**February 8, 2024**

**Project No.15**

# SMART HELMET WITH LIGHT INDICATORS FOR BRAKES & TURNS

by

Jasmehar Kochhar

Sanjivani Sharma

William Salazar

``

TA: Nithin Shanthini Praveena Purushothaman

## Abstract

We are addressing the safety challenge faced by motorcyclists today by increasing their visibility to other vehicles. This is achieved by incorporating turn and brake signals onto a motorcycle helmet, which being required in many places by law, is both physically as well as legally positioned to serve better than existing turn lights. We outline the usage of components required for this product, the wireless communication between light sensors and microcontrollers, as well as ethics and safety guidelines we have abided by in the development of our product.

# Contents

# 1. Introduction

Motorcycle riders account for 14% of all traffic fatalities, despite the fact only 3% of all registered vehicles are motorcycles. "The number of motorcyclist fatalities in 2021 increased by 8 percent from 2020, from 5,506 to 5,932." [1] According to the National Highway Traffic Safety Administration (NHTSA) of the United States Department of Transportation, "More than other vehicle drivers, motorcyclists must remain visible at all times, and anticipate what might happen." We want to address this safety problem. Lane splitting is a common practice endorsed by American Motorcyclist Association, wherein a motorcycle's narrow width can allow it to pass between lanes of stopped or slow-moving cars on roadways where the lanes are wide enough to offer an adequate gap.

We believe to address all the above: visibility to other vehicles, aiding lane splitting and reducing fatality, it is essential to remove ambiguity about the motorcyclist's path and make turn signals and braking more visible.

## 1.1   Problem

The biggest challenge faced by motorcyclists on the road today is their visibility to other vehicles. Left turn crashes due to oncoming traffic being unable to assess their turn intention (turn signals are mostly situated on the back of motorcycles) are exceedingly common. In 2021, 42 percent (1,158) of motorcycles-vehicle collisions were due to the other vehicles turning left while the motorcycles were going straight, passing, or overtaking other vehicles [1]. Furthermore, many countries of the world are still using hand signals to communicate turn intention!

Turn signals are obviously critical for successful maneuvering of a motorcycle. An important and often used practice of motorcycle riders that requires the highest level of safety is the practice of lane splitting, which is legal in 17 US states. Lane splitting involves weaving in and out of traffic lanes to reduce the risk of cars running into the back of the motorcycle in stop-and-go traffic. More states are considering legalizing this practice, which, if carried out with proper training and safety, can help reduce crashes from the back. However, lack of turn signal/braking visibility can make this practice incredibly dangerous for the rider.

## 1.2 Solution

Our solution increases the visibility of motorcyclists through a simple, yet practical addition: by integrating turn and brake signals onto helmets using strong LED strips. 18 US states make it illegal to ride motorcycles without helmets, and many others require young riders to do so as well, making our solution enabled by law. The LED lights are visible from the rear, the sides, and the front of the helmet, vastly improving a motorcycle rider's visibility to surrounding parties. The helmet will is able to communicate wirelessly with the turn and brake signals of the motorcycle vehicle in real time, and a combination of light sensors, microcontrollers with Bluetooth modules and LED lights are used to help us achieve this real time communication.

## 1.3 Visual Aid



**Figure 1: LED placement on helmet.**
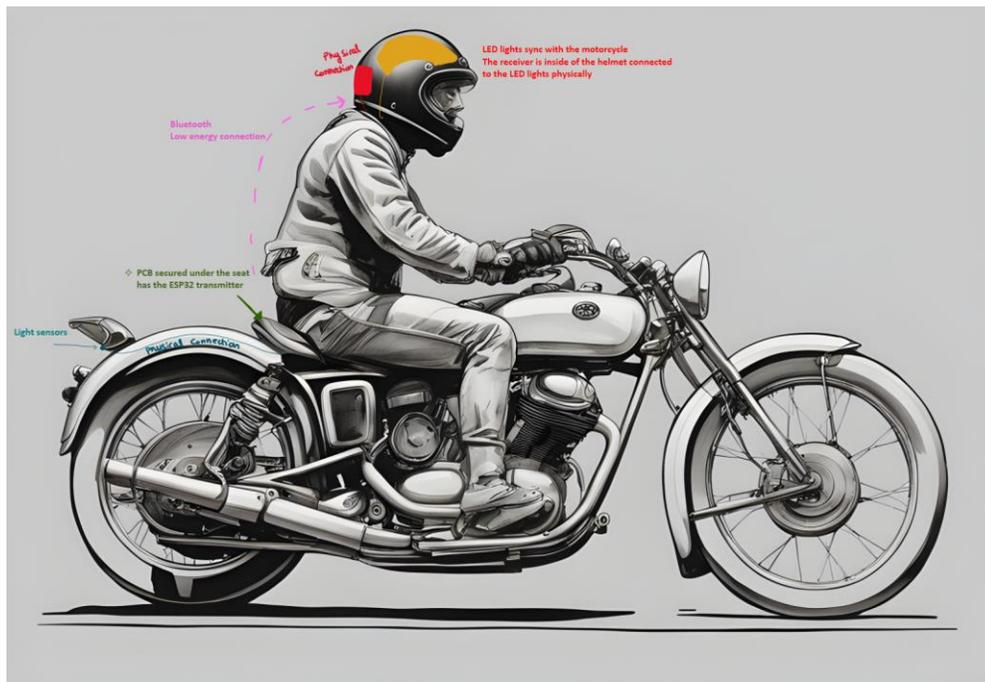


**Figure 2: LED and circuitry placement relative to motorcycle.**

## 1.3   High-level Requirements

The following high-level requirements below highlight successful functionality of our product:

1. When the motorcycle's right turn signal illuminates and blinks, the helmet's right LED should illuminate and blink. The same relationship should apply to the left LED.
2. When the motorcycle applies its brakes and its brake lights illuminate, the helmet's brake light should illuminate. When the brakes are released, the LED should turn off.
3. The latency for the helmet LED lighting up should not be above 0.5 seconds to communicate in real time precisely the moment a light is illuminated on the motorcycle.
4. Battery should indicate when the it is below 20%.

# 2 Design

## 2.1 Block Diagram
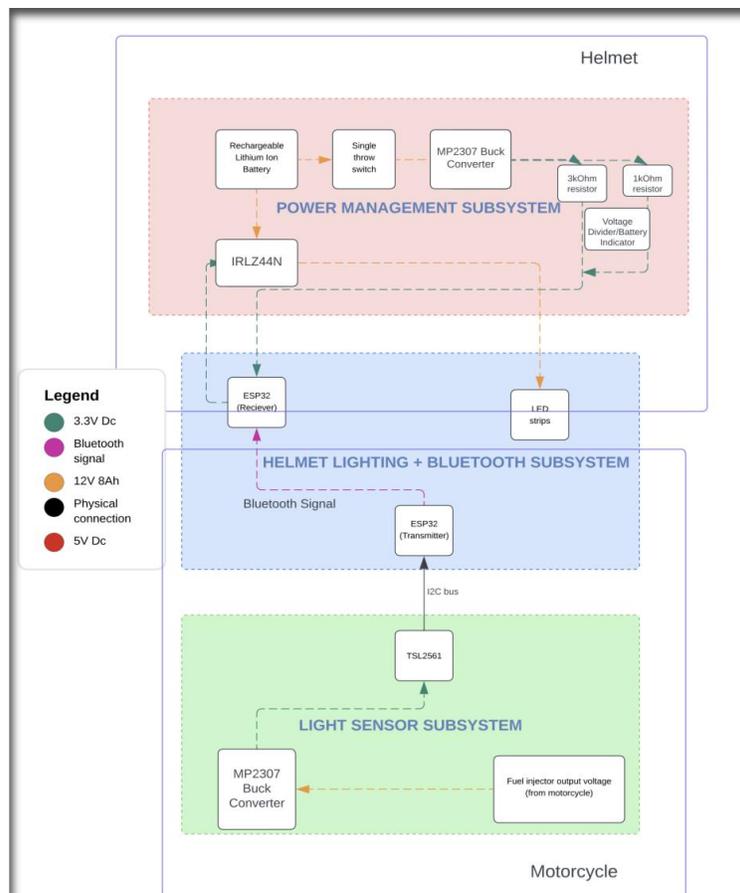


**Figure 3: Block Diagram for the lighting system**

The 3 main subsystems that make up our system consist of the Power Management Subsystem, the Helmet Lighting & Bluetooth Subsystem, and the Light Sensor Subsystem. The Power Management Subsystem is responsible for providing and regulating power to the LEDs and microcontrollers. The Light Sensor Subsystem is responsible for sending signals to the helmet of when the motorcycle's turn signals & brake light illuminate and the Helmet Lighting and Bluetooth Subsystem is responsible for communicating with the motorcycle to illuminate the LEDs on the helmet.

