location but that proves to not satisfy our engineering requirements. Therefore, we are utilizing the API that Google has so that we don't need to invent something that is already available to us.

5.6.3 Mobile Application

In order for our project to be functioning as our group proposed, Mobile application has to be connected to the main module. The first goal of our design is to keep driver safe by preventing him or her from touching his or her phones while driving. Thus, user interface is needed to facilitate mobile device connection to the main module so driver can use the features that we are designing in this project.

The first task that need to be completed in our B.R.A.I.N. HELMET mobile application is to check whether the driver device is connected via Bluetooth to the main module so data can be sent from Mobile device and received by the Microcontroller that is initially connected to the Bluetooth module. The Mobile Application will have a button that user should click to check device connectivity. After that user should see a screen with devices that available for connection. B.R.A.I.N. HELMET device should be one of them. User should choose it in order to be connected to it. After that user will see a screen with the device connection statutes and settings. Then the navigation part should work properly after connecting mobile device with the main controller.

The navigation part of the B.R.A.I.N. HELMET's Application is based on collaborating with Google's Maps. The main idea is to set Application to work with Google's maps properly. Thus, the application is supposed to pull data from Google's Maps such as current location, direction, notifications, and the expected arrival time. These dates has to modified and sent to the microcontroller which in turn has to modify it and send to the user either to the LCD displayer or speakers.

5.6.3.1 Development Environment

B.R.A.I.N. HELMET's Application will be developed using Android operating system. Android is one of the most powerful operating systems for building Mobile Applications. Android Studio is the platform used to build Android Apps. It has many high-quality features that facilitate and accelerate building apps for Mobile Applications developers and designers. Integrating Intellij IDEA to Android Environment has added many good properties to coding and running applications. These good properties include instant run, fast rich emulator, and intelligent code editor. Android Packet Kit (APK) can be generated for all device types because of Android Studio Gradle's based property. Gradle- based builds provide Robust and Flexible systems that can be optimized for all Android devices. Moreover, developer can code with more self-confidence since Android provides many code template, sample apps, and testing tools. In addition to the robust lint checks that has 280 different checks across the entirety of the developer app as demonstrated on Android's official website.

5.6.3.2 Platform Choice

Android is the most used operating systems in the market nowadays. In addition to the rich feature development environment for Android which is Android Studio, Android is an open source Linux community which made it preferable for developers and customers. The other choice that our group have considered is iOS which is the operating system used for mobile devices manufactured by Apple Inc. This operating system requires developers to purchase a license that costs $99.00 each year. If the developer choose to purchase Apple Developer Enterprise Program to get access to more devices, the price will be more expensive ($299.00 per year) as published on Apple's Company website. Moreover, to develop an iOS Application designer must have on an Apple computer while Android developers can work on any platform. The official language for Android development is Java which is an easy and clean programming language. The main parts of Android is developed using Java and its Application Programming Interfaces (API) are designed to be called from Java. The programming language of iOS and Mac OS is C and Objective-C. The developer needs to write almost twice the code written for Android Apps to get the same result.

Moreover, this powerful environment allows the designers to integrate with version control tools such as GitHub. Thus, this environment is very sufficient for our project team members so we can share code and check updates. Also, our software team can modify and continue working on building our project application from any place they want. The combination of flexibly and efficiency for Android makes it the best choice for our project application which fits all the features and capabilities that our App must include.

5.6.3.3 Use case Diagram

The main goal of B.R.A.I.N. HELMET's Application is to establish connection with the main module to enable data to be sent from mobile device to LCD displayer and speakers. Once the connection is established, driver can receive notifications for the desired direction. These notifications can be heard on the speakers or shown on the display that is placed on front of the helmet. Driver can drive safely by watching signals on the LCD screen or hearing it on the speakers without touching mobile device.

First of all when user open B.R.A.I.N. HELMET's Application, three options will be displayed on the screen. User needs to connect device to the main module to enable the functionality of the App. By tapping on connect to device, user should see a screen that has all Bluetooth connection devices. B.R.A.I.N. HELMET's device should be one of them, so user should select it. Now the Application will be ready to send sound streams or string streams to main module. Main module will be responsible after that to transfer these data to LCD and speakers.

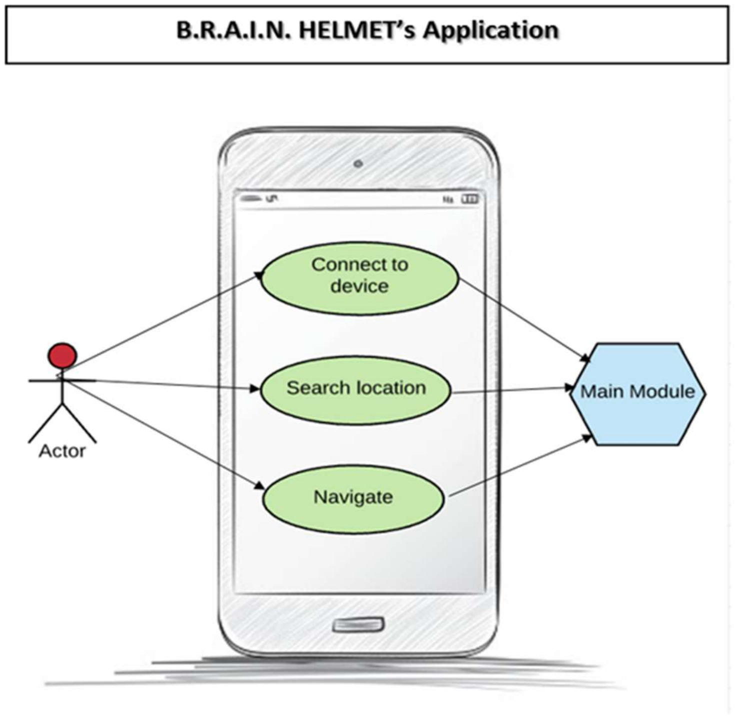


Figure 42 - BRAIN Helmet's Application Use Case Diagram

5.6.3.4 API Levels

In order to build a useful Application for users API version has to be chosen carefully. Each new project on Android Studio has certain setups that must be done in initial setup for the project. One of these important settings is selecting the form factor that your App will run on. For this project we have chosen our Application to be run on phones and tablets. After that designer has to choose the level of API that identifies the framework API revision offered by a version of the Android platform. The basic role for choosing the API level is that lower levels of API target most devices but have less features available. Most recent versions have more features, so less code and testing will be needed. Developer should design an Application that can be used by many users. Also, designers should not use the very old versions of Android since not many users still have these devices. Moreover, newest API levels are not held by many users yet. Thus, choosing the median of these levels will be a smart choice since developers do not want to write any useless Applications. For example, if developer choose API 8, 100% of devices will be covered involving good features for developer. API framework's includes a set of Intents, a set of XML elements and attributes, a set of permissions that applications can use, and a core set of packages and classes. API updates should be included in each consecutive version of the Android platform.

Design of UI of HUD

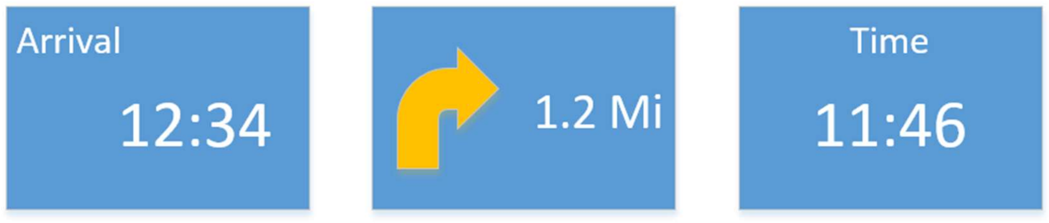


Figure 43 - Original HUD GUI Sketches

The three images above are another rough sketching the minimalistic themed GUI the BRAIN Helmet users will have. It will only display a few items to reduce the time the driver takes his or her eyes away from the road and look at the screen. The information will take the entirety of the screen. When we start programming the interface, a single button will be implemented and placed next to the screen to allow the user to change screens.

5.7 Summary of Design

The design for the Android application will be minimalistic and will help users of any age to be able to connect to the helmet, get directions to their destination, and drive there. The interface for the HUD display will have as little information as possible so that the user can take the minimal amount of time looking at the screen and keep the eyes on the road.

6.0 Project Prototype Construction and Coding

The design of the system was a demanding task, but the following section will be a brief description of the final design and software design of how it will be implemented into the BRAIN helmet. The integrated schematic was done on the program Eagle, made by Autodesk. The software is on the Microcontroller Processor and the Android App Developer.

6.1 Integrated Schematics

The design for the BRAIN helmet schematic was done in the AutoDesk program EAGLE. This program is used to build schematics for PCB design as well as the design for the board. The program is easy to use and very efficient when it comes to designs of your own system. Designing the system, the libraries for each individual piece had to added into eagle in order to get that part within the system. Figuring out the pin layout was the most tedious task, but it was accomplished. This design will be tested with breadboard testing and that will be explained later within the paper.

When coming with the design for the BRAIN helmet, the first objective was to figure out the pins for each set up and how we were going to transfer data between each piece of equipment. In our design, we have the RN52 Bluetooth module, the Atmega2560, a charger for a Li-Po battery, a 3500 mAH Li-po battery, a OLED display (0.96 inches), speakers, electret microphone, and switches and LED's. Each device has its own purpose within our system. The bluetooth module will handle the wireless communication between the android phone and the app from the android phone. The atmega will handle the signals read from the bluetooth module and display what is needed on the OLED display. The LED's in the system will be implemented to be able to know if the bluetooth module is connected, or has been connected with a wireless device. The charger, will allow the charging of the power battery for the system.

Designing this schematic, we had to keep in mind what pins will be connected in order to communicate with each other. The hardest connection was the connection between the bluetooth module and the microprocessor. We didn't know if we want to use UART or GPIO pins between the two. Once we figured this out, the connections between the two devices was made and signals would pass through. The system will be powered by a Li-Po battery that would be producing 3.7V-4.2V(even though the most we need was 3.3V) to the bluetooth module, mic, Microcontroller, and the OLED display. This battery should last over 30 hours on a complete charge; though the possibility of varying due to future features is a likely future plan. We will be integrating this schematic onto a printed circuit board inside of a 3-D printed case and attaching it to the exterior of the motorcycle helmet. The below figure is the final Schematic.

6.2 PCB Vendor and Assembly

The final design of our system will be implemented into a Printed CIrcuit Board. The use of a printed circuit board is so that the connections between the main devices will stay soldered together and will last a decent amount of time. Usually these boards are designed through software such as Eagle Cad, and then sent to PCB Vendors who then print the board. The exact holes and connections from your eagle file will be implemented exactly the way it is sent on your file. The vendors do tests on the connections from one slot to another but as in if the system we are implementing works will be dawned upon ourselves. There are many different vendor and assemblers for printed circuit boards. There are two main PCB manufactures that were compared. Once is from elecrow, and the other mypcbway. Both of these manufacturers you send the files of the PCB Design and then depending on how many you want, they will send back your specified PCB.

