Smart Motorcycle Helmet Device

A Major Qualifying Project Report

Submitted to the Faculty of Worcester Polytechnic Institute in partial fulfillment of the requirements of the Degree of Bachelor of Science by:

Lindsay Ambrosino, Rosina Comatas, and Alexander Mondro

Date: 4/28/2022

Advisors: Professor Stephen J. Bitar & Professor Songbai Ji

Acknowledgments

We would like to thank Professors Stephen Bitar and Songbai Ji for their help throughout this project. Without their wisdom and constant encouragement, this project's success would not have been possible. Our weekly meetings with them kept us on the right track and offered new angles when we got stuck on a problem. We would also like to thank Professor Fiona Levey and Meredith Merchant for their help in setting up our pendulum testing rig. They were instrumental in guiding us through the procedures of locating, setting up, and using the device in a safe and effective manner.

Abstract

This project aims to reduce the fatality of motorcycle crashes through early detection and automatic notification. An accelerometer, photoplethysmograph, and Bluetooth modules were integrated onto a motorcycle helmet to sense crash-related impacts and then notify emergency services with a victim's GPS location and heart rate using a smartphone app. Testing indicates that the device is capable of reliably detecting impacts above a defined threshold as well as accurately monitoring heart rate. It is recommended that future work should focus on designing a crash-proof enclosure for the device as well as refining the crash detection algorithm.

Executive Summary

Introduction and Background

According to the National Highway Traffic Safety Administration (NHTSA), motorcycle riders are 27 times more likely to experience a fatal accident than car drivers (iii, 2021). Additionally, it was confirmed that longer EMS response times are associated with higher rates of crash mortality, as visualized in Fig. E1. It is also more common for longer response times in rural areas where crashes take longer to be reported (Byrne, 2019). By reducing the response time to an accident, mortality rates may decrease.

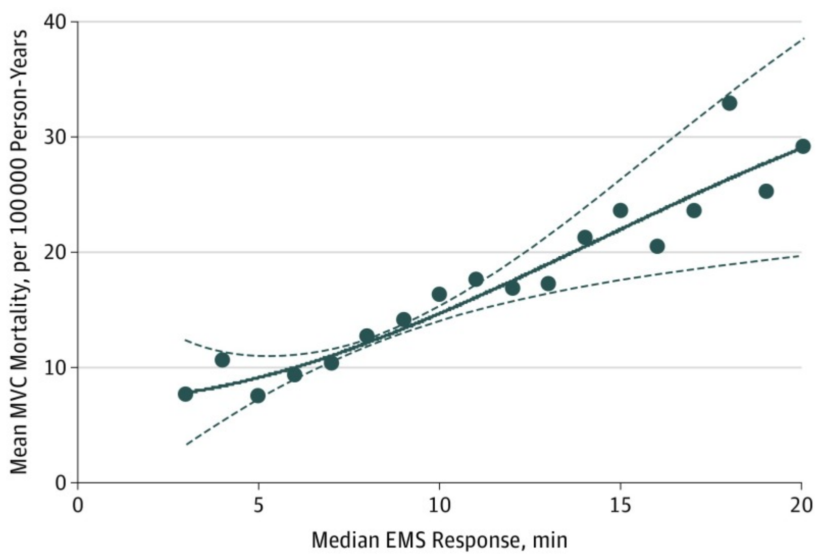

Figure E1: Graph depicting the trend of the mortality of motor vehicle accidents increasing as the response time of emergency services increases (Byrne, 2019).

To determine if a crash has occurred, a MEMS accelerometer can be implemented to measure changes in acceleration through the piezoelectric effect of crystal vibrations within the sensor (Goodrich, 2013). To aid in monitoring the vitals of crash victims before emergency services arrive, their heart rate can be found through photoplethysmography sensors (PPGs) that measure the change in light as blood moves through the veins (Ghamari, 2016). Our project aims to combine these two sensors to both detect crashes and then alert emergency services of the crash and wellbeing of the victim through a device mounted to the back of a motorcycle helmet.

Methods

In order to create a device capable of detecting and reporting motorcycle crashes, the team developed a four-pronged approach to finalize the design. The components needed to complete the project were various input power and regulators, creating an algorithm for the analog PPG, setting a threshold for the accelerometer in SPI, and creating an app to take in the data from the sensors and communicate it to emergency services. Once implemented, the components were tested on their accuracy and reliability. This design is depicted in the block diagram of the device in Fig. E2 below.

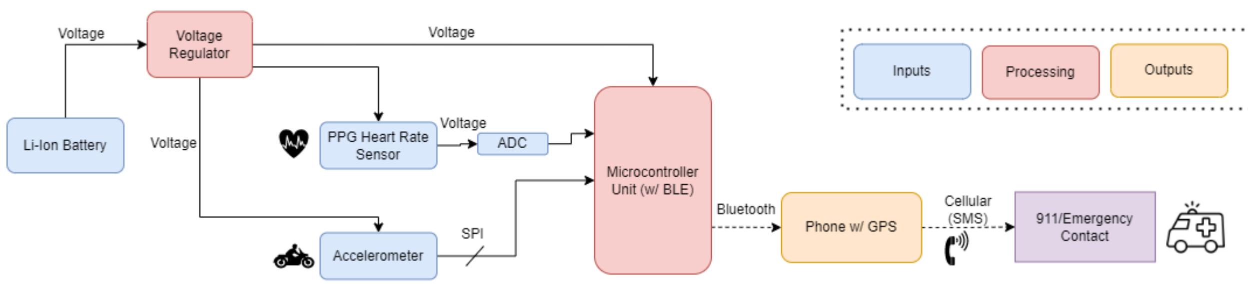

Figure E2: Block diagram of the inputs, processing components, and outputs of the smart motorcycle helmet device.

Input Power Hardware

To power the device, a constant input of 3.3V is required for both sensors and the Nordic Semiconductors nRF52840 Bluetooth microcontroller. A lithium ion battery operating at 7.2V, 3250mAh, and 23.4Wh was selected as the main power source as it has a large power capacity, can be recharged, has decreased safety risks, and is easily portable. To reduce the input voltage to 3.3V, a linear voltage regulator was utilized.

Accelerometer

To determine the acceleration of the user, a 3-axis accelerometer was utilized where the sensor outputs the different accelerations experienced in hex. The accelerometer is able to detect acceleration of +/-100Gs while sampling at a rate of 1024Hz. Coding in SPI, we are able to set an acceleration threshold that needs to be passed in order for a crash to be detected. Based on previous literature on the forces experienced during motorcycle braking and crashing, we set our threshold to 8Gs in all axises of acceleration. To test both the accuracy and reliability of the accelerometer and crash detection, a pendulum loaded with up to 15lbs was swung at the motorcycle helmet. Accuracy was calculated by determining the difference between the expected and experimental results while the reliability of the crash detection was tested by seeing how many times a crash was detected after setting our threshold.

PPG

An analog PPG was implemented to determine the heart rate of the user. The PPG rests inside of the helmet against the temple of the user. The algorithm created samples of the heart rate for ten seconds every minute to determine the average peak reached. It then counts the amount of times that 80% of the average is reached and multiples that value by six to determine the beats per minute. Different materials over the PPG as well as different pressures of the PPG against the skin were tested to determine ways to enhance user comfort. Additionally, the algorithm we created to determine heart rate was tested against the Apple Watch algorithm to determine the accuracy of our PPG device.

Mobile App Development

To transmit the data collected from the sensors to emergency services, we developed an app for iOS using Swift. The data from the sensors were collected by the Bluetooth microcontroller where the data is then sent to the app when a crash is detected. When a crash is detected, the app sends a notification to emergency services and the user's emergency contact through a third party API, Twilo, of the location of the user using the phone's GPS as well as the user's heart rate. To test the effectiveness of the app's ability to transmit data collected, the team sent data from the app to our personal phones to determine if the information could be received in a way that is helpful to both emergency responders and the user.

Results

The accelerometer was tested in a series of 14 trials, with 7 trials adding no additional weight to the pendulum, 4 trials adding an additional 5lbs to the pendulum, and 3 trials adding

10lbs to the pendulum. For each of the trials, the pendulum was released at an angle of about 50° before colliding with the helmet as the accelerometer collects data. For each of the trials run, the accelerometer correctly registered the collision through the spikes in acceleration in the X and Y axis, as seen in Fig. E3. Following these preliminary tests of determining the functionality of the accelerometer, the device's ability to register a crash was tested. The same procedure was repeated, but if the acceleration surpassed a set threshold of 5Gs, the microcontroller would be notified that a crash had occurred. In five out of five of these trials, the device successfully registered a crash as occurring. Based on these results, the accelerometer is acceptably and reliably functioning for use in the device.

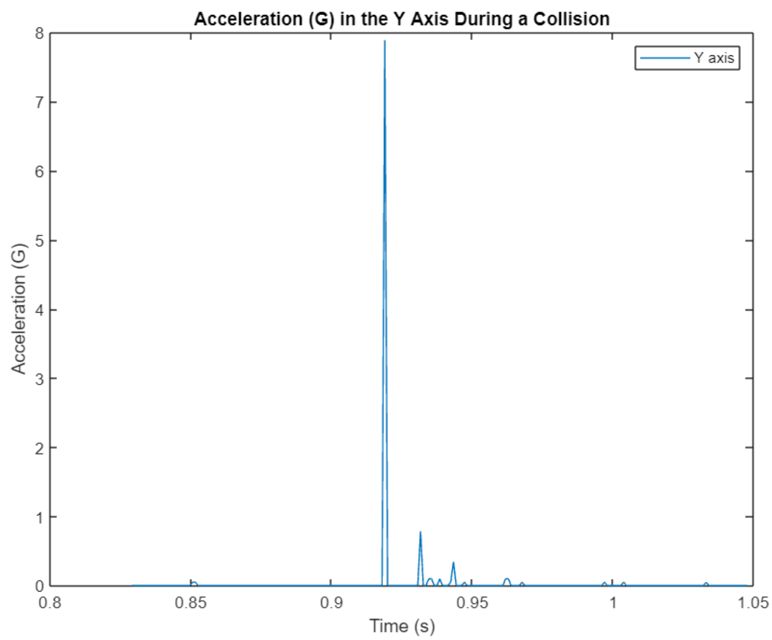

Figure E3: Graph depicting the acceleration (G) of a motorcycle helmet during a collision with a pendulum weighing 10lbs.

To test the accuracy of the PPG, the heart rate of the user collected from the device was compared against the heart rate collected from an Apple Watch. Both the user's resting heart rate and the user's heart rate after a minute of vigorous exercise were tested in three trials each. For every trial, the heart rate from our device was within 5 BPM of the Apple Watch's heart rate. For the purpose of the device, the accuracy of the PPG heart rate algorithm is acceptable for use in the device. Additionally, for user comfort, different materials were placed as barriers between the device and skin. It was found that single layers of lighter fabrics, such as white cotton or linen, allowed an accurate heartbeat signal to be generated while darker and thicker fabrics created noisy heartbeat signals. Based on these findings, a layer of white cotton between the device and the user's skin can be used in the final implementation of the device to enhance user comfort while wearing the device.

Recommendations and Conclusions

Overall, a successful prototype for a smart motorcycle helmet device was created over the course of the completion of this project. Both sensors implemented work accurately and reliability. Moving forwards, recommendations can be made to clean up the prototype into a finalized and marketable device. The most pressing task for future work is in creating a more compact version of the device that is able to fit within a crash-proof enclosure. In its current state, the device is very bulky and can easily be damaged. By creating a printed circuit board (PCB) of the device, a small crash-proof enclosure can be created to encase the device and attach to any helmet. By creating a device that is attached to any helmet, motorists are able to keep their beloved helmets while also cutting costs, as similar products are a smart device embedded within an entirely different helmet. Another recommendation is for future work to devise a more sophisticated crash detection method involving multiple accelerometers and a gyroscope to account for the numerous types of crashes that can be experienced during a motorcycle crash.

The prototype created at the completion of this project is a device that the team is incredibly proud to have created. The goal of this project is to design a device that will easily attach to any commercial motorcycle helmet and help motorcycle drivers get help faster during a crash by detecting the crash, alerting Emergency Services, and reporting the heart rate of the device wearer to first responders. The intention is for 911, or other emergency services or contacts, to be alerted of a crash faster than typical when a motorcyclist crashes without another car around or if any other drivers are also hurt and unable to call. Through the recommendations listed above, the device has real potential to save many lives as motorists experiencing accidents in more rural areas now have a reliable method to get help in the event of a crash.

Table of Contents

Table of Figures

1.0 Introduction

The purpose of this project is to reduce the fatality of motorcycle crashes through early crash detection and automatic alerts to emergency services. This will be achieved through the implementation of an accelerometer, PPG sensor, and Bluetooth modules mounted to the back of any motorcycle helmet. The accelerometer coding in SPI is used to track the acceleration of the motorcycle, and, if the acceleration surpasses a set threshold of 8gs, detect crashes. The analog PPG will be mounted to the inside of the helmet against the temple to track the rider's heart rate. In the event of a crash, the rider’s heart rate and the updated status of a motorcycle crash occurring are sent over Bluetooth to the user’s smartphone app for iOS, which then alerts emergency services and emergency contacts of the accident.

