Electric Longboard Safety Suite

ECE 445 Design Document - Spring 2021

Project # 17

Pouya Akbarzadeh, Alexander Krysl, Patrick Stach

Professor: Jonathon Schuh

TA: Evan Widloski

Contents

1 Introduction

1.1 Problem

Electric Skateboards and Longboards have skyrocketed in popularity for personal transportation in urban cities & towns [1]. Their nimble and speedy characteristics allow users to easily navigate long distances of congested vehicle or foot traffic, yet are small and lightweight enough to be carried around indoors. Despite their value as a useful transportation and recreational tool, it is clear from the relatively simple motor and user interface designs that nearly all consumer electric skateboards and longboards lack seemingly paramount safety features. Note that we will use the terms "electric skateboard" and "electric longboard" interchangeably for the rest of this document. To begin, no consumer electric longboard attempts to discern whether the user is physically on the longboard or has jumped/fallen off. Unfortunately, electric skateboard speed controllers do not make this distinction, and so it allows for highly dangerous scenarios where a user falls off the electric longboard while inadvertently maintaining the throttle. These electric longboards may then accelerate out of control —possibly toward pedestrians —having lost the weight of the user. Next, no consumer electric longboard includes mechanisms that attempt to mitigate wheel-slip, which is the result of the user providing too much throttle input, more than what the electric skateboard wheels' traction can handle. This can occur when traveling over uneven/gravel-like terrain, performing extremely hard turns, or even applying severe braking throttle while going at speed, and more. In each of these cases, the electronic speed controllers will allow the powered wheels on the electric skateboard to overcome static friction, continually "slipping" across the ground. Wheel-slip is a significant safety risk as it dissolves the user's control of the board. It may upset the balance of the user, it may cause the electric longboard to slide out from under the user, and it may prevent the user from braking/slowing down the electric skateboard appropriately. While the statistics of electric longboards have not been formally studied, in the case of electric bicycles, roughly 30% of accidents were a result of wheel-slip [2].

1.2 Solution

To help remedy these safety concerns, we will implement a twofold plan. First, we will integrate weight sensors within the trucks of the longboard. Second, we will develop & utilize wheel-revolution sensors across each of the four wheels. We will then use that sensor data to identify whenever wheel-

slip or user-ejection has occurred; when it is identified, we will interface with the motor controllers to allow us to reduce motor power to the responsible wheel(s). Whenever we do identify a scenario where the user has fallen off of the board, we will ensure that the electric longboard is not allowed to accelerate more than a trivial amount. Further, whenever we identify that the rear, powered wheels are spinning considerably faster than the front, unpowered wheels, we will attempt to regain traction by dampening the throttle input until the difference is minimized.

1.2 Visual Aid

Figure 1: A common style of electric longboard

All commercially available electric longboards utilize remote controllers as their user interface design. For our project, we will not deviate from this, as seen on label A in Figure 2. Our board microcontroller will receive user throttle input from the remote via 2.4GHz RF signal, in addition to an input from a dead man switch as described above, such that if the button is released, motor power is disengaged.

On a fundamental level, our longboard starts as a basic, non-electric longboard, including a wooden deck, skateboard trucks, and urethane wheels. This is seen as label B on Figure 2. This establishes the fundamental control for the user, as he or she will stand upon the longboard and