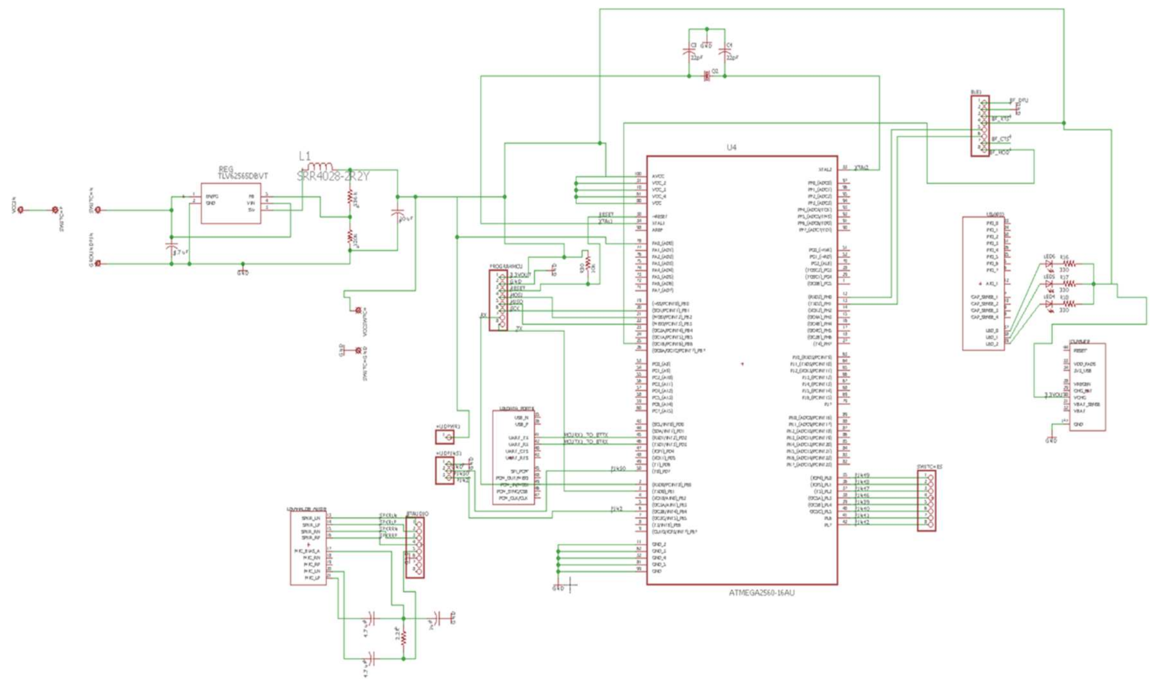


Figure 44 - Final Schematic

Elecrow is a PCB manufacturer that operates online. They have a wide range of options when it comes to PCB assembly. They have a 4-6 day lead time once the order is sent in. The steps to fabricate a PCB are simple, just following a step by step instructions from the website. First you must design your gerber files on eagle, then you send it in. You must decide the dimensions of the pcb after this, when selecting more than 50 fabrications, it would be wise to choose a default size of PCB. In our case, this doesn't matter because at most we will order 3 PCB.Then you can pick the color of the PCB that it will be printed on, in our case we will be using green. Then you can choose what panelization that will be done on the design file. The swift return of the PCB from elecrow would be the main reason we would use this PCB assembler.

PCBway is another PCB manufacturer who has been in the business for up to 10 years. They are known for the easiest and most interactive PCB design company in the world. There is an available instant PCB quote. They also have a system to watch your PCBs on route to the designer, as well as updates on the work on your PCB is going. PCBway uses many different inspection equipment when testing such as Flying Probe Tester, X-ray, and Automated Optical Inspection Machine. The facility has many engineers working to make sure the PCBs are good for even their work. They are also known for their short lead time, due to this they have a dependable on time ship time. The price of on PCBway is a fraction of the cost compared to american and european PCB assemblers. There is even a refund and return policy for hiccups within their exports.

We plan to use one of these two PCB assemblers, they have both great reviews and lead times so depending on price quotes when we finish the PCB design. We are leaning more towards PCBway, but it can change in Senior Design II.

6.2.1 PCB Design

The design of the Printed Circuit Board of our system will be created in the program mentioned in a previous section Eagle. We will use this software to design the final schematic of the circuit as well as go through the trials of designing a printed circuit board. In order to test the multiple features we wanted to include on the BRAIN Helmet, we designed and tested three separate PCBs. The first of which was with the original RN52 Bluetooth module. Following our change into using two modules; Bluetooth Low Energy and Bluetooth Classic, we decided to then move onto our second printed circuit board. The second printed circuit board consisted of the BC127, Atmega2560, buck convertor voltage regulator, and buttons. The final pcb consisted of breaking the second pcb into a main pcb and a button pcb to be placed on the side of the helmet. These board files are known as GERBER files and are different from the schematic files in Eagle. These are the actual files that will be sent to a PCB assembler such as PCBway, and then they will assemble our design of the PCB so we can test our system. These files lay out the certain specifications for board manufacturers. The GERBER files are written in a readable ASCII format, that will be used to develop certain features of the system that we need on the PCB. We will be using the latest version of layout program known as RS247X. The difference between the RS247D and RS247X is the specificity of each type, the "X" version is the newer, and more specific on the design. So, in our case it would be better to go with this layout due to the system to be more precise, which in turn lead to a better result. This standard includes, embedded format, embedded apertures, custom aperture definitions, film control statements, multiple layers embedded into a single file, and special polygon fill commands.

The composition of a PCB and the different layers on it, play a huge part into how the design of a PCB is manufactured. There are many different layers when it comes to a PCB, such as the copper layer, where the actual signals will be conducted throughout the board. The insulation layers that protect these copper layers, and then they are covered with something that is called the solder mask. This solder mask is what gives the color to the board itself, such as green, red or blue. Most manufacturers, use different colors based on what is available, for example it is free for a PCB to be blue or green with the manufacturer known as PCBway, but it costs extra to have either red or black boards printed. The last and final layer of the PCB is the silkscreen layer, which is used to add text and logos on top of the PCB.

Within the program, we are using to design our PCB, it has defaults for these different layers. Here is a table of the layer, its color, and its purpose.

Table 16 - PCB Layer Design Descriptions

Color of Layer	Layer Name	Purpose of Layer
Red	Top	This is the Top layer of Copper
Blue	Bottom	Bottom layer of the Copper Layer
Light Green	Pads	Through Hold pads, it will be exposed copper on both top and bottom
Light Green	Vias	Holes that are covered by solder mask, has copper on both top and bottom
Dark Yellow	Unrouted	Rubber band like lines that show which pads need to be connected
Grey	Dimension	Outline of the Board
Yellow	bPlace	Silkscreen printed on the bottom of the PCB
White	tPlace	Silkscreen printed on the Top of the PCB
Grey	tStop	Stopmask, is where solder mask SHOULD NOT be applied (top)
Grey	bStop	Stopmask, is where solder mask SHOULD NOT be applied (bottom)

The different layers of the PCB include the copper top and bottom, the solder mask top and bottom, and the silkscreen top and bottom. The copper layer is an important layer of the PCB due to it being the conductor or “wire” of the PCB. The copper layer is usually the first layer to be recreated from the GERBER files. For the top copper files, the extension is .cmp, and for the bottom copper files, it is.sol. There are different methods for the copper layer of the PCB manufacture, there are large volume, small volume and Hobbyist. The difference in these methods are the number of items being put into the PCB itself. For PCB’s with bigger features, the silk screen printing method is used for large volume. For small volume, usually laser resist ablation, and PCB milling. For the Hobbyist level of PCB manufacturing, there is a different approach, and that is laser printed resist. This method is used to use heat transfer with an iron to bare laminate.

There are other methods of PCB manufacturing such as subtractive, additive and semi-additive process. The subtractive process removes copper from the entirely copper coated board, to make sure there are no unwanted copper. In the additive method, the copper pattern is electroplated onto a bare substrate. The good thing about the additive process uses less material and less waste is produced. Semi-additive is used in most PCB manufacturers, where the un-patterned board has a thin layer of copper already on it. A reverse mask is then applied, and then addition copper is then plated onto the board in the unmasked areas. Then the copper may be plated to any desired weight, Tin-lead and other plating are then applied afterwards. The drilling of a PCB is also necessary for certain applications of the board. The PCB will be drilled with a drill bit coated with tungsten carbide. This element is recommended since many of the PCB materials are abrasive and drilling requires high amounts of RPM. These holes are usually drilled by computers that are reading computer generated files known as NCD files. The NCD file is basically the blueprint that describes the location and size of holes that must be drilled into the PCB. These holes can be made to be conductive, by electroplating the metal eyelets, to connect board layers together. Most conductive holes are intended for the insertion of a through-hole lead. When vias are required, they aren't drilled using the conventional method, they are drilled using lasers. For multi-layer boards with three or more layers, drilling will produce smear of decomposition products when the hole is plated through. Etch-back is used chemically with a potassium permanganate. This etch-back process removes resin and glass fibers so that the copper layers extend into the hole.

A solder mask layer or solder stop mask Is a thin layer of polymer that is applied over the copper layer on a printed circuit board. This mask layer protects the copper layer from oxidation and to prevent solder bridges from forming between closely placed solder pads. A solder bridge is an unintended electrical connection between two conductors, by the means of solder paste between two pads. Printed circuit boards use solder masks to prevent this from occurring. These masks aren't uses for breakout boards, but used for mass produced boards that are soldered automatically using reflow or another soldering technique. A solder mask comes in different media, depending on the application. The lowest cost solder mask is an epoxy liquid that is silkscreened through the pattern onto the PCB. Others include liquid photo imageable solder mask or dry film photo imageable solder mask. These processes typically go through a thermal cure after the pattern is defined. This solder mask is used to protect the copper layer of the PCB in order to keep the solder and connections between the solder and copper from every causing a short or shunt within the system.

6.2.2 PCB Fabrication

Once the GERBER files that were described in layers in the previous section, the actual fabrication process will take place at a PCB manufacturer. This section will explain the basic manufacture process that takes place. The first step will be choosing a PCB manufacturer. In our case, we will be going with PCBway. How the PCB manufacturer starts is usually with a transfer of the files that are sent from us the designer. The manufacturer uses a DMF check, then they laser print using very precise printing technology to create highly detailed film of the PCB design. After the precision prints are created, then comes the copper layering. In this step, cleanliness is of upmost importance due to the fact that the copper is where the current passes through the system so this layer is crucial for the whole system to not create any shunts anywhere. Once the copper is laid out, then the board receives blasts of UV light onto it. This light will pass throughout the board hardens the photo resist that is underneath the copper. Once the board is done, it is washed in an alkaline solution that will remove any left photo resist material. Then onto the drilling explained earlier for the holes of the circuit. To remove any unwanted copper from the board, there is a copper solvent that removes any excess that isn't covered by photo resist. Then another solvent is used again to wash the board one last time, and now a board with a perfect protected copper layer is left. The alignment of holes must be ensured to make sure that the drilling process doesn't ruin the board. Once the layers are placed together, it is impossible to correct ant errors on other layers. The last step of PCB fabrication is when all the separate layers of the PCB are bonded together. They are inspected by a technician, then can be bonded together. This process is known as PCB prepreg. During this process, the layers are placed over each other and then pressed together. It is basically a PCB sandwich. The board is done with the actual assembly of the board. The rest of the process deals with steps with for aesthetic purposes such as labeling of certain parts on the PCB. This process depending on how big the PCB can take up to 5 days at certain manufacturers.