Through the completion of this project, the lives of thousands of motorists can be saved. By reducing the time to alert emergency services through automatic reporting of motorcycle accidents, the fatality of these crashes can be decreased as motorists receive the assistance they desperately need at a quicker rate. This report will go throughout the process taken to complete this project, starting with the required background knowledge on all of the sensors and Bluetooth modules used. The objectives and constraints are then listed to aid in understanding the thought behind the methodology in the following section. The implementation of the methodology as well as the testing and results of the device components are then included. Finally, the conclusions of the report and areas for future work are discussed.

2.0 Background/Literature Review/Applicable Standards

To understand why this project is worth being undertaken, it is first important to look into the safety issues currently plaguing motorcyclists. This section examines the data surrounding motorcycle crashes and safety as well as general information on the sensors necessary to complete the project.

2.1 Motorcycle Crashes

According to the National Highway Traffic Safety Administration (NHTSA), motorcycle riders are 27 times more likely to experience a fatal accident than car drivers, and in 2019 there were 5,014 motorcyclists in the US who died in motorcycle crashes (iii, 2021). There are many factors that can contribute to the fatality of these crashes, and one major factor is the Emergency Medical Services (EMS) response time. A study was done to analyze the correlation between EMS response times to crash site and fatality using data obtained in the calendar years 2013-2015 across 2268 counties in the US. After accounting and adjusting for possible confounding variables, such as rurality, emergency medical service on-scene EMS, transport times, trauma resources, and traffic safety laws, it was confirmed that longer EMS response times are associated with higher rates of crash mortality (Byrne et. al., 2019). Figure 1 below shows the crude association between these variables

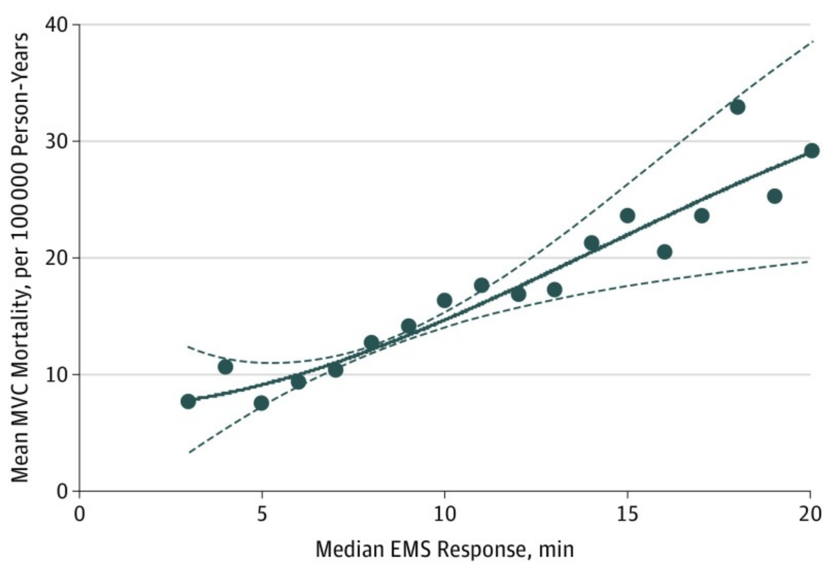

Figure 1: Mean MVC Mortality vs. Median EMS Response (Byrne et. al., 2019)

Some of the aforementioned possible confounders, on-scene times, transport times, access to helicopter EMS, and traffic safety laws were not proven to correlate with mortality to the same degree that EMS response time did. Response time did, however, vary between rural/wilderness counties and urban/suburban counties, with emergency services taking three additional minutes on average to reach a crash site in more rural areas.

While the study focused on the response of EMS once they had been contacted, any delay in contacting emergency services would likely have the same effect on mortality, as the study identified the cause of the correlation as being the time-dependency of severe trauma. Consequently, it can be concluded that immediate contact of EMS after a crash gives those involved the best chance for survival.

An additional study in Australia investigated the effectiveness of Automatic Crash Notification (ACN) in the reduction of road fatalities. The “golden hour” is often discussed in the medical field, which indicates that the first sixty minutes following a multiple trauma injury are crucial, and the rate of mortality increases by up to 50% if restorative care is not delivered within this initial time frame. ACN has the ability to decrease the response time of EMS and therefore increase the likelihood of crash victims receiving care within the golden hour. In comparison with a manual call to emergency services, ACN eliminates both the decision and contact phases. The study defined the decision phase as the time between the initial crash and the decision of a driver, passenger, or witness that medical services are needed. The contact phase is then defined as the time period following this decision when a phone or other communication method is located to be used. When these phases become automatic, the time is greatly decreased and the The study reported that using ACN could prevent nearly 11% of all passenger vehicle occupant fatalities currently reported on Australian roads (Lahausse et. al., 2008).

2.2 Current Solutions

The idea of a smart motorcycle helmet, as well as related technology, has been executed by engineers in the past. One relevant example is a smart motorcycle helmet created by Ty Uehara, a student at the University of Hawaii at Manoa, in response to his friend being unable to call for help after a nearly-fatal motorcycle crash (Ebrahimji, 2019). His design, the ConTekt helmet, uses a GPS sensor, accelerometer, gyroscope, and pressure sensors to detect impact and alert emergency services when a motorcyclist is unable to access their phone to do so. The helmet uses the multiple sensors to detect crashes in different ways – changes in acceleration, velocity, orientation, etc. – and requires at least two of them to be triggered for emergency services to be notified (Leech, 2019).

Another smart helmet was developed in the Philippines as the popularity of motorcycles has increased due to their cost-efficiency and speed navigating through traffic. Similar functionality was included in the helmet with GPS tracking, crash detection, alerting to EMS and family members, and the addition of an alcohol level sensor and alert system. The helmet prototype was realized using both a Raspberry Pi and an Arduino and also featured a anti-theft capability using the external GPS module (Tayag et. al., 2019).

Another relevant example of similar technology comes from studies on measuring heart rate and SpO2 from the forehead inside a helmet for military applications. In a comparison between PPG data from the wrist and from the forehead, the signals obtained from the forehead were overall stronger and less noisy, while also requiring far less contact pressure (Mendelson & Pujary, 2003). The effect of motion artifacts on the SpO2 signal from the forehead were also investigated; subjects were fitted with a helmet and the signals were analyzed during rest, talking, head movements and vehicle motion. When compared to identical measurements being taken at the finger, jaw, and chin, the forehead provided the least variability throughout all movements and was most accurate in measuring SpO2 from a helmet-embedded device (Johnston et. al., 2004).

2.3 Sensors

In the event of a crash, it is important to know several key factors of the accident involving both the health and safety of the cyclist as well as the severity of the crash in order to best assist at the scene of the accident. Through a series of sensors, data on the speed the cyclist was going, the change in acceleration at the impact of the crash, and the heart rate variability and blood pressure of the cyclist can be collected to better understand the full scope of the motorcycle accident

2.3.1 Crash Detection with Accelerometers

When determining if a crash has occurred, the most crucial detail is the change in speed at the moment of collision. One way of measuring the change of speed is through the use of an accelerometer. An accelerometer is a sensor that measures either static or dynamic acceleration through the piezoelectric effect, where static acceleration is found in constant forces like gravity and friction while dynamic acceleration is found in vibrations or shocks (Goodrich, 2013). The piezoelectric effect occurs when microscopic crystals are put under stress from accelerative force, creating voltage. This voltage is processed by the accelerometer to aid in determining both the velocity and orientation of the measurand (Omega, 2021). The voltage created is amplified and then calibrated to gravitational units, a unit that is proportional to force (Brown, 2011). Accelerometers typically come as either high or low impedance devices. The high impedance accelerometers are connected directly to the device and produce an electric charge. The charge produced usually requires manipulation through both code and instrumentation to properly function. Additionally, this type of accelerometer can tolerate high heats. The low impedance accelerometers are more common in the market. They function similarly to the high impedance option, but come with a built in micro-circuit and transistor to convert the high charge to a lower voltage that a wider range of electrical devices can tolerate (Omega, 2021).

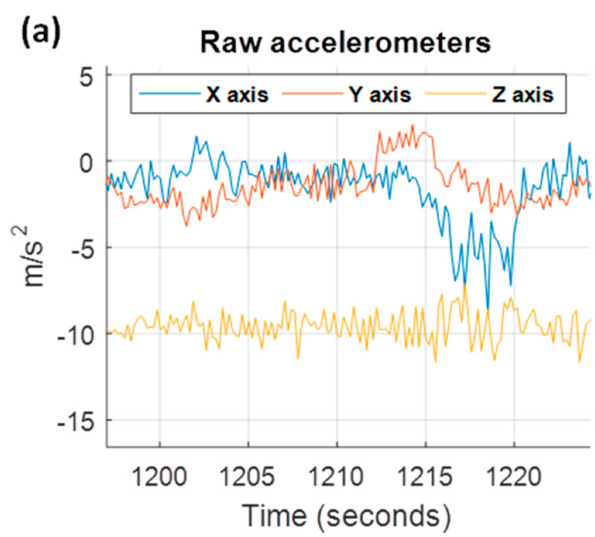

Figure 2: the raw signal from an accelerometer attached to a vehicle as it makes two turns at around 1215s, as depicted by the decrease in acceleration in the X and Y axis (Hernández, 2018).

Accelerometers can be used to detect acceleration on the three axises of X, Y, and Z, as pictured in Fig. 2. As the subject slows in a certain axis of motion, the acceleration detected for that axis will decrease. The same applies inversely if the subject increases in a specific axis.

Velocity can also be determined from the accelerometer by taking the integral of the acceleration signal, due to acceleration being the first derivative of velocity and second derivative of displacement (Brown, 2011).

2.3.2 Heart Rate Monitoring with PPG

Heart rate sensors are an innovative and discrete method for researchers and physicians alike when monitoring heart rate variability of subjects. A photoplethysmograph, or PPG, is able to monitor both by measuring the change in light as the blood moves through the veins. As blood is moved through the body, the pressure inside of the veins change, causing the skin to pulse with each heartbeat. By reflecting light off of the skin, the PPG is able to detect the change in light refraction as the skin moves with each heartbeat (Ghamari, 2018).

The PPG can receive the altered light while in either transmission mode or reflectance mode. Transmission mode is when tissue such as the finger or earlobe separates the light source and the detector. In this mode, the amount of pressure applied to the site is crucial as too much pressure can slow the peripheral blood volume which leads to inaccurate readings. Reflectance mode has the light source and detector on the same side of the tissue, with the light source bouncing off of the tissue to the detector. Reflectance mode is often used on the forehead, wrist, ankle, forearm, or torso (Ghamari, 2018).

PPG's utilize two different types of light sources: infrared LEDs (IR-LED) and green LED's. IR-LEDs are most often used to measure blood flow in areas with a greater concentration of blood. Green LEDs are the most common in PPGs as the light wavelength can penetrate more deeply into the body's tissues, generating more accurate measurements. These measurements can be used to determine the heart rate variability, blood pressure, and the amount of oxygen in the blood (Ghamari, 2018).

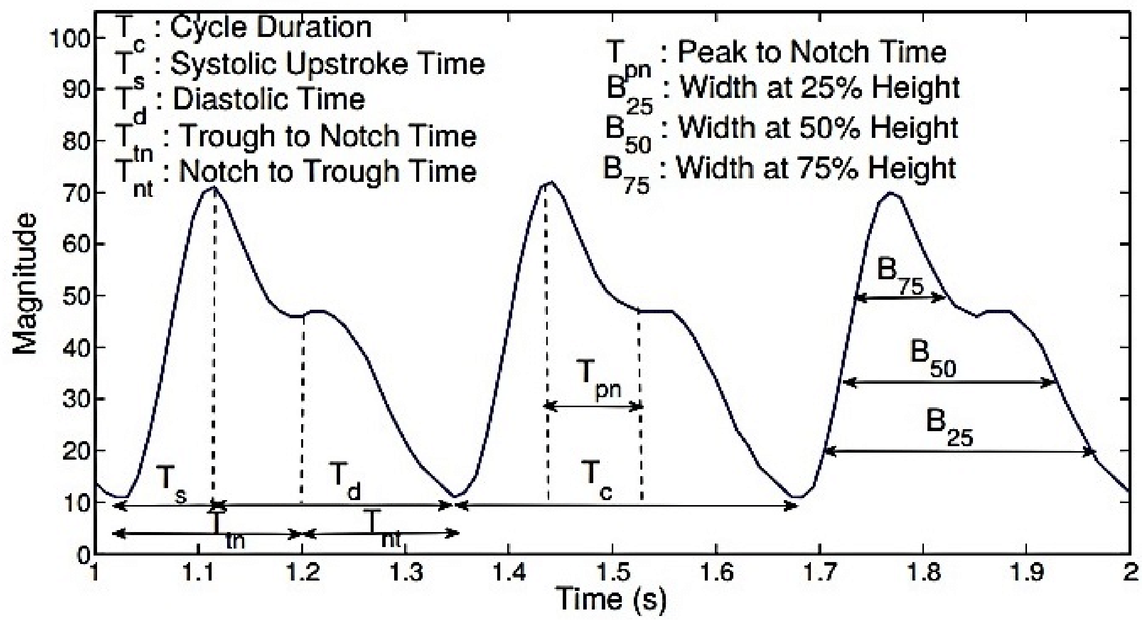

Figure 3: The signal from the PPG depicts the cycle duration, diastolic time, trough, diastolic notch, and bandwidth at varying peak amplitudes (Tjahjadi, 2020).

