Directional Impact Sensing Helmet (DISH)

ECE 445 Design Document

Team 5: Patrick Sear, Saathvik Narra, Ryan Josephson
Professor: Viktor Gruev
TA: Ugur Akcal
Spring 2023

Contents

1 Introduction

Providing the problem statement and how we plan on addressing it and a visual diagram about how our solution will work.

1.1 Problem

In the NFL, many athletes suffer from concussions or other conditions as a result of repetitive, severe head trauma. These events can lead to long-term effects on the athlete's health and significantly contribute to the reduced lifespan of professional football players. The average professional football player dies younger than age 60 [1], no doubt accelerated by frequent, intense head trauma. This problem can be helped by making accurate collision data immediately available to medical personnel for making game-time decisions and helmet manufacturers so that they can make informed design choices based on real, in-game data.

1.2 Solution

The Directional Impact Sensing Helmet (DISH) is an electronic system that can be attached inside any existing football helmet. It can determine where on an athlete's head a collision occurs, as well as how hard the hit is. The data collected will be quickly available to personnel on the sidelines so that they can make real-time decisions to ensure the safety of their players. The information will be visualized for ease of use of the team medical staff so that they can best inspect the player.

The data will also be useful for designing the next generation of helmets. Over the course of a season, all hits can be tracked and analyzed so that newer models can be tailored precisely to protect from hits that occur in a real, game-time environment. The helmets could even be designed differently for each position, prioritizing the most common hits each role receives.

For the DISH to work properly, we can group our project into three modules: power, control, and data reception and visualization.

1.3 Visual Aid

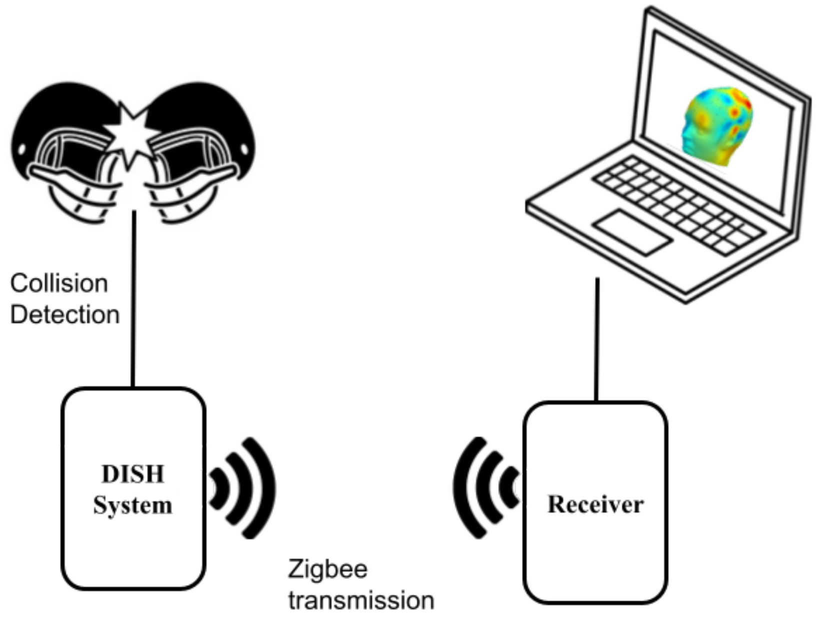

Figure 1: High-Level Overview of DISH System

Upon detecting a collision through signal output from the force sensitive resistor array, the onboard microcontroller begins transmitting data through the Zigbee module. This data will be received and passed to the personal computer, in which it will be analyzed and displayed for the user.

1.4 High-Level Requirements

- The helmet must track the location and severity of each collision within <20% error.
- The receiver must know that a dangerous collision has occurred within 5 seconds of the collision at least 90% of the time.
- The in-helmet system must be able to comfortably fit within a standard football helmet.

2 Design

2.1 Block Diagram

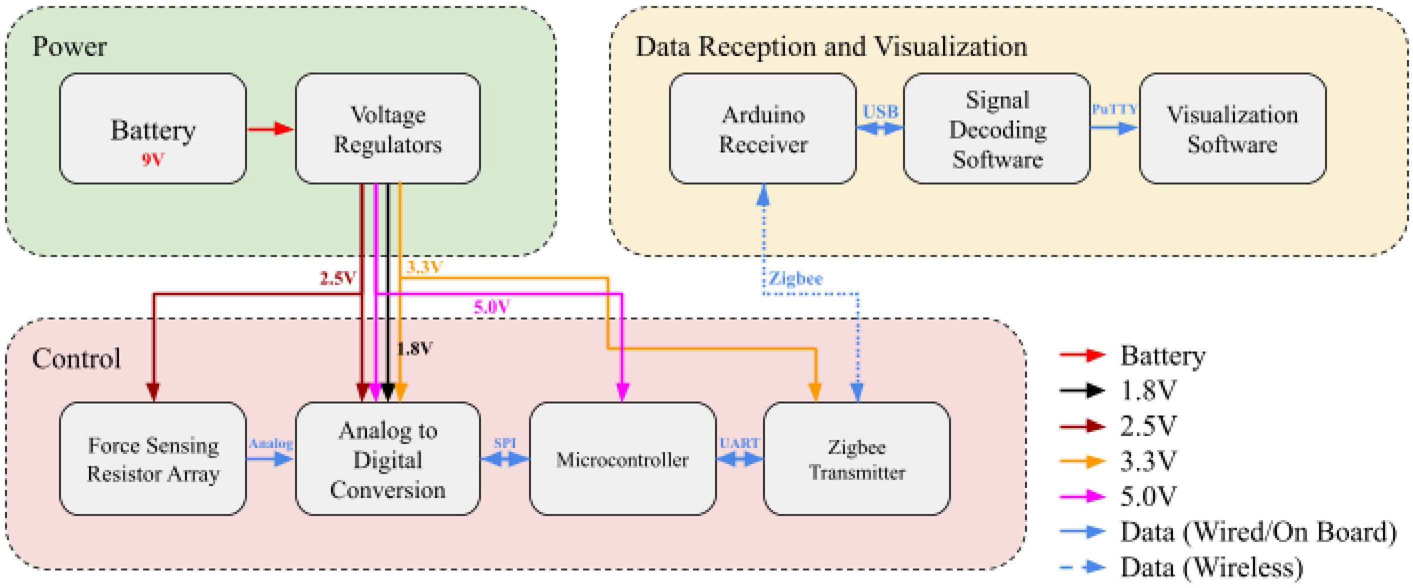

Figure 2: Block diagram

2.2 Physical Design

Within the helmet, our design will be encased in a case. To fully enclose our circuit and the battery, we are opting for a box similar to the P-2409TX potting box from Polycase [2], as shown in *Figure 3*. This encasement is approximately 4 x 2.13 x 0.9 in, so we can fully enclose the entire system within the helmet. Additionally, this case should be small enough that it can fit within the helmet comfortably. Protruding from the encasement will be a 16-pin connector that connects to the force sensing resistors within the helmet.

Figure 3: P-2409TX from Polycase

The physical placement of the force sensing resistors is important. Within the helmet, we will place two towards the front of the helmet, one on either side of the helmet, two towards the back of the helmet, and two on the roof of the helmet. The placement is shown in *Figure 4*. Each colored circle is the anticipated placement of an FSR, and the dashed-line circle is the placement of the last remaining force sensing resistor placed on the opposite side of the helmet. This positioning aims to capture the most amount of data from regions where the player is expected to be hit while still maintaining strong coverage and minimizing dead spots.