6.3 Final Coding Plan

This section contains the main functions that we are planning to include in the code for our Application. There are three main features that we are planning to integrate on our application. These three features are Bluetooth connection, navigation, and phone calls. First of all the Bluetooth connection must be established in order for the remaining features to function. On Android Studio every new task added is called Activity. For example the main screen of the application is an Activity. Then if the user click the button of Bluetooth Connection a new activity will show up.

6.3.1 Coding Plan for Bluetooth Connection

This function is responsible for setting the Bluetooth connection for the application, so Mobile device will have a wireless connection with main module. This method will start by asking the user if this application can have access to the device. Then it will ask the user if the application can turn the device Bluetooth on if it is turned off. Then this method should look for all the available devices and list it on a screen for the user. User should be available to pick any device from the devices that are listed on the screen.

First of all the Bluetooth permission must be declared in order to perform any type of Bluetooth communication such as looking for a connection, receiving and transmitting data. Thus, developer must add both BLUETOOTH and BLUETOOTH_ADMIN permissions in order to use Bluetooth features and set device discovery. After that, Bluetooth Adapter must be initialized which is required for all Bluetooth activities. Then the Bluetooth must be checked by calling isEnabled () function which returns a Boolean value either true or false (enabled or disabled).

Then in order for your mobile device to scan and look for devices, startLeScan () method must be added to the code. This method looks for available devices for connection and let you know the result by implementing LeScanCallback () to return the scan results. After that programmer must set the connection with Generic Attribute Profile (GATT) server to read and write BLE Attributes. This is our software team coding plan to establish Bluetooth connection which can be reviewed on Android official Website with all steps and required functions. Google's play services SDK kit must be installed.

6.3.2 Coding Plan for Navigation

Our navigation function is based on Google's Maps. Android Studio is prepared with all the necessary development Environment to build Applications that depends on Google's maps for finding directions. There is a ready Google Map Activity on the main screen for activities.

Purpose for navigation feature in our application is to help driver get notification that direct him or her to the desired destination. Our Software team has already started writing the code for our B.R.A.I.N Helmet project application. They have decided to choose an empty activity on Android studio and build the navigation part by part. First, they have written the code to integrate Orlando's map to the main screen. Then, they have designed a menu that contains three main option which are: location, destination, and calculating the root. User should enter his or her current location on the specified field. Then, user should enter the address of the desired destination. Finally, the user should choose calculating the root from the desired menu on order to calculate the root for him or her. Finally, user should be able to see a blue line starting from user current location connected until the final location. After that, driver should get a turn by tur notifications which are displayed on the LCD displayer and transmitted to the speakers. Software team is working to get this part done which means they are done with programing this part of the application so they can move on and start programing receiving and answering phone calls.

6.3.3 Coding Plane for Phone Calls

There are built-in application for phone calls provided by Android. For this part of the application we need Android CALL_PHONE permission. Also, we need ACTION-CALL and ACTION-DIAL to activate built in phone call functionality accessible in Android devices. This is the main things that we are planning to integrate in our Application. For sure, many other things have to be manipulated and added to the code in order for everything to work together correctly at the end.

6.3.4 Code Plan Summary

All the steps that are needed to set the Bluetooth, Navigation and any other feature that we are planning to include in our Application are listed and demonstrated on Android official website. Software team is trying now to train themselves and follow all perform all tutorials listed on Android Website in order to be prepared to write an efficient and functional Application. This activity has google_maps_api.xml and the MapsActivity.java files ready to edit. There are certain instructions to get Google Maps API key to access Google maps Server. This is our plan to develop this part of the application. Android official Website has all the important guidelines and steps if we need any further assistance.

7.0 Project Prototype Testing Plan

The BRAIN helmet design once implemented needed to be tested. The parts individually were tested before being implemented into the system as well as the system as a whole. The design was tested in phases, such as the bluetooth module pairing, Microcontroller and OLED displaying, and android app communication. These had to be tested before being integrated together. The following is an explanation of how each device is tested and in different environments.

7.1 Hardware Test Environment

The hardware for the BRAIN Helmet system needs to be tested before being implemented into the entire PCB. The following section is a detailed description of how we tested each main device before testing it as an entire system. Many of these tests took place in UCF labs; including open labs and the Senior Design Lab.

7.1.1 Initial Development Test Environments

The BRAIN Helmet hardware primarily consists of the electronic group and a casing that attaches to the helmet. Prior to the development of the final PCB build, the subsystems of the electronic group will be constructed on breadboards in order to ensure proper connection between pins and test of appropriate voltage ranges and proper interaction over the Bluetooth connection between the Bluetooth module and the android device. During this initial designing phase, subsystem group members conducted tests at labs at UCF with available digital multimeters, power supplies, and breadboards. Additional testing took place at personal residences with mini-breadboards and handheld digital multimeters. After subsystems tests prove conclusive, the breadboard version of the BRAIN helmet's electronic group will be assembled in a UCF lab to continue testing, fine-tuning, and finishing the necessary BRAIN Helmet android application and microcontroller programming for the Heads-UP Display.

7.1.2 Post PCB Test Environments

Once the PCB is fully built and assembled with the motorcycle helmet, the helmet will have its integrated features tested in multiple environments. The first of these environments will be the senior design lab at UCF. Shortly after, tests will be conducted in moving vehicles with the BRAIN Helmet in the passenger seat and on bicycles. After repeating successful tests, the helmet will be tested on an actual motorcycle.

7.2 Hardware Specific Testing

To make our system work we have to manufacture a way to test the system to see if it meets our requirements. For each subsystem explained we have a way to figure out ways to see if the system works.

7.2.1 Power Supply Unit

The power supply unit has been monitored for proper voltage and current values. This entails the initial voltage coming directly off the battery from ~3.7V - 4V and the regulated voltage once tied to the voltage divider at 3.3V. In terms of purely the power supply unit, the current being drawn into the battery was additionally monitored through the Micro-USB to JST Lithium-Ion battery charger to ensure the Adafruit 328 was not being charged at a current rate that would cause damage to the battery. Therefore, the charging current into the Adafruit 328 was monitored to be less than or equal to 500mA. Once the power supply unit is entirely connected to the rest of the BRAIN Helmet's electronic group, multiple tests will be conducted to ensure the BRAIN Helmet provides users with an ample amount of usage for at least 6 or more hours.

7.2.2 Bluetooth Unit

To test any of the commands or other uses of the Bluetooth Module, we must first be able to connect it to our device. We do this buy powering on the Bluetooth Device via 3.3 V from a power supply, and then we should see the LED's hooked up for configuring. The two LED's are set up for when activated, flashing or off signifies a certain action is available with the Bluetooth Device. The following is a table that shows what the LED's are doing for certain actions:

Table 17 - LED Interface

LED	Status	Description
LED0 and LED1	Flashing	The RN52 is discoverable to a device
LED0 only	Flashing	The module is Connected

LED	Status	Description
LED1 only	Flashing	The module is Connectable

The Bluetooth Module must be tested to make sure when connected wirelessly the pins are correctly sending to the MCU and then the MCU to the HUD for us to test this, we must simply hook up the RN52 to a MCU and give it commands such as, play, pause, or stop, when the Bluetooth device is paired with a phone, these commands will then tell when to play, pause, or stop the music streaming from the phone. The testing for the Bluetooth Module will also include the pairing of both android and iOS devices. We do this by connecting an android phone, in our case a google Pixel, and an iOS phone, an iPhone 7 Plus. These phones connecting would mean the profiles within in the firmware of the Bluetooth Module are up to date and able to pair with any big phone manufacturer. When testing the speakers of the system, we first pair the phone, then play music to the device. Once the music is playing, we cover the speakers with cloth in order to simulate how it would sound inside the helmet, to see if the sound quality was good enough to even hear through the thick layering of the Motorcycle Helmet. When testing the speakers, we can apply an external amplifier to clear the static that might be being outputted. Qualitatively, the speakers sounded well when not moving, but also since the user will be moving, we tested the Sound quality on a bicycle to test if the sound of the music was any different. To test the microphone of the system, we just wire the Bluetooth System, and to make sure the microphone itself is working, we use an oscilloscope to read the waves that are being generated from our voice. We also must test if the microphone will have lots of noise and how it deals with the noise, because this mic will be placed inside of a motorcyclist's helmet, who is moving at a high velocity, which means the air around will cause interference when it comes to noise around what really needs to be picked up. So, when testing we ride a bike with the microphone covered to see how the riding affects mic quality. Also, we connect to the Bluetooth Device and to check if the mic works and phone calls can be answered, we use the Bluetooth connected device to call another phone and if we can go through with a regular phone call than we should be able to hear within the helmet. We must make sure to test that the mic would be able to be heard when covered due to it also being under a layer of cushion that will be layered for the safety inside of the motorcycle helmet.

7.2.1 Navigation HUD

To test the navigation system on the app, the display should change based on the current location on the user. While the user is driving, the display should poll the data from the app and discern the code received from the app. When the user pressed the button, the display should change and update the display based on the variables on the software.

7.3 Software Test Environment

Android Studio is the official building environment for Android Applications. Android Studio is designed to give the programmers a great experience by providing many features and testing methods. There are two different types of testing that can be made using Android Studio. Programmer can test the code by creating a JUnit test that runs on the local JVM. Also, the Application can be tested directly on mobile device.

In order to test our software application, software team will do continues testing on Android Studio emulator. Also, they will do some direct test on the mobile device itself between each new stage and progress they will make on the application.

7.3.1 Test types and location

There are different locations for tests that can be made on Android Studio depending on the type of test that programmer needs to perform. The source code and source sets are provided by Android Studio on their official website. The two types of tests that Android contributors can implement are:

7.3.1.1 Local Unit Tests

These tests can be performed on the programmer's computer by (JVM) Java Virtual machine. It is located at "**module-name**/src/test/java/." as demonstrated on Android Studio Website. These tests are useful when there is no limitation on Android Framework to minimize the execution time of the program. One of the standard mocking libraries is used by this test since a modified version of android.jar is used which includes stripped off final modifiers. These tests are designed for testing programs that required integration and functional user interfaces to test user interaction.

7.3.1.2 Instrumented Tests

These kinds of tests are the ones running on an emulator or hardware device. The Instrumentation APIs must be accessed to enable information access to some main parts on the Application. Also, it enables the programmer to control the Application under test from your test code. These tests are located at module-name/src/androidTest/java/.as demonstrated on Android Studio Website. It has its separate AndroidManifest.xml file since it's built into a different APK from the application APK.