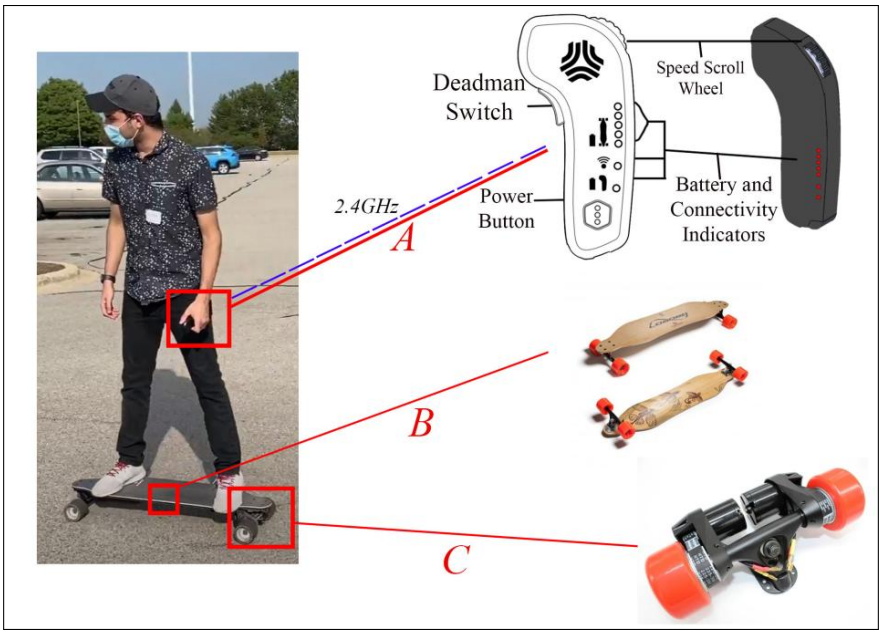

Figure 2: Visual aid of assembly and use of an electric longboard

turn by leaning in the corresponding direction.

Then, to begin the conversion of our basic longboard into an electrically powered one, we will then attach the following to the rear trucks and rear wheels: motor mounts, brushless outrunner motors, and a belt & pulley system. This is seen as label C on Figure 2.

After all the important physical components are installed, we will complete the installation of the electronic components to the longboard, which include the batteries, electronic speed controllers, safety microcontrollers, wheel/rotational speed sensors, and weight sensors. An example of a fully assembled longboard can be found on Figure 1.

1.3 High Level Requirements

To consider our project successful, our safety suite must fulfill the following:

1. Our electric longboard is able to sense if the weight atop the board is significantly less than that of a normal user (a threshold value of 40 pounds); when sensed for 0.5 seconds or more, motor power is reduced to a maximum speed of 5 mph or less.

2. Our electric longboard is able to measure/record the rotational speed of each of the front, unpowered wheels on the longboard to a tolerance of $\pm$ 5% of the true value.

3. Our electric longboard will sense whether the powered wheels are slipping in the opposite direction of the unpowered wheels (as would be the case under extreme braking); when sensed, our electric longboard will automatically dampen the reverse-throttle value until traction is recovered.

2 Design

2.1 Physical Design

For the front wheel RPM sensing, we will be using the assembly show in Figure 3. The system consists of 4 parts: the wheel, the magnet ring, the Hall sensors attachment, and the trucks. The magnet ring is secured to the wheel so that as the wheel freely spins the magnet ring spins with it. The Hall sensor attachment is mounted onto the trucks and is fixed in place. The Hall sensor attachment is lined up close enough to the magnetic ring so that the Hall sensors can detect the magnetic field of a magnet passing by.