## 2.2 Subsystem Overview

### 2.2.1 SUBSYSTEM 1: LIGHT SENSOR SUBSYSTEM
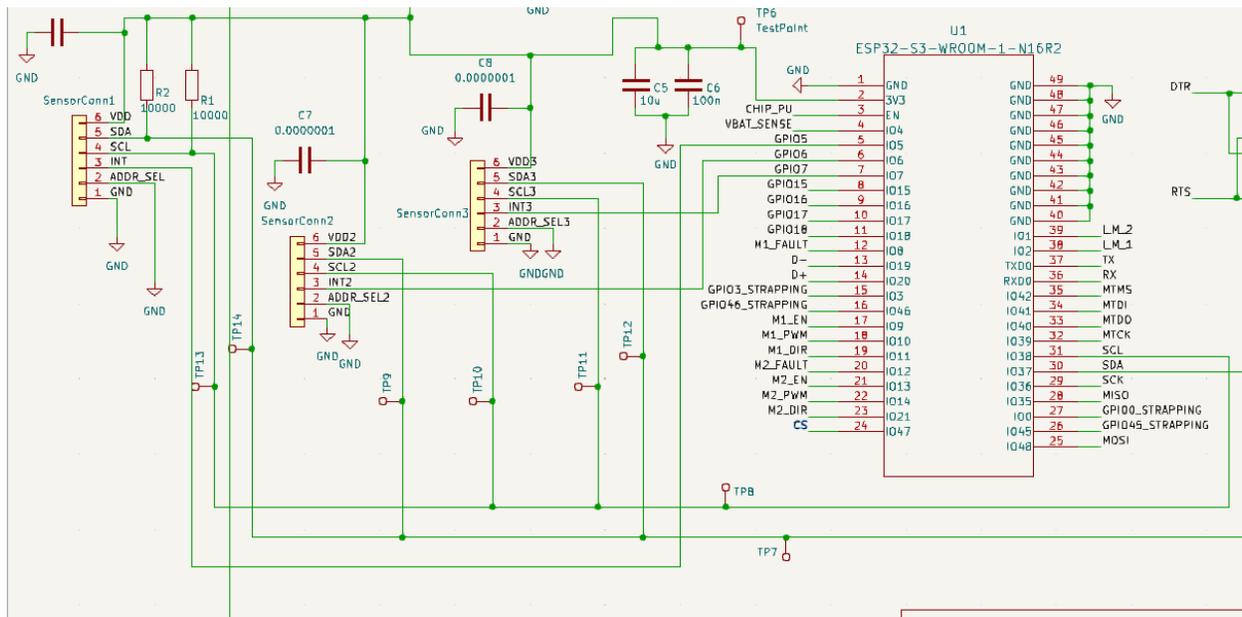


**Figure 4: Light Sensor Subsystem – Motorcycle Schematic**

The TSL2561 sensor communicates via I2C (multi-master, multi-slave) bus with the ESP32, and allows us to read the light intensity data from the turn and brake signals. This is affixed to our PCB in the motorcycle itself (can be accommodated under the seat discreetly) and receives 12 volts of power from the fuel injector on the motorcycle. This subsystem communicates with the Bluetooth subsystem to communicate sensor data as described in the next section.

## 2.2.2 SUBSYSTEM 2: BLUETOOTH SUBSYSTEM - HELMET & MOTORCYCLE COMMUNICATION
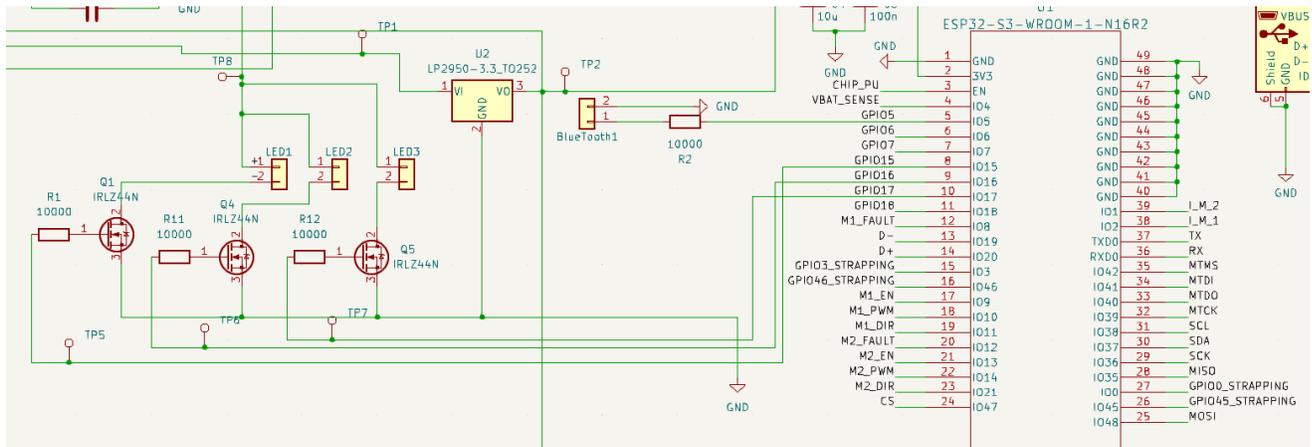


**Figure 5: Bluetooth Subsystem – Receiver – Helmet Schematic**
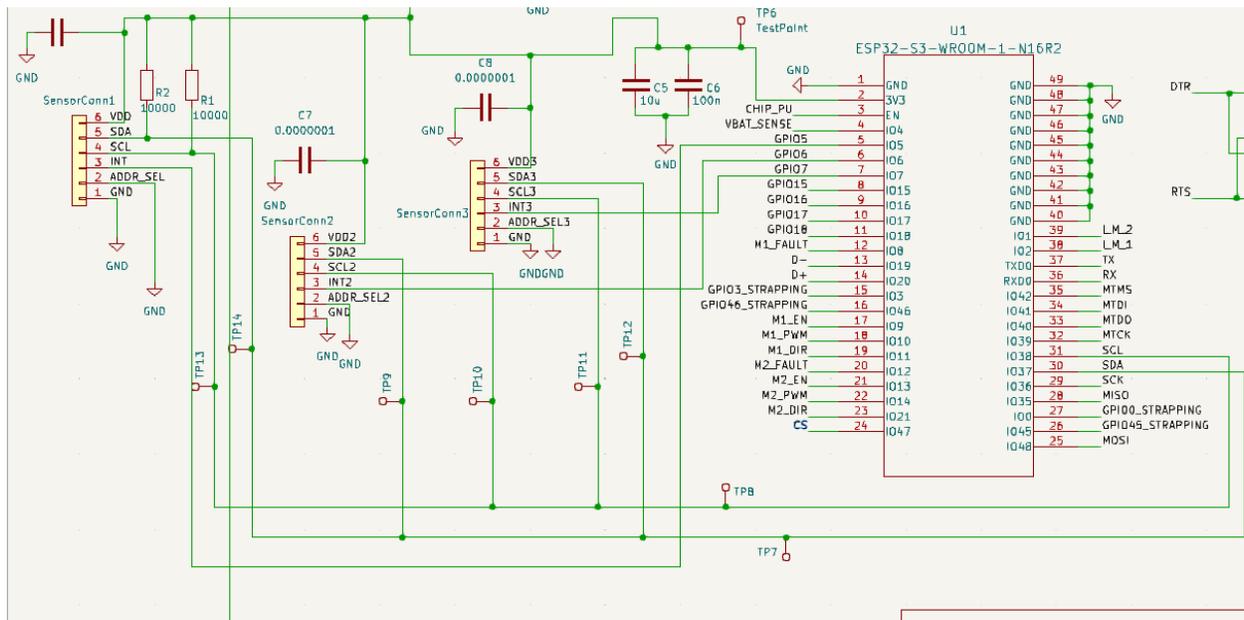


**Figure 6: Bluetooth Subsystem – Transmitter – Motorcycle Schematic**

We are using the ESP32 for its Bluetooth communication capabilities, which eliminates the need for an additional Bluetooth module. We use BLE (Bluetooth Low Energy) to keep our power usage efficient. It is used both as a transmitter (motorcycle) and a receiver (helmet). One is affixed to our motorcycle PCB, and the other is fixed to the helmet PCB to transmit light sensor data from the motorcycle to the helmet. On our chip, we have assigned our light sensors to use GPIO pins 5, 6, and 7 on our motorcycle PCB. On the helmet PCB, we have assigned GPIO pins

15, 16, and 17 to the corresponding LEDs which appear in the helmet lighting subsystem. We have assigned GPIO pin 5 on the helmet PCB to be used for the Bluetooth indicator LED, which communicates Bluetooth connection status.