The signal received from the PPG as the light source is refracted to the detector is depicted in Figure 3. In this graph, the heart rate variability is found as either the time between the two peaks or the time between the troughs, labeled as Tc. The increase of the signal from the minimum in the trough to the peak to the diastolic notch, the small notch between the minimum and maximum points, corresponds to the systolic phase while the decrease from the diastolic notch to the trough of the signal corresponds to the diastolic phase. Other measurements can be determined from this figure by deriving the curve generated. Research indicated that with the second derivative, the user is able to more easily determine any abnormalities in the patient, such as hypertension or risk of heart disease (Elgendi, 2012).

2.4 Batteries and Power Management

When designing an electronic device, it is important to keep in mind the ways in which the device can be powered as well as how to monitor the amount of power the device is using. When determining the type of power source, engineers may weigh factors such as expenses, longevity, and overall voltage output of the power source. When the device receives power, it is also important to regulate the amount of voltage going into the device to ensure that the specific components within the device are receiving their ideal levels of voltage to properly function. As the device uses power, to establish an easy user experience, many engineers consider ways in monitoring the battery level of their power source to communicate the longevity of the power source with the user.

2.4.1 Battery Choices

When determining which power source to use, there is the option of using a rechargeable lithium ion battery or 9 volt dry cell batteries to be changed out by the user. Lithium batteries work by ionizing lithium atoms in the anode of the battery to separate the electrons from the atom during the discharge cycle. The ions then move from the anode to the cathode where the ions then regain their electrons and neutralize. A lithium ion battery delivers up to 3.6 volts of electricity. The cost of an Li-ion battery can range from about $10 to $20 on websites such as Digikey and can deliver voltages almost three times greater than other battery technologies such as Ni-Cd or Ni-MH batteries (*Lithium-Ion*, 2020). Some downsides to Li-ion batteries are that they are fairly hazardous if handled improperly. Due to their reactive nature, there are many restrictions and guidelines on shipping Li-ion batteries, both alone or inside of products. The main safety issues of the batteries can be caused by being used at high voltages, overexerting the battery to the point of an unsafe temperature, high pressure build up, or sending too high of a current through the battery. By controlling the temperature, pressure, internal voltage, and current, users and designers are able to reduce the chance of battery failure (Semeraro, 2021). If these factors are not properly monitored and regulated, the battery can lose its power capacity and may fail after a few years.

Another possible power source is a regular dry cell battery. These batteries typically range in output voltage of 1.2V to 9 volts. They function by a graphite rod in the center of the battery which serves as a cathode. Around the cathode is a zinc anode. These layers are both in contact with a mixture composed of carbon and magnesium dioxide which helps in moving electrons from the anode to the cathode. While these batteries are not rechargeable, they are easy to be replaced by the user and cost anywhere from $0.60 to $3 for one battery with the price rising as the voltage output increases (Lazarski, 2015).

2.4.2 Voltage Regulation

As discussed in the previous section, an excess of voltage or current from the power source to circuit components can damage the components, impacting their functionally and lifespan. One method of increasing the life of these electronics is by reducing the amount of voltage sent to each component through voltage regulators, which are electronic devices that maintain voltages within the specified range of the component. These regulators function similarly to variable resistances as they are able to decrease in resistance when the load is heavy and increase in resistance when the load is reduced in order to minimize any potentially damaging current variation. Additionally, voltage regulators are fairly inexpensive, with prices ranging from about $2 to $7 on Digikey (Britannica, 2019).

There are two types of voltage regulators: linear and switching regulators. Linear regulators use a high gain differential amplifier to employ the use of active pass devices, such as BJT or MOSFET in either series or shunt. This allows the device to adjust its output voltage as it compares the output voltage to a reference voltage. Typically, the noise from a linear regulator is lower than that of a switching regulator. A switching regulator is able to convert an input DC voltage to voltage applied to power MOSFETs of BTJ switches. The new filtered power is sent back to the circuit through a power switch that turns on and off, allowing output voltage to remain constant. While filtering the power, unfortunately some power losses can occur when turning the MOSFET on and off continuously. This power loss is important to consider when researchers are deciding which regulator best fits their circuit design. Some other key factors to consider when choosing which voltage regulator will fit the circuit design are the desired input and output voltages as well as the output current. Additionally, it is important to consider the output noise, the load transient response, ripple voltage, and efficiency. With proper circuit design, the efficiency can range anywhere from 50% to up to 90% efficiency (Analog, 2009).

2.4.3 Battery Monitoring

As an electrical device is used, the capacity of its battery is depleted. It is important to understand the capacity of the battery, specifically when knowing the capacity can save lives. Additionally, understanding the capacity of the battery prevents the user from overcharging their battery. This is partially crucial for Li-Ion battery applications as these types of batteries will experience a reduced lifespan if consistently overcharged (Avvari, 2015). A battery fuel gauge is a device that is able to monitor the batteries' voltage, current, and temperature through the integration of functions using a Coulomb counter. Many battery fuel gauges communicate through an I2C or SPI interface, allowing the user to visualize the battery life by generating graphs and figures (*Battery*, n.d.). A figure of the voltage and current of a battery can be depicted in Fig. 4, where the voltage slowly decreases as the device is used until the amount of usable power is depleted. A battery fuel gauge also has the ability to estimate the time until the device shuts down and the remaining useful life of the battery. These estimations are very important as they allow the user to gain a better understanding of the longevity of their device until the next charge (Avvari, 2015).

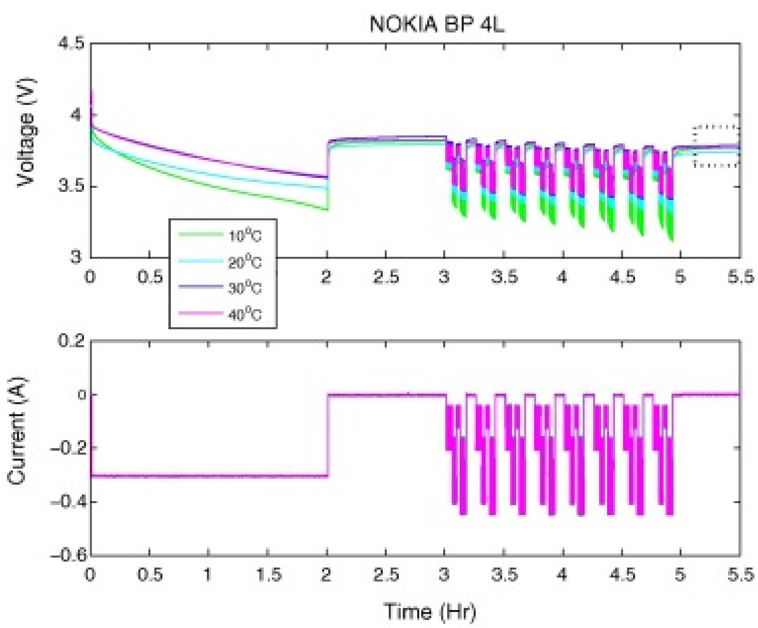

Figure 4: Graphs generated from a battery fuel gauge of a Nokia battery depicting the amount of voltage and current produced during continuous use (Avvari, 2015).

2.5 Microcontrollers

A microcontroller is utilized in many electronic devices as the central decision-maker and controller of internal and external signals in the device. There are many things to consider when choosing a microcontroller unit (MCU) for a project, including factors such as cost, power, peripherals, size, I/O ports, and experience with the product (Peterson, 2020). The main distinguishing factor for a project like the Smart Motorcycle Helmet Device is for the MCU to have Bluetooth capabilities. This is for the functionality of connecting to a driver's phone and communicating both that emergency services need to be contacted and PPG data from the device. The alternative to using Bluetooth and a phone for this communication is to have an on-board cellular module that sends an SMS directly from the hardware of the device. The Bluetooth option has been chosen due to multiple factors, including experience working with BLE on microcontrollers, the ability to use the data plan and GPS capabilities of the driver's phone, and the simplicity of the hardware.

2.5.1 Microcontroller Choice

The chosen microcontroller for the project is the Nordic Semiconductor nRF52840, a System-on-Chip that uses a 32-bit ARM® Cortex™-M4 CPU with floating point unit running at 64 MHz. It supports Bluetooth Low Energy, Bluetooth mesh, NFC, Thread and Zigbee, provides ample memory in RAM and FLASH, integrates numerous peripherals, and has an on-chip power management system. Nordic Semi is one of the leading manufacturers of BLE-capable MCUs, and the nRF52840 is the most sophisticated of the nRF52 Series (Nordic Semiconductor, 2021). ON Semiconductor and Texas Instruments also offer Bluetooth-capable devices in the NCH-RSL10-101Q48-ABG and CC26xx series, respectively (Peterson, 2020). While the main driving factor of the MCU selection is the prior experience of the group working with the Nordic device and the possession of an nRF52840 development kit, the extensive documentation and support provided by Nordic Semi was also a driving factor.

2.5.2 Bluetooth Stack Protocol

There are two main types of Bluetooth used in projects: Bluetooth Classic and Bluetooth Low Energy. Bluetooth Classic is typically used for audio applications, such as speakers, headphones, and car wireless connections, while Bluetooth Low Energy is the main choice for data transfer in low-energy devices, such as battery-powered wearable devices. For the Smart Motorcycle Helmet Device, Bluetooth Low Energy (BLE) is the clear choice for its energy efficiency, integration with microcontrollers, and typical application (Bluetooth, 2021).

BLE has distinct protocol layers that form a stack at different levels of abstraction, from the physical implementation to the app-level design. Figure 5 below diagrams the typical BLE stack.

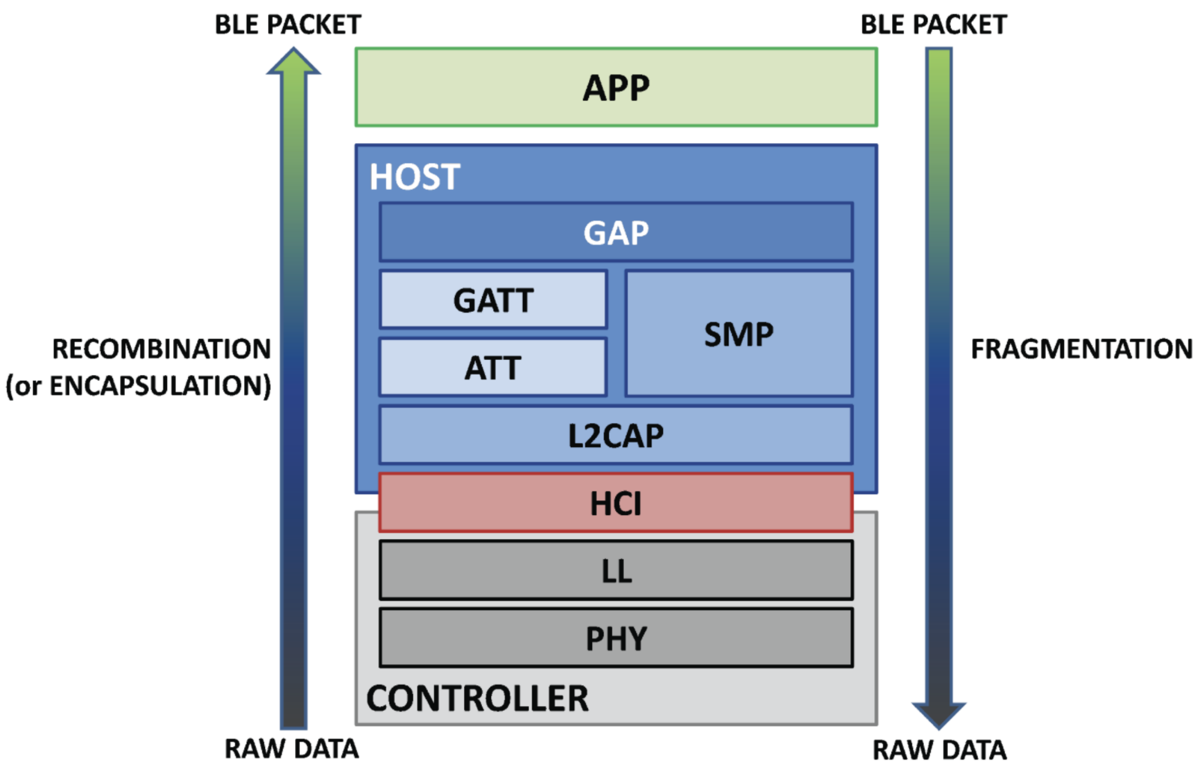

Figure 5: Typical Bluetooth Low Energy Stack Protocol (Tawil, 2018)

At the controller level, the Physical and Link Layer are involved with the radio frequency connections and transmission of raw data between BLE-connected devices. The Host-Controller Interface creates a link between the physical controller operations and the Bluetooth protocol layers and data within the Host. In the Host layers, the L2CAP, ATT, GATT, SMP, and GAP all work to facilitate connections between BLE devices, advertise relevant parameters, access real data, and more. The application layer is where the code controlling all of the layers is written, such as within a MCU. With a microcontroller that serves as a BLE SoC (System on Chip), the controller, host, and application layers are all present in the MCU without any peripheral chips (Tawil, 2018).

A Bluetooth device begins to advertise when to let other Bluetooth devices, which are scanning, know that it is available to connect. While advertising, the device broadcasts information about itself, such as its name and connection specifications. Once connected, the device transfers data through various characteristics that are part of various services. This structure is defined in the GATT Profile (Generic Attribute Profile). An individual service groups multiple characteristics together with one focused purpose, such as a command service or a data service. Each characteristic is set up for one or more data directions (i.e. master-to-slave and/or slave-to-master), a size of data packet to be transmitted, and is for a more specific purpose, such as one specific category of data transfer.