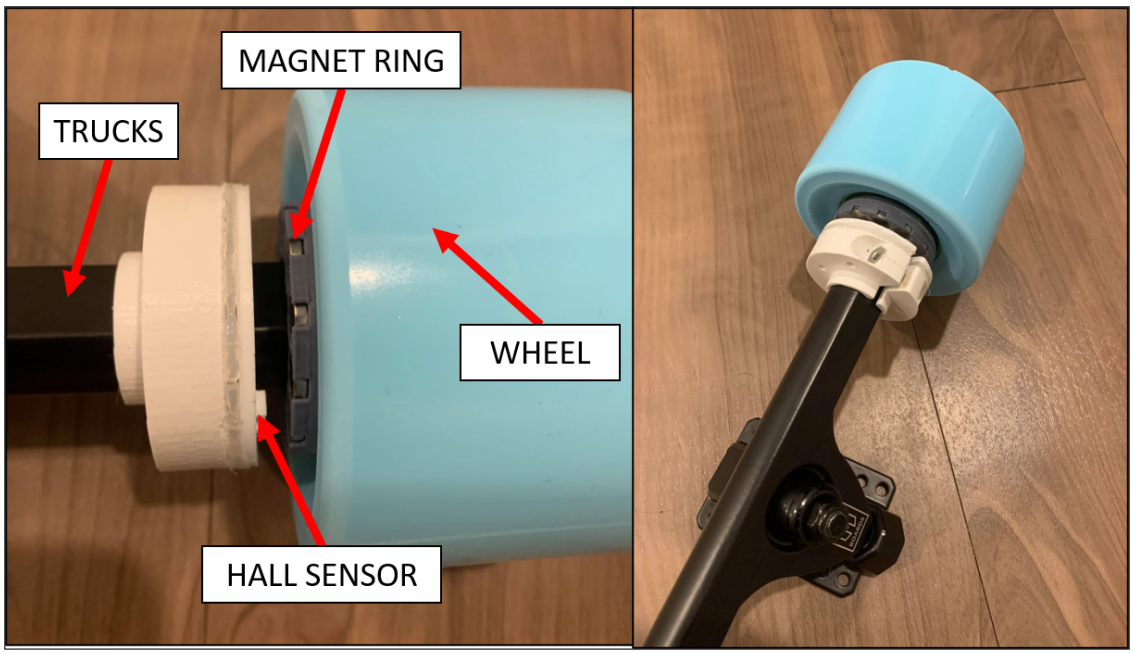

Figure 3: Assembly for front wheel RPM measurements with a Hall effect Sensors

For weight detection, we will put a pressure sensor in-between the both trucks and the bottom of the deck. When a user is on the board, as seen in Figure 4 on the left, the user's weight will be distributed between the front and back trucks, strongly compressing the sensor. When there no one is on the board, as seen in Figure 4 on the right, only the weight of the deck (negligible to a person) will distributed to the front/back trucks, minimally compressing the sensor in between the deck and the board.