## 2.2.3 SUBSYSTEM 3: HELMET LIGHTING SUBSYSTEM
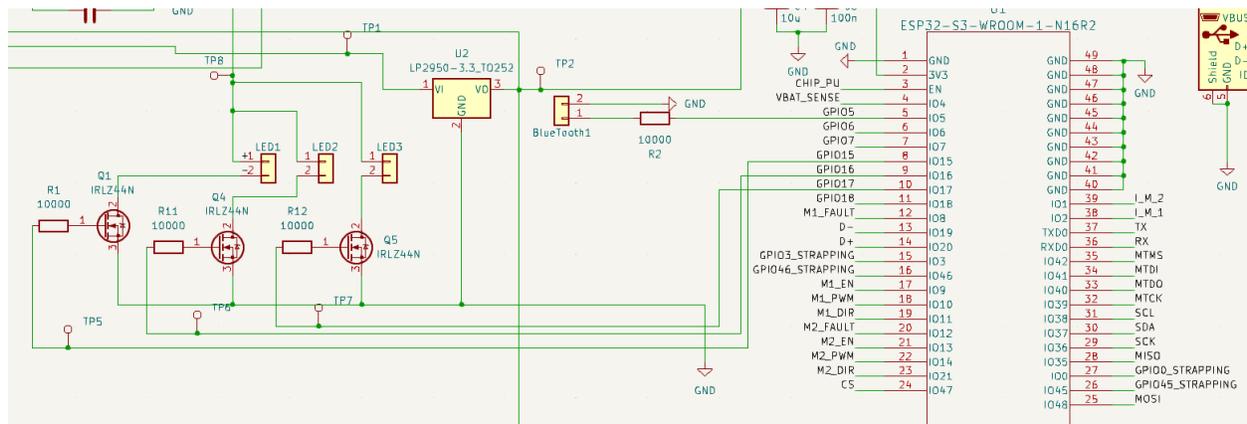


Figure 7: Helmet Lighting – Helmet Schematic

As a part of the Bluetooth subsystem, this subsystem receives the light sensor data that is used on the helmet. The Helmet LEDs are connected to the ESP32 in the helmet which act as a receiver from the motorcycle PCB. The turn signal LEDs are on the upper side of the helmet so that they don't obstruct the peripheral view of the rider. Most road accidents relating to lights on the motorcycle are due to left turns, so we made sure that the LED would be visible from the front as well. The brake light on the other hand only needs to be visible from the back.

- LEDs: Red and amber LED strips are affixed to the helmet, compliant with Illinois law. To avoid compromising the structural integrity of the helmet, we used strong, weather resistant glue to attach the LEDs to the helmet.

## 2.2.4 SUBSYSTEM 4: POWER MANAGEMENT SUBSYSTEM
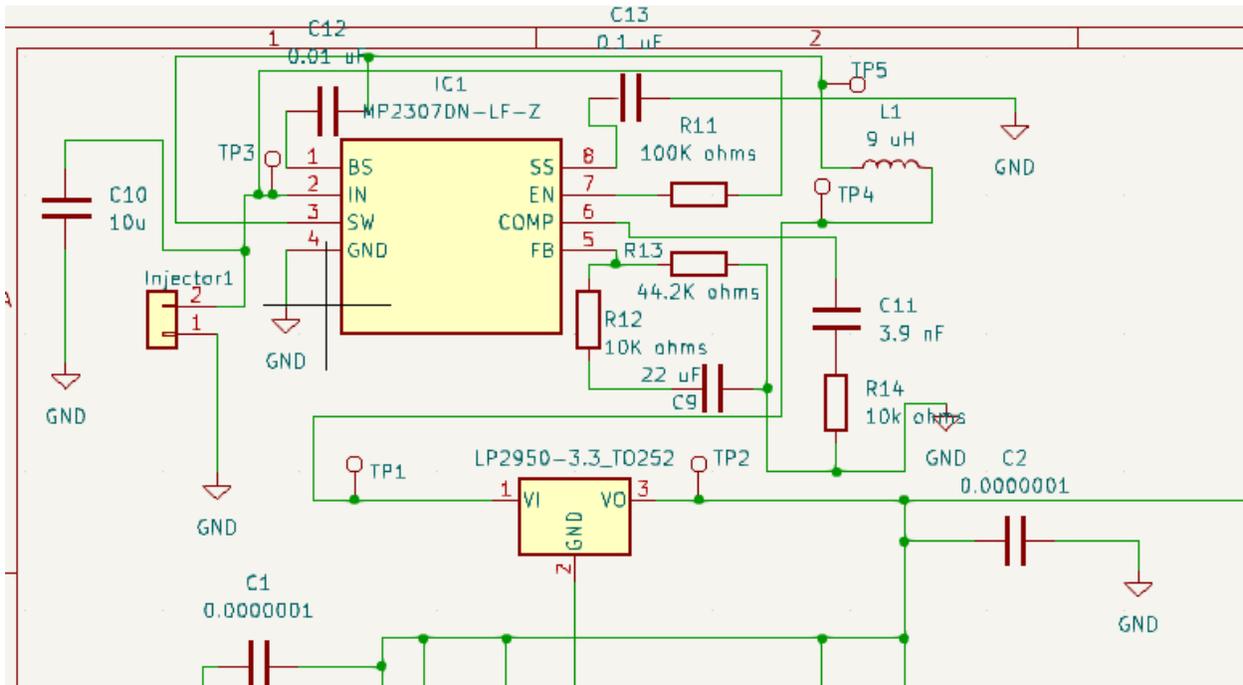
**Figure 8: Power Management – Step-down circuitry**
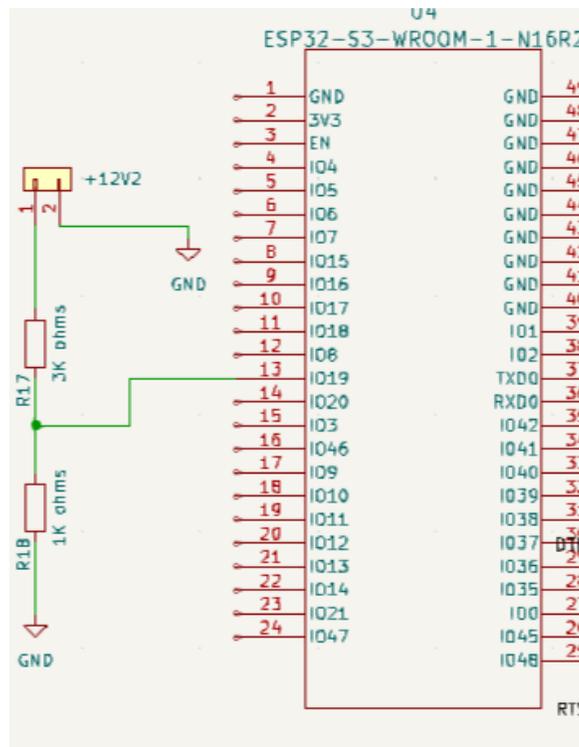
**Figure 9: Power Management – Voltage Divider**

For components connected to the motorcycle, they receive voltage from the motorcycle's fuel injector output, which only supplies power when the motorcycle is on. Thus, the system does not drain power when the motorcycle is not in use. A rechargeable battery is present inside the helmet to power up the ESP and the LEDs.

Due to the possibility of the battery heating up and to maintain the safety of the helmet, the battery pack is in cased in flame retardant material.

## 2.3 Subsystem Requirements

### 2.3.1 SUBSYSTEM 1: LIGHT SENSOR SUBSYSTEM

The light sensors must initially output lux values of the light they are exposed to. They are then calibrated according to the lux values of the motorcycle indicator lights to respond only to the lights in those ranges, and not below (too dim) or above (too bright, for example sunlight during daytime). The light sensors are attached close to the motorcycle lights to make sure they are the primary input source for lux data. Once we fix the functioning range of the sensors, we test them in varying light conditions.
The light sensor must also communicate with the ESP32 through the SDA and SCL lines. This communication is tested using a serial monitor. If the light reading is 0, we know that the sensor is not reading.

Refer Appendix C Table 1

For results, refer Appendix B

### 2.3.2 SUBSYSTEM 2: BLUETOOTH SUBSYSTEM - HELMET & MOTORCYCLE COMMUNICATION

Comprehensive testing scenarios should include pairing, connection initiation, reconnection after disconnection, and handling of potential connection errors or interruptions. Signal strength and range testing must be conducted to guarantee reliable communication under various environmental conditions, such as interference from other devices or obstacles between the motorcycle and the helmet.

Data integrity checks and error correction mechanisms should be implemented to detect and mitigate transmission errors that may occur due to noise or signal degradation.