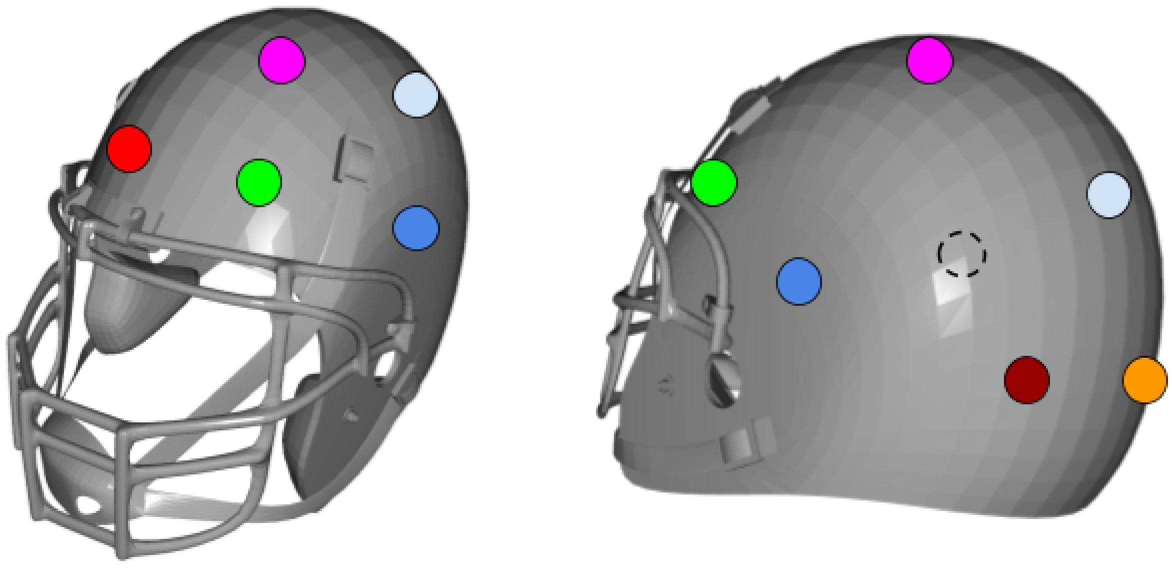

Figure 4: Force Sensing Resistor Placement

2.3 Power Subsystem

2.3.1 Overview

The power subsystem is where the power sources of the DISH are generated. Within the power subsystem is the battery and the voltage regulators.

The DISH internally uses four different voltage levels: 1.8V, 2.5V, 3.3V, and 5V. For this reason, we need a way to generate these voltages. We do this with the use of SPX1117 voltage regulators. Per the datasheet, the overlapping input voltage ranges of each of these voltages is the range of 6.4V-10V [3]. For this reason, the battery must have a voltage level within this range. It is important to recognize that we are using one of each voltage regulator, except for the 2.5V rail, which will have two regulators. This is because we want to reduce the noise as much as possible for the reference voltage for the analog to digital converter to ensure accuracy. The schematic for the power subsystem is shown in *Figure 5*. Each of the five voltage regulators has capacitors on the input and outputs, per the datasheet [3]. Also, as can be seen in *Figure 5*, each power level is decoupled with 1uF capacitors proportional to the number of voltage pins it is supplying. This will help filter out voltage surges and noise and ensure that each component receives clean power.

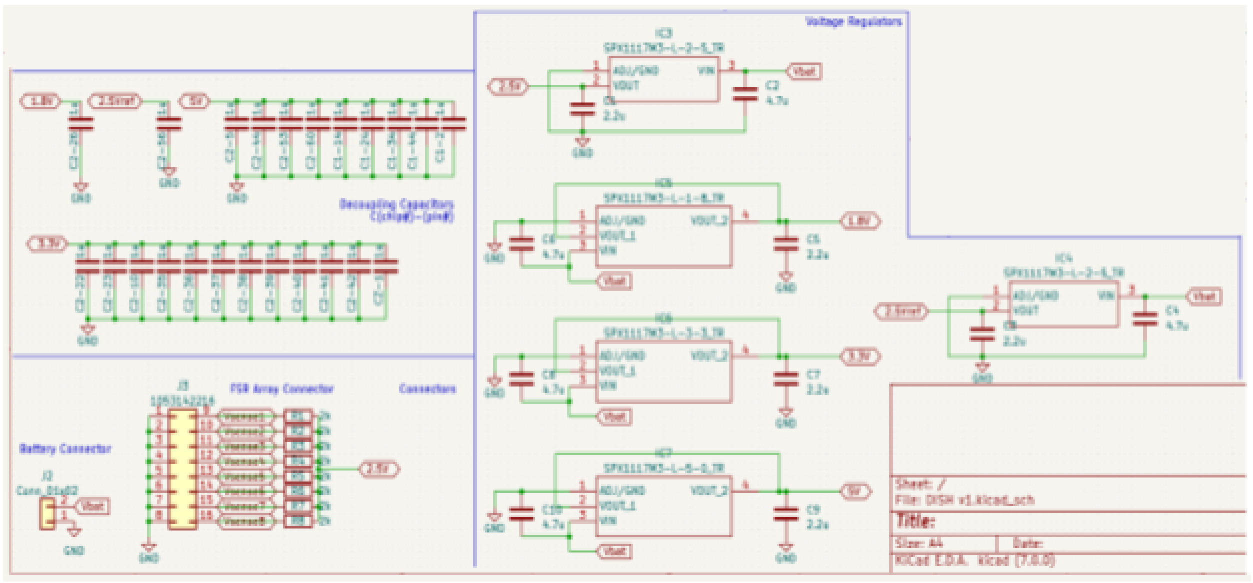

Figure 5: Power Subsystem Schematic

The battery we are using is a rechargeable 9V battery. This battery is different from a standard 9V battery; it comes packaged with additional benefits. For one, this battery falls within the aforementioned voltage range. Also, this battery has a capacity of 1300 mAh. This should allow us to run the DISH for more than long enough for our demo. Additionally, this battery is rechargeable and has built-in overcharge protection, so, for testing and debugging, we can use this battery rather early on without needing to keep track of the charge or buying more batteries.

Because we are using a battery packaged like a 9V battery, we can use a standard 9V battery connector to connect this power supply to our circuit.

2.3.2 Interfaces

- Battery
  - The battery directly connects to the input voltage pins of each of the SPX1117 voltage regulator chips.
- Voltage Regulators
  - The 1.8V voltage regulator supplies power to the analog to digital converter.
  - One of the 2.5V voltage regulators supplies power to the force sensing resistors. The other 2.5V voltage regulator acts as a voltage reference for the analog to digital converter chip.
  - The 3.3V voltage regulator supplies power to the analog to digital converter and to the in-helmet Zigbee module.
  - The 5V voltage regulator supplies power to the microcontroller and the analog to digital converter.

2.3.3 Requirements

The power subsystem must satisfy the following requirements:

1. The battery must supply a constant voltage between 6.4V and 10V and supports up to 350mA of current draw.
2. The 1.8V rail must maintain a voltage between 1.65V and 1.95V per [4] at a current of at least 17mA.
3. The 2.5V rail must maintain a voltage between 2.475V and 2.525V at a current of at least 20mA.
4. The 3.3V rail must maintain a voltage between 2.7V and 3.6V per [4] and [5], at a current of at least 220.5mA.
5. The 5V rail must maintain a voltage between 4.75V and 5.25V per [4] at a current of at least 91mA.

To verify these requirements, the procedures outlined in *Table 1* will be employed.