7.3.2 Add a New Test

Tests can be created for a specific class or method. The new tests can be either a local unit test or an instrumental test. There are some steps that have to be followed in order to create a new test:

- Open the java file that you want to test it

- Determine the part of the code that you want to test and click (Ctrl+Shift+T)
- Choose to create new test from the menu that appears
- Create test dialog will appear, select any methods to generate and edit any fields then click ok.
- Choose Destination Directory dialog will appear, for instrumented test click androidTest and for local unit test click test then click ok

7.3.3 Run a test

There are some steps that have to be followed in order to run a test:

- Click on Sync Project on the toolbar to make sure that your project is synchronized with Gradle.
- Run your test by either: right click a test and click run in the project window, or by right click a class or method in the test field from the code editor and click run to test all methods in the class.

7.3.4 Import Test Results

On the Android Studio Website, there is description on how developer can view test results. There are two simple steps that must be performed which are:

- Choose Import Test Results.
- Choose the file that you want to import from the drop-down menu which will display the results on the Run window.

7.3.5 Export Test Results

The test results can be exported in XML or HTML format by doing the following steps:

- Choose Export Test Results.
- Edit the settings as you need from the Export Test results dialog, and click "ok"

7.3.6 Android App Testing

To test our application a real Android Mobile device will be used. Any android device that is 5.1 or higher will be good to test our application since the API we are choosing is 21 API level. A real-time testing on the Android phone will be done continuously to check our application. Moreover, the application will be connected to the main module via Bluetooth and tested too. By testing application on the Android App directly developer can see the actual design of the application and do a real-time testing by interacting with the application by tapping, entering, and changing information on the application. Our application will display three main features on the main screen the user can tap and choose what he or she needs. For the navigation part, the user is required to enter his or her current location and enter the desire destination. After that the user is required to tap into the calculation route field from the navigation menu in order to get the directions. By doing these tests and interact with the application directly developer can improve the application and debug it easily since they can try and see all the mistakes that must be fixed or avoided.

In order to test Application on the Android Mobile device directly some easy steps must be followed to configure the phone and set it ready to test the application. These steps are listed below:

- The USB debugging must be (turned on) on the Android Mobile Device. This feature is mostly used for Applications debugging. Thus, the user can keep this feature on all the time, but if the user is concern about security, Bluetooth debugging can be turned off while not debugging applications.
- Double click the AndroidManifest.xml file in the current project's of the package explore.
- Then choose True from the debugging drop down menu which will enable Android tools from checking the run of the application.
- Then select file and save the new AndroidManifest.xml file.
- Based on your type of computer set it to communicate with the Android Mobile Device.
- Connect the Mobile Device to the configured computer by a USB cable.
- Finally, choose the device and run the project from the green play button.

The developer should see the actual application displayed on the mobile devive. Then the developer can directly interact with application and test all the features it has. If the application crashes for any reason the developer can set the debugging point at any line as needed to test the part that need to be fixed. The developer can disconnect the device from the computer to stop connection. To safely remove the device from your computer do not just pull out the cable. Instead open finder on your computer and search for the device under devices, then click on the eject button next to the device name. When the device disappears, pull out the cable from the device.

7.3.7 Software Testing Summary

After having a good understanding of these two-different kind of tests that are provided by Android Studio. Software team has decided to perform Instrumental Tests since it's more relevant to our application. The software application that we are planning to design require user interaction. Thus, the tests will be performed on an emulator or on the mobile device itself. All of this information are demonstrated and explained on android studio

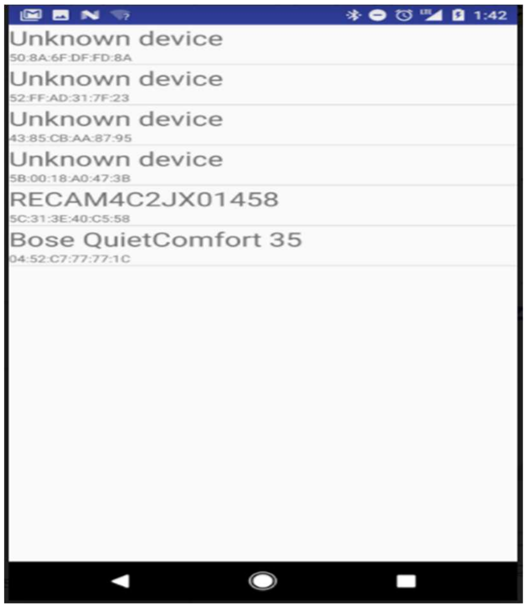
7.4 Software Specific Testing

Software Application must be tested before submitting final product to be evaluated by the faculty. In order to test this Application Android Studio will be used as demonstrated on the previous section. Software team will perform continuous tests by using emulator provided by android studio. Also, Software team will purchase or use one of the team members Android mobile device since it is one of the good testing environments.

Since we are integrating many important features into our application, Software team has started building Application. Building application will require them to have good experience writing code in java and using Android studio. If the coder has good experience with java building application will be much easier since Android studio is based on java with some more functions and coding tricks that developer must be familiar with. Our Software team could successfully finish one little part of the application which is creating the main activity that has our application main features which are: bluetooth Connection, Navigation, and phone calls. We could successfully set the Bluetooth discovery as demonstrated on the following section. The remaining functions will be tested using the same methodology by using Android mobile device and checking the functionality of the part being tested.

7.4.1 Bluetooth Device Scanning Testing

This is an initial testing for Bluetooth scanning functionality. After the Bluetooth get access to the device and making sure that Bluetooth is turned on. The next thing is searching and scanning for devices. As shown on the figure, all the available devices are listed on the screen. Thus, we can say that this part of the application is working correctly. Our main module Bluetooth is not ready yet or it should appear as one of the available devices on the screen, then user should choose our B.R.A.I.N Helmet device in order to establish connection. After that data can be received and sent forth and back between Mobile device and main module. Driver will be able to get navigation notification and phone calls efficiently.



```
Unknown device
50:8A:6F:DF:FD:8A
Unknown device
52:FF:AD:31:7F:23
Unknown device
43:85:CB:AA:87:95
Unknown device
5B:00:18:A0:47:3B
RECAM4C2JX01458
5C:31:3E:40:C5:58
Bose QuietComfort 35
04:52:C7:77:77:1C
```

Figure 45 - Scanning Bluetooth Devices

7.5 B.R.A.I.N User Troubleshooting Tips

These are some of the tips that we are provided for our B.R.A.I.N. HELMET's device users, so they can have some suggestion to solve some problems and difficulties that they might face. This will assist not only the development and testing of the BRAIN Helmet but additionally work towards the future final troubleshooting guide for BRAIN Helmet users.

Table 18 - Troubleshooting Table

Issue	Troubleshooting Steps
The device does not power on	✓ Open the BRAIN Helmet Electronic Group to check all wires and make sure of the connections. This is to rule out faulty hardware ✓ Disconnect and replace the Lithium-Ion rechargeable battery via the JST Connection ✓ Make sure the battery is working with the required voltage; LEDs not turning on will be a full sign of failing voltage.
The Mobile Application does not connect to the device via Bluetooth or does not sync in a reasonable time	✓ Check the Bluetooth connection settings on the application. ✓ Disconnect the device and try to connect it again ✓ Disable existing Bluetooth connections to any media capable devices already connected. ✓ Turn off the device and then restart the android device and attempt to reconnect ✓ Restart the BRAIN Helmet
The LCD displayer does not display the signals.	✓ Check connection and any loose wires ✓ Check the power connection ✓ Refer to battery replacement above
The speakers are not clear	✓ Check the helmet padding for obstructions

8.0 Administrative Content

This section includes all the administrative parts that are related to our project. Starting from the milestones and how we divide that main tasks and responsibilities between us. Followed by project management methodology that we have followed in order to ensure success for our project. Finally, we have discussed finance and budget that we have proposed for our project. All of these sections will be constantly updated through Senior Design One and Two as the BRAIN Helmet proceeds through design, implementation, and testing.

8.1 Milestone Discussion

This section demonstrates the main tasks that we have to complete in order to reach to our final goals for the BRAIN Helmet. These tasks have been divided between the members of this group based on many factors that includes skills and major. It is common practice that the Hardware Team will work together on their tasks and the Software team additionally on their tasks together. Also, there are tables provided to demonstrate these tasks based on certain deadlines that must be completed on or before these dates.

8.1.1 Milestones

In this part, we will discuss our individual milestones tables that we have created at the beginning of senior design one. These tables will basically show the responsibilities and achievements that each group member will accomplish based on clearly determined dates that group members have to follow. This also includes some plans that we have made for Senior Design Two, so we can go ahead and start working since there is a lot of work that has to be done. The due dates are not determined yet for Senior Design Two. However, we as a team know the basic parts that we can start working on it. Thus, being a little bit ahead can give as a chance to start working on the PCB layout.

Milestones table will show the tasks which each project member has to achieve and the due dates for the completion of each task. Before determining milestones, we have met as a group many times to determine the project idea we did a lot of research until we have decided on one idea that we like as one team. Also, as a group, we met to discuss our project name and the features that we are supported by this project at the end of this project. We have also met many times to decide which parts do we need and who will be in charge of each part which ended up by dividing responsibilities between us and creating our milestones' tables.

Table 19 - Jordan Yamson's Milestones

Objectives	Duration (Days)	Achievement Date
Research Bluetooth modules RN52, HUDs, Batteries, and Power	14	June 15th, 2017
Research accessories to recharge batteries	14	June 15th, 2017
Assemble the Bluetooth Module RN52 with 4-ohm speakers and power supply as input voltage	2	June 24th, 2017
Test the Bluetooth Module RN52 with 4-ohm speakers and power supply as input voltage	2	June 28th, 2017
Purchase 16-ohm speakers to test additionally	1	June 17th, 2017
Solder wires to Bluetooth module RN52's breakout board for further testing	2	June 29th, 2017

Table 20 - Ryan Mortera's Milestones

Objectives	Duration (Days)	Achievement Date
Research Bluetooth modules RN52, and Power	14	June 15th, 2017
Assemble the Bluetooth Module RN52 with 4-ohm speakers and power supply as input voltage	2	June 24th, 2017
Test the Bluetooth Module RN52 with 4-ohm speakers and power supply as input voltage	2	June 28th, 2017
Purchase Bluetooth Module	1	June 17th, 2017
Solder wires to Bluetooth module RN52's breakout board for further testing	2	June 29th, 2017

Table 21 - Stephan Morale's Milsetones

Objectives	Duration (Days)	Achievement Date
Assemble the Bluetooth Module RN52 with 4-ohm speakers and power supply as input voltage	2	June 24th, 2017
Test the Bluetooth Module RN52 with 4-ohm speakers and power supply as input voltage	2	June 28th, 2017
Purchase 16-ohm speakers to test additionally	1	June 17th, 2017
Solder wires to Bluetooth module RN52's breakout board for further testing	2	June 29th, 2017

Table 22 - Nada Algharabawi's Milestones

Objectives	Duration (Days)	Achievement Date
Research Wireless Pieces and Building Software Applications	14	June 15th, 2017
Research Software Prototypes	14	June 15th, 2017
Deciding which environment to use for Software Application	2	June 28th, 2017
Design the prototype for Mobile Application	1	June 17th, 2017

We have also placed some of our future plans for Senior Design Two. We have determined the steps that need to be accomplished in chronological order and assigned each of us the necessary tasks that need to be completed individually or as one group. In Senior Design Two, the hardware team and software team will work more separate than in Senior Design One. Goals will be set for both teams and it will be to the team's discretion who works together or separate on that indivdual task. Group meetups and discussions will take place regularly in person and electronically to ensure proper progress and to do cumulative testing of all subsystems and eventually the final product. The following two tables are currently under work but are the tables to be updated in Senior Design Two.