 Robust error handling mechanisms should be in place to handle situations where the light sensor data transmission fails or encounters errors, ensuring that the system can recover gracefully and maintain functionality.

Refer Appendix C Table 2

For results, refer Appendix B

### 2.3.3 SUBSYSTEM 3: HELMET LIGHTING SUBSYSTEM

The positioning of the turn signal LEDs on the upper side of the helmet, away from the rider's direct line of sight, prevents distraction and maintains the rider's focus on the road ahead. Similarly, ensuring that the brake light LED is visible from the back alerts following vehicles of the rider's intention to slow down or stop, reducing the risk of rear-end collisions.

The secure fixation of LEDs to the helmet is crucial to prevent detachment during normal riding conditions, which could pose a safety hazard to the rider and other road users. The use of strong adhesive strips provides a reliable method for affixing the LEDs to the helmet without compromising its structural integrity or compromising safety standards. We will perform physical tests to assess the durability and stability of the attachment mechanism under various environmental conditions, including exposure to vibrations, wind forces, and temperature fluctuations.
Refer Appendix C Table 3

For results, refer Appendix B

### 2.3.4 SUBSYSTEM 4: POWER MANAGEMENT SUBSYSTEM

The step-down circuit consisting of the buck converter (MP2307) and voltage regulator (LP2950) must bring voltage down from the input of 12V from the motorcycle injector to 3.3V required for the ESP32's functioning. These are tested using a multimeter on a breadboard, and then on a PCB.

The voltage divider circuit must split voltage up in a ratio of 3:1 so that the user is alerted when power supply falls from 12V to 8V or below. This is verified by using a multimeter to measure the voltage values after choosing appropriate resistances, which were 3k Ohms and 1k Ohms for our circuit. In addition, we verified this by using the ADC on the ESP32 which reads the input voltage. At 8V the Bluetooth light started blinking.

Refer Appendix D Table 4

For results, refer Appendix B

## 2.4 Design Alternatives

Since our initial design document, we have ended up making a lot of changes to the block diagram and our systems to be able to get our design to work properly. The final design is what we used for our final round PCB order to be able to achieve our high-level requirements. The most major change was the voltage divider circuit that was used for measuring the current battery level via the ESP32.
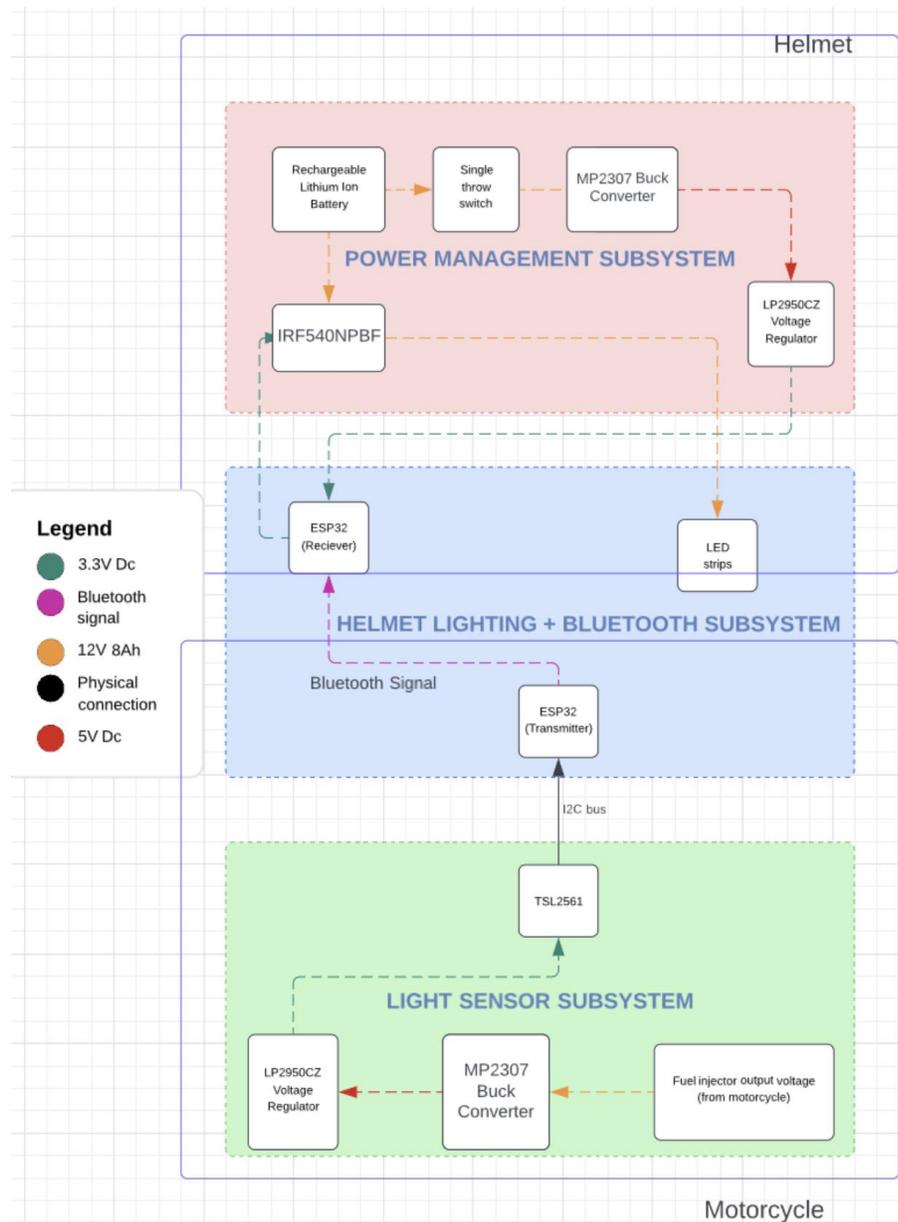


**Figure 3: Block Diagram of Old System**
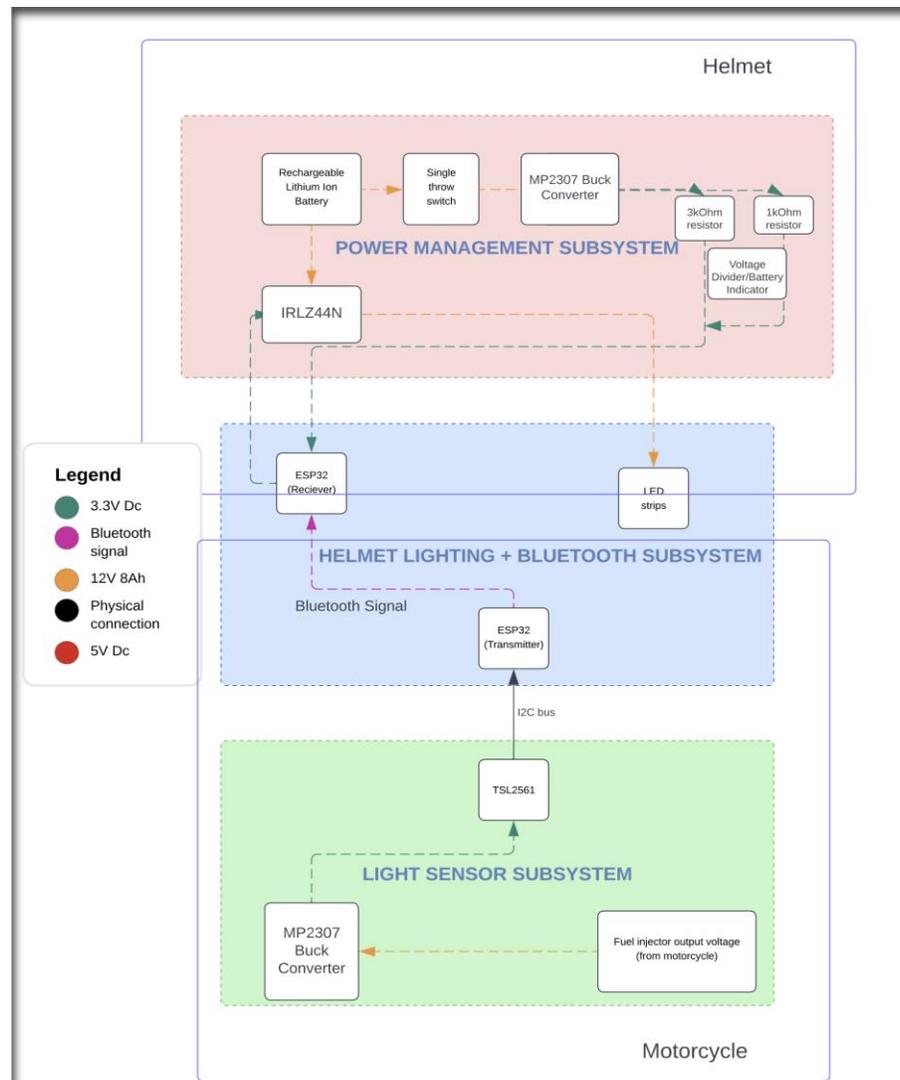
**Figure 4: Block Diagram of New System**

From testing our final iteration of PCB, we have quite a few changes that we would make to mitigate the challenges that we faced during assembling the final design. The following are the challenges that we faced and the alternatives that we would implement to come up with a better and more functional product.