Requirements	Verification
• The battery must supply a constant voltage between 6.4V and 10V and supports up to 350mA of current draw.	• Connect wires to the cathode and anode of the battery. Do not solder directly onto the battery. • Connect the wires to an electronic load. • Connect a digital multimeter across the battery in voltage-read mode. • Set the electronic load to constant current mode with a load of 350mA. • Verify that the voltage read across the battery is between 6.4V and 10V while supplying 350mA to the electronic load.
• The 1.8V rail must maintain a voltage between 1.65V and 1.95V per [4] at a current of at least 17mA.	• Solder a wire on each of the pins of the 1.8V SPX1117 chip. • Connect a power supply across the input voltage and the ground pins. • Connect a digital load across the output voltage and the ground pins, set to a resistance of 115Ω or less. • Connect a digital multimeter in voltage-read mode across the output voltage and the ground pins. • Set the power supply to 9V and power it on. • Verify that the voltage read across the resistive load falls between 1.65V and 1.95V and that the current supplied by the power source is greater than or equal to 17mA.
• The 2.5V rail must maintain a voltage between 2.475V and 2.525V at a current of at least 20mA.	• Solder a wire on each of the pins of the 2.5V SPX1117 chip. • Connect a power supply across the input voltage and the ground pins. • Connect a digital load across the output voltage and the ground pins, set to a resistance of 127Ω or less. • Connect a digital multimeter in voltage-read mode across the output voltage and the ground pins. • Set the power supply to 9V and power it on. • Verify that the voltage read across the resistive load falls between 2.475V and 2.525V and that the current supplied by the power source is greater than or equal to 20mA.

● The 3.3V rail must maintain a voltage between 2.7V and 3.6V per [4] and [5], at a current of at least 220.5mA.	● Solder a wire on each of the pins of the 3.3V SPX1117 chip. ● Connect a power supply across the input voltage and the ground pins. ● Connect a digital load across the output voltage and the ground pins, set to a resistance of 16.5Ω or less. ● Connect a digital multimeter in voltage-read mode across the output voltage and the ground pins. ● Set the power supply to 9V and power it on. ● Verify that the voltage read across the resistive load falls between 2.7V and 3.6V and that the current supplied by the power source is greater than or equal to 220.5mA.
● The 5V rail must maintain a voltage between 4.75V and 5.25V per [4] at a current of at least 91mA.	● Solder a wire on each of the pins of the 5V SPX1117 chip. ● Connect a power supply across the input voltage and the ground pins. ● Connect a digital load across the output voltage and the ground pins, set to a resistance of 57Ω or more. ● Connect a digital multimeter in voltage-read mode across the output voltage and the ground pins. ● Set the power supply to 9V and power it on. ● Verify that the voltage read across the resistive load falls between 4.75V and 5.25V and that the current supplied by the power source is greater than or equal to 91mA.

Table 1: Power Subsystem RV Table

2.4 Control Subsystem

2.4.1 Overview

The control subsystem manages how the data is processed and sent inside the DISH until transmitted to the receiver. Within the control subsystem is the force sensing resistors, analog to digital converter, microcontroller, and the Zigbee transmitter.

The microcontroller we are using is an ATmega32U4 [6]. This is because we will be able to program it with the Arduino bootloader, and then use the Arduino IDE to implement software. This microcontroller will be in direct communication with the off-chip simultaneous sampling ADC and the Zigbee transmitter. Using the microcontroller, we will make a trigger that, if triggered, would pass data from the force sensing resistor array to the receiver. The connections between the microcontroller and some of its peripherals can be seen in *Figure 6.*

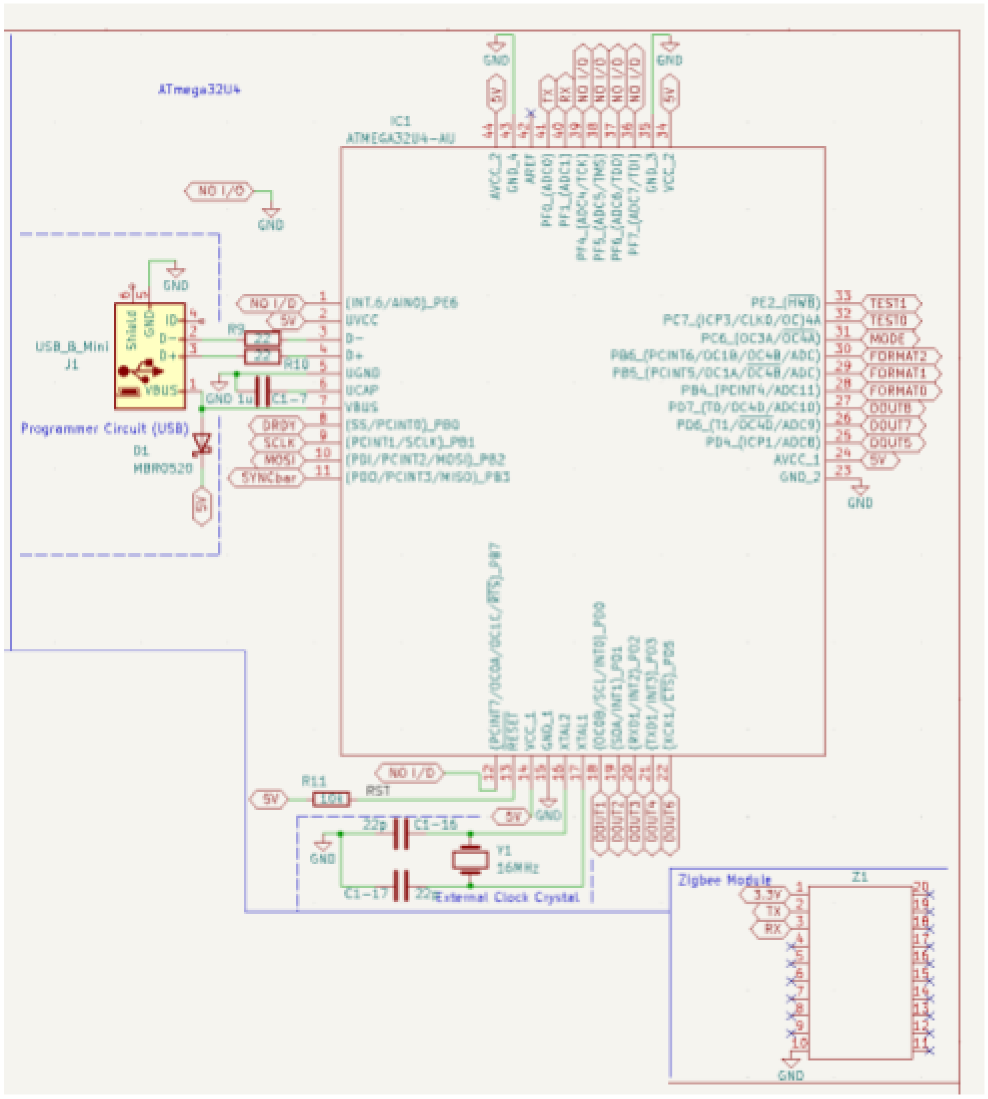

Figure 6: ATmega32U4, Zigbee, Programmer schematic

Part of the connections to the microcontroller includes a programmer circuit, which can also be seen in *Figure 6*. This circuit should allow us to connect the microcontroller to a computer via USB to more rapidly program it.

To measure the force applied to the user, there will be an array of strategically placed force sensing resistors placed within the helmet. They will be used to measure how much force is received to the head. To be able to measure relevant data from the force sensing resistors, we will construct voltage divider circuits for each force sensing resistor such that we can measure voltage changes when a force is applied to the sensor.

Force sensitive resistors rated for the intensity of collision seen in a football game can be quite expensive. For the sake of this project, we are going to use force sensing resistors that are rated for a lower amount of force. If we were to change these sensors for the more expensive alternatives, very few changes would be required. Effectively, the entire system would be the same, save for some resistor values and the calibration of some measurements. For this reason, we believe it is acceptable to use force sensing resistors with a lower force rating.

Due to how the ATmega32U4 MUXes its analog inputs and its low resolution, we are going to use an off-chip 16-bit simultaneous sampling ADC. We plan on using the ADS1178IPAPR chip, since it can take up to eight simultaneous analog inputs and communicate via SPI, so we can use fewer pins in the microcontroller. It also provides a higher resolution digital conversion and a faster sampling rate than is available on the ATmega32U4. The ADC pin settings can be seen in *Figure 7*, and its interconnections to the microcontroller can be compared with the schematic in *Figure 6*.