Table 23 - Senior Design Two Documentation and Presentation

Senior Design Two		
Documentation and Presentation	**Complete Date**	**Person in charge**
CDR Presentation	Sept. 22nd	Group
Conference Paper	Nov 17th	Jordan Yamson
Middle Term Demo	Nov. 1st	Group
Final Paper	Dec. 4th	Jordan Yamson
Final Demo	Nov. 29th	Group
Final Presentation	Nov. 29th	Jordan Yamson
Exit Interview	Dec. 5th	Group

Table 24 - Senior Design Two Fabrication and Evaluation

Senior Design Two		
Fabrication and Evaluation	**Complete Date**	**Person in charge**
Software Prototype	9/30/2017	Software Group
Hardware Prototype	9/30/2017	Hardware Group
Assemble	10/31/2017	Group
PCB Design	8/31	Hardware Group
PCB Layout	8/31	Hardware Group
Revisit Specs	9/27	Software Group

Senior Design Two		
Fabrication and Evaluation	**Complete Date**	**Person in charge**
Robustness Testing	Oct. 30th	Group
Improvements	Oct. 30th	Group

8.2 Project management

In This section, we will go over roles distribution in our group and the structure of our project. Also, we will cover our individual milestones for Senior Design one and Senior design two in tables that show the tasks and the proposed completion date for each one. In this section, we will also take a look at how we managed our project-based on Finances, budget, and division of labor.

8.2.1 Management Plan

Project management is one of the important basics that any group needs to understand and follow properly. A project management plan should be determined by the beginning of the project, and all project members should understand it and follow it continually in order to succeed. All successful project management discuss group member's roles, project finances, project constraints and many other essential building blocks. One of the advantages of project management is splitting the responsibilities and roles of the project between project members. In this way, each member of the group will be assigned some work he or she has to complete in a certain time of a clearly determined schedule. Having a clearly determined schedule will motivate members to work hard in order to keep up with submission dates which will ensure that the project will be complete at the determined time. However, each group member has some strengths that make him or her qualified for some responsibilities more than the other group members. Thus, group members have to discuss their skills, strengths, and weaknesses so they can split the work based on this information. Some team members might have common skills and strengths especially if they are studying the same major. In this case, they can share some of the responsibilities, or the can divide it between them depending on what they like to do more. Also, some of the group members more cooperative than the others, so they do not mind doing some extra work that helps the project to proceed well. Moreover, in every new project, there will be some new things that group members do not know well or have to teach themselves how to do it or use it. In this case, these responsibilities have to be distributed between them even if they do not have any experience with it. Thus, not all work will be based on something they like or have excellent skills on it, but it also will be on something that project requires or need in order to be done.

Another important thing that needs to be covered in the project management plan is the finances. Finances are one of the important things that have to be determined clearly at the beginning of the project. Each group member has a different amount of money that he or she is willing to pay for the project depending on their income and other reasons related to their social lives. Thus, after doing some research on the required parts they need to be ordered, manufacturing costs, and calculating everything they need in order to complete the project they can determine the project's budget. Team members have to follow project's budget, or the project will not finish as planned. Group members have to decide who will take care of the finances during the project time, so he or she can make sure that they do not exceed the budget and the limits they put on finances based on their research and calculations. This member has to know exactly when and where money must be spent and the amount that has to be spent. In this way, project members can avoid breaking up their budget and losing their project.

8.3 Division of Labor

For Senior Design One we have decided to divide up the responsibilities between as based on our skills and strengths. Since two of our group members are Computer Engineers and the other two are Electrical Engineers we ended up having Software team and Hardware team. Of course, we as a team will put the things together at the end and help each other along the two semesters, we divided responsibilities, but this is still a group work that can be done in a great way if all team members do hard work and respect each other. Computer Engineers will do most of the software work that is related to the project such as microcontroller module, Mobile Application, Bluetooth module, Navigation module, Connections, Software Power and such things.

Hardware team will work on Hardware parts of the project and then put everything together with the software team to complete the project. Hardware work includes but not limited to Hardware Power Management, Batteries, PCB Design, and Power Testing. Hardware Power Management contains power supply for the Microcontroller, Speakers, and backup battery for PCB. After doing an excellent research on the necessary parts that the project needs, we have purchased the necessary parts that we think are the best for our project based on power, efficiency, price and other factors. Before deciding to buy any device or part of the project, we compared it with other parts that are available in the market, and we picked the ones that we think are better for our project. We also considered buying some other parts such as a microcontroller. We have ordered three of them to avoid problems such as releasing new versions or stopping producing the old ones or timing issues such as taking a long time to arrive which is very risky since we have a schedule and certain dates that we always have to consider and follow.

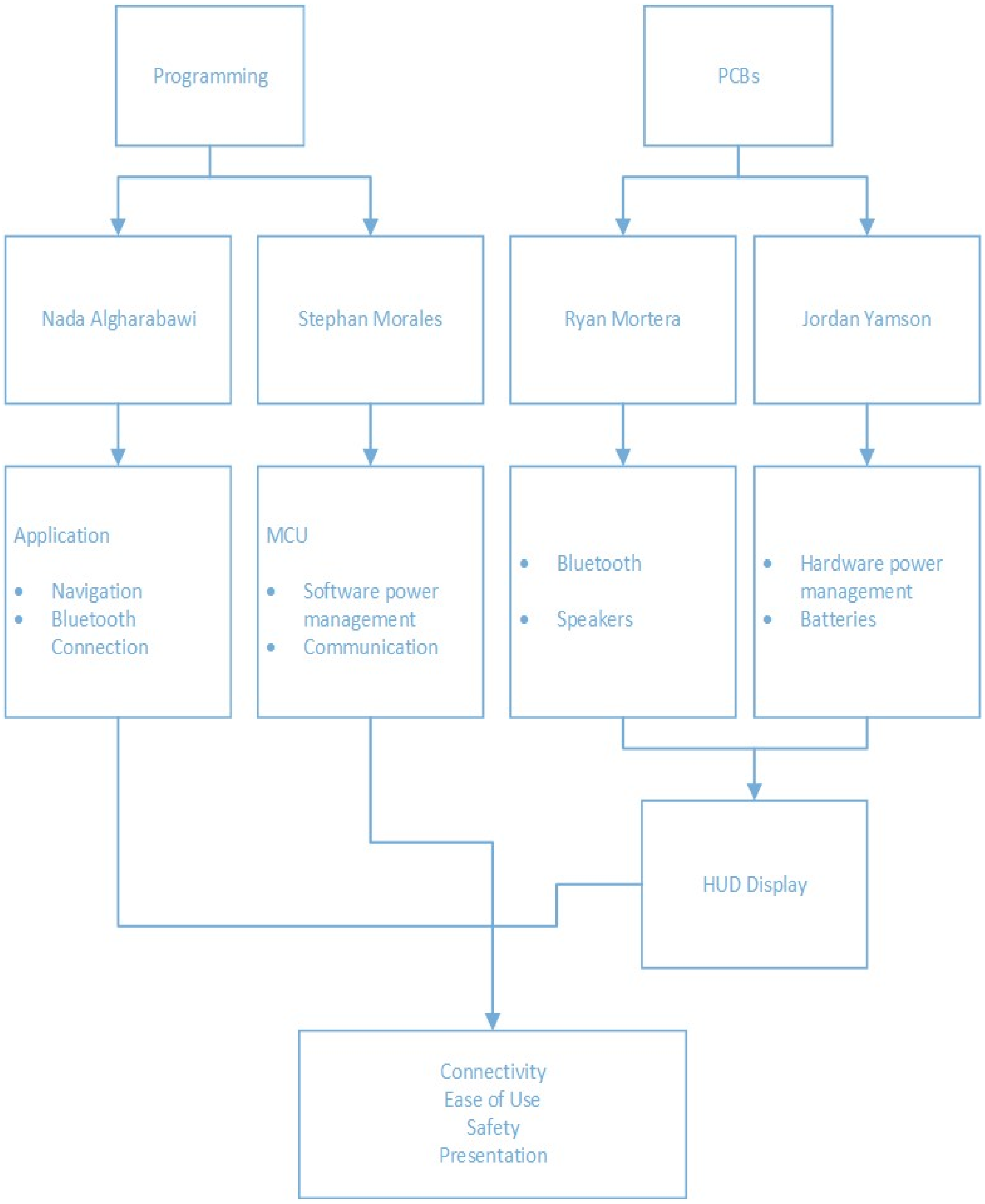


Figure 46 - Labor Division

8.4 Team Organization

To ensure reaching final goal, a plan must be designed and followed by all team members. This plan has to be followed and maintained by all group members to ensure that work is being accomplished. Each team member has some responsibilities that must be done by himself or herself. Moreover, there is some work need to be done by the whole group. In order to ensure work progress, group member has to be in touch to know exactly how the work is going on. These communications can be done by different ways depending on the task and the reason for meeting.

8.4.1 Communication

One of the important things for any team to success is having excellent communication methods. Any team member must be able to reach, share, and review any part of the work related to the project. Our group is basically divided to two teams: hardware team and software team. Some of the work need to be done individually and some of it need to be done by the whole group. In order for the group to continue progressing well in the project, team members have to get access to all the project files, codes, designs, schematic, resources, documentations, and anything else that is necessary for the project. All group members has each other emails and phone numbers. However, that is not enough, group should use something to share files, codes, and reports so they can keep knowing the progress of the entire project parts. Moreover, sometimes it is difficult for some members of the team to meet face-to-face every week because of scheduling conflicts, or traffic issues. Thus, we have decided as one group to choose some communication applications that can facilitate communication between group members. We have created a group messages on our phone so we can remind each other with the important deadlines and schedule meetings. Also, to work on our reports and documents we have used Google's Docs so we can work together on one file or uploading files then review it or take a look on it.

There is a fixed every week meeting that all team members should attend. This meeting is scheduled to be every Thursday two hours before class time on Engineering One. Also, we have scheduled a meeting every week on Friday two hours before class for Senior Design Two. The place can vary during Senior Design Two since there are many things need to be completed. For example, sometimes the meetings will be done on the lab to perform some tested related to the final project deliverables. Other meeting can be done on Engineering One or the library to work on the reports, presentations, and demo.

8.4.2 Information Sharing

In order to be informed with new updates and share information between all group members, we are using Google Drive to upload and share files. Thus, all group members can have access to all files related to the project. In this way we as one group can keep track of all available resources, data, and accomplishments we have done to complete our goal. Also, we can ensure that all required work is going to be accomplished by or before its deadline, so group members can move to the next step or next part. Also, Google Docs gives us a chance to work on documents together so we can add, edit, and review files from any place we want so we can save time and finish work as fast as possible. For Senior Design One most of the group paperwork and researched files have been completed using Google Drive. Also, we are planning to use the same methodology on Senior Design Two to complete our paperwork, presentations, sharing test's results and everything we need to have as one group.