1. The ESP32 antenna was blocked by copper as seen in our final PCB design. Copper absorbs radio waves coming out of the esp32, hence sending complex values through the esp32 led to corrupted values on the other side. To solve this problem, we reduced the load and which allowed the microcontroller to notify the other esp32 that the lux values have been changed. Something we would change here is to remove the copper behind the antenna area and
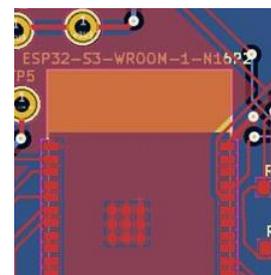


**Figure 5: Antenna with copper layer**

move it away from the inductor in the power system area which could also interfere with the signals.

2.  Our USB to UART bridge inverted the RX to TX connections to the ESP32 which made communication with the ESP32 impossible, evidenced by no boot message when powered. To fix this temporarily we used an external USB to UART bridge and connected it to the esp32 with wires and the correct lines. In the future just switching the lines would allow boot messages to show up and to communicate directly using a micro-USB.

3.  When calculating buck converter output voltage values, we did not consider the draw of current from the ESP32 or other connected components, the current changes the output value. As seen from appendix A part 7 with the buck converter calculations, the draw of the current is an important factor in determining how much voltage the buck converter would output. Unfortunately, we did not correctly access the output current it would draw and so our buck converter was not outputting the correct values. This is just something that we would need to keep testing to be able to find the correct value or we can put in a potentiometer to be able to change the output voltage during testing.

4.  We required 3.4 volts due to the extra components, hence the esp32 undervolted when the voltage regulator was connected. When using 3.3 volts the esp32 was outputting the error brown boot which means that it is undervolted. This would be because the 3.3 volts were also powering certain sensors and LEDs which would leave less than 3.3 for esp32 to boot up. To change this, we would use a linear voltage regulator with a range of 3.4 to 3.5 volts.

5.  Very low resistance for voltage sensing wasted a lot of power. Since it is a voltage divider circuit a certain amount of current is directly being drained which would lower the number of hours the helmet could be used, thus to avoid power wastage we would attach larger resistances in the power circuit.

# 3. Cost & Schedule

## 3.1 Cost Analysis

### 3.1.1 Labor

$87,267 Average Salary = $41.96/Hour

$41.96/Hour x 2.5 x 45 Hours = $4720.5

$4720.5 * 3 Engineers = $14,161.5

### 3.1.2 Parts

Refer Appendix D Table 6

### 3.1.3 Sum

Part Total Cost = $101.28

Labor Cost = $14,161.5

Total Sum = $101.28 + $14,161.5 = $14,262.78

## 3.2 Schedule

Refer Appendix D Table 5

# 4. Ethics and Safety

In developing the smart motorcycle helmet, we recognize that it is important to address ethical & safety considerations during the development and the lifecycle of the product, on and off the road. Throughout our time as students and soon as professional engineers, we will adhere to the ethical principles outlined in both the IEEE and ACM Codes of Ethics. These codes will be the basis of principles in our journey and will guide us through ethical decisions we face. By addressing the following, we are committed to designing and developing a motorcycle helmet that improves rider safety and upholds ethical principles.

## 4.1 Ethics

### 4.1.1 User Safety

Our biggest concern is the safety of the helmet user. Our commitment is to design a helmet that creates a safer environment for every motorcycle rider that uses it, without compromising ethical principles. As stated in the IEEE code, we must prioritize public safety in professional activities. Additionally, we also value transparency and will make certain that users are fully informed of all capabilities of the helmet.

### 4.1.2 Accessibility

Another focus of our agenda focuses on designing an inclusive project that can be used by all motorcycle riders, regardless of technological understanding or physical capabilities. As seen in the IEEE code, there is immense value in promoting diversity and accessibility in all engineering projects.

### 4.1.3 Professionalism

As a group, we will work to uphold the highest level of professionalism throughout our product's development. We will accurately and honestly communicate our product's capabilities, in line with the principles of integrity in professional practice outlined in the IEEE and ACM codes.

## 4.2 Safety

### 4.2.1 Regulation, Durability, & Integration

We will work to make sure the helmet complies with federal safety regulations as outlined by the United States Department of Transportation and with industry standards related to safety. This will involve materials testing and padding implementations to ensure the helmet provides proper protection in case of a crash. We will also adequately integrate the helmet with existing subsystems of the motorcycle to ensure compatibility and reliability of the product. This will culminate in real-world test rides performed by a licensed driver to verify performance.

### 4.2.2 Product Instruction

We will ensure comprehensive product training to users to ensure proper use of the helmet to mitigate safety concerns related to user error.

# 5. Conclusion

ECE 445 was an extremely valuable course and allowed us to gain real hands-on experience in product development. As we move forward towards our career as engineers, we are now equipped to bring the many lessons we learned in the course with us on our journeys towards success.

## 5.1 Accomplishments

In the result of our project, we ended up satisfying all of the defined high-level requirements and the subsystem requirements. Some of our testing is shown as part of appendix B, however since a lot of our testing and verification is not qualitative, we do not have a lot of tabular data. On our breadboard we were able to have our sensors recognize lux values and have a sufficiently high gain which allows for a differentiation between on and off states for the lights.

Furthermore, the esp32 can send data reliably even between walls and for around 20 meters of distance and can reconnect. We can quickly and efficiently send signals without any delay. Finally, we have an indicator that warns the user about the discharging battery.

## 5.2 Uncertainties

We could not demonstrate fully independent functionality due to 2 wires touching and drawing a large amount of current through the circuit which burnt the esp32 and the sensors that were connected to the PCB. To mitigate as much damage as possible when large current draws happen due to accidental short circuits, connect fuses onto the PCB to prevent larger damage. For extra protection the wires should be taped up with an electrical table so there are no exposed leads. Due to the limited sensors, ultimately decided to simulate it and we do know that it currently works with user input, and we were able to satisfy our higher-level requirements and our subsystem requirements.

## 5.4 Future Work

To improve our product, we would like to center our focus on the packing, design, and the overall aesthetic of the helmet. This would involve finding thinner LED lights that have a sleeker fit to the surface of the helmet and creating a smaller PCB & case to fit on the helmet in a less protruding way.

# Appendix A: Equations and Calculations

1. For the light sensor TSL2561, the two main configurations that can be sources of errors are the style of interrupts chosen and the lux value outputted by the sensor.

   1.1 Interrupt styles are determined by the INTR field in the Interrupt Register and the primary purpose is to detect a "meaningful" change in intensity. This can be defined both in terms of light intensity and time, or persistence, of that change in intensity. We can define a threshold above and below current light level or specify a number of conversion cycles for which a light intensity exceeding either interrupt threshold must persist before actually generating an interrupt. This can be used to prevent transient changes in light intensity from generating an unwanted interrupt. This value can range from 1 (interrupt occurs immediately whenever either threshold is exceeded) to 15 (15 consecutive conversions must result in values outside the interrupt window for an interrupt to be generated).

   For e.g., if N is equal to 10 and the integration time is 402 ms, then an interrupt will not be generated unless the light level persists for more than 4 seconds outside the threshold. Using functions defined in a TSL2561 driver procured online [15], we are currently implementing a polling mechanism to detect light intensity values. We read the irradiance values, and we use predefined thresholds to restrict functionality of these sensors depending on external light conditions.

   1.2 As mentioned in subsystem requirements, the integration time and gain settings directly impact the accuracy of light intensity reading. These are fixed using the TIMING register.
   INTEG FIELD VALUE "00": Integration time is 13.7 ms with a scale factor of 0.034.
   INTEG FIELD VALUE "01": Integration time is 101 ms with a scale factor of 0.252.
   INTEG FIELD VALUE "10": Integration time is 402 ms with a scale factor of 1.
   INTEG FIELD VALUE "11": Not applicable (N/A), as this setting is used for manual timing control and stops the integration cycle when writing a 0.
   These settings allow for adjustment of the sensor's sensitivity and response time to changes in light intensity. The scale factor adjusts the sensitivity of the sensor to light, with a higher scale factor (closer to 1) providing greater sensitivity.