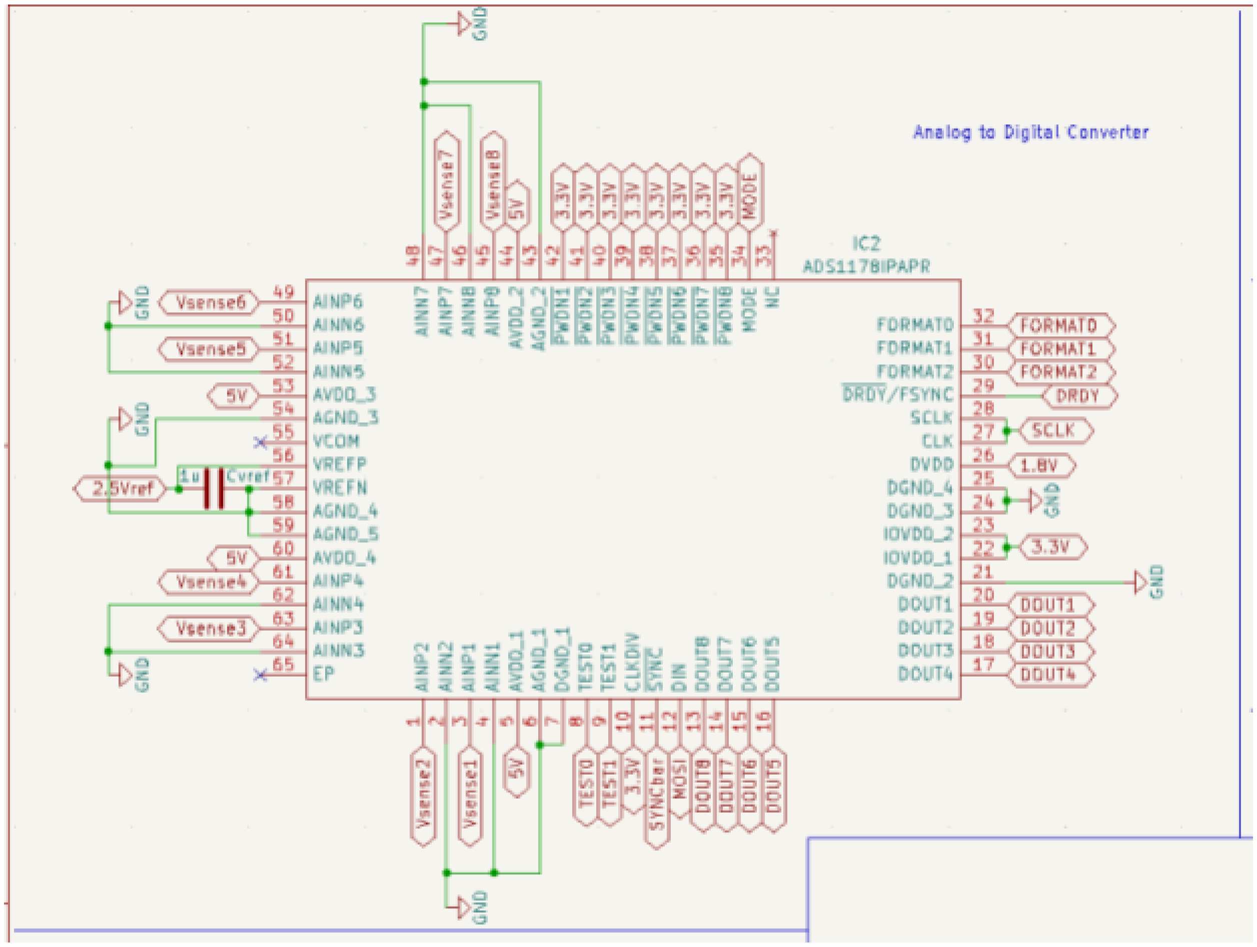

Figure 7: ADC settings and interconnections schematic

The microcontroller in the helmet will receive the data from the ADC and will transmit the data to the receiver via a Zigbee transmitter. Zigbee is an optimal communication protocol because it is cheap and has a range that is usable for our purposes. To ensure the wireless connection between the microcontroller and the receiver is intact, the microcontroller will send a verification signal periodically to the receiver. The specific module we are going to use is the ZB Series S2C XBee Pro Module. The Zigbee transceivers operate on the IEEE 802.15.4 transmission protocol. It sends 127-byte packets, 68 bytes of which are reserved for the payload data [7]. The remaining 59 bytes are reserved as shown in *Figure 8.*

6 bytes PHY	9 bytes MAC	22 bytes NWK	9 bytes APS	3 bytes ZCL	68 bytes Payload	10 bytes MIC

Figure 8: Zigbee Packet Structure

The remaining 59 bytes are to be used for network security/encryption and error detection/correction. We will be transmitting a 2-byte digital voltage reading from each of the 8 sensors along with one byte of additional metadata per sensor reading. This makes our total payload size 24 bytes which will easily fit within the 68 bytes allocated.

2.4.2 Interfaces

- Force Sensing Resistor Array
  - The force sensing resistor array receives 2.5V from the 2.5V voltage regulator.
  - The force sensing resistor array conditions an analog signal that is read by the ADC with a direct connection.
- Analog to Digital Converter
  - The analog to digital converter receives 1.8V, 2.5V, 3.3V, and 5V from the corresponding voltage regulators.
  - The ADC reads the analog data from the resistor array and converts it into digital data which is then read on the microcontroller using SPI.
- Microcontroller
  - The microcontroller receives 5V from the 5V voltage regulator.
  - The microcontroller takes the data coming from the ADC and computes whether the values from the force sensors indicate a collision.
  - The microcontroller packages the data along with identifying monikers, and gives it to the Zigbee transmitter.
- Zigbee Transmitter
  - The Zigbee Transmitter receives 3.3V from the 3.3V voltage regulator.
  - The Zigbee transmitter receives the packets from the microcontroller and sends it to the receiver using a Zigbee transmission.

2.4.3 Requirements

The control subsystem must satisfy the following requirements:

1. The microcontroller must be able to comprehend data from the analog to digital converter via SPI over 90% of the time.
2. The microcontroller must be able to correctly identify when a collision occurs with more than 90% accuracy.
3. The microcontroller must be able to send and receive signals to and from the remote receiver via the Zigbee transmitter more than 90% of the time.
4. The off-chip simultaneous sampling ADC must be able to measure all signals from the force sensing resistor array concurrently more than 90% of the time.
5. The Zigbee transmitter must be able to properly send a signal up to 40 meters away.
6. The signal must be able to be sent through the helmet and through adverse conditions with an accuracy of over 90%.

To verify these requirements, the procedures outlined in *Table 2* will be employed.

Requirements	Verification
● The microcontroller must be able to comprehend data from the analog to digital converter via SPI over 90% of the time.	● Construct a completed DISH without putting the circuit inside the helmet or plugging in the force sensing resistor array connector. ● Program the microcontroller to communicate with the ADC via SPI to read the data from the first FSR channel, and to repeat it in another I/O port. ● Connect a power supply across a digital load and connect these across where one of the force sensing resistors would be. ● Connect an oscilloscope to the designated output I/O port to read the voltage. ● Turn on the power supply to 2.5V, the digital load to a 100Ω load, and verify that the I/O port is expressing 7FFFh digitally. ● Change the power supply to 1.25V, and verify that the I/O port is expressing 3FFFh digitally, with an error margin of 2%.
● The microcontroller must be able to correctly identify when a collision occurs with more than 90% accuracy.	● With a fully constructed DISH helmet, connect an oscilloscope across one of the I/O pins. ● Program the microcontroller to read the data from the analog to digital converter as intended, except when a sharp voltage change is detected, output a high voltage to the I/O pin the oscilloscope is reading. ● With a hammer, tap where one of the force sensing resistors are on the outside of the helmet. ● Review the oscilloscope and confirm that the voltage level is constantly high, signaling a collision occurred.
● The microcontroller must be able to send and receive signals to and from the remote receiver via the Zigbee transmitter more than 90% of the time.	● Fully construct a DISH helmet. There is no need to connect the force sensing resistor array for this testing. ● Connect the other Zigbee module to an Arduino and a laptop. ● Program the microcontroller to send a known-test signal periodically via Zigbee. Program the Zigbee-Arduino setup to receive the Zigbee signal and print it to the Arduino console. ● Place each setup at least 10 feet apart. ● Send a test signal from the DISH to the Zigbee-Arduino receiver.