8.4.3 Web Based Git

In order for our team to get continues access to the software part of the project, we will use GitHub. GitHub is a Web-Based Git or a Git repository housing service. This web based graphical interface has many collaboration features include but not limited to building software, reviewing codes, managing projects and it's a free repository at the same time. Our software team members are collaborating effectively using this interface. Thus, they can, access code to review it and edit it from anyplace they want. This saves a lot of time, effort, and for sure any scheduling conflicts. Our work will continue smoothly by having a central location that we can share our project's software code. Also, team member can keep track of the changes and updates so they can always know the progress of the work that they have done. GitHub supports many formats and features beside source code such as Documentation, Graphs, Issue Tracking, Emoji's, and Photoshop. For our project purpose, we will use it mostly for our source code.

One of the Software Team has created an account on GitHub to post the code and all updates related to Software coding. This repository contains all the coding parts related to our application. Since Software team has already started programming our project application, they have added the parts that they have done. Each time they modify or add a new section they always add it to our project GitHub account. Also, they leave a comment next to the new update with the exact time at which they have done these modifications. These comments include a brief description to describe the new updates that has been added recently such as adding a new feature or a new class, or start working on a new part of the application, and fixing some mistakes on one of the classes. In this way, they stay organize and safe their work from any problems that might happen.

8.5 Budget and Finance Discussion

This section will introduce the main issues we have considered when we determined the budget that will cover our project. There is a table that demonstrate all the parts that we have purchased in addition to the price for each part.

8.5.1 Finance

In order to determine the budget, we as a group have done extensive research on the parts that we need for our project. For instance, the person who is responsible for developing Bluetooth module has done research on that part and figure out which device is more applicable for our project based on price, efficiency, safety and other factors. Also, the team member who is in charge of system battery has done research and find out which battery is more practical for our project based on the time it lasts, safety, size, and price. After all of that, we have discussed all options and decided for sure the parts that we are going to integrate into our project. We have calculated the final cost for the project which is in the limits that we have expected. One of the important aspects that we have tried to accomplish in our project is making sure that production cost is low with excellent quality and efficiency so the final product can be affordable to different levels of customers in the market.

8.5.2 Purchased Parts

The below table is a detailed record of the cost analysis for the parts bought for the BRAIN Helmet.

Table 25 - Cost Analysis

	Part Number	Unit Cost ($)	Quantity	Total Cost
Processor	MSP430F5529	8.06	1	8.06
	ATmega2560	11.85	1	11.85
BT Modules	BC127	26.95	3	80.85
	BC127 Breakout	44.95	1	44.95
	Bluefruit LE Friend	15.00	1	15.00
Speaker (x2)	00050016FP035A	3.50	2	7.00
	Com-10722	0.51	6	3.06
Microphone	BOB-12758 Breakout	2.95	2	5.90
Battery	Adafruit 328	15.96	1	15.96
	Ofeely 458292	13.59	1	13.59
LiPo Battery Charger	PRT-102177	7.95	2	15.90
Helmet	SH-FF0016	42.99	1	42.99
Buttons	COM-09339	1.95	0	0
	Tactile Button	0.50	0	0
Potentiometer	COM-09288	0.95	0	0
	COM-09117	2.95	0	0
HUD	LCD-13003	14.95	1	14.95
	DiyMall I2C OLED	8.50	2	17.00
Voltage Regulator	COM-00526	1.95	3	5.85
Tools	Mini-Breadboards	1.00	6	6.00
	Solder-Kit	25.50	1	25.5
	Solder Tip Cleaner	9.14	1	9.14
			SUM =	**343.55**

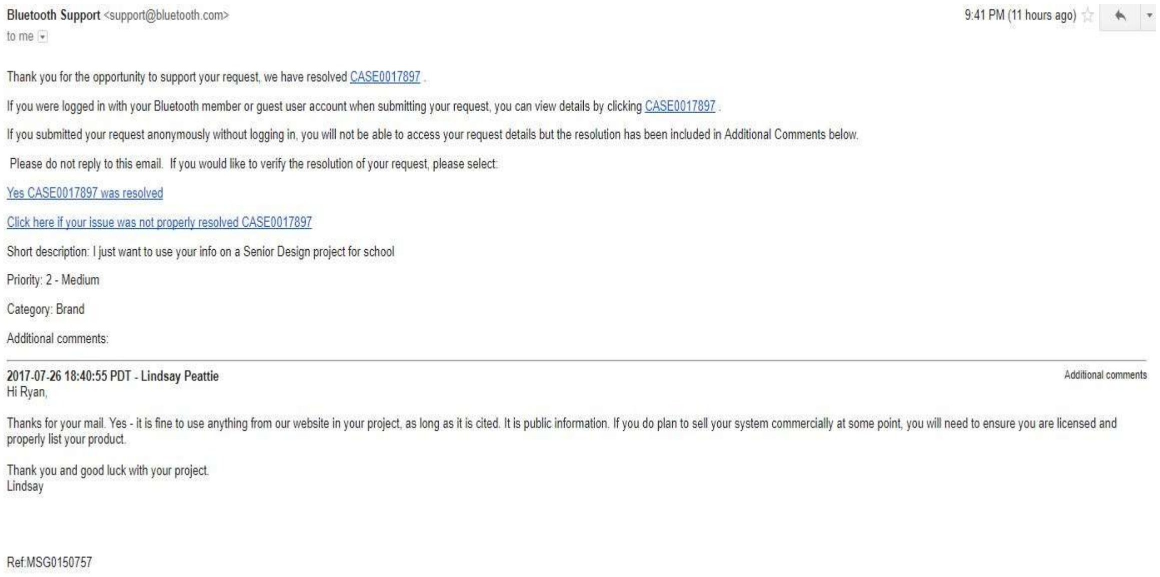
9.0 Conclusion

Overall, the main issue that we are addressing is to ease the user of hands-free navigation and talking while keeping focused on the road. Just like the user operating a motor vehicle that comes with navigation and hands-free talking standard, we offered a cheaper alternative to those who prefer riding on motorcycles. Based on our research, we can make a system that will connect to the user's cellular device to the helmet to listen to music, answering phone calls, voice navigation all into one rechargeable system attached to the back of the helmet. Products in the market today do offer a similar design of what we want to implement, but at a price that is questionable based on our parts selection. The battery life should last for several days of normal use and can also be used as a normal helmet without powering it on. The user interface of the display is a minimalistic style to offer the operator the minimal amount of time looking at the screen to see information available to him. The buttons and potentiometer will be placed around the helmet at spots that the user can easily remember and will be tactile enough to sense the pressed button through the user wearing gloves. Even though there are standards that we must conform to when we design our system in addition to the constraints we encounter, we hope that this system will allow further progression of the advancement of addressing convenient features to motorcyclists.

Appendices

Appendix A - Copyright Permissions

Bluetooth.com Permissions



Bluetooth Support <support@bluetooth.com>
to me

9:41 PM (11 hours ago)

Thank you for the opportunity to support your request, we have resolved CASE0017897 .

If you were logged in with your Bluetooth member or guest user account when submitting your request, you can view details by clicking CASE0017897 .

If you submitted your request anonymously without logging in, you will not be able to access your request details but the resolution has been included in Additional Comments below.

Please do not reply to this email. If you would like to verify the resolution of your request, please select:

Yes CASE0017897 was resolved

Click here if your issue was not properly resolved CASE0017897

Short description: I just want to use your info on a Senior Design project for school

Priority: 2 - Medium

Category: Brand

Additional comments:

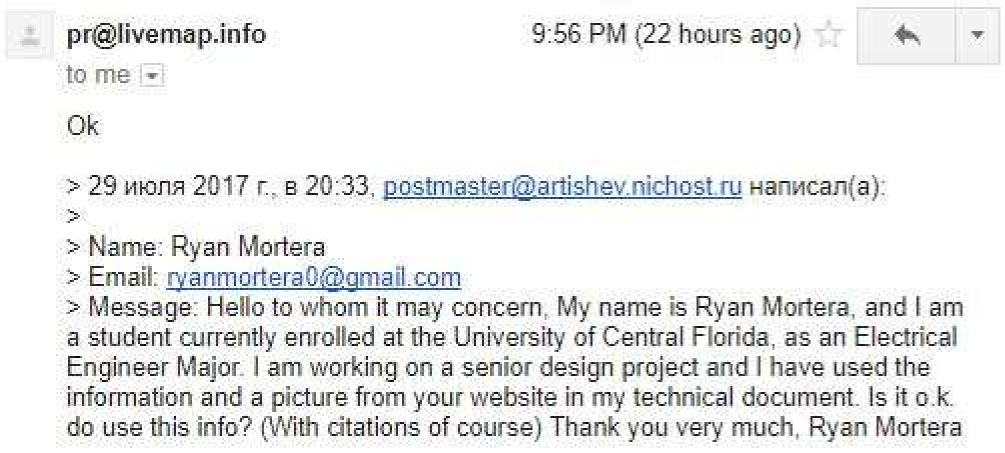
2017-07-26 18:40:55 PDT - Lindsay Peattie Additional comments
Hi Ryan,

Thanks for your mail. Yes - it is fine to use anything from our website in your project, as long as it is cited. It is public information. If you do plan to sell your system commercially at some point, you will need to ensure you are licensed and properly list your product.