2.6 Phone Application Software

Once connected through Bluetooth to the microcontroller, software must be written to use the smartphone's GPS and cellular capabilities. This software will consist of a lightweight smartphone application. The software application will have two main components. First, it must be able to take in data from the Bluetooth connection to the microcontroller. This will include data about the helmet's battery, the heart rate of the user, and information about if a crash has been detected. The second job of the application is to be able to send an SMS message to emergency services or other contacts when a crash detection occurs with the user's GPS location and heart rate data. Fortunately, these are common features in other applications, so examples to help implement them are readily available. An additional consideration is which operating system the application is developed for, as the time constraints of the project only allow time to create one app.

2.6.1 Smartphone Operating Systems

There are two major smartphone operating systems for which the application could be developed. The first is Android, which is a free and open-source mobile operating system by Google. Android offers many benefits to being the chosen OS for development. First, it has about 2.5 billion devices running it, including 70% of all smartphones. This would mean software developed for it will reach the most users possible. Another benefit is that the official development language for Android is Java, which all the members of this project have experience with. Additionally, it is possible to develop using other non-official languages like Kotlin, C#, Python, and C++ to develop features for applications even though they are not natively supported. The integrated development environment (IDE) for developing Android apps is Android Studio, which is available on both Mac and Windows machines, and allows for emulation of Android phones within its environment for testing.

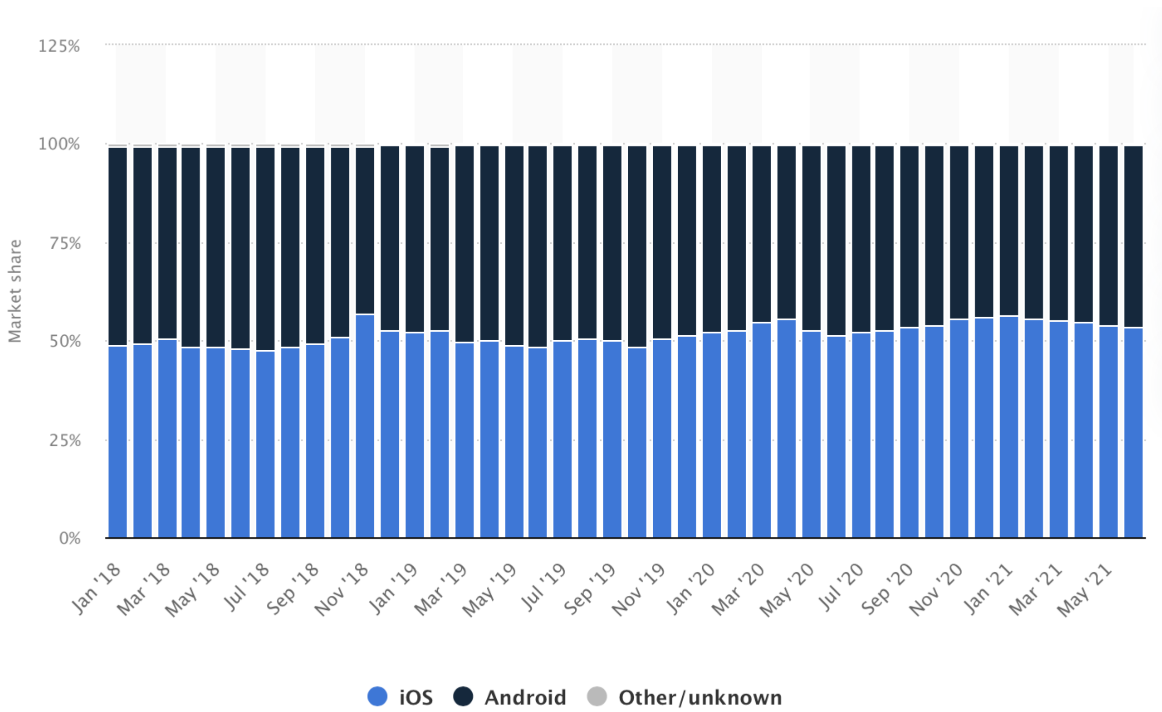

Figure 6: Percent market share of iOS and Android devices in North America from Jan '18 - May '21 (StatCounter, 2021).

The second operating system is iOS, which was created by Apple, and is used on all iPhones and iPads. One benefit to developing on iOS is that it is the most popular smartphone operating system in North America with 53.66% of all devices running it, compared to 45.99% running Android (*Mobile OS share in North America...* 2021). iOS applications are coded using objective-C (an extension of C) or Apple's Swift language which was released in 2014. One downside to using this language is that none of the group members have used Swift or objective-C before (although they have some experience in C/C++). Like Android, other languages can be used for backend features like python, C#, and C++. One more factor in using iOS is that the group members of this project are more familiar with iOS and have iOS devices.

2.6.2 Implementing Application Features

Receiving data from the microcontroller over Bluetooth requires using Android or iOS's Bluetooth APIs. For Android, on startup, the app will check if the Android device supports Bluetooth and if Bluetooth is enabled. If both are true, then it will check for paired devices and instantiate the microcontroller's Bluetooth module as a BluetoothDevice object. Then, on a separate thread[1], it will use this BluetoothDevice to create a BluetoothSocket object, which is what Bluetooth uses to transfer data between the devices. Finally, on another thread, the app uses the BluetoothSocket connection to establish an input and output stream to read from and write to the microcontroller respectively (Wirsing, 2014). iOS has a similar process for connecting to Bluetooth devices. Swift contains a "Core Bluetooth" framework with objects for connecting and sharing information between peripheral Bluetooth devices (*Core Bluetooth Framework*).

Sending SMS messages is an even simpler problem as both Android and iOS have special handlers for that task. For Android, first, you must ask permission from the user when the app first starts to allow the app to send SMS messages. Once the user allows permission, an Android object, 'smsManager', has a method called 'sendTextMessage' which will allow a message to be sent from the user's phone to any phone number. For iOS, it is impossible to send a message from the user's phone number to another directly. iOS only allows for message composing, but the user must confirm the message before it is sent. This is an issue as in the event of a crash the user may not be able to confirm the message if he or she is badly injured. However, it is possible to automatically send messages from a third-party phone number using a REST API called Twilio. In this case, a request would be made to the Twilio API to send a message for the user. The downside to this is there would be reliance on a third party to deliver the messages instead of the user's phone. This requires an internet connection as well as reliance on a third-party server. If the third-party server stopped working, messages would not be able to be sent.

[1] A thread (or thread of execution) is a set of information for a CPU to execute a stream of instructions. Multiple threads can run simultaneously allowing multiple sets of instructions to be processed at the same time.

2.7 Physical Components of Design

One of the decisions that must be made on this project is how to mount the sensors and circuitry to the helmet. Two possible options have been considered. The first is to directly embed the electronic components inside the motorcycle helmet. This would allow for the components to be protected by the helmet itself and would be easier to secure. The second option is to create a capsule and an attachment system to add the electronics to any existing helmet. This would be desirable for riders who already have a helmet and just wish to upgrade it without purchasing an entirely new one.

2.7.1 Attaching Components to Helmet

If adding the electronics to an existing helmet, there would need to be a way to attach the components to the helmet. This can take two forms, a strong adhesive, or straps. Adhesives might be the more elegant solution. However, helmets can be made of many different materials including carbon fiber, kevlar, polycarbonate plastic or thermoset resins. (Grayen). An adhesive might stick well to some of these materials but poorly to others (Rossi, 2021). Use of a strong multipurpose adhesive like a cyanoacrylate adhesive may be a viable option, but this will need to be tested further (*How to Bond Carbon Fiber* 2020). Straps made of Velcro would allow the user to remove the device more easily for recharging purposes but would look less professional. Also, additional testing would be needed to test how securely each of these options would attach the device to the helmet as it must be able to sustain connection through high-impact crashes.

2.7.2 Helmet Ergonomics

Finally, the ergonomics of the helmet must be considered, particularly with weight distribution and the comfortable placement of the photoplethysmography (PPG) sensor. One of the most important aspects of wearing a helmet is its comfort when worn for longer durations. Adding components to the helmet will inevitably add additional weight to the

Figure 7: Counterbalance weight on helmet base

helmet as well. For this reason, it will be critical to choose lightweight components, as well as distribute the weight so that it will not cause neck discomfort. Placing components at the back of the helmet near the bottom is one location that could help comfort. Helicopter pilots routinely place counterweights in this area to reduce neck load when wearing goggles (Van den Oord et al., 2012). Adding hardware components to this area could produce the same effect as the counterweights for the rider and may improve weight distribution of the helmet overall. It will also counterbalance the PPG sensor, which would likely be placed forward in the helmet. Additionally, the PPG sensor adds a challenge for comfort as it must be firmly pressed against the skin to get an accurate reading. As discussed in the sensor reading section, the forehead is the ideal area to mount the sensor for an accurate reading. Incorporating padding around the sensor would reduce the pressure the sensor will induce by being pressed against the forehead.

2.8 Testing the Device

After creating prototypes of devices, engineers need to ensure that the device is properly working with both accuracy and reliability. In terms of motorcycle crashes, some factors that may need testing are the durability of the helmet upon impact and its ability to stay on the head of the cyclist. In regards to the possible sensors used, some tests may need to occur on the accuracy of the accelerometer and photoplethysmograph.

2.8.1 Motorcycle Helmet Crash Testing

While in a crash, a cyclist can hit an object at speeds as high as 100mph. To ensure the safety of the rider, the helmet must cover the frontal and temporal sections of the head. Additionally, the helmet must maintain its structure while reducing possible injuries to the head upon impact of the accident. To test the structural integrity of the helmet, researchers can drop the helmet onto a flat surface and detect any physical changes to the helmet. Additionally, researchers can drop heavy objects of about 10kg onto the helmet and then assess any possible damages. While undergoing these tests, a fake head is strapped into the helmet with five accelerometers attached in the formation pictured in Fig. 8. The acceleration of the head is determined based on the accelerometer data with the goal being to reduce the amount of acceleration experienced by the head in relation to the helmet. Additionally, while undergoing these tests, the position of the artificial head is monitored to determine how well it remains inside the helmet during the simulated crashes (*Crash*, 2017).

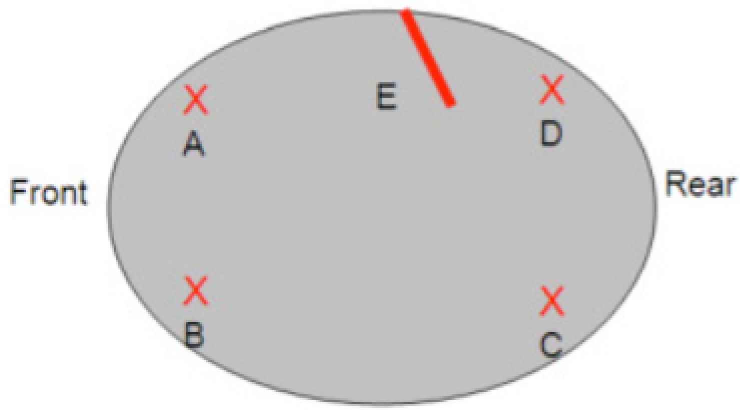

Figure 8: A diagram of a helmet depicting where accelerometers should be placed to determine the acceleration of the head during simulated crashes (Crash, 2017).

2.8.2 Testing Sensors

When using a PPG, the engineer wants to derive the heart rate variability, blood pressure, oxygen saturation in the blood, and the respiratory rate. In order to determine the heart rate variability, a series of tests must be performed using both an ECG and a PPG to monitor the patients. Over a period of five minutes, data from the ECG and PPG would be collected from a patient sitting down and then from the same patient walking around or doing low impact activity. After refining the data collected from the two sensors, the designer will compare the heart rates of the ECG and the PPG using the P-P and R-R intervals of each signal to determine the accuracy and reliability of the PPG in the device. When performing this test, it is expected for the PPG signal to have a slight delay from the ECG signal, but the intervals between the two should have very little deviation. Additionally, the researcher can use a blood pressure cuff to manually compute the blood pressure of the test subject. This manual calculation can be compared to the data retrieved through the raw PPG to determine the accuracy of the PPG when measuring blood pressure (Ghamari, 2016).

When using a series of multiple accelerometers, a researcher is able to determine both the speed at which the subject is moving as well as its relative position through speed calculations. In order to test if the accelerometers attached to the helmet are outputting the correct acceleration of the helmet, a series of tests have been derived. Using the pendulum located at WPI, the helmet will be attached to a crash dummy set on a track. A pendulum will then be loaded with weights up to 15lbs, be brought up to an angle of about 50°, and then released to hit the helmet. Data will be collected both before, during, and after the collision of the pendulum to the helmet in the X, Y, and Z axis. The data collected will then be compared to calculations done prior to determine the speed and height the helmet can reach. In order to perform these calculations, the mass of the helmet with the sensors must be determined. Additionally, the height at which the helmet is dropped and the length of rope used must be known before performing an analysis on the force of gravity and tension on the helmet.

After determining the accuracy of the accelerometer, researchers will then gauge the abilities of the accelerometers to determine a crash. To determine the change in velocity as the an object goes from motion to rest after impact, the helmet attached to the pendulum will be dropped from a height and swing until it reaches contact with a solid object, whether that be a wall, a mat, or some other object to halt the movement of the pendulum. The data from the accelerometer can be used to determine the speed at which the helmet hit the wall as well as its deflection from the wall after impact. Undergoing several trials of this at different initial speeds and heights will help in fine tuning what the device will register as a crash (Gabauer, 2010).