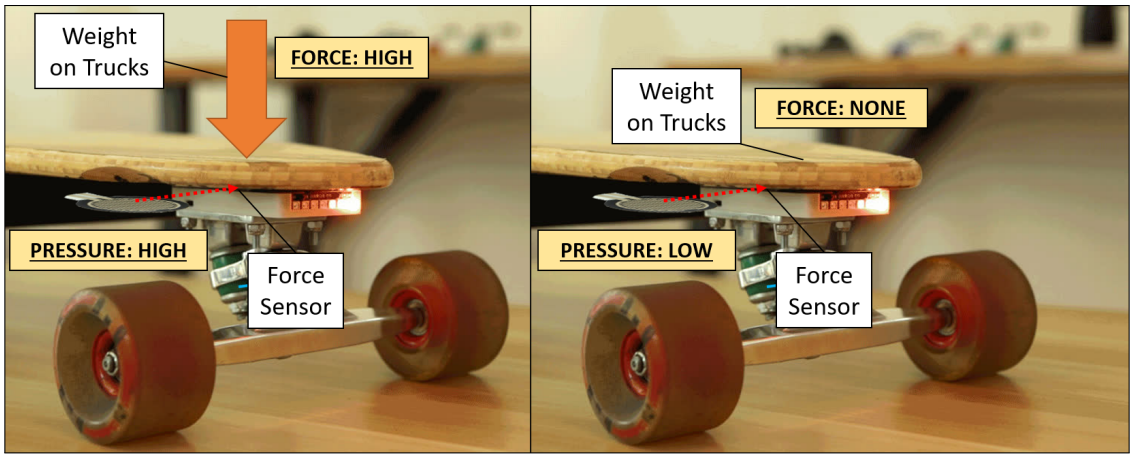

Figure 4: Assembly for weight detection on board using pressure sensor

2.2 Block Diagram

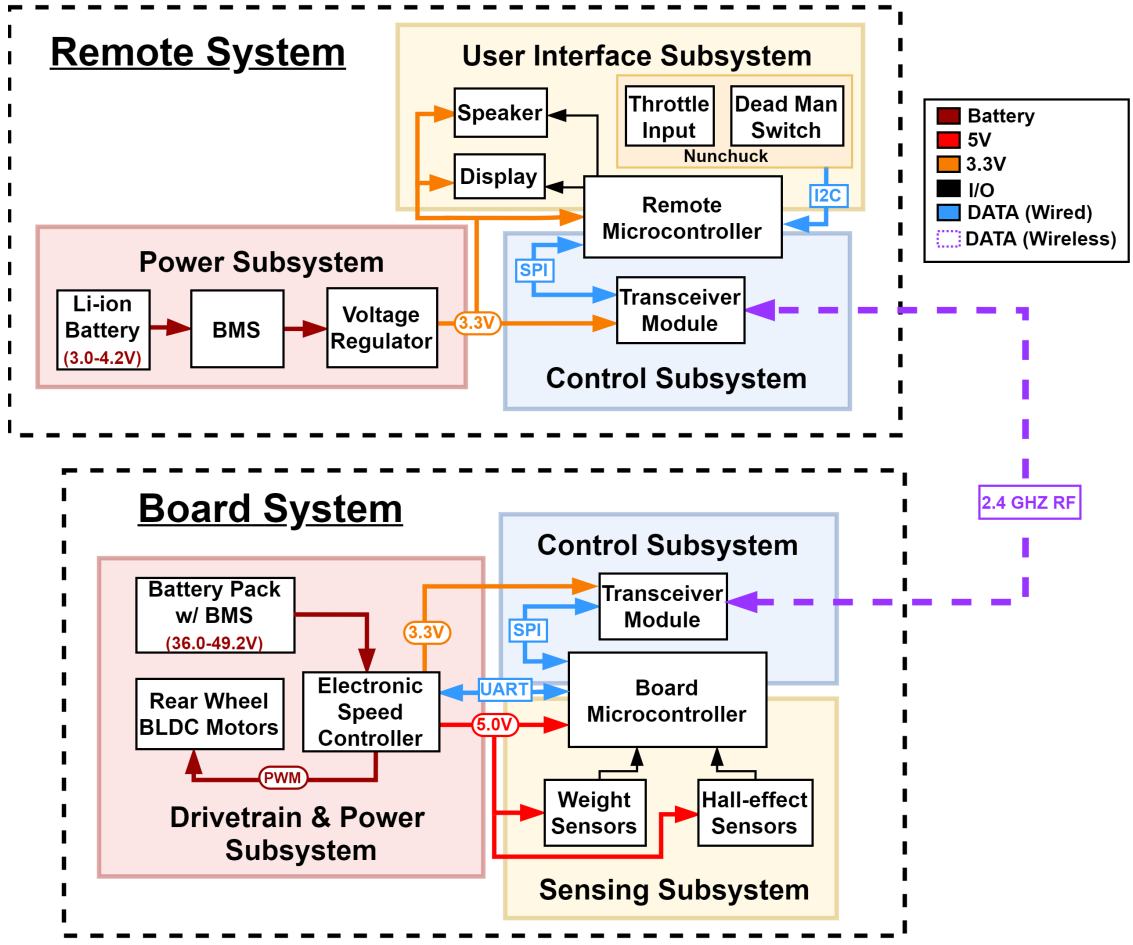

Figure 5: Electric Longboard Safety Suite Block Diagram

2.3 Functional Overview & Block Diagram Requirements

2.3.1 Board Control Subsystem

The Board Control Subsystem is responsible for receiving transmissions from the remote, taking data from the sensing subsystem, and commanding the electronic speed controllers. Based on these inputs, the board control subsystem will command the electronic speed controllers to power the motors accordingly. The Board Control Subsystem determines whether the board will accelerate or decelerate and will implement our wheel-slip mitigation algorithm. Our Board Control Subsystem is made up of an ATmega328p microprocessor that interfaces with the electronic speed controllers via UART and with the nRF24L01 transceiver module via SPI. The transceiver module will then wirelessly communicate with the remote. These modules will be driven by the Board Drivetrain and Power Subsystem via 5 volt and 3.3 volt outputs from the electronic speed controller module. For more information on the software design of the Board Control Subsystem, please refer to Section 2.5.1. To ensure that the Board Control Subsystem is fulfilling its responsibilities for receiving transmissions from the remote, taking data from the sensing subsystem, and commanding the electronic speed controllers, a requirements & verification table can be found below. Note that the verification tests in both Table 1 & Table 2 each require that there is enough space for the user to use the board at speeds of at least 12 miles per hour for at least 20 seconds.