	● Verify that the intended Zigbee signal matches the data shown in the Arduino console. ●
● The off-chip simultaneous sampling ADC must be able to measure all signals from the force sensing resistor array concurrently more than 90% of the time.	● Fully construct a DISH helmet. Do not connect the force sensing resistor array for this testing. ● Program the microcontroller to read the data from the analog to digital converter normally, except have it print all data to an I/O pin directly after receiving the data. ● Connect a power supply across a digital load. ● Connect an oscilloscope to the designated I/O pin. ● Connect each of the eight data lines across the power supply and digital load. ● Power on the power supply to read 2.5V, and the digital load to be a load of 100Ω. ● Verify that the oscilloscope is reading constant logic 1's.
● The Zigbee transmitter must be able to properly send a signal up to 40 meters away.	● Connect each of the Zigbee modules to an Arduino and a laptop. ● Create Arduino code for one Zigbee-Arduino setup to act as a transmitter and the other to act as a receiver. ● Send a test signal from the transmitter to the receiver at a close range to ensure proper operation of the transmission. ● Move the Zigbee-Arduino setups such that they are exactly 40m away. Measure this distance using a tape measure. ● Test the test signal once more, and verify that the transmitted signal is received.
● The signal must be able to be sent through the helmet and through adverse conditions with an accuracy of over 90%.	● Connect each of the Zigbee modules to an Arduino and a laptop. ● Create Arduino code for one Zigbee-Arduino setup to act as a transmitter and the other to act as a receiver. ● Send a test signal from the transmitter to the receiver at a close range in open air to ensure proper operation of the transmission. ● Place a water bottle between the Zigbee-Arduino setups to simulate rain. ● Test the signal again, and verify that the transmitted signal is received. ● Move one Zigbee-Arduino setup behind a wall.

	● Test the signal once more and verify that the transmitted signal is received.

Table 2: Control Subsystem RV Table

2.5 Data Reception and Visualization Subsystem

2.5.1 Overview

This subsystem is responsible for collecting the data transmitted by the helmet and displaying it in an easily digestible format to the user.

The package will be sent by a Zigbee transmitter on the DISH helmet system. The receiver will consist of a personal computer, Arduino Uno Rev3 and Zigbee Digi XBee S2C 802.15.4 RF Module [5]. The Xbee will receive this package and transmit data directly to the Arduino board. Upon receiving a signal from the XBee module, the Arduino will output the package to the computer's serial port via USB. The serial port input will be processed using PuTTY serial console and stored as a text file. The text file can be taken as an input to the visualization software which will display it as a heatmap over a model highlighting the location and severity of the impact. The visualization software is programmed in Python using the free open-source software package Open3D [8]. The displayed model can be manipulated in space by the user, as seen in *Figure 9*.

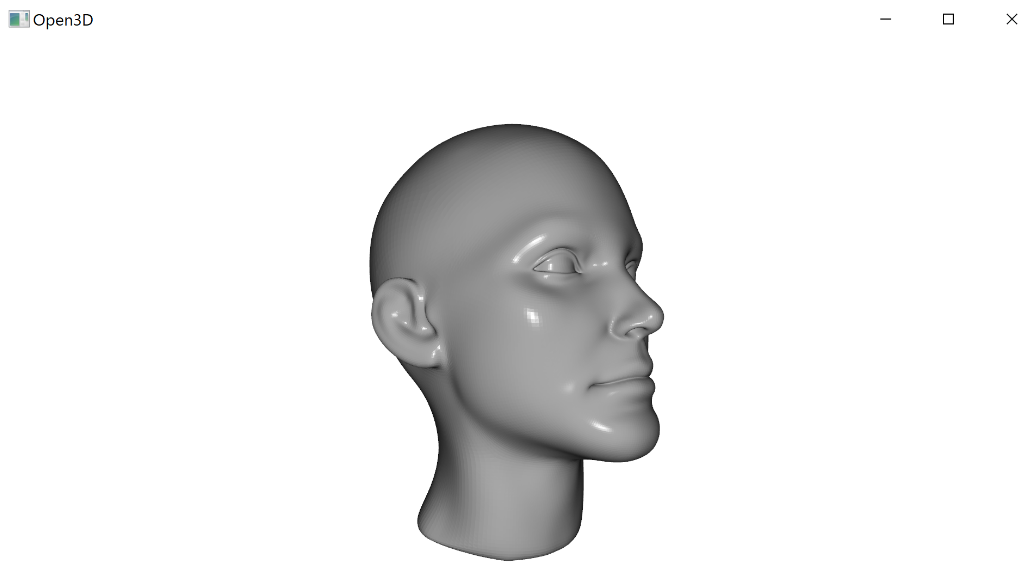

Figure 9: The received data will be displayed on a model as shown above.

It is important to recognize that the end user will have access to two separate outputs from the system. One output will be a notification that a potentially dangerous collision has occurred. The other output will be the actual visualization of the received data. These two separate outputs allow for a faster decision on whether a hazardous collision may have occurred, since this is the main information needed to decide if a player needs to be pulled or not. This allows for more time to actually visualize the data, since the data can be visualized in the time it takes to remove the player from the field.

2.5.2 Interfaces

- Zigbee
  - The Zigbee module wirelessly receives data from the microprocessor and transmitter on the helmet.
  - The Zigbee module communicates to the Arduino through UART.
- Arduino
  - The Arduino receives data from the Zigbee receiver through UART.
  - The Arduino sends data to the personal computer serial port through USB.
- PuTTY
  - PuTTY catches serial port data as it is sent from the Arduino and outputs the data to text file logs.
- Python
  - The visualization program accesses text file outputs from PuTTY on the local disk.
  - The software outputs the result of data analysis to the monitor in the form of a color-coded 3d model.

2.5.3 Requirements

The reception/visualization subsystem must satisfy the following requirements:

1. The receiver must receive packets sent with a maximum error rate of 10%.
2. The signal received by the Arduino must be delivered to the serial input exactly as it is seen by the hardware.
3. The visualizer must be able to identify the impact within 20% of where it actually occurred.
4. The data must be visualized within 30 seconds of packet reception.

To verify these requirements, the procedures outlined in *Table 3* will be employed.

Requirements	Verification
● The receiver must receive packets sent with a maximum error rate of 10%.	● Set up a DISH helmet and receiver in a typical use case environment– outdoors, ~40m separation. ● Program the on-board microcontroller to send a fixed series of packets in the standard Zigbee format. ● Compare the transmitted and received signals, verify that the error rate is less than 10%.
● The signal received by the Arduino must be delivered to the serial input exactly as it is seen by the hardware.	● Program the receiver to send fixed packets to the computer over the serial port. ● Directly read the digital output of the Arduino using an oscilloscope at the serial output pins. ● Read the data saved in the text file by PuTTY. ● Verify that the data is exactly the same as all points.
● The visualizer must be able to identify the impact within 20% of where it occurred.	● Set up a completed DISH helmet and receiver in close proximity to ensure signal integrity. ● Impact the helmet at clearly defined points using a hammer, both directly on a sensor and between sensor locations. ● Ensure that the calculated impact location is within 20% of the true location.
● The data must be visualized within 30 seconds of packet reception.	● Set up a DISH helmet and receiver in close proximity to ensure signal integrity. ● Program the on-board microcontroller to send a fixed series of packets in the standard format. ● Program the Arduino to turn on an LED upon packet reception. Start timing when the LED turns on. ● Run the visualization software when the data has been transmitted. ● Ensure that all times between reception and display on the screen are less than 30s.