Thank you and good luck with your project.
Lindsay

Ref:MSG0150757

Live Map Permission:

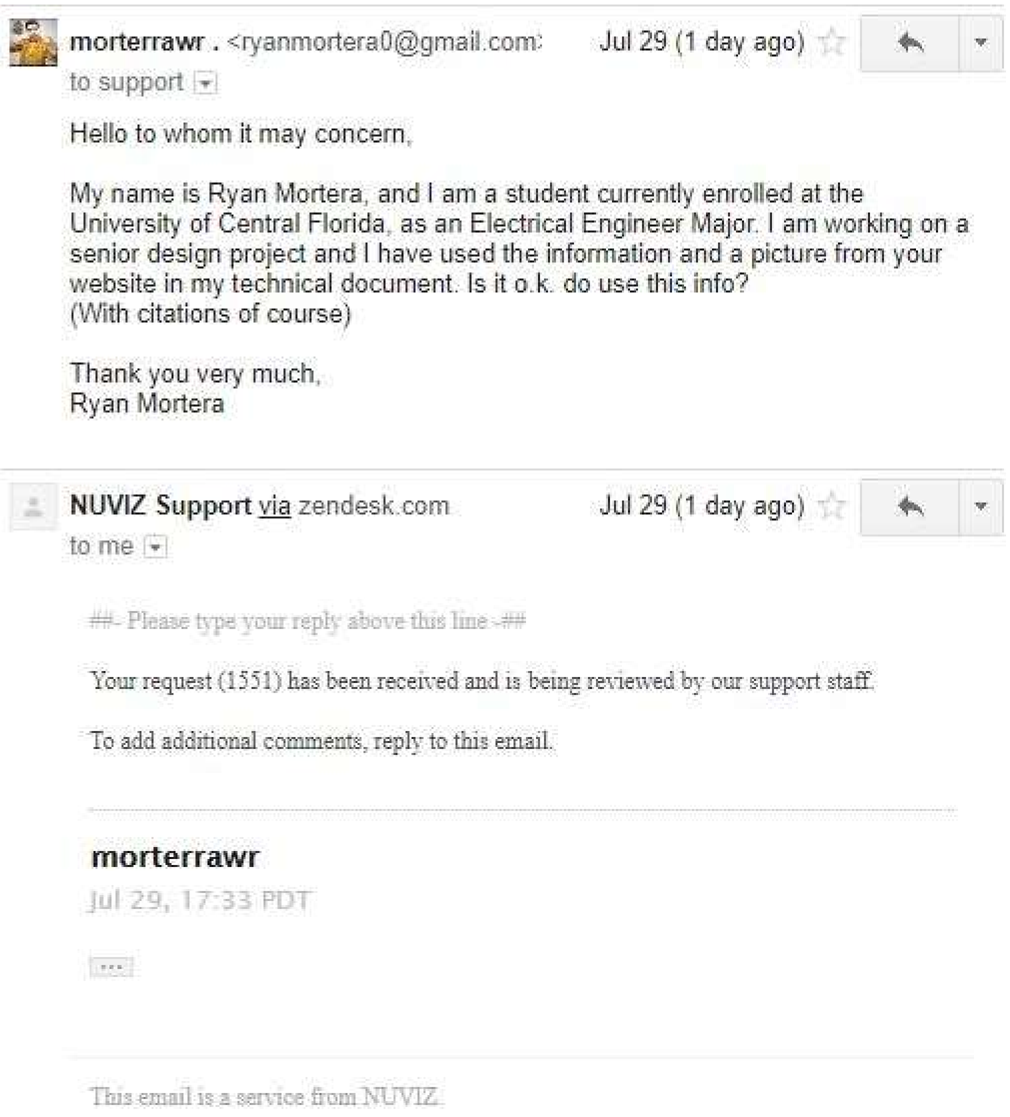


pr@livemap.info 9:56 PM (22 hours ago)
to me

Ok

> 29 июля 2017 г., в 20:33, postmaster@artishev.nichost.ru написал(а):
>
> Name: Ryan Mortera
> Email: ryanmortera0@gmail.com
> Message: Hello to whom it may concern, My name is Ryan Mortera, and I am a student currently enrolled at the University of Central Florida, as an Electrical Engineer Major. I am working on a senior design project and I have used the information and a picture from your website in my technical document. Is it o.k. do use this info? (With citations of course) Thank you very much, Ryan Mortera

NUVIZ Permission:


morterrawr . <ryanmortera0@gmail.com>
Jul 29 (1 day ago)
to support

Hello to whom it may concern,

My name is Ryan Mortera, and I am a student currently enrolled at the University of Central Florida, as an Electrical Engineer Major. I am working on a senior design project and I have used the information and a picture from your website in my technical document. Is it o.k. do use this info?
(With citations of course)

Thank you very much,
Ryan Mortera

NUVIZ Support via zendesk.com
Jul 29 (1 day ago)
to me

##- Please type your reply above this line -##

Your request (1551) has been received and is being reviewed by our support staff.

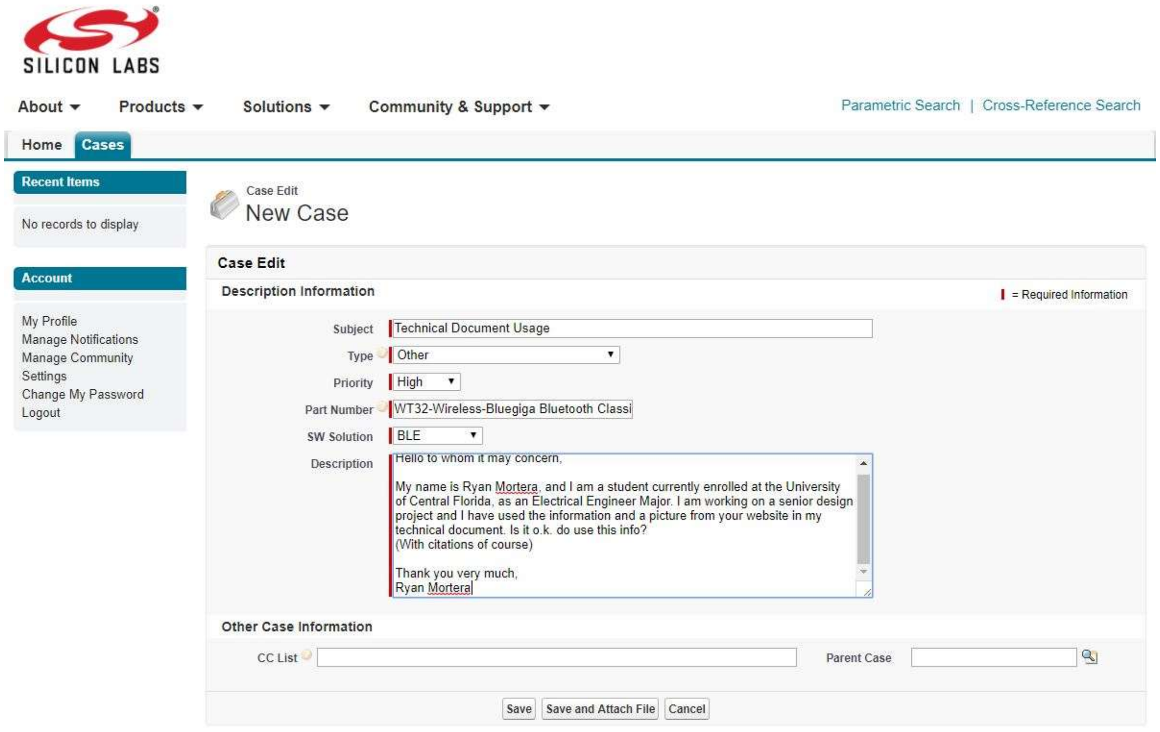
To add additional comments, reply to this email.

morterrawr
Jul 29, 17:33 PDT

...

This email is a service from NUVIZ.

Silicon Labs Permission:



Case Edit

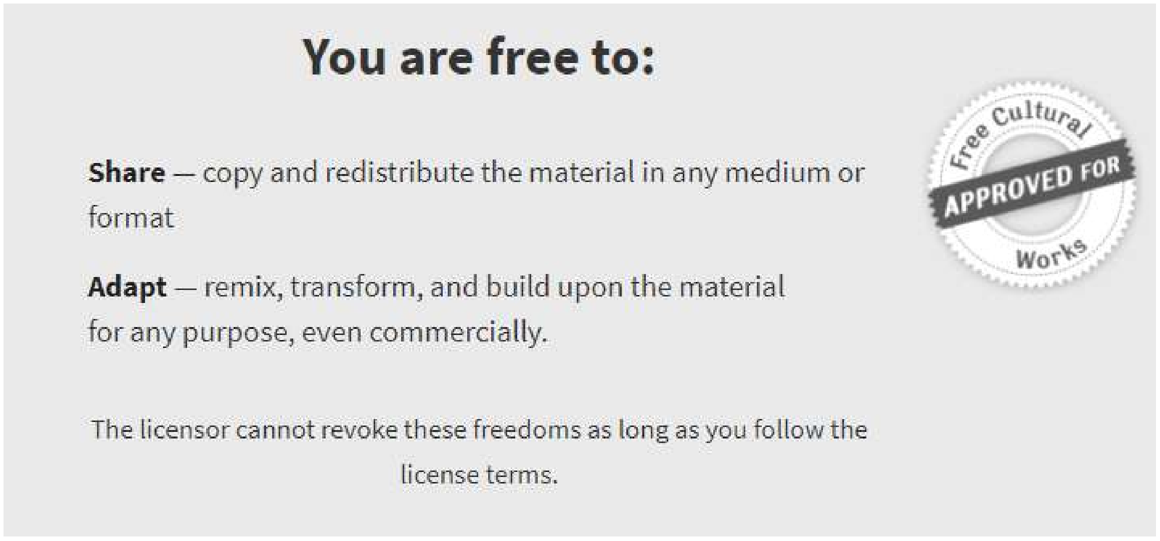
Description Information | = Required Information

Subject	Technical Document Usage
Type	Other
Priority	High
Part Number	WT32-Wireless-Bluegiga Bluetooth Classi
SW Solution	BLE
Description	Hello to whom it may concern, My name is Ryan Mortera, and I am a student currently enrolled at the University of Central Florida, as an Electrical Engineer Major. I am working on a senior design project and I have used the information and a picture from your website in my technical document. Is it o.k. do use this info? (With citations of course) Thank you very much, Ryan Mortera

Other Case Information

CC List		Parent Case	

Sparkfun’s images and tutorial material are all CC BY-SA 4.0:



You are free to:

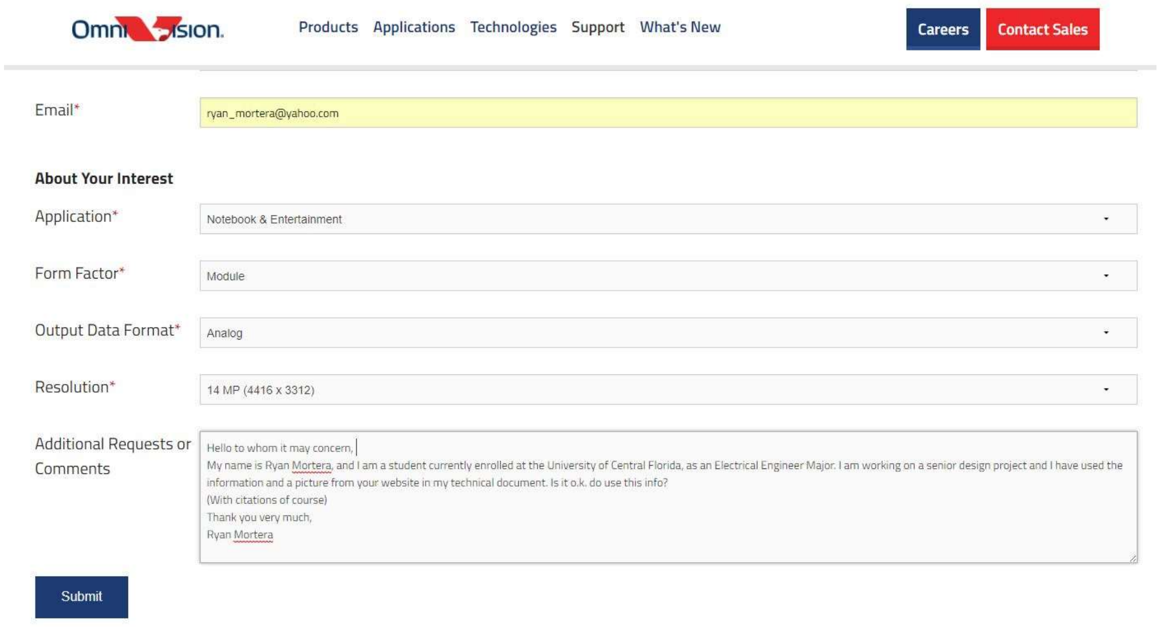
Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