Table 1: Board Control Subsystem – Requirements & Verification Pt. 1

Requirements	Verification
• When the Board Control Subsystem detects a fatal communication failure and detects that the user is on the board, the Board Control Subsystem must command the electronic speed controllers to coast.	• Ensure the board is at an idle state such that the user is on top of the board and the remote is wirelessly connected to the board. • Then, under normal operating conditions, reach a speed of at least 12 miles per hour via remote throttle input. • Then, turn off the remote while the user remains on top of the board. Confirm that the board coasts and does not attempt to command the electronic speed controllers to halt.
• When the Board Control Subsystem detects a fatal communication failure and detects that the user is no longer on the board, the Board Control Subsystem must command the electronic speed controllers to stop.	• Ensure the board is at an idle state such that the user is on top of the board and the remote is wirelessly connected to the board. • Then, under normal operating conditions, reach a speed of at least 12 miles per hour via remote throttle input. • Then, the user must safely eject off of the board and turn off the remote. Confirm that the board begins to decelerate.
• When the Board Control Subsystem detects that the user is no longer on the board and that the remote dead-man switch is not pressed, the Board Control Subsystem must command the electronic speed controllers to stop.	• Ensure the board is at an idle state such that the user is on top of the board and the remote is wirelessly connected to the board. • Then, under normal operating conditions, reach a speed of at least 12 miles per hour via remote throttle input. • Then, the user must safely eject off of the board and release the dead-man switch. Confirm that the board begins to decelerate.
• When the Board Control Subsystem detects that the user is no longer on the board and that the remote dead-man switch is pressed, the Board Control Subsystem allows the throttle input to take effect up to a maximum speed of 5 miles per hour or less.	• Ensure the board is at an idle state such that the user is not on top of the board and the remote is wirelessly connected to the board. • Then apply maximum remote throttle input. Confirm that the maximum speed achievable by the board is no more than 5 miles per hour (i.e. that of walking pace).

Table 2: Board Control Subsystem – Requirements & Verification Pt. 2

Requirements	Verification
• When the Board Control Subsystem detects that the user is on the board and that the remote dead-man switch is not pressed, the Board Control Subsystem must command the electronic speed controllers to coast.	• Ensure the board is at an idle state such that the user is on top of the board and the remote is wirelessly connected to the board. • Then, under normal operating conditions, reach a speed of at least 12 miles per hour via remote throttle input. • Then, release the dead-man switch while the user remains on top of the board. Confirm that the board coasts and does not attempt to command the electronic speed controllers to halt. Confirm that the board does not respond to throttle input.
• When the Board Control Subsystem detects that the user is on the board, the remote dead-man switch is pressed, and the rotational direction of the front wheels are opposite the wheel rotational direction of the motorized rear wheels, the Board Control Subsystem will adjust the throttle value commanded to the electronic speed controllers such that the motorized wheels regain static frictional traction.	• Load a build of the board microcontroller such that the wheel-slip mitigation algorithm is ignored (RE-TRACTION state is identical to RUN state, as seen in section 2.5.1). • Ensure the board is at an idle state such that the user is on top of the board and the remote is wirelessly connected to the board. Then, under normal operating conditions, reach a speed of at least 12 miles per hour via remote throttle input. • Then, apply as much reverse throttle to cause the rear wheels to spin in a direction opposite of the forward momentum of the board. Record the braking throttle value. • After, load a release-level build of the board microcontroller such that the wheel-slip mitigation algorithm is active and implemented. • Ensure the board is at an idle state such that the user is on top of the board and the remote is wirelessly connected to the board. Then, under normal operating conditions, reach a speed of at least 12 miles per hour via remote throttle input. • Then, apply the same or more reverse throttle as was recorded previously. Confirm that the rear wheels regain traction at least one instance during braking.