   The gain setting (low gain at 1× or high gain at 16×) directly influences the sensor's light sensitivity. High gain settings can allow for better detection in low-light conditions but may lead to saturation under bright conditions. The chosen integration

time and gain setting must be matched to the expected light levels for accurate light intensity measurements.

Motorcycle lights are generally between 35 and 90 lux [3], and for this range we will test with low gain (1x) first, especially in a moderately lit environment. If the sensor fails to differentiate light levels adequately at this setting, we will switch to high gain (16x).

2. In microcontrollers, floating point operations are often not supported, or have poor performance, so the lux calculation must be done without floating point operations. Since floating point has been removed, scaling must be performed prior to calculating illuminance if the integration time is not 402 ms and/or if the gain is not 16. This is explained in detail in the TSL2561 software integration portion of the datasheet.

3. **For the TSL2561 light sensor:**
   Supply Current: Active mode 0.24 to 0.6 mA, Power-down mode 3.2 to 15 μA.
   Voltage: Output low voltage at 3 mA sink current is 0 to 0.4 V and at 6 mA sink current is 0 to 0.6 V.
   Leakage Current: -5 to 5 μA.
   **For the ESP32 microcontroller:**
   Input Voltage (VIH/VIL): High-level input voltage is 0.75 times the supply voltage to the supply voltage plus 0.3 V. Low-level input voltage is -0.3 V to 0.25 times the supply voltage. [4]
   Output Voltage (VOH/VOL): High-level output voltage is 0.8 times the supply voltage. Low-level output voltage is up to 0.1 times the supply voltage. [4]
   Sink Current (IOL): Low-level sink current is up to 28 mA for output drive strength set to the maximum. [4]
4. The ESP32's VIH and VIL levels must match the TSL2561's logic levels for proper communication. Since TSL2561 operates with logic levels based on its supply voltage (2.7V to 3.6V), we will ensure that the ESP32's GPIO pins, when configured for I2C communication work with these levels.
5. Both the ESP32 and TSL2561 have specified operating temperature ranges. The devices' performance, including the accuracy of the TSL2561's (operates between 243 K to 343 K) light measurements and the ESP32's (operates between 233 K to 398 K) [4] processing capability, may vary with temperature.
6. **Heat tolerance analysis:** The main sources of heat dissipation in our project are:

   **ESP32**: The products sealed in moisture barrier bags (MBB) should be stored in a non-condensing atmospheric environment of < 40 °C and 90%RH. The module is rated at the moisture sensitivity level (MSL) of 3.

After unpacking, the module must be soldered within 168 hours with the factory conditions $25 \pm 5$ °C and 60%RH. If the above conditions are not met, the module needs to be baked.

**Voltage Regulators**: We will calculate the maximum power dissipation and thermal resistance as such:

$$Pdmax = (Vin - Vout)* Iout + Vin * Iq$$

Where P*dmax*: Maximum Power Dissipation, V*in*: Input Voltage, Vout: Output Voltage I*out*: Output Current, I*q*: Quiescent Current

We will calculate the thermal resistance (Rt) from the junction to ambient to ensure the junction temperature (Tj) does not exceed the maximum rated junction temperature (Tjmax) using:

$$Tj = Ta + Pd * Rt, \text{ where Ta is ambient temperature}$$

LP2950 has a thermal resistance junction-to-ambient (Rt) of 180°C/W for the SOIC-8 package, and the maximum junction temperature (TJ) is +150°C.[11]

**Buck Converter**: We are using an MP2307 buck converter to step down our voltage from 12V to 5V. The operating input voltage should be between 4.75V to 23V, and the operating output voltage is between 0.925V to 20V. The device works in an operating temperature range of –40°C to +85°C

The switching frequency of the buck converter determines the performance of the converter. On a higher switching frequency, the capacitor and inductor needed are smaller.

The main power loss sources are switching loss, conduction loss and driver loss. [15]: Switching Loss: Switching losses occur due to transition of the switch from an ON state to OFF state and vice versa.

In a single cycle, power loss is:

$$E = 2 \times \int_0^{tcross} \frac{V(t)}{(t)} dt = Vdsmax * Idmax * tcross$$

Where Vdsmax is voltage across switch (when it is OFF)

Idmax is current through it when it is ON

Tcross is crossover time during turn on and turn off respectively

Power loss proportional to switching frequency is :

$$P = Vin \times Idmax \times tcross \times fsw$$

Where Vin is the input voltage and

fsw is switching frequency

Conduction Loss: Conduction losses occur in the power processing interval in the converter. These losses depend not on its frequency, but on its duty cycle:

$P = Irms^2 \times Rds$

Where Irms is the RMS switching current

And Rds is the on-resistance of the mosfet

Driver Loss: Driver loss occurs when the interelectrode capacitances charge and discharge. The controlling equation is:

$P = Vdrive \times Qg \times fsw$

Where Vdrive is the gate drive voltage

And Qg is the gate charge factor (proportional to the effective input capacitance and gate drive voltage)

**Batteries:** We are using the Turnigy Graphene Panther 4S 1300mAh 75C LiPo Battery. We will monitor the battery temperature to prevent heating over around 60°C (140°F) which is the maximum recommended temperature for LiPo batteries [14].

7. **Buck Converter Calculations**:
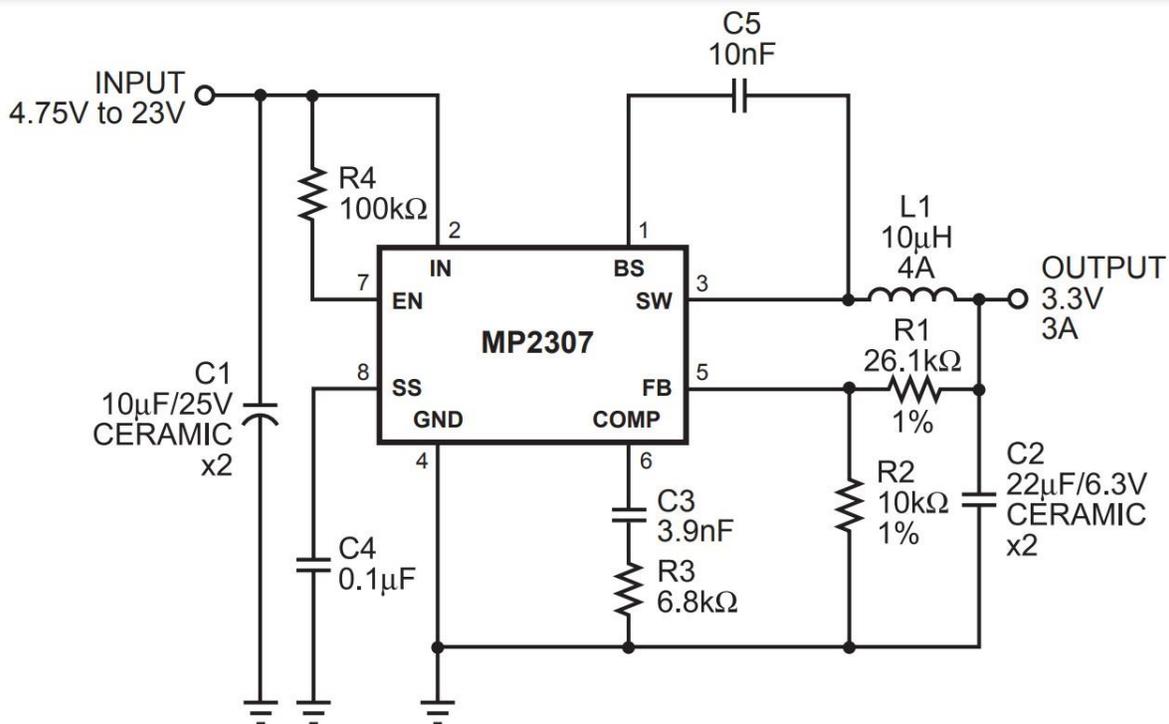


**Figure 6: Example Buck Converter Circuit with 3.3 Vout**

Component selection for the buck converter is conducted as follows:

R1 and R2: to set the output voltage, chosen as 44.2kOhms and 10kOhms respectively according to datasheet [16]