3.0 Problem Statement/Goals & Specifications

This chapter focuses on determining a list of objectives and constraints for the project. Determining the essential criteria for the project will allow the team to more easily assess the methods that will be needed to create a working prototype.

3.1 Goals Statement

According to the National Highway Traffic Safety Administration (NHTSA), motorcycle riders are 27 times more likely to experience a fatal accident than car drivers. When an accident occurs, the high likelihood of injury makes it vital that emergency services are called as soon as possible. In an accident, the drivers involved may not be able to call for help due to injury.
This is particularly important in rural areas where it is reported to have 42% more fatal crashes in rural areas than urban areas due to the lack of bystanders to witness the crash. The goal of the project is to design a device that will easily attach to any commercial motorcycle helmet and help motorcycle drivers get assistance faster during a crash by detecting the event of the crash through accelerometer sensor data, alerting Emergency Services through a developed app, and reporting the heart rate of the device wearer measured through a PPG sensor to first responders through bluetooth to the app. The intention is for 911, or other emergency services and contacts, to be alerted of a crash faster than typical when a motorcyclist crashes without another car around or if any other drivers are also hurt and unable to call.

3.2 Objectives and Constraints

The team outlined a series of criteria to be met while designing the device. Below is the collective list of constraints and objectives.

Objectives

- Objective 1: Detect a motorcycle crash

  Being able to detect the moment of a crash is important so that the data on the user's location and heart rate can be sent to authorities to get the help quickly.
- Objective 2: Monitor the users' heart rate and acceleration

In order for the device to properly determine if the user has endured a crash as well as determine the overall wellbeing of the user, it is pertinent for the device to measure the acceleration of the user through an accelerometer and the heart rate of the user through PPG.

- Objective 3: Report data from sensors to app

  While the device is in use, data on the acceleration and heart rate of the user will be transmitted to the app through bluetooth. In doing this, emergency services will have access to knowledge on the general well being of the user as well as an estimate of the intensity of the accident.

- Objective 4: Be able to alert authorities

  The device must alert emergency services of the accident as well as send the GPS location of the user through the app so that authorities can quickly provide assistance in the event of an accident.

- Objective 5: Be easy for the consumer to use

Constraints

- Constraint 1: Must be safe for the user

  The safety of the user is the top priority of the project. All wiring and electrical components should be in accordance with the safety standards for electrical devices.

- Constraint 2: Must attach to any motorcycle helmet

  The device should be able to fit to any motorcycle helmet so that the user can continue to use their tried and true helmet without

- Constraint 3: Must not accidentally alert authorities

  In the event that the user has not endured a serious crash or simply no crash at all is detected, the user should have the ability to cancel

4.0 Methodology

After determining the goals and constraints of the project, the team devised a method of firmware, hardware, software, and mechanical fixtures to create the working prototype. This chapter presents the general system level architecture of the project, as well as a detailed breakdown of each system.

4.1 General Architecture

The design for the device has five main prongs being the input hardware, processing, output hardware and software, app development, and mechanical fixtures. The input hardware includes components such as power through Li Ion batteries and two sensors: a PPG sensor and a 3-axis accelerometer. The signals collected from the sensors are sent to the microcontroller through an analog signal (PPG) or SPI (accelerometer) while the battery level and input voltage is regulated and monitored to keep the device safe. After signal processing, the data is sent to the app through bluetooth. Additionally, LEDs and buzzers are utilized to improve the user experience. An app is developed to send the data collected from the device to the authorities as well as the GPS location of the user. The mechanical fixtures designed for this device are to help easily mount the device to the helmet as well as comfortably place the PPG sensor on the forehead.

4.2 Input Hardware

In order for the user to interact with the device, they must have input methods to change the status and actions of the helmet. This section covers all of the inputs the device will take in.

4.2.1 Power

When designing the device to fit a motorcycle helmet, it was important to determine the best method of a traveling power source. After considering different options of replaceable batteries, rechargeable lithium ion batteries, and poly lithium ion batteries, the team decided on selecting a lithium ion battery operating at 7.2V, 3250mAh, and 23.4Wh due to its large power capacity and its decreased safety risk. The battery is required to power the PPG and accelerometer sensors as well as the microcontroller and assorted accessories of LEDs and buzzers. However, the sensors and microcontroller only require 3.3V of the 7.2V delivered by the battery. To help decrease the voltage output of the battery, a 3.3V 3A linear voltage regulator was purchased. The regulator received a 7.2V input and consistently decreased the output to 3.3V to safely power the components.

4.2.2 Sensors

The device measures both the heart rate of the user as well as the relative acceleration of the user. To accurately measure both these features, two different sensors were required: a PPG and an accelerometer. When choosing the sensors, the team carefully considered the platform required to get data from the sensors as well as purchasing sensors with an evaluation board for easier installation and testing. The three-axis accelerometer uses SPI to collect and transmit data to the microcontroller, while the PPG sensor outputs an analog signal that is processed by the ADC in the microcontroller. The specificity required in the data from the sensors is much higher for the acceleration, which is why these interfaces were chosen, respectively.

4.2.3 PPG

When designing the device, it was important for there to be a method to obtain the general vitals of the user so that they may be transmitted to first responders in the event of a crash. Measuring the heart rate of the user is the most efficient and easiest method to get a general sense of the vitals of the user. There are generally two sensor methods to obtain a heart rate through either an ECG or a PPG. An ECG sensor typically requires multiple electrode sensors and leads while a PPG only requires one main sensor. Due to this, the PPG is a more compact option that would be able to fit inside of the helmet without requiring leads to go out from the helmet to the torso or limbs. Only one lead to the forehead is required to transmit power to the sensor as well as deliver the output signal to the microcontroller. The PPG chosen for the project was selected to have the same operating voltage as the microcontroller at 3.3V and outputs an analog pulse signal. The sensor utilizes two green LEDS and one infrared LED, with green LEDs collecting the strongest PPG signals over other colors of LEDs.

4.2.4 Accelerometer

The other sensor required for the device is a three axis accelerometer. In order to detect the event of a crash as well as the intensity of the crash, the changes in acceleration of the cyclist must be measured and analyzed. A sensor that collects the acceleration of the subject in both the x, y, and z plane is the three axis accelerometer. When choosing which accelerometer to purchase, the required voltage, interface, and amount and sensitivity of g's measured were crucial. The sensor obtained requires the same 3.3V input as the PPG and microcontroller as well as operating on an SPI interface. In the event of a crash, a motorcycle undergoes G's ranging from 50 to 150. In order to best measure acceleration across this large range, the accelerometer selected can detect up to ±400 Gs. This wide range allows for the entity of the crash to be measured, including all sharp peaks of acceleration that exceed the typical 150G.

4.2.5 Start and Abort Buttons

To enhance consumer usability of the device, power and abort buttons will be added to the device. In order to use the device, the user must first activate the device using the power button. This allows for the user to save on battery and also be in control of when the device is active. The other button is an abort button. This option allows for users to stop the device from contacting the authorities if they are safe and do not require assistance in the event of an accident. One feature discussed later in the chapter is the alert buzzer on the device that notifies the user that the authorities will be connected momentarily. In the event that the user does not want to contact the authorities, be it that the accident was really not that bad or that help is already there, the user can push a button to stop the device from notifying emergency services. One of the concerns of this button on the device is the increased chance of the user accidentally hitting the button during a real emergency. This could occur if the helmet hits something during the accident, switching the button off, or if the button gets accidentally nudged. To avoid an accidental turn off, a cover was placed over the button that the user must manually pop open in order to access the abort button. This cover allows for the user to have a safer option to halt notifying the authorities.

4.3 Processing

To simplify the design as much as possible and eliminate the need for level-shifting of the SPI buses, all components were chosen to use a 3.3V power supply. This requires voltage regulation of the 7.2V lithium ion battery. Typically, voltage regulation from a higher to lower value is down one of two ways: with a linear voltage regulator or a buck converter. Buck converters use a switching mechanism that eliminates power losses (increasing efficiency) but adds extra complexity. For the initial prototyping of the circuit, the losses involved with using a linear voltage regulator are worth the simplicity it brings to the circuit, with only three pins and simple to implement. For prototyping purposes, the LT1085CT-3.3#PBF was selected, a 3.3V linear voltage regulator capable of outputting 3A of current and taking input voltage up to 20V. While the device will use far less current and have a far lower input voltage, the regulator was chosen to leave a buffer and work without having to predict the power requirements of the circuit.

In using the lithium ion battery as the power source for the device, it will become important to know the status of the battery at all times. The user must be notified when the battery reaches a critically low level and needs to be charged, particularly because the device serves a safety purpose. Oftentimes, the state of a battery can be read and monitored with an external integrated circuit called a fuel gauge, which can relay information about the battery's voltage, current, and temperature. The fuel gauge serves as an intermediate between the battery and the microcontroller, which processes this information and makes decisions and changes outputs based on it. The exact field gauge to be used for the project will be chosen later, once more final decisions on a battery and communication protocol have been solidified, as power management isn't the most critical aspect of the design for its functionality and testing.

The nRF52840 microcontroller was chosen to do the majority of the processing for the device, with two major components of its firmware being the Bluetooth and SPI modules. The MCU has a Soft Device that can be used as a Bluetooth peripheral, while a phone app will serve as the Bluetooth master. Nordic firmware examples provide a baseline for Bluetooth implementation. These examples and availability of Nordic support, along with the previous experience of the team with MCU, are the main reasons this particular microcontroller was chosen.

Within the Bluetooth Low Energy (BLE) modules, the various custom services and their characteristics can be initialized. Then the device can begin advertising. Once the application is able to connect with the MCU, commands, statuses, and data can be transferred back and forth between the two devices. Specifically, a command service, status service, and data service will be created for these functions. The directionality of communication for each characteristic within these services will be set and a custom protocol for the meaning of different commands and statuses will be set.

SPI was decided to be the main communication protocol between the microcontroller and accelerometer, as opposed to others like I2C. This is due to the prevalence of SPI devices and the simplicity of the Nordic firmware examples using SPI. The Nordic examples again create a baseline of the communication protocol, but the various registers, addresses, timing, and more for both the PPG sensor and accelerometer need to be customized into the firmware.

4.4 Software

The first choice for software was whether to develop the smartphone app for Android or iOS. Android is an attractive choice as it uses java, which our group was more familiar with the syntax of than Swift. It also allows for messages to be sent automatically on behalf of the user, meaning third party API's wouldn't need to be used to send text messages. Finally, there are more Android devices in the world than iOS devices, so theoretically more people could use our helmet initially. One con of developing for Android is that the java development involves more overhead code than the newer designed swift language. Another is that none of our group members had access to an Android device, so we would likely have to purchase one for testing.

iOS was attractive because the swift language is newer and more modern. The code is easier to read and understand and tends to require less overhead than java. Another positive was that our group was more familiar with ios devices and had ready access to them for testing. Additionally, there are more ios devices than Android devices in the US, which would likely be our initial target market. One of the main downsides to using ios is that it is impossible to send a text message on behalf of the user directly. The user is required to approve the message before it is sent, which may be problematic if the user was in an accident where they could not get to their phone. To get around this we would have to use the Twilio API, which costs $0.0075 per text message. Another con of ios is that our group members were less experienced in it's syntax than java.

After testing Android studio and XCode, the default IDE's for Android and iOS respectively, our group ended up choosing to develop for iOS due to its ease of use. Additionally, having iPhones already meant that we could begin testing immediately without having to order an expensive Android device. The group also found that although they were familiar with java, using java to develop an Android app was much different than writing typical programs. This meant that Swift might actually be easier to learn to develop with even though our group has some experience with java.

Once the type of device was selected, the group started by adapting a simple example bluetooth messaging application. When the app is first opened, it immediately begins scanning for the uuid of the BLE controller on the helmet. Once connected, it displays the device name and moves to the next screen. In the second screen messages can be exchanged with the BLE device. There is a small text box to transmit data to the controller and a text view that will show any received transmissions. In this way the user of the app can send instructions to the helmet or receive data about when crashes occur. Once data can be exchanged from the app, we will have a trigger that will send a message when it receives a message of a crash. User friendliness of the application was not considered at this stage of the design. The application is meant to be a simple proof of concept to show the hardware functioning properly.

4.5 Outputs

To keep the user informed as to the status of the device, output hardware must be considered. This includes visual and auditory feedback mechanisms. This section explores possible feedback mechanisms.

4.5.1 LEDs

Providing feedback about the state of the device is important to the user experience. To give feedback when the helmet is off, one or more LEDs will be used in various patterns. A red LED can be used to signal low power, telling the user that the device needs to be charged soon. A blinking LED can show the device is in pairing mode and is broadcasting bluetooth to add a device. A solid green light can represent that the device is paired and is charged. The team will work to expand on these initial ideas to give the user an idea about the state of the helmet at all times. These LEDs will be controlled with the microcontroller.

4.5.2 Speakers

While LEDs will give the quality information while the helmet is not being worn, they will be impossible to see when the helmet is on. For this reason, a secondary form of feedback is required in the form of speakers. The speakers will play chimes that can give the user feedback about the state of the system. For example, a chime can play letting the user know that the battery is low or that it disconnected from their phone. Perhaps the most important application of the speaker will be to play a warning sound when a crash is detected. This allows the user to know that emergency services are about to be notified. This is important because it notifies the user, so they can cancel this action by clicking the abort button. This will stop unwanted notifications of emergency services if a crash is erroneously detected.