2.3.2 Board Sensing Subsystem

This subsystem is responsible for sensing both the relative weight atop the board as well as the rotational speed and direction of both front wheels. Our weight sensors are implemented with a resistor divider circuit, utilizing the velostat material as the load sensor. Velostat is a material which decreases in resistance as pressure is applied to it. Our front RPM sensors are implemented with Hall-effect switches that face towards a ring of magnets attached to the front wheels. For more information regarding the physical design of this subsystem, refer to Section 2.1. To ensure that the Board Sensing Subsystem is fulfilling its responsibilities for sensing both the relative weight atop the board as well as the rotational speed and direction of both front wheels, a requirements & verification table can be found below. Our board microcontroller will then take in the data from the sensing sub-modules and calculate the actual weight atop the board and front wheel velocity. For more information regarding the software design of the RPM sensors, refer to Section 2.5.2

Table 3: Board Sensing Subsystem – Requirements & Verification

Requirements	Verification
• Taking the weight sensors as input, the sensing subsystem must determine whether the weight atop the board is greater than a user weight threshold, as determined from testing (30-50 pounds).	• Ensure the board is at an idle state such that no weight is added on top of the deck. Record the boolean value for weight threshold sensing as read from the board microcontroller via serial debugging. • Next, add weight atop the deck that is greater than 30 pounds. Record the boolean value for weight threshold sensing as read from the board microcontroller via serial debugging. Confirm the value read is different from the default reading. • After, remove the weight atop the deck. Record the boolean value for weight threshold sensing as read from the board microcontroller via serial debugging. Confirm the value read is the same as the default reading.
• Taking the wheel RPM hall effect sensors as input, the sensing subsystem must determine the RPM of the left front wheel and the right front wheel, within a tolerance of ±5% of the RPM value.	• Ensure the board is at an idle state such that the wheels are not spinning. Record the left front wheel RPM value as read from the board microcontroller via serial debugging. Confirm this value is 0 (within a tolerance of +/- 5 RPM). • Next, manually spin the left front wheel to a non-zero RPM. This may be accomplished by attaching a belt between the left front wheel and the rear, powered wheel, then commanding the powered wheels to move. Record the left front wheel RPM value as read from the board microcontroller via serial debugging. Confirm this value is greater than 0 (within a tolerance of +/- 5% RPM). The true RPM value may be queried from the appropriate motor controller. • Then, ensure the right front wheel is not spinning. Record the right front wheel RPM value as read from the board microcontroller via serial debugging. Confirm this value is 0 (within a tolerance of +/- 5 RPM). • Next, manually spin the right front wheel to a non-zero RPM. This may be accomplished by attaching a belt between the right front wheel and the rear, powered wheel, then commanding the powered wheels to move. Record the right front wheel RPM value as read from the board microcontroller via serial debugging. Confirm this value is greater than 0 (within a tolerance of +/- 5% RPM). The true RPM value may be queried from the appropriate motor controller.

2.3.3 Board Drivetrain & Power Subsystem

The Board Drivetrain & Power Subsystem is responsible for supplying power to all electronic components on the board, and provide the means to control the rear motors on the electric skateboard. For more information regarding our choices of the commercial components that make up the Board Drivetrain & Power Subsystem, please refer to section 2.6. In the particular context of our safety system for electric longboards, the Board Drivetrain & Power Subsystem must transmit the RPM data of each powered wheel, and must accept commands from the Board Control Subsystem to set the relative speed of the motors.

Table 4: Board Drivetrain & Power Subsystem – Requirements & Verification

Requirements	Verification
• The left electronic speed controller and right electronic speed controller must transmit their respective motor RPM values to the board control subsystem upon request.	• Ensure the board is at an idle state such that the wheels are not spinning. Record the left and right RPM value as read from the board microcontroller via serial debugging. Confirm that both values are near 0 (+/- 5 rpm). • Then, command the both electronic speed controllers to set a motor current of 10 Amps. While both wheels are spinning, physically restrict the right wheel such that it rotates at a slower rate compared to the left wheel. • Record the left and right RPM value as read from the board microcontroller via serial debugging. Confirm that the RPM value for the left wheel is higher than that of the right wheel.
• The electronic speed controllers power the motors at the command of the board control subsystem. The commands may be to set duty cycle, set current, or set RPM.	• Ensure the board is at an idle state such that the wheels are not spinning. Then, send a command to both ESCs from the board microcontroller to either set a nonzero duty cycle, current, or RPM. Confirm that both motors begin spinning.