L1: inductor chosen as 10uH using the inductor selector tool [17] and verifying using datasheet equations. We are using a 340 kHz inductor, with Vmax as 12V and Vmin as 8V. Our desired output voltage is 5V. Iout value was chosen according to the ESP32.
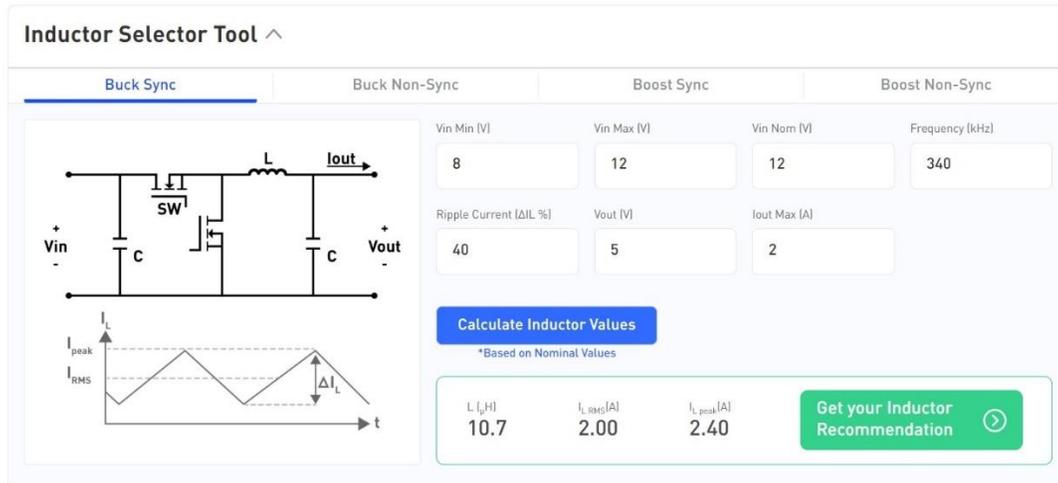


**Figure 7: Inductor Value Calculation**

C1: Input Capacitor chosen as 10uF. The datasheet says: "When using ceramic. capacitors, make sure that they have enough capacitance to provide sufficient charge to prevent excessive voltage ripple at input." This was the same value as the example circuit.

C2: Output Capacitor chosen as 22uF, also the same value as the example circuit.

C3 and R3: Compensation components chosen as 3.9 nF and 10kOhms.

# Appendix B: Simulations and Results

Following are images for verification of working sensors and esp32 results. Since most of our verification is not based on hard values instead are based on qualitative data such as real-world testing.

```
Break light is on!
luminosity_Break From Break Light: 65536 lux
Right light is on!
luminosity_Right From Right Light: 65536 lux
Left light is on!
luminosity_Left From Left Light: 65536 lux
Break light is on!
luminosity_Break From Break Light: 65536 lux
Right light is on!
luminosity_Right From Right Light: 65536 lux
Left light is on!
luminosity_Left From Left Light: 65536 lux
```

**Figure 8: values showing lux values from testing**

This is the data from the lights being on and where we had set out threshold (65000) for bright days. For indoors the values went as low as 500 lux so for indoors our threshold was around 800 lux only. The gain values are set to automatic so the threshold can be changed through a mapped value looking at the average value of the lux that is being picked up by the sensors.

Since ultimately, we did not have our sensors, we ended up using user input on the serial monitor using the code below, this is showcased in our demo video, but we were not able have the code up in time for the demo since it was short notice.

```
while (Serial0.available() == 0) {
}
int sensor_light = Serial0.parseInt();
Serial0.println(sensor_light);

  if(sensor_light == 4){
  BreakLightCharacteristics.notify();
  }
  if(sensor_light == 2){
  RightLightCharacteristics.notify();
  }
  if(sensor_light == 3){
  LeftLightCharacteristics.notify();
  }
```

**Figure 9: code for user input to notify**

As for the power system, we stepped down from 12 volts to $5 \pm 0.1$ volts using our buck converter with the exact capacitors, resistors and inductors given in the calculations. From there we stepped it down to 3.3 volts for the exact voltage required for the ESP32.

Figure 10: 12 volts from the input          Figure 11: 5 volts to the output
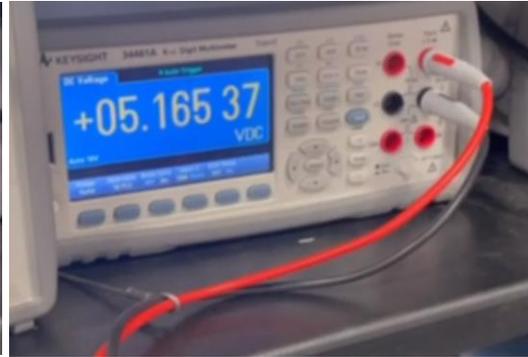


Figure 12: 3.3 volts from the linear voltage regulator.

As for our Bluetooth subsystem, since we had the indicator light, we could be sure of when the Bluetooth was connected however initially when testing we could connect to the serial monitor to be able to tell us about the connection and when it got disconnected.
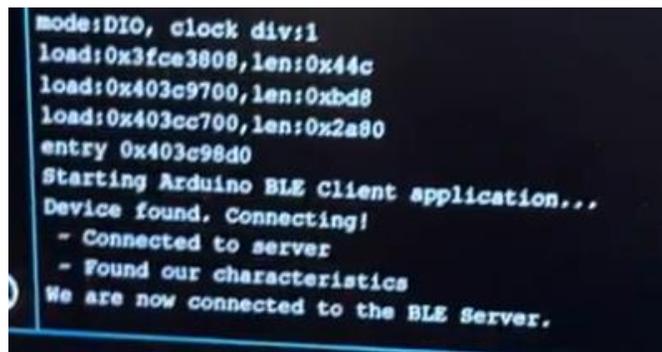


Figure 13: client-side connection on serial monitor.

Finally to make sure that the motorcycle server and Hemet client only connect to each other we have specific UUIDs for the server for the client to seek out.

```
#define bleServerName "Motorcycle_server"
static BLEUUID MotorcycleServiceUUID("7107eb72-77e7-411b-8dcb-af365300a59c");
```

Figure 14: client-side identifier

```
#define bleServerName "Motorcycle_server"
#define Motorcycle_UUID "7107eb72-77e7-411b-8dcb-af365300a59c"
bool deviceConnected = false;
```

Figure 15: server-side identifier

# Appendix C: Requirements and Verifications

| Requirements | Verification |
|---|---|
| **1.** Hardware set-up requirements [2]:<br>**1.1** The power supply lines must be decoupled with a 0.1 µF capacitor placed as close to the device package as possible. This bypass capacitor should have low effective series resistance (ESR) and low effective series inductance (ESI), such as the common ceramic types, which provide a low impedance path to ground at high frequencies to handle transient currents caused by internal logic switching.<br>**1.2** Pull-up resistors are required for the SDAH and SCLH lines at a high level, and a separate pull up resistor between 10 kOhm and 100kOhm is required for the interrupt (INT) line.<br><br> | W will ensure that a 0.1 µF capacitor is placed as close as possible to the power supply pins of the TSL2561 to minimize noise and stabilize the power supply.<br><br>We will physically inspect the PCB to confirm correct placement of the decoupling capacitor and pull-up resistors.<br><br>We will use a multimeter to verify the correct resistor values and the continuity of the connections and ensure there are no short circuits between the power supply lines and ground.<br><br>An oscilloscope can be used to check the integrity of the signals on the SDA, SCL, and INT lines (if there are sharp transitions without excessive ringing, indicating good decoupling and correct pull-up resistor values) |
| **2.** Sensitivity and Performance requirements:<br><br>The sensor should be able to differentiate between ambient light conditions and the motorcycle's turn and brake lights. For this, the integration times and gain settings need to be correctly configured to optimize response for light intensity changes. | After setting up the sensor, we will perform tests under controlled light conditions to calibrate the sensor. We will adjust the integration times and gain settings until we can reliably detect the difference between ambient light and the lights of the motorcycle. By analyzing the ratio of visible to infrared light (using channel 0 and channel 1 readings), we will program the ESP32 to recognize light patterns emitted by the motorcycle's indicators. The tests will look as follows. We can see that the light reading value changes from values like 324 -> 26 -> |

| | |
|---|---|
| | 1623 from ambient light conditions, to dim lighting, to bright light. |
| |  |
| | **Figure 10: Example values for lux values while polling** |
| **3.** Communication with ESP32:<br><br>The light sensor should be detected by the ESP32 and should periodically obtain a light reading from it, displaying it on the console. One TSL2561 device will be connected to two GPIOs on the ESP32 (I2C SDA and SCL).10 kOhm resistor from each GPIO to the 3.3V supply are connected to act as pull-up resistors. | The light sensor is directly connected to the ESP32 via GPIO pins. We will set upper and lower thresholds based on the expected light intensity values when the motorcycle indicators are on. These values are programmed into the TSL2561's interrupt threshold registers to trigger an interrupt when it detects light intensity from the indicators. |