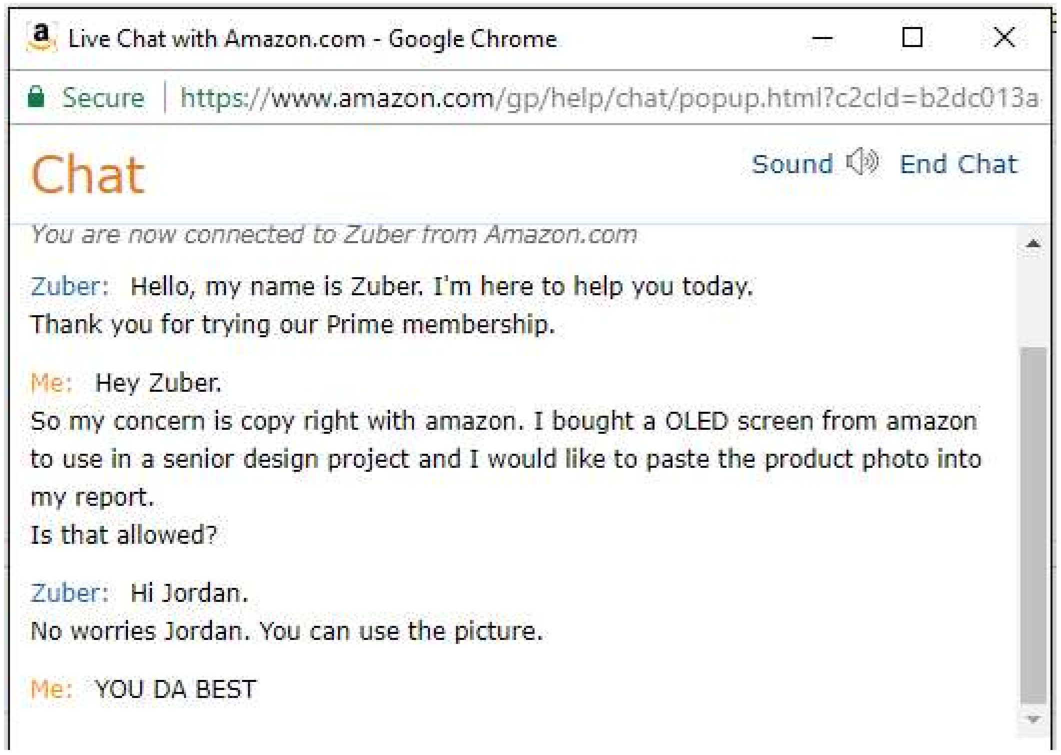
Free Cultural
APPROVED FOR
Works

Omni Vision Permission:



Email*	ryan_mortera@yahoo.com
About Your Interest	
Application*	Notebook & Entertainment
Form Factor*	Module
Output Data Format*	Analog
Resolution*	14 MP (4416 x 3312)
Additional Requests or Comments	Hello to whom it may concern, My name is Ryan Mortera, and I am a student currently enrolled at the University of Central Florida, as an Electrical Engineer Major. I am working on a senior design project and I have used the information and a picture from your website in my technical document. Is it o.k. do use this info? (With citations of course) Thank you very much, Ryan Mortera

Amazon Permission:


Live Chat with Amazon.com - Google Chrome
Secure
https://www.amazon.com/gp/help/chat/popup.html?c2cId=b2dc013a
Chat
Sound
End Chat

You are now connected to Zuber from Amazon.com

Zuber: Hello, my name is Zuber. I'm here to help you today.
Thank you for trying our Prime membership.

Me: Hey Zuber.
So my concern is copy right with amazon. I bought a OLED screen from amazon to use in a senior design project and I would like to paste the product photo into my report.
Is that allowed?

Zuber: Hi Jordan.
No worries Jordan. You can use the picture.

Me: YOU DA BEST

Appendix B - Datasheets Appendix

Roving Networks RN52 Datasheet
OVC3860 Datasheet
WT32 Datasheet
ATmega2560 Complete Datasheet

ATmega2560 Summary Datasheet
DIYMALL 0.96” OLED Display Datasheet

Appendix C - Table of Abbreviations

Acronym	Definition
IntelliJ IDEA	java Integrated Development Environment for developing computer Software
SDK	Software Development Kit
JVM	Java Virtual machine.
QEMU	Quick Emulator
ADT	Android Development Tools
IDE	Integrated Development Environment
JetBrains	Which is a software development company whose tools are targeted towards software developers and project managers.
GPS	Global Positioning System
ARM	Advanced RISC Machine
NDK	(Native Development Kit) is a tool that allows you to program in C/C++ for Android devices
AVD	Android Virtual Device
PCB	Printed Circuit Board
HOQ	House of Quality
LCD	A liquid-crystal display
APK	Android Packet Kit

Acronym	**Definition**
API	Application Programming interfaces
HUD	Heads-Up Display
GUI	Graphical User Interface
Li-Po	Lithium Ion Polymer/Lithium Polymer
BLE	Bluetooth Low Energy
GATT	Generic Attribute Profile

Appendix D - References

Aydin, M. (2012). Android 4. [electronic resource]: *new features for application development; develop Android application using the new features of Android Ice Cream Sandwich*. Birmingham, UK: Packt Pub., 2012.

Bluetooth Technology from https://www.bluetooth.com/what-is-bluetooth-technology/how-it-works/le-broadcast

Burd, Barry. HOW TO TEST AN ANDROID APP ON A REAL DEVICE. (30 July 2017) Retrieved from http://www.dummies.com/programming/java/how-to-test-an-android-app-on-a-real-device/

Chee Oh, C., Yilun, H., & Hoe Kyung Jung1, h. (2016). Augmented Reality navigation System on Android. *International Journal of Electrical $ Computer Engineering (2088-8708)*, 6(1), 406-412.

Code Style for Contributors. Retrieved from https://source.android.com/source/code-style

Erb. T. O., & Doda, N. (1989). *Team Organization: promise-practices and possibilities.* Washington, DC: National Education Association, c1989.

FCC standards from https://www.fcc.gov/oet/ea/rfdevice#block-menu-block-4

Freedom of Information act for Government Documents https://www.hq.nasa.gov/office/pao/FOIA/

Garrett, J. J. (2003). *Elements of user experience: User-centered design for the web*. New York: American Institute of Graphic Arts; Indianapolis: New Riders, 2002.

Google Maps API. Retrieved from https://developers.google.com/maps/documentation/directions/

GOT Standards from https://one.nhtsa.gov/people/injury/pedbimot/NoMigrate/fmvss218.htm

Hicks, B. J., Medland, A. J., Mullineux, G., "The representation and handling of constraints for the design, analysis and optimization of high speed machinery", *Artificial Intelligence for Engineering Design, Analysis and Manufacture (AIEDAM)*, **20** (2006) 313-328.

IEE standards from IEEE standards website http://standards.ieee.org/ (Must sign up and use account to reach standards)

Marcus, A. SIGGRAPH 93 tutorial notes: Graphic Design for User Interfaces. August 1993.

Microchip processor product page retrieved from
http://www.microchip.com/wwwproducts/en/ATmega2560

PCB Design Process retrieved from
https://en.wikipedia.org/wiki/Printed_circuit_board#Copper_patterning

PCB Manufacturing Process retrieved from
https://www.pcbcart.com/article/content/PCB-manufacturing-process.html

Product page for Arduino development board retrieved from
https://store.arduino.cc/usa/arduino-mega-2560-rev3

Radio Frequency Radiation from
https://hps.org/hpspublications/articles/rfradiation.html
Shari Lawrence Pfleeger and Joanne M. Atlee, "Software Engineering: Theory and Practice", 4th Edition, Prentice Hall, 2010.

Rouse, Margaret. constraint (project constraint). Retrieved from
http://whatis.techtarget.com/definition/constraint-project-constraint

Snef standards retrieved from
http://www.smf.org/docs/articles/dot

Solder Mask definition retrieved from
https://en.wikipedia.org/wiki/Solder_mask

Sollenberger, Kyle. (2012, August 7). 1o User Interface Design Fundamentals. Retrieved from http://blog.teamtreehouse.com/10-user-interface-design-fundamentals.

Texas Instruments MSP430F5529 Product page retrieved from
http://www.ti.com/tool/msp-exp430f5529lp

Texas Instruments MSP432P401R Product page retrieved from
http://www.ti.com/tool/msp-exp432P401R

Texas Instruments WEBENCH Power Archite (2017)
https://webench.ti.com/webench5/power/webench5.cgi?origin=ti_panel&app=powerarchitect&lang_chosen=en_US

Virgillito, Dan. (2016, May 18). How to Choose the Right UI Design Color for Your WordPress Site. Retrieved from https://www.elegantthemes.com/blog/tips-tricks/how-to-choose-the-right-ui-design-colors-for-your-wordpress-site.

Wienclaw, R. A. (2015). Project Management. *Research Starters: Business (online Edition),*

Williams, John. Moore. (2017 May 30). 10 essential UI (user – interface) Design Tips. Retrieved from https://webflow.com/blog/10-essential-ui-design-tips.

(2016, January 19). 9 Differences between IOS and Android App Development. Retrieved from https://www.cleveroad.com/blog/9-differences-between-ios-and-android-app-development

(2016). Atmel Studio 6 Integrated Development Environment. Retrieved from http://ww1.microchip.com/downloads/en/DeviceDoc/8487B-Studio6-E-A4-0912_LR.pdf

(25 July 2017) Wikipedia C++ Standards retrieved from https://en.wikipedia.org/wiki/C%2B%2B

(25 July 2017) Wikipedia C++03 Standard retrieved from https://en.wikipedia.org/wiki/C%2B%2B#Standardization

(25 July 2017) Wikipedia C++11 Standard retrieved from https://en.wikipedia.org/wiki/C%2B%2B11

(25 July 2017) Wikipedia C++14 Standard retrieved from https://en.wikipedia.org/wiki/C%2B%2B14

(25 July 2017) Wikipedia C++17 Standard retrieved from https://en.wikipedia.org/wiki/C%2B%2B17

(27 July 2017) Wikipedia Software prototyping retrieved from https://en.wikipedia.org/wiki/Software_prototyping

(27 July 2017) Helmet design by Matej Pezer Retrieved from https://grabcad.com/library/helmet-design-by-matej-pezer-1

(18 June 2017) User Interface Guidelines Retrieved from https://developer.android.com/design/index.html

(18 June 2017) AOSP Java Code Style for Contributors Retrieved from https://source.android.com/source/code-style

(31 July 2017) Activity Lifecycle Retrieved from https://developer.android.com/reference/android/app/Activity.html#ActivityLifecycle

(4 July 2017) Learn Git and GitHub without any code! Retrieved from https://github.com/