4.5.3 Texting Service API

When developing for iOS, it is impossible to automatically send a text message on behalf of the user without any user input. This is because iOS requires the user to manually approve any message before it is sent. However, during a crash the user will likely be unable to interact with their phone. The solution to this issue is to use the Twilio API which sends text messages through their servers instead of from the user's phone directly. Each message sent through Twilio API costs $0.0075. This will be practical when the product is in use as crashes are infrequent for users, so the cost of sending the messages will be minimal. However, for testing where we need to send many messages this cost may be unreasonable. For that reason, during testing messages will simply be displayed in a text box in the smartphone app. Implementing the API to make it an actual text message will be the last feature we focus on. This makes sense as the important technology to focus on is the helmet itself.

4.6 Mechanical Components

While this project is about an electronic device, there are mechanical considerations that must be made when constructing a physical device. Ensuring the mechanical components are considered will increase the reliability of the system overall. This section explores those mechanical components.

4.6.1 Attaching Board to the Helmet

In order to attach the board, batteries and sensors to the helmet, there must be a strong bond formed between the two components. First, the team will create the enclosure for the electronics. This will need to protect the components from breaking in the event of a collision. Using materials with high impact resistance such as polycarbonate plastic will be tested for this enclosure. Next, the enclosure with the components will have to be adhered to the helmet. This will be done through using strong adhesives. The team will test multiple adhesive types like epoxy or a strong multipurpose adhesive like cyanoacrylate.

4.6.2 PPG Sensor Comfort

The PPG sensor will be mounted to the forehead pad of the helmet, as this is the best location to ensure consistent firm contact and get an accurate reading. However, considerations must be made to increase comfort and reduce the pressure of the sensor against the user's forehead. To do this, the team will be adding foam padding around all parts of the sensor except the light emitter and receiver. This will disperse the pressure and add cushion instead of feeling metal and plastic pressed against the forehead. To increase comfort where solid foam cannot be added (the light and receiver), the team will experiment with adding clear or semi clear film. This will continue to allow the sensor to function while adding a bit of comfort and protection to the led itself. Testing will be conducted with multiple film types to ensure the accuracy of the readings do not change with the film's addition.

5.0 Implementation/Design

To determine the overall effectiveness of the device, individual components of the device were tested. These components include the power supply, the AC to digital converter, the PPG and our heart rate algorithm, and the accuracy of the accelerometer. It is through the combination of testing each competency individually that allowed the team to debug each stage before all the components are added together to form the overall device. Once all components were combined onto the helmet, a final series of testing was conducted to determine if the different pieces worked cohesively together.

5.1 Final Design

Figure 9 depicts the final block diagram of the design, with the blue boxes representing the input signals for the device, red boxes representing the processing of the signals for the device, and yellow boxes depicting the outputs of the device.

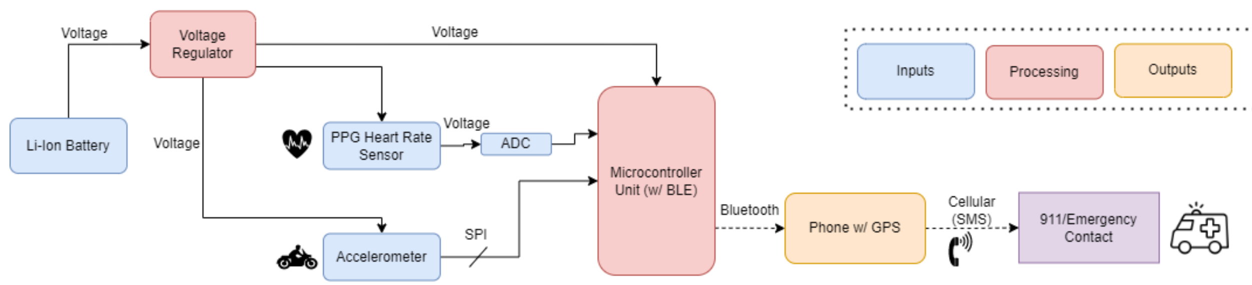

Figure 9: Final block diagram of the device depicting all input signal sources, processing components, and outputs of the device.

The device power is supplied by a 7.2V Li ion battery with a linear voltage regulator. The voltage regulator decreases the voltage to 3.3V which is input to both the PPG, accelerometer, and the microcontroller. The PPG is mounted to the inside of the helmet and makes contact with the forehead where an optimal pulse is collected. The analog signal is then processed through the ADC and the microcontroller uses the digital signal to find a value for heart rate. Through SPI, the G forces from the accelerometer experienced in the X, Y, and Z directions are sent to the microcontroller via SPI.

The microcontroller collects all of the data sent from the PPG sensor and accelerometer. To determine the beats per minute (BPM) from the PPG, the average peak value over an interval of 10 seconds is calculated. Then, the number of times in which the signal reaches that value while increasing is found and the BPM is calculated. When G forces greater than 8 are detected along any axis, a crash is detected. At this instance, all data on BPM and acceleration are transmitted to the user's app on their cell phone through Bluetooth. Emergency services are then alerted with the user's location and vitals using the user's cell phone, with the location coming from the cell phone's GPS.

5.2 Team Testing

To ensure that each of the individual components of the device are reliable, we conducted a series of tests on possible points of failure. This ensured the accuracy and proper functioning of the device. This section explores those methods of testing.

5.2.1 Power

To determine if adequate voltage was being delivered to the components of the device, the voltage from the battery and voltage regulator were tested. The battery used is a 7.2V Li-ion battery. The battery voltage reliably delivered 7.2V after the voltage was read several times. The voltage regulator was then tested with a digital multimeter to determine if it reliably regulates the voltage to 3.3V since all of the components run on 3.3V. Both alone and with resistors, the regulator reliably delivered 3.3V for all trials of various resistance values.

5.2.2 PPG and Skin Contact

To deliver a comfortable user experience, the amount of contact to the skin required to generate a recognizable signal was tested through layering different colored fabrics across the sensor as well as the amount of pressure applied by the sensor. In terms of pressure applied to the sensor, when aggressive pressure was applied, the signal became very noisy and warped with no discernible BPM. When some pressure was applied, the signal did have some noise, but with some filtering and threshold setting described in the previous sections, BPM should be easily determined. When the PPG was held just above the skin without actual contact, there was some noise and the voltage of the signal was greater than the previous trials. However, like the trial with some pressure, a signal can still be discerned with some filtering and thresholds. As long as the pressure applied from the PPG to the skin is not excessively harsh, the heart rate signal will result in reliable BPM readings. In terms of comfort while wearing, fabrics of different colors and thickness were tested as a barrier between the skin and PPG. Darker and thicker fabrics resulted in noisy signals while lighter and thinner fabrics resulted in clearer signals. Additionally, more layers of the thinner fabrics also resulted in more noise. Based on these results, a layer of thin white cotton fabric will be used as a barrier between the skin and PPG for increased user comfort while wearing the device.

5.2.3 Analog to Digital Converter

To read the analog PPG signal, an ADC was used to translate the signal to the microcontroller 10 times per second. In order for the device to accurately report the BPM of the user, the ADC must correctly convert the signal to a digital format for the microcontroller to read. To test the accuracy of the ADC, the converter was fed a constant DC voltage. This signal was then sent to the microcontroller where the output was read. As the input voltage was increased or decreased, the microcontroller output the same voltage as input to the ADC.

5.2.4 BPM Counting and Accuracy

Now that accurate readings were available from the PPG sensor, an algorithm was needed to use this data to calculate the heart rate of the user in beats per minute (BPM). The algorithm first would sample for 5 seconds (or 50 samples) and find the peak value the PPG sensor measured. This was done to establish the amplitude a heartbeat reaches on the sensor, since this may vary depending on varying contact levels with the sensor. From this value, a threshold was set at 80% of this peak value. This is illustrated in Figure 10 below.

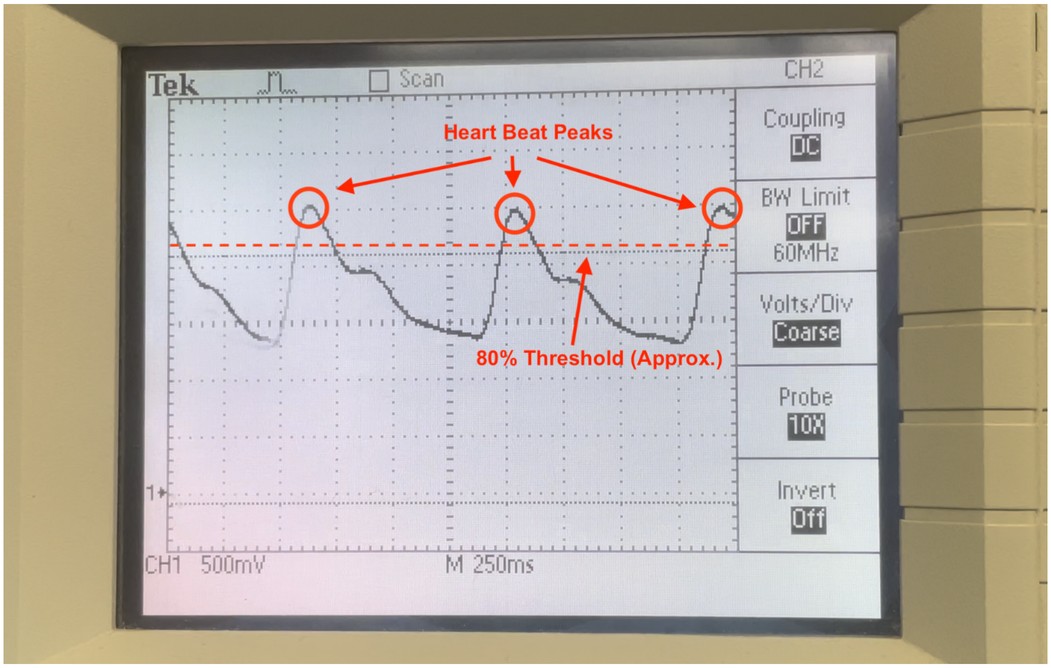

Figure 10: PPG sensor visualization with labels for each heart beat and the 80% threshold set by the algorithm.

When the value measured rises above the threshold, a heartbeat is recorded. It cannot register another heartbeat until it goes below the threshold again. The algorithm will count how many beats there are within a 15-second time frame, and then multiply that by 4 to get the BPM of the user.

To test the accuracy of the BPM calculated from the PPG sensor, a 2-minute long sample was taken from the PPG. This was done while pressed against different pulse areas such as the finger, wrist, temple and forehead. The BPM calculated from this trial was then compared to the BPM from an Apple Watch, a device with reliable heart rate sensing. This was repeated three times for each area, two at resting heart rate and once at an elevated heart rate (obtained by jumping up and down for 30 seconds). From the PPG, it was found that the BPM was always within 5 BPM of the Apple Watch at all locations and for both testing and elevated heart rate.

5.2.5 Accelerometer Crash Detection Accuracy

To determine both the accuracy and reliability of the accelerometer and crash detection, a pendulum loaded with up to 15lbs was swung at the motorcycle helmet. The helmet was mounted to a stand on a sliding track that moved backwards upon collision with the pendulum. The accelerometer was attached to the back of the helmet where it measured acceleration in the X, Y, and Z axis. The accelerometer was tested in a series of 14 trials, with 7 trials adding no additional weight to the pendulum, 4 trials adding an additional 5lbs to the pendulum, and 3 trials adding 10lbs to the pendulum. For each of the trials, the pendulum was released at an angle of about 50° before colliding with the helmet as the accelerometer collects data. For each of the trials run, the accelerometer correctly registered the collision through the spikes in acceleration in the X and Y axis, as seen in Fig. 11. Following these preliminary tests of determining the functionality of the accelerometer, the device's ability to register a crash was then tested. The same procedure above was repeated, but if the acceleration surpassed a set threshold of 5Gs, the microcontroller would be notified that a crash had occurred. In five out of five of these trials, the device successfully registered a crash as occurring. Based on these results, the accelerometer is acceptably and reliably functioning for use in the device.

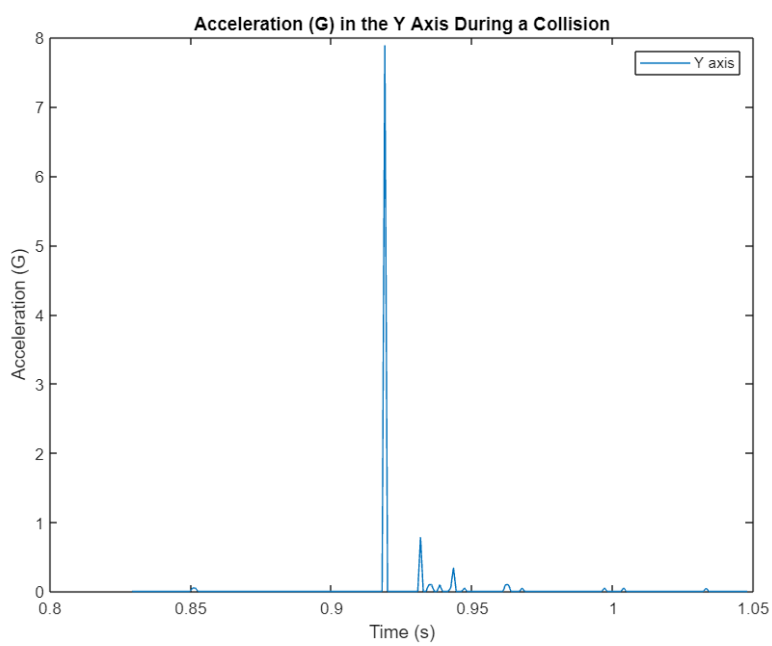

Figure 11: Graph depicting the acceleration (G) of a motorcycle helmet during a collision with a pendulum weighing 10lbs.