Table 3: Reception and Visualization Subsystem RV Table

2.6 Tolerance Analysis

We'll need to ensure that we can obtain a sufficient digital resolution when converting the analog signals for use in the microcontrollers. This problem is exacerbated by the inherent nonlinearity of the force sensing voltage divider and force sensitive resistor. The voltage divider is to be constructed as shown below in *Figure 10*.

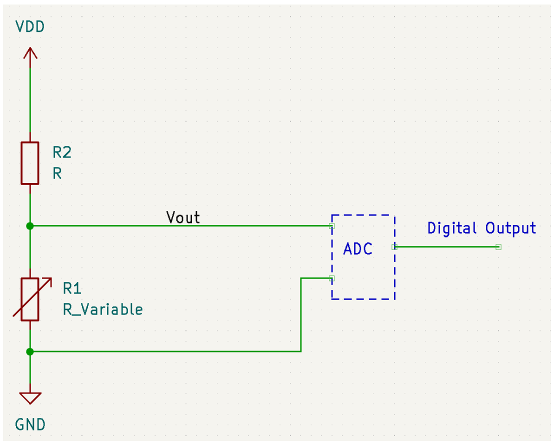

Figure 10: Voltage divider schematic

With this schematic, the analog output will be governed by a nonlinear relationship to R1, which is itself already a nonlinear dependance on force. The output voltage potential is as seen below in *Equation 1*.

$$V_{out} = V_{DD} * \frac{R1}{R1+R2} \qquad (Eq.\ 1)$$

Simulating the output of the voltage divider using the force sensitive resistor dependency highlights the potential problems with this setup. For the expression shown in *Equation 2*, R_f is the resistance of the force sensing resistor, R_0 is an initial resistance value with no force applied, and $F_{applied}$ is the applied force. The -1.3 exponential is an estimation based on the datasheet of our chosen force sensing resistor. A force vs output voltage plot can be seen below in *Figure 11*.

$$R_f = R_0 * F_{applied}^{-1.3} \qquad (Eq.\ 2)$$

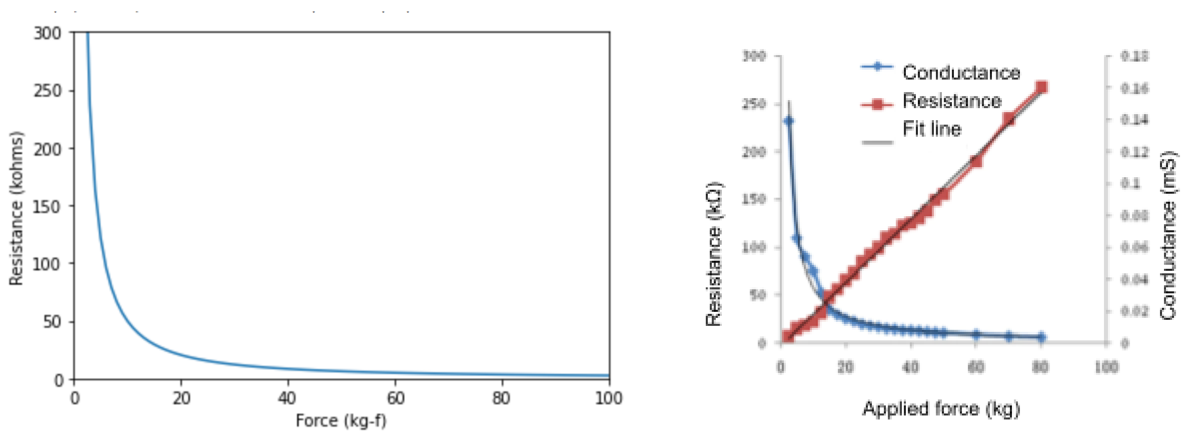

Figure 11: Comparison of estimated resistance equation with manufacturer datasheet [9]

Because the resistance of the force sensing resistor is nonlinear, we may encounter some trouble with precisely measuring the force input. Using the voltage divider circuit as seen in *Figure 10*, we can strategically select our R2. By doing this, we can experience more noticeable changes from the force sensing resistor when experiencing forces of magnitude that are of interest. This can help with linearization of data to minimize other calculations. We have selected 2kΩ as our secondary resistor sizing. This choice was made since at a large force applied, the force sensitive resistor will show resistances of ~5kΩ-2kΩ. As shown in *Figure 12*, this balances linearity with output swing, allowing us to receive the most evenly-spaced and accurate data.

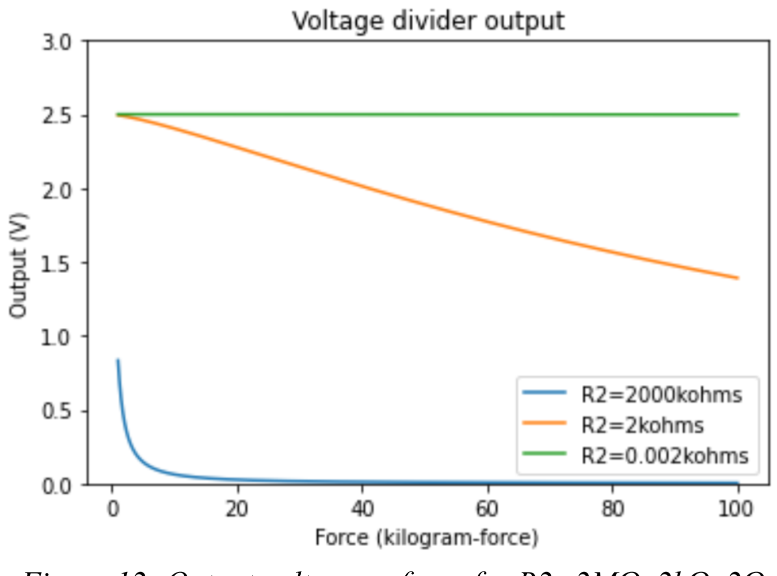

Figure 12: Output voltage vs force for R2=2MΩ, 2kΩ, 2Ω

Another aspect to keep in mind is the resolution of our data. As seen in *Figure 13*, if we have too low of a resolution, we will have trouble with accuracy.

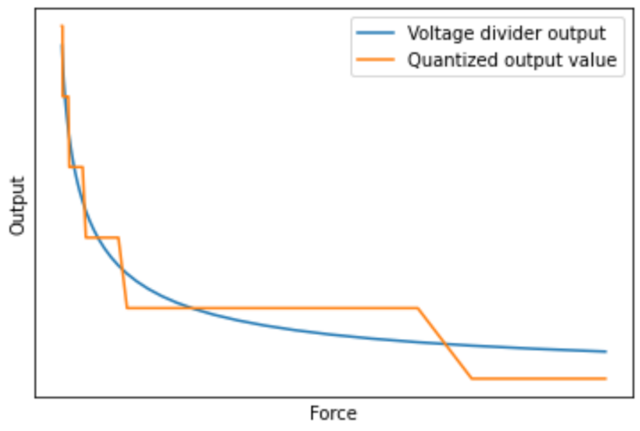

Figure 13: Risk of Quantization at 5 Level Resolution

Clearly, this system has too few voltage steps, and will not be able to digitally differentiate analog signals at high force levels. We plan to use a 16-bit ADC, so we do not expect this to be a problem. However, it is still a concern to be aware of during the design process. The simulated 16-bit resolution overlaid with analog voltage output as a function of force can be seen below in *Figure 14*.