**Table 1: Requirements and Verifications – Light Sensor Subsystem**

| Requirements | Verification |
|---|---|
| **1.** The ESP32 must establish a Bluetooth Low Energy connection between the helmet and the motorcycle. | We will use a blinking led indicator to signal whether Bluetooth connection was successful or not, as in standard Bluetooth devices. |
| **2.** The Bluetooth subsystem must reliably transfer light sensor data between the helmet and motorcycle. | This will be verified by a polling mechanism which polls at a fixed rate to receive light sensor information in varying brightness conditions. |
| **3.** The Bluetooth system should be able to remain connected during travel | We will send data messages between the devices to ensure an active connection. If a timeout occurs (no acknowledgement of the data message), it will attempt to reconnect. |

**Table 2: Requirements and Verifications – Bluetooth Subsystem**

| Requirements | Verification |
|---|---|
| **1.** Turn signal LEDs must be visible from the front & back and the brake light LED must be visible from the back | An observer will perform a visibility test of the helmet at different angles and in different lighting conditions of the LEDs. |
| **2.** LEDs must be securely fixed to the helmet | Perform an adhesion test by applying a pulling force by hand to the LEDs. Perform test in a dry condition, after applying water, and while applying heat. |
| **3.** The LED light must not restrict the motorcyclist vision | While wearing the helmet, assess the helmet's vision by identifying markers placed within view of the helmet at different angles. |
| **4.** Latency of motorcycle indicator turning on and helmet LED turning on should not exceed 3 seconds. | We will conduct tests to ensure that the latency remains below our fixed value using a timer when the motorcycle indicator lights are turned on. If the latency is above 3 seconds, we will revisit the polling mechanism. |

**Table 3: Requirements and Verifications – Helmet Lighting Subsystem**

| Requirements | Verification |
|---|---|
| **1.** Components directly connected with the motorcycle must receive power from the fuel injector voltage output only when the motorcycle is on | Using a multimeter, measure 12 volts output from the fuel injector when the motorcycle is on and 0 volts when the motorcycle is off. |
| **2.** The system should only use/drain power when the motorcycle is on | Using a multimeter, connect the probes in series with the output of the MP2307 buck convert and measure 5 volts output when the motorcycle is on. Additionally, measure the output of LP2950CZ voltage regulator and measure 3.3 volts output when the motorcycle is on. When the motorcycle is off, our measurements should read 0 volts for both components. |
| 3. The system should have a voltage divider to alert user when the battery supply has fallen from 12V to 8V. | This was verified by using the ADC on the ESP32 which read the input voltage. At 8V the Bluetooth light started blinking. |

**Table 4: Requirements and Verifications – Power Management Subsystem**

# Appendix D: Relevant Tables

| Week | Task | Will | Sanji | Jas |
|------|------|------|-------|-----|
| **2/19** | 1. Complete Design Document<br>2. Retrieve Motorcycle Helmet | 1 | 1, 2 | 1 |
| **2/26** | 1. Finalize Bill of Materials and Order Parts<br>2. Complete Draft of PCB for Review | 1, 2 | 2 | 1, 2 |
| **3/4** | 1. Finalize PCB Design and Submit Order<br>2. Determine Hardware Configuration within Helmet and Deliver Helmet to Machine Shop<br>3. Program Microcontrollers<br>4. Perform Functionality Test using Breadboard Circuit and Sensors | 1, 2, 4 | 2, 3, 4 | 1, 3, 4 |
| **3/11** | Spring Break | Spring Break | Spring Break | Spring Break |
| **3/18** | 1. Perform PCB Functionality Tests<br>2. Test Physical Layout of PCB within Helmet<br>3. Perform Bluetooth Connectivity Tests with Sensors | 1, 2 | 1, 3 | 1, 3 |
| **3/25** | 1. Submit Final PCB Order<br>2. Perform Durability & Functionality Test on Helmet | 2 | 1 | 2 |
| **4/1** | 1. Entire System Integration Testing with Motorcycle and Helmet<br>2. Conduct Test Drive | 1, 2 | 1, 2 | 1, 2 |
| **4/8** | 1. Perform Final Debugging<br>2. Prepare Mock Demo | 1, 2 | 1, 2 | 1, 2 |
| **4/15** | 1. Perform Mock Demo<br>2. Prepare for Final Demo | 1, 2 | 1, 2 | 1, 2 |
| **4/22** | 1. Perform Final Demo<br>2. Finalize Final Paper<br>3. Prepare Final Presentation | 1, 2 | 1, 2 | 1, 2 |
| **4/29** | 1. Perform Final Presentation | 1, 2 | 1, 2 | 1, 2 |

**Table 5: Schedule**

| Part | Manufacturer | Quantity | Cost |
|------|--------------|----------|------|
| 101020030-TSL2561 | Seeed Technology Co. | 3 | $10.90 <br> 101020030 - TSL2561 |
| ESP32-S3-WROOM-1-N16R2 | Espressif Systems | 2 | $3.62 <br> ESP32-S3-WROOM-1-N16R2 |
| RC Airplane Module Mini 360 DC-DC Buck Converter Step Down Module MP2307 | HiLetgo | 2 | $1.92 <br> RC Airplane Module Mini 360 DC-DC Buck Converter Step Down Module MP2307 |
| 10118194-0001LF | Amphenol (ICC) | 2 | $0.45 <br> 10118194-0001LF |
| IRLZ44NPBF | Infineon Technologies | 3 | $1.28 <br> IRLZ44NPBF |
| Adafruit Flexible Silicone Neon-Like LED Strip | Adafruit Industries | 3 | $13.95 <br> Flexible Silicone Neon-Like LED Strip |
| SP0503BAHTG | Littelfuse Inc. | 2 | $0.86 <br> SP0503BAHTG |
| DS04-254-2L-01BK | CUI Devices | 1 | $0.66 <br> DS04-254-2L-01BK |
| SS8050-G | Comchip Technology | 4 | $0.29 <br> SS8050-G |
| B3S-1000 | Omron Electronics | 4 | $0.66 <br> B3S-1000 |
| RC0805FR-0710KL | YAGEO | 14 | $0.10 <br> RC0805FR-0710KL |
| RC0805FR-071KL | YAGEO | 3 | $0.10 <br> RC0805FR-071KL |
| RC0805FR-07100KL | YAGEO | 4 | $0.10 <br> RC0805FR-07100KL |
| C0805C104K9RAC7800 | KEMET | 5 | $0.19 <br> C0805C104K9RAC7800 |
| C0805C106K9RAC7800 | KEMET | 2 | $0.48 <br> C0805C106K9RAC7800 |
| C0805C105K9RAC7800 | KEMET | 4 | $0.18 <br> C0805C105K9RAC7800 |

**Table 6: Parts**

# Appendix E: Abbreviations

ADC……………………………………………………………Analog-to-Digital Converter

BLE…………………………………………………………………Bluetooth Low Energy

GPIO……………………………………………………………General Purpose Input/Output

I2C………………………………………………………………Inter-Integrated Circuit

LED………………………………………………………………Light Emitting Diodes

PCB……………………………………………………………..Printed Circuit Board

SCL………………………………………………………………Serial Clock Line

SDA………………………………………………………………Serial Data Line

UART………………………………………… Universal Asynchronous Receiver / Transmitter

USB………………………………………………………………Universal Serial Bus

# References

[1]   https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813466.pdf

[2]   *https://cdn-shop.adafruit.com/datasheets/TSL2561.pdf*

[3]   https://herrmans.eu/news/bike-light-buying-guide-how-many-lumens-or-lux-do-i-need/

[4]   https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

[5]   https://www.link-labs.com/blog/bluetooth-vs-bluetooth-low-energy

[6]   https://www.ieee.org/content/dam/ieee-org/ieee/web/org/about/corporate/ieee-code-of-ethics.pdf

[7]   https://www.acm.org/code-of-ethics

[8]   https://randomnerdtutorials.com/esp32-pinout-reference-gpios/

[9]   https://grainger.illinois.edu/academics/undergraduate/majors-and-minors/electrical-engineering - :~:text=The average starting salary for,average signing bonus of %249,554.

[10]  https://www.picoauto.com/library/automotive-guided-tests/injector-voltage/ - :~:text=The injector is an electromechanical,is switched via the ECM.

[11]  https://www.onsemi.com/pdf/datasheet/lp2950-d.pdf

[12] https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf#i2c

[13] https://www.jauch.com/blog/en/6-important-parameters-for-the-design-in-of-lithium-polymer-batteries/#:~:text=By%20default%2C%20lithium%20polymer%20cells,prevail%20when%20charging%20the%20cells.

[14] https://github.com/lorsi96/TSL2561-ESP32-Light-Sensor-Driver/tree/master

[15] https://www.ti.com/lit/an/slvaed3/slvaed3.pdf?ts=1711720655042&ref_url=https%253A%252F%252Fwww.google.com%252F

[16] https://cdn-shop.adafruit.com/datasheets/MP2307_r1.9.pdf

[17] https://www.monolithicpower.com/en/mp2307.html