6.0 Analysis & Results

After gaining the raw data from our testing, we had to analyze our results. When conducting this analysis, it was vital to have a control or hypothesis by which to compare our results to. This section makes sense of the raw data from the testing and makes conclusions about the efficacy of the heart rate and crash detection functionality.

6.1 Heart Rate Results

Our heart rate monitoring feature performed up to our expectations. We were able to consistently calculate heart rates through the PPG sensor at multiple contact parts on the body including the finger, wrist, temple and forehead. Our basis for comparison was the heart rate monitor on an Apple Watch, which has a reliable PPG heart rate system. In our testing, we had consistent heart rate calculations that were within 5 BPM of the Apple Watch readings.

One design element we had to change to increase accuracy was to include a calibration stage at the beginning of the readings. Originally, a hard coded threshold was set to specify what would be considered a heartbeat and what was not. However, it became apparent that this was problematic since there was variability as to the amplitude of the beat peaks due to factors like location on the body and firmness of the contact with the sensor. This meant that the threshold would sometimes be too high, and no beats would be detected, or too low, and false beats were detected. The solution was to adaptively set the threshold based on how large the heartbeat peaks were. In the first 5 seconds, the algorithm measures the peak value, and sets the threshold to 80% of that value. This ensured that the threshold was appropriate to measure all beats correctly.

However, there were still some metrics that our heart rate sensor could improve on. In its current form, it takes 20 seconds to get an initial reading. It takes 5 seconds to calibrate, and then 15 seconds to count heart beats. It also has shortcomings when updating, as it updates the calculation every 15 seconds. Using more complicated algorithms that use a rolling average, or using the average time between beats would decrease the time it takes to calculate and update the BPM. The code for the heart rate algorithm module can be seen in appendix A.

6.2 Crash Test Results

The accelerometer's crash detection was able to detect a crash for all of the trials with different weights of 10lbs and 15lbs being swung at the helmet. Based on its ability to consistently and accurately detect crashes once an acceleration threshold has been reached, the accelerometer met the team's expectations for its functionality as a crash detector.

One design element that was changed throughout the testing process was the sampling rate of the accelerometer. Initially, the sampling rate was only 200Hz as the device experienced severe lag while logging all of the acceleration data points. At a sampling rate this low, it is possible for crashes to not be detected by the device as quick spikes in acceleration may not be sampled. However, after saving the data to memory rather than a live log, the sampling rate was able to be increased to 1024Hz without any lag in the data log. The is an industry standard sampling rate, at about 1 sample every 1ms.

7.0 Recommendations

The Smart Motorcycle Helmet Device has been developed to a preliminary proof-of-concept prototype with many areas to be further expanded and improved upon. The current design uses development boards and prototyping materials to create a functional, but minimal, version of the final design. The team is providing numerous recommendations for further work.

7.1 Device Hardware Footprint

Due to the use of a solderless breadboard, MCU development kit, sensor breakout boards, and discrete wires and components, the current footprint of the device is relatively large compared to what it could be when condensed to a singular printed circuit board. Particularly for the application of the device and its need to withstand high-acceleration crashes, a small and durable board is required for the design. Aside from the lithium-ion battery, all components used in the prototype either already have a small footprint or could easily be replaced by a smaller and functionally equivalent component. Therefore, the entire device could be condensed to a PCB with the approximate same dimensions as the lithium-ion battery.

7.2 Attachment to Helmet

In order for the device to be functional, it must be able to withstand forces exceeding what could be expected from the most extreme motorcycle crashes. The goal for the device is to be easily attachable to any commercial helmet, so significant development would have to go into both the protection of the electronics and the attachment mechanism. It would become important not only for the PCB and battery to be protected from damage, but also secure from movement. In order to accurately record acceleration and heart rate data, the sensors must remain as still as possible with respect to the helmet and driver, eliminating any noise or interference.

7.3 Hardware User Interface

Small hardware additions to the device would greatly enhance its usability for riders. The first of these features would be various LEDs to alert the driver of various states of the helmet. Ideally, these LEDs would be mounted in a place that is visible, though not distracting, to the driver, and include indications for power, battery level, and crash detection. Along with the LEDs, there would also be a few buttons/switches for the device. A power switch could be turned on as soon as the helmet was placed on the driver's head, and the main other button would be an abort button in the case of a false crash detected.

While the algorithm for crash detection would be developed and tested further for accuracy prior to the device being commercially available, the risk would still remain that a crash could be detected incorrectly. In the case when a crash is detected by the device but the driver is entirely unharmed, they should have the option to cancel the alert to emergency services. The mechanism for including this feature would include both an LED and sound indication that a crash has been detected, and there would be a short delay in the app sending an alert. During this delay of 10-15 seconds, a conscious and healthy driver would be able to easily press the abort button and stop the alert from being sent altogether. It would be critical for the button to be in an extremely accessible place, but not at the risk of it being accidentally pressed in a high-acceleration crash.

7.4 Battery Management

The device currently uses a 7.2V rechargeable lithium ion battery as its main power source, which is then regulated down to a 3.3V source for the microcontroller and sensors. In its current state, there are no supplementary battery regulation chips to handle the charging and status of the battery. Moving forward into development, at least two IC's would be introduced to the design: a battery charging IC and a fuel gauge IC. The battery charging IC serves to control and regulate the speed at which the battery charges and when it charges. This would also require the addition of a charging plug to the device and additional control circuitry. The fuel gauge IC is used to monitor the status of the battery, reporting information such as its state of charge (SOC) and state of health (SOH). This information would then be digitally transferred to the microcontroller and seen by the user on both the smartphone app and through LED indicators on the helmet.

7.5 Data Collection

Currently, data is collected by the microcontroller from the accelerometer at 1024 Hz and from the PPG sensor at 10 Hz. These values were initially chosen because they allowed for usable results while still being able to log values and avoid any timing errors. Going forward, as the algorithms for each sensor are created to be more accurate, the rate at which data is sampled from these sensors would have to be increased as well. This would require more advanced scheduling and interrupts than are currently implemented, but would ensure no data is being lost and the results are as accurate as possible.

7.6 Processing Algorithms

Accuracy of both the crash detection algorithm and the heart rate measurement algorithm are critical to the functioning of the device. For the crash detection algorithm, a very primitive method is being used to trigger a crash alert when the acceleration in any axis exceeds a set threshold. While this detection method works reliably in controlled testing environments, it could lead to multiple false positives, in which the device incorrectly indicates a crash has occurred. While the addition of an aforementioned abort button would act as a safety net for these occurrences, crash detection should be as accurate as possible to eliminate the need to use such a feature. Additional accelerometers and gyroscopes could be implemented into the device to develop a more complete understanding of the linear and rotational accelerations involved in a crash and their implications.

7.7 Bluetooth Communication

In order to save power and increase the efficiency of communication between the microcontroller and the phone, the way in which the two communicate over Bluetooth could be refined. Currently, as the smartphone is acting as the BLE master, it needs to continuously request a crash status from the microcontroller at regular intervals. Once the MCU communicates that a crash has been detected, the app then requests the heart rate and receives the value. Then it is able to send the alert to emergency services. To save power and increase efficiency, this system could be altered to eliminate the continuous requests from the app. The peripheral device could be set to send notifications to the master, in which case no communication would occur until a crash has been detected.

8.0 Ethical Considerations

When creating a safety device, it is especially important to consider the ethical consequences of the design and its implications. By creating a device that is supposed to save people's lives in the event of a crash, the device must be held to an extremely high reliability standard. If for some reason people were relying on the device to keep them safe and it failed, then that could potentially result in the serious injury or death of a user. As the creators of such a device, it is our responsibility to ensure that the probability of failure during a life and death event be as miniscule as possible. Many companies have had difficulties with liability issues like this in the past. For example, car manufacturers have come up with prototypes for drowsy driver detection systems. However, very few of these have become commercially available, since there is a liability issue for the companies. There is a concern that if a driver were to fall asleep and get injured or killed, then people would sue the company for promising drowsy driver detection that did not work.

Another consideration is the effect an additional safety device may have on rider behavior. For example, if we oversell the benefits and safety features of the device, it may give riders a heightened sense of security. A heightened sense of security may lead drivers to attempt riskier behaviors since they feel like they have a safety blanket that will save them no matter what. While evidence of this is difficult to find, there have been some studies that have made this assertion. For example, from 1963-1973 there were significant advances in formula one car safety, but the number of casualties remained unchanged (Potter, 2011). While it is unlikely that this increase in risk taking would outweigh the safety benefits of the devices, it is important to be careful not to over-promise the device's capabilities to encourage reckless behaviors.

9.0 Conclusion

Overall, a full end to end system was created that achieved the goals of adding crash detection, heart rate monitoring, and automatic alertion of emergency services to a motorcycle helmet. To detect crashes, a MEMS three-axis accelerometer was used to measure the acceleration the helmet experiences(and by extension the rider's head) up to +/-100g's. The accelerometer communicated with the Nordic microcontroller using the SPI digital protocol and was sampled at 1024 Hz. When an acceleration of over a set magnitude is reached (set at 5 g's in our tests), an alert is sent over Bluetooth to the custom build app on the user's smartphone. The microcontroller then switches to reading the heart rate of the user. The heart rate is measured through an analog PPG sensor positioned inside the helmet on the user's temple. The microcontroller samples the sensor using an analog to digital converter at 10 times a second, and uses a custom algorithm to extract a heart rate from the data. When the iOS app receives the crash alert, it is able to automatically send a text message to emergency services through the Twilio API. To test the system, a pendulum was used to create consistent crash level impacts. In our testing, this system was able to successfully detect a crash and send an alert over Bluetooth to the smartphone app 5 out of 5 trials. While there are many further steps that can be taken to refine the design, this prototype has demonstrated how a system like this can be used to add crash detection functionality to motorcycle helmets. Such functionality has promising applications to help increase rider safety and potentially save lives.

References

Analog Devices. (2009). *Understanding how a voltage regulator works*. Understanding How a Voltage Regulator Works | Analog Devices. Retrieved October 14, 2021, from https://www.analog.com/en/technical-articles/how-voltage-regulator-works.html.

Apple Developer. (n.d.). *Core Bluetooth Framework*. Apple Developer Documentation. Retrieved October 14, 2021, from https://developer.apple.com/documentation/corebluetooth.

Avvari, G. V., Pattipati, B., Balasingam, B., Pattipati, K. R., & Bar-Shalom, Y. (2015). Experimental set-up and procedures to test and validate battery fuel gauge algorithms. *Applied Energy*, *160*, 404–418. https://doi.org/10.1016/j.apenergy.2015.09.048

Battery Fuel Gauge. STMicroelectronics. (n.d.). Retrieved October 14, 2021, from https://www.st.com/en/power-management/battery-fuel-gauge.html.

Bluetooth Technology Overview. Bluetooth. (2021). Retrieved October 14, 2021, from https://www.bluetooth.com/learn-about-bluetooth/tech-overview/.

Brown, R. N. (2011). 8 - Accessories . In *Compressors: Selection and sizing* (3rd ed., pp. 344–414). essay, Elsevier Science. Retrieved October 14, 2021, from https://www.sciencedirect.com/science/article/pii/B9780750675451500109.

Byrne, J. P., Mann, N. C., Dai, M., Mason, S. A., Karanicolas, P., Rizoli, S., & Nathens, A. B. (2019). *Association Between Emergency Medical Service Response Time and Motor Vehicle Crash Mortality in the United States*. JAMA surgery, 154(4), 286–293. https://doi.org/10.1001/jamasurg.2018.5097

Crash protection. CRASH. (n.d.). Retrieved October 14, 2021, from https://crash.org.au/crash-protection.html.

Ebrahimji, A. (2019, December 7). *He almost lost his friend to a motorcycle accident. So he created a helmet that calls 911*. CNN. Retrieved October 14, 2021, from https://www.cnn.com/2019/12/07/us/911-helmet-trnd/index.html.

Elgendi, M. (2012). On the analysis of Fingertip Photoplethysmogram Signals. *Current Cardiology Reviews*, *8*(1), 14–25. https://doi.org/10.2174/157340312801215782

Facts + Statistics: Motorcycle crashes. iii. (2021). Retrieved October 14, 2021, from https://www.iii.org/fact-statistic/facts-statistics-motorcycle-crashes.

Fibricheck beat-to-beat accuracy compared with wearable ECG in broad dynamic range. FibriCheck. (n.d.). Retrieved October 14, 2021, from https://www.fibricheck.com/fibricheck-beat-to-beat-accuracy-compared-with-wearable-ecg-in-broad-dynamic-range/.