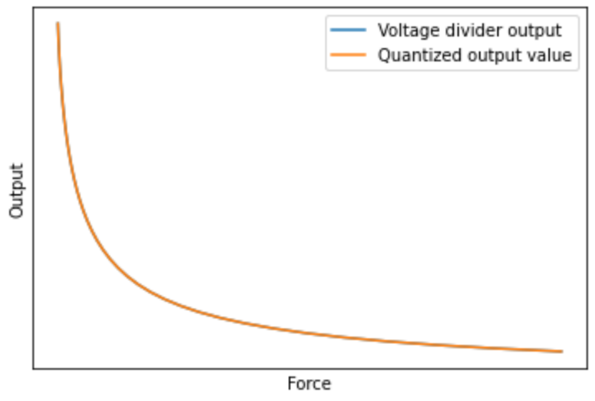

Figure 14: Risk of quantization at 16-bit resolution

Numerically, combining *Equation 1* and *Equation 2* to get *Equation 3*, using 2kΩ as R2 and 2.5V as VDD, we find that the difference in voltage output between $F_{applied} = 99\,kgf$ and $F_{applied} = 100\,kgf$ (the maximum rated force) is 8.05 mV.

$$V_{out} = V_{DD} * \frac{R_0 * F_{applied}^{-1.3}}{R_0 * F_{applied}^{-1.3} + R2} \quad (Eq.\ 3)$$

Our ADC has 16-bit resolution and has an input range of -2.5V to 2.5V with a constant voltage step increment of ~76.3uV as calculated from *Equation 4* where *k* is the number of bits of resolution..

$$V_{step} = \frac{V_{max} - V_{min}}{2^k} \quad (Eq.\ 4)$$

Using *Equation 5*, we see that this results in ~105 steps in between 99 and 100 kgf.

$$N_{steps} = \frac{V_1 - V_2}{76.3uV} \quad (Eq.\ 5)$$

Clearly, 105 steps per kilogram of force at the region with the least analog resolution in the force sensing resistor is sufficient to differentiate on a digital scale.

Also, some additional consideration needs to be made for the thermal characteristics of the voltage regulators. Because the voltage regulators used in the circuit are linear rather than switching, the voltage drop realized within the regulators is power lost as heat. We need to ensure the voltage regulators are still within their operating temperatures. According to the datasheet, each of the voltage regulators can operate up to 125 °C and have a junction to ambient thermal resistivity of 46 °C/W [3]. So, the temperature increase for each voltage regulator can be calculated using *Equation 6*.

$$P_{heat} = (V_{input} - V_{output}) \cdot I_{draw} \quad (Eq.\ 6)$$

With a 9V battery, per *Equation 6*, the 1.8V rail dissipates $(9 - 1.8)(0.017) = 0.1224W$, corresponding to a temperature rise of $0.1224 \cdot 46 = 5.63\,°C$. The worst-case 2.5V rail dissipates $(9 - 2.5)(.02) = 0.13W$, which corresponds to a temperature rise of $0.13 \cdot 46 = 5.98\,°C$. The 3.3V rail dissipates $(9 - 3.3)(0.2205) = 1.257W$, which corresponds to a temperature rise of $1.257 \cdot 46 = 57.68\,°C$. The 5V rail dissipates $(9 - 5)(0.091) = 0.364W$, corresponding to a temperature rise of $0.364 \cdot 46 = 16.744\,°C$. It can be assumed that the ambient temperature is 25 °C, so the most problematic voltage rail is definitely the 3.3V rail, which is likely to reach a temperature upwards of 83 °C. This is still operational, but it is a high temperature that needs to be considered when both designing the PCB and the encasement.

3 Cost & Schedule

3.1 Cost Analysis

3.1.1 Parts Cost Analysis

The total cost for parts as seen in *Table 4* is $174.09. Notably, more parts were ordered than required. This is done in case parts are damaged during testing or to obtain better deals.

Description	Part Number	Unit Price	Quantity	Cost for Quantity
ADC	ADS1178	$18.95	2	$37.90
Microcontroller	ATmega32U4	$5.68	1	$5.68
XBee Transmitter	XBee Module - ZB Series S2C - 2mW with Wire Antenna	$22.95	3	$68.85
1.8V regulator	SPX1117M3-L-1-8/TR	$0.53	3	$1.59
3.3V regulator	SPX1117M3-L-3-3/TR	$0.53	3	$1.59
5.0V regulator	SPX1117M3-L-5-0/TR	$0.53	3	$1.59
Force Sensing Resistors	CN1501541594	$24.44	1	$24.44
Male Conn	WM15179-ND	$4.68	1	$4.68
Female Conn	WM15025-ND	$1.03	1	$1.03
2.5V regulator	SPX1117M3-L-2-5/TR	$0.53	4	$2.12
9V battery	n/a	$16.99	1	$16.99
2kΩ Resistor	RNCP0805FTD2K00	$0.073	20	$1.46
10kΩ Resistor	SR1-0805-310	$0.073	10	$0.73
1uF Capacitor	CL10A105KA8NNNC	$0.031	40	$1.24
4.7uF Capacitor	CL10A475KQ8NNNC	$0.039	10	$0.39
22pF Capacitor	06035A220JAT2A	$0.10	4	$0.40
USB-mini-B	GMSB0530112C1HR	$0.46	2	$0.92
16MHz Crystal	ECS-160-20-3X-TR	$0.35	3	$1.05

Schottky Diode	LMB12S-TP	\$0.48	3	\$1.44
Total				\$174.09

Table 4: Itemized List of Parts

3.1.2 Hours of development and Labor Costs

Our group consists of 2 Electrical Engineers and 1 Computer Engineer. Average pay for Electrical Engineers with a bachelor's degree is \$80,000 and Average pay for Computer Engineers is \$100,000 for computer engineers according to the Grainger College of Engineering [10, 11].

Category	**Estimated Hours**		
	Saathvik	Ryan	Patrick
Circuit Design	0	20	5
Board Layout and Components Check	10	25	20
Full Stack System Monitor	40	4	30
Soldering	6	15	8
Prototype And Debug	60	60	60
Documentation and Logistic	25	30	25
Total Hours	**141**	**154**	**148**

Table 5: Hours of Development

Labor Cost:

Saathvik: $\$50\ (hourly\ rate)\ \cdot\ 2.5\ \cdot\ 141\ (total\ hours)\ =\ \$17625\quad (Eq. 7)$

Ryan: $\$40\ (hourly\ rate)\ \cdot\ 2.5\ \cdot\ 154\ (total\ hours)\ =\ \$15400\quad (Eq. 8)$

Patrick: $\$40\ (hourly\ rate)\ \cdot\ 2.5\ \cdot\ 148\ (total\ hours)\ =\ \$14800\quad (Eq. 9)$

Total Labor Cost = \$47825

3.1.3 External Resources

- Machine Shop: The machine shop is going to be used to modify a physical enclosure for our project to encase the electronics. The generic enclosure we get will need to be modified in the workshop to have holes for power and a hole for connectors to connect force sensors with PCB. It was estimated modifications to our enclosure would take three hours.

- Senior Design Lab Workshop: We need the lab in order to do soldering and testing using the oscilloscope, a Digital Multimeter, and DC Power Supply.