Gabauer, D. J., Kusano, K. D., Marzougui, D., Opiela, K., Hargrave, M., & Gabler, H. C. (2010). Pendulum testing as a means of assessing the crash performance of longitudinal barrier with minor damage. *International Journal of Impact Engineering*, *37*(11), 1121–1137. https://doi.org/10.1016/j.ijimpeng.2010.03.003

Ghamari, M. (2018). A review on wearable photoplethysmography sensors and their potential future applications in health care. *International Journal of Biosensors & Bioelectronics*, *4*(4). https://doi.org/10.15406/ijbsbe.2018.04.00125

Ghamari, M., Soltanpur, C., Cabrera, S., Romero, R., Martinek, R., & Nazeran, H. (2016). Design and prototyping of a wristband-type wireless photoplethysmographic device for heart rate variability signal analysis. *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. https://doi.org/10.1109/embc.2016.7591842

Goodrich, R. (2013, October 1). *Accelerometers: What they are & how they work*. LiveScience. Retrieved October 14, 2021, from https://www.livescience.com/40102-accelerometers.html.

Grayen, M. (n.d.). *What Material Choices Are There in Motorcycle Helmet Construction?* Motorcycle ID. Retrieved October 14, 2021, from https://www.motorcycleid.com/articles/experts/michael-g.html.

Helmet With Counterweight. (n.d.). Retrieved October 14, 2021, from https://m.media-amazon.com/images/I/61beOXh63qL._AC_SL1298_.jpg.

Hernández Sánchez, S., Fernández Pozo, R., & Hernández Gómez, L. (2018). Estimating vehicle movement direction from smartphone accelerometers using Deep Neural Networks. *Sensors*, *18*(8), 2624. https://doi.org/10.3390/s18082624

How to Bond Carbon Fiber. Permabond. (2020, April 21). Retrieved October 14, 2021, from https://www.permabond.com/materials_bonded/how-to-bond-carbon-fiber-2/.

Johnston, W.S., Branche, P.C., Mendelson, Y., & Pujary, C. (2004, April 17). *Effects of motion artifacts on helmet-mounted pulse oximeter sensors.* IEEE. Retrieved October 14, 2021, from https://ieeexplore.ieee.org/document/1300071

Lahausse, J. A., Fildes, B. N., Page, Y., & Fitzharris, M. P. (2008). *The potential for automatic crash notification systems to reduce road fatalities*. Annals of advances in automotive medicine. Association for the Advancement of Automotive Medicine. Annual Scientific Conference, 52, 85–92.

Lazarski , A. (2015). *Basics of Electrochemistry*. UW CEI Secondary Use of Lithium-Ion. Retrieved October 14, 2021, from https://mdmurbach.github.io/cei-demo-tutorial/electrochemistry-and-batteries.html.

Leech, S. (2019, December 10). *Contekt concept could revolutionise helmet safety*. INFO MOTO. Retrieved October 14, 2021, from https://infomoto.com.au/news/contekt-concept-could-revolutionise-helmet-safety/.

Lithium-Ion Battery. Clean Energy Institute. (2020, September 25). Retrieved October 14, 2021, from https://www.cei.washington.edu/education/science-of-solar/battery-technology/.

Mendelson, Y., & Pujary, C. (2003, September 17). *Measurement site and photodetector size considerations in optimizing power consumption of a wearable reflectance pulse* *oximeter*. IEEE. Retrieved October 14, 2021, from https://ieeexplore.ieee.org/abstract/document/1280775/

Mithun, S. K., Umesh, S., & Pathan, R. (2013). Conceptual Design Of Motor Cycle Helmet To Meet The Requirement Of Thermal Comfort, Ergonomics And Safety. SAS Tech Journal, 12(1).

nRF52840. Nordic Semiconductor. (2021). Retrieved October 14, 2021, from https://www.nordicsemi.com/Products/nRF52840.

Peterson, Z. (2020, April 8). *Bluetooth 5 Microcontroller Selection Guidelines*. Octopart. Retrieved October 14, 2021, from https://octopart.com/blog/archives/2020/04/bluetooth-5-microcontroller-selection-guidelines.

Potter, J. (2011). Estimating the Offsetting Effects of Driver Behavior in Response to Safety Regulation: The Case of Formula One Racing. *Journal of Quantitative Analysis in Sports*, *7*(3). https://doi.org/10.2202/1559-0410.1276

Rossi, J. (2021, March 25). *7 Ways on How to Mount Gopro on Motorcycle Helmet - Complete Guide*. Sporty Journal. Retrieved October 14, 2021, from https://sportyjournal.com/how-to-mount-gopro-on-motorcycle-helmet/.

Omega Engineering Inc. (2021, March 15). *How to Measure Acceleration?* https://www.omega.com/en-us/. Retrieved October 14, 2021, from https://www.omega.com/en-us/resources/accelerometers.

Semeraro, C., Olabi, A.-G., & Dassisti, M. (2021, February 24). *Sustainability issues in manufacturing and operation of second-generation flow batteries*. Reference Module in Materials Science and Materials Engineering. Retrieved October 14, 2021, from https://www.sciencedirect.com/science/article/pii/B9780128157329000760.

StatCounter. (2021, June 30). *Mobile OS share in North America 2018-2021*. Statista. Retrieved October 14, 2021, from https://www.statista.com/statistics/1045192/share-of-mobile-operating-systems-in-north-america-by-month/.

Tawil, Y. (2018, September 29). *Bluetooth Low Energy (BLE) 101 Tutorial: Intensive Introduction*. Atadiat. Retrieved October 14, 2021, from https://atadiat.com/en/e-bluetooth-low-energy-ble-101-tutorial-intensive-introduction/.

Tayag, M. I., & Asuncion De Vigal Capuno, M. E. (2019). Smart motorcycle helmet: Real-time crash detection with emergency notification, Tracker and anti-theft system using internet-of-things cloud based technology. *International Journal of Computer Science and Information Technology*, *11*(03), 81–94. https://doi.org/10.5121/ijcsit.2019.11307

Tjahjadi, & Ramli. (2020). Noninvasive blood pressure classification based on photoplethysmography using K-Nearest Neighbors Algorithm: A feasibility study. *Information*, *11*(2), 93. https://doi.org/10.3390/info11020093

Van den Oord, M. H. A. H., Frings-Dresen, M. H. W., & Sluiter, J. K. (2012). Optimal helmet use and adjustments with respect to neck load: The experience of military helicopter aircrew. *International Journal of Industrial Ergonomics*, *42*(1), 73–79. https://doi.org/10.1016/j.ergon.2011.09.001

Wirsing, B. (2014, March 26). S*ending and Receiving Data via Bluetooth with an Android Device*. Retrieved October 14, 2021, from https://www.egr.msu.edu/classes/ece480/capstone/spring14/group01/docs/appnote/Wirsing-SendingAndReceivingDataViaBluetoothWithAnAndroidDevice.pdfvingDataViaBluetoothWithAnAndroidDevice.pdf.

Appendix A - Heart Rate Algorithm

```
#include <stdbool.h>
#include <stdint.h>
#include <stdio.h>
#include <string.h>
#include "nrf.h"
#include "nrf_drv_saadc.h"
#include "nrf_drv_ppi.h"
#include "nrf_drv_timer.h"
#include "boards.h"
#include "app_error.h"
#include "nrf_delay.h"
#include "app_util_platform.h"
#include "nrf_pwr_mgmt.h"

#include "nrf_log.h"
#include "nrf_log_ctrl.h"
#include "nrf_log_default_backends.h"

#define SAMPLES_IN_BUFFER 10
volatile uint8_t state = 1;
volatile int beatlock = 0;
volatile int beats = 0;
volatile int threshold = 0;
volatile int threshtest = 0;
volatile int reads = 0;
volatile int BPM = 0;

static const nrf_drv_timer_t m_timer = NRF_DRV_TIMER_INSTANCE(0);
static nrf_saadc_value_t     m_buffer_pool[2][SAMPLES_IN_BUFFER];
static nrf_ppi_channel_t     m_ppi_channel;
static uint32_t              m_adc_evt_counter;


void timer_handler(nrf_timer_event_t event_type, void * p_context)
{

}
```

```
void saadc_sampling_event_init(void)
{
    ret_code_t err_code;

    err_code = nrf_drv_ppi_init();
    APP_ERROR_CHECK(err_code);

    nrf_drv_timer_config_t timer_cfg = NRF_DRV_TIMER_DEFAULT_CONFIG;
    timer_cfg.bit_width = NRF_TIMER_BIT_WIDTH_32;
    err_code = nrf_drv_timer_init(&m_timer, &timer_cfg, timer_handler);
    APP_ERROR_CHECK(err_code);

    /* setup m_timer for compare event every 100ms */
    uint32_t ticks = nrf_drv_timer_ms_to_ticks(&m_timer, 100);
    nrf_drv_timer_extended_compare(&m_timer,
                                   NRF_TIMER_CC_CHANNEL0,
                                   ticks,
                                   NRF_TIMER_SHORT_COMPARE0_CLEAR_MASK,
                                   false);
    nrf_drv_timer_enable(&m_timer);

    uint32_t timer_compare_event_addr =
nrf_drv_timer_compare_event_address_get(&m_timer,
                                                                  NRF_TIMER_CC_CHANNEL0);
    uint32_t saadc_sample_task_addr   = nrf_drv_saadc_sample_task_get();

    /* setup ppi channel so that timer compare event is triggering sample task in SAADC
*/
    err_code = nrf_drv_ppi_channel_alloc(&m_ppi_channel);
    APP_ERROR_CHECK(err_code);

    err_code = nrf_drv_ppi_channel_assign(m_ppi_channel,
                                          timer_compare_event_addr,
                                          saadc_sample_task_addr);
    APP_ERROR_CHECK(err_code);
}


void saadc_sampling_event_enable(void)
{
```

```
   ret_code_t err_code = nrf_drv_ppi_channel_enable(m_ppi_channel);

   APP_ERROR_CHECK(err_code);
}


void saadc_callback(nrf_drv_saadc_evt_t const * p_event)
{

   if (p_event->type == NRF_DRV_SAADC_EVT_DONE)
   {
      ret_code_t err_code;

      err_code = nrf_drv_saadc_buffer_convert(p_event->data.done.p_buffer,
SAMPLES_IN_BUFFER);
      APP_ERROR_CHECK(err_code);

      //troubleshooting for Saadc
      //int i;
      //nrf_saadc_value_t adc_val;
      //NRF_LOG_INFO("ADC event number: %d", (int)m_adc_evt_counter);
      //for (i = 0; i < SAMPLES_IN_BUFFER; i++)
      //{
      //   adc_val= p_event->data.done.p_buffer[i];
      //   nrfx_saadc_sample_convert(0,&adc_val);
      //   NRF_LOG_INFO("ADC Value: %d", adc_val);//p_event->data.done.p_buffer[i]);
      //   NRF_LOG_INFO("Volts :" NRF_LOG_FLOAT_MARKER
"",NRF_LOG_FLOAT(adc_val*3.6/1024));
      //}


      int i;
      NRF_LOG_INFO("ADC event number: %d", (int)m_adc_evt_counter);

      for (i = 0; i < SAMPLES_IN_BUFFER; i++)
      {
         NRF_LOG_INFO("%d", p_event->data.done.p_buffer[i]);
         if(threshtest==0){
           if(reads == 5) {
             threshtest = 1;
```

```
          threshold = threshold*0.7;
          reads = 0;
          NRF_LOG_INFO("THRESHOLD: %d", (int)threshold);
        }
        else if (p_event->data.done.p_buffer[i]>threshold) {
          threshold = p_event->data.done.p_buffer[i];
        }
      }
      else{
        if (p_event->data.done.p_buffer[i]>threshold&&beatlock==0){
          NRF_LOG_INFO("BEAT"); // print BEAT
          beats++;
          beatlock=1;
        }
        else if (p_event->data.done.p_buffer[i]<(threshold)){
          beatlock=0; //heartrate lock out
        }
      }
    }
    reads++;
    if(reads==15&&threshtest==1){
      BPM = beats*4;
      NRF_LOG_INFO("NEW Calculated BPM = %d",BPM);
      reads=0;
      beats=0;
    }
    NRF_LOG_INFO("BPM = %d",BPM);
    NRF_LOG_INFO("ADC event number: %d", (int)m_adc_evt_counter);
    m_adc_evt_counter++;
  }
}


void saadc_init(void)
{
  ret_code_t err_code;
  nrf_saadc_channel_config_t channel_config =
    NRF_DRV_SAADC_DEFAULT_CHANNEL_CONFIG_SE(NRF_SAADC_INPUT_AIN0);

  err_code = nrf_drv_saadc_init(NULL, saadc_callback);
```

```
    APP_ERROR_CHECK(err_code);

    err_code = nrf_drv_saadc_channel_init(0, &channel_config);
    APP_ERROR_CHECK(err_code);

    err_code = nrf_drv_saadc_buffer_convert(m_buffer_pool[0], SAMPLES_IN_BUFFER);
    APP_ERROR_CHECK(err_code);

    err_code = nrf_drv_saadc_buffer_convert(m_buffer_pool[1], SAMPLES_IN_BUFFER);
    APP_ERROR_CHECK(err_code);

}


/**
 * @brief Function for main application entry.
 */
int main(void)
{
    uint32_t err_code = NRF_LOG_INIT(NULL);
    APP_ERROR_CHECK(err_code);

    NRF_LOG_DEFAULT_BACKENDS_INIT();

    ret_code_t ret_code = nrf_pwr_mgmt_init();
    APP_ERROR_CHECK(ret_code);

    saadc_init();
    saadc_sampling_event_init();
    saadc_sampling_event_enable();
    NRF_LOG_INFO("SAADC HAL simple example started.");

    while (1)
    {
        nrf_pwr_mgmt_run();
        NRF_LOG_FLUSH();
    }
}
```