3.1.4 Total Costs

Category	**Estimated Cost**
Parts	$174.09
Labor	$47825.00
Total	$47999.09

Table 6: Total Costs

3.2 Schedule

Week	**Task**	**Person**
2/20	Start circuit schematic design, component selection	Ryan
	Research XBee/Arduino interface research, impact localization algorithm from sensor data	Saathvik
	Visualization software research, initial stage development	Patrick
	Design Document, Team Contract	**All**
2/27	Complete schematic design, begin PCB layout, finalize component selection and ordering	Ryan
	Begin writing receiver Arduino code for XBee interface,	Saathvik
	Expand existing software visualization, full heatmap functionality, streamlined file IO	Patrick
	Design review	**All**
3/6	Finalize PCB layout, confirm communication/plans with machine shop	Ryan
	Complete Arduino code, begin testing with XBee modules, PCB verification	Saathvik
	Log serial communication to txt file, define	Patrick

	mappings from raw data to meaningful info, PCB verification	
	1st round PCB orders	**All**
3/13	Purchased component validation	Ryan
	Confirm Zigbee communications	Saathvik
	Streamline serial → text file → visualization process	Patrick
3/20	Assemble PCB, validate on-board functionality	Ryan
	Integrate XBee module with microcontroller on board, ensure communication	Saathvik
	Validate timing, error requirements of software, PCB validation	Patrick
3/27	PCB revisions, component re-selection	Ryan
	PCB revisions, verify wireless connectivity/range requirements	Saathvik
	PCB revisions, demo full information flow helmet → software	Patrick
	Second round PCB orders	**All**
4/3	Full PCB/encasement/helmet assembly, modify as needed	Ryan
	Impact localization algorithm testing/modification	Saathvik
	Impact localization algorithm implementation with display	Patrick
4/10	Secondary PCB validation, enclosure modifications	Ryan
	Secondary PCB validation, verification of high-level requirements	Saathvik
	Secondary PCB validation, complete integration of software	Patrick
	Team Contract Fulfillment	**All**
4/17	Prepare demo, hardware troubleshooting	Ryan

	Prepare demo, communications troubleshooting	Saathvik
	Prepare demo, software troubleshooting	Patrick
	Mock Demo	**All**
4/24	Prepare demo, presentation, report	Ryan
	Prepare demo, presentation, report	Saathvik
	Prepare demo, presentation, report	Patrick
	Final Demo, Mock Presentation	**All**
5/1	Prepare presentation, final report	Ryan
	Prepare presentation, final report	Saathvik
	Prepare presentation, final report	Patrick
	Final Presentation	**All**

Table 7: Schedule

4 Discussion of Ethics and Safety

Given that this device is designed to be used in a high-impact environment, it is critical that it is designed with only the highest standard of safety in mind. The IEEE Code of Ethics states that it is our responsibility "to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design" [12]. This is especially relevant since our device will be located within a helmet, very close to the head of the user.

It is important that the physical footprint of the device is as small as possible so that it will not pose any extra risk to the wearer. Existing in-helmet devices like quarterback transceiver modules can serve as a model of safe physical design choices. A small, plastic encasement with additional padding will help to mitigate any risk presented by the device.

Another area for concern is the network of force sensing resistors placed within the helmet. We must ensure the addition of our device does not reduce the safety the football helmet already provides to the user. For this reason, we must ensure all padding is maintained within the helmet. With this, we must also make sure all cables placed within the helmet pose no safety hazards, including extra pressure felt on the head and possibly loose cables.

We also must consider the inherent risk of a battery system. We must consider both battery placement, as well as the durability of the battery. There is a risk of placing a battery so close to the head and in a heavy contact environment. The batteries must be well encased and have no possibility of failing in a dangerous way due to a collision. Per OSHA guidelines [13], potential hazards to lithium batteries include “dropping, crushing, and puncturing,” each of which are possible within a football game. The device will also be located in the bottom corner of the helmet, somewhere collisions are less frequent, so the circuit and battery will be under reduced stress. We must also ensure there is minimal risk of our battery exploding, as certain lithium batteries have this possibility. According to the OSHA [13], to reduce risk to the user, the batteries must be inspected “for signs of damage, such as bulging/cracking, hissing, leaking, rising temperature, and smoking before use” in case thermal runoff is imminent.

For this reason we have chosen to use a rechargeable 9V 1300mAh Lithium-Polymer battery. While not as robust as a LiPO4 or NiMH battery, the Lithium-polymer battery does offer some advantages over a traditional Lithium-ion battery. The primary difference is that Lithium-polymer electrolytes are contained in a solid or gel polymer rather than liquid carbonate electrolytes as in a Lithium-ion battery. This comes with many advantages such as low flammability, mechanical stability, and zero electrolyte leakage [14]. To expand further upon our design, we would ideally use a more durable battery chemistry. However, for the scope of this prototype, the safety features of the Lithium-polymer battery should be more than sufficient.

Due to the nature of working with lithium-based battery systems, we have investigated the proper safety precautions for maintenance and use of lithium batteries in an electrical system. For more passive safety precautions, we will be sure to store the battery in dry, cool locations when not in use. It is equally important that it is maintained in good condition even when not in use. Since it is a rechargeable battery, we will ensure to remove it from the power source when it is fully charged. As previously mentioned, the battery has built-in overcharge protection, which should prove to be a safety bonus as well. Before and after running any experiments in which the battery will be used to power the DISH circuit, it will be carefully inspected to ensure that there is no cracking, burn marks or any other sign of external damage. When testing the full system, due to the nature of the product, it will be essential to not place the battery in any location that may put it at greater risk than if the helmet system is being used as intended.

Additionally, the testing process for the DISH helmet has potential risks if not executed properly. Whenever possible, the helmet will be tested with a dummy or model head, not a human head. While the helmets, of course, are designed to protect from collision, there always exists a risk of injury. This is especially the case if the wearer is not properly trained or acclimated to the impacts.

These safety precautions will allow for the rapid and safe development of a system with the potential to create a safer environment for many more athletes in the future.

5 Citation

[1] V. T. Nguyen et al., “Mortality Among Professional American-Style Football Players and Professional American Baseball Players,” JAMA Network Open, vol. 2, no. 5, p. e194223, May 2019, doi: https://doi.org/10.1001/jamanetworkopen.2019.4223.

[2] “P-2409TX TX Series Plastic Potting Boxes for Electronics,” *Polycase*. https://www.polycase.com/p-2409tx.

[3] MaxLinear, “800mA Low Dropout Voltage Regulator,” SPX1117 datasheet, Sept. 2018, https://assets.maxlinear.com/web/documents/sipex/datasheets/spx1117.pdf.

[4] Texas Instruments, “Quad/Octal, Simultaneous Sampling, 16-Bit Analog-to-Digital Converters,” ADS1174/ADS1178 Datasheet, Sept. 2008, https://www.ti.com/lit/ds/symlink/ads1178.pdf.

[5] Digi, “Digi XBee S2C 802.15.4 RF Modules,” XBee S2C Datasheet, https://www.digi.com/resources/library/data-sheets/ds_xbee-s2c-802-15-4.

[6] Atmel, “8-bit Microcontroller with 16/32K bytes of ISP Flash and USB Controller,” ATmega16U4/ATmega32U4 Datasheet, Apr. 2016, https://ww1.microchip.com/downloads/en/devicedoc/atmel-7766-8-bit-avr-atmega16u4-32u4_datasheet.pdf.

[7] Silicon Labs, “Zigbee Mesh Network Performance,” AN1138, https://www.silabs.com/documents/login/application-notes/an1138-zigbee-mesh-network-performance.pdf.

[8] Q.-Y. Zhou, J. Park, and V. Koltun, ‘Open3D: A Modern Library for 3D Data Processing’, arXiv:1801. 09847, 2018.

[9] “1kg 3kg 5kg 10kg 20kg 30kg 50kg 100kg 150kg Film Pressure Sensor Resistive Force Sensitive Plantar Flexible Robot Fsr Insole - Sensors - AliExpress,” aliexpress.com. https://www.aliexpress.us/item/2251832774382148.html.

[10] “Electrical Engineering,” grainger.illinois.edu. https://grainger.illinois.edu/academics/undergraduate/majors-and-minors/electrical-engineering.

[11] “Computer Engineering,” grainger.illinois.edu.

https://grainger.illinois.edu/academics/undergraduate/majors-and-minors/computer-engineering.

[12] "IEEE Code of Ethics," IEEE.org, 2023, https://www.ieee.org/about/corporate/governance/p7-8.html.

[13] Occupational Safety and Health Administration, "Preventing Fire and/or Explosion Injury from Small and Wearable Lithium Battery Powered Devices," June 2019, https://www.osha.gov/sites/default/files/publications/shib011819.pdf.

[14] Z. Chen *et al.*, "4-V flexible all-solid-state lithium polymer batteries," *Nano Energy*, vol. 64, p. 103986, Oct. 2019, doi: https://doi.org/10.1016/j.nanoen.2019.103986.