2.3.4 Remote Control Subsystem

The Remote Control Subsystem is responsible for passing the user interface input data to the Board Control Subsystem; this specifically includes the input throttle thumbwheel/thumbstick and the dead-man switch. The Remote Control Subsystem must package this information into radio communication messages to the Board Control Subsystem. Additionally, the Remote Control Subsystem will need to respond to messages sent from the board. If any fatal communication failure is detected, the Remote Control Subsystem must alert the user through the Remote User Interface

Subsystem. For more information regarding the software design of the Remote Control Subsystem, refer to section 2.5.3.

Table 5: Remote Control Subsystem – Requirements & Verification

Requirements	Verification
• The remote control subsystem must send the throttle and dead man switch values, as read from the microcontroller, to the board control subsystem.	• Ensure the board is at an idle state such that the remote is wirelessly connected to the board. Record the throttle value & dead man switch value as read from the remote microcontroller via serial debugging. • Then, record the throttle value & dead man switch value as read from the board microcontroller via serial debugging. Confirm that these two sets of values are identical.
• The remote control subsystem must indicate when a fatal communication failure has been detected.	• Ensure the board is at an idle state such that the remote is wirelessly connected to the board. Record the connection status as read from the remote microcontroller via serial debugging. Confirm that the connection status indicates that the remote is connected to the board. • Then, turn off power to the board. Record the connection status as read from the remote microcontroller via serial debugging. Confirm that the connection status indicates that the remote has lost connection with the board.

2.3.5 Remote User Interface Subsystem

The Remote User Interface Subsystem is responsible for allowing the user to control the electric skateboard, within the confines of our safety suite. The throttle must have at least one axis of motion that is self-centering, as this allows the user to intuitively understand how much throttle they are applying at any given moment. The Remote User Interface Subsystem must also include a dead-man switch, a button that must be pressed for the throttle to take effect. Further, the Remote User Interface Subsystem must provide sensory output to the user, such as a speaker or vibration motor, to communicate back to the user of any errors.

Table 6: Remote User Interface Subsystem – Requirements & Verification

Requirements	Verification
• The throttle input must provide at least one self-centering axis of motion.	• Push the throttle wheel/thumbstick to its limit in one direction on the axis, then release. Confirm that the throttle wheel/thumbstick returns to the default center position. • Next, push the throttle wheel/thumbstick to its limit in the other direction on the axis, then release. Confirm that the throttle wheel/thumbstick returns to the default center position.
• The remote microcontroller must be able to interpret the position of the throttle input.	• Leave the throttle wheel/thumbstick at its default position. Record the throttle value as read from the remote microcontroller via serial debugging. • Then, push the throttle wheel/thumbstick to its maximum position on the axis. Record the throttle value as read from the remote microcontroller via serial debugging. Confirm that this value is greater than the value from default position. • Then, push the throttle wheel/thumbstick to its minimum position on the axis. Record the throttle value as read from the remote microcontroller via serial debugging. Confirm that this value is lesser than the value from default position.
• The remote microcontroller must be able to interpret the status of the dead man switch.	• Leave the dead man switch unpressed. Record the dead man switch value as read from the remote microcontroller via serial debugging. • Then, press and hold the dead man switch. Record the dead man switch value as read from the remote microcontroller via serial debugging. Confirm that this value is different from the value from default. • Then, release the dead man switch. Record the dead man switch value as read from the remote microcontroller via serial debugging. Confirm that this value is the same as the value from default.
• A sound must emit from the remote if the remote microcontroller detects a fatal transmit/connection failure.	• Load a build of the microcontroller firmware where the fatal transmit/connection failure is set to always active. Confirm that the remote produces a noise to alert the user.

2.3.6 Remote Power Subsystem

The Remote Power Subsystem is responsible for supplying power to all electronic components on the remote, most importantly the microcontroller and the transceiver module. The subsystem must be able to safely charge the the battery with external 5V USB power, in addition to regulating the variable battery output voltage to 3.3V. For more information on the hardware design of the remote power subsystem, refer to section 2.4.2.

Table 7: Remote Power Subsystem – Requirements & Verification

Requirements	Verification
• Must be able regulate battery voltage to power components throughout the discharge cycle of the battery and automatically cutting out power when battery voltage drops too low..	• Connect input of voltage regulator to voltage supply. Connect output of voltage regulator to programmable load. Set voltage supply to maximum battery voltage, 4.2V. • Check voltage reading with multimeter to make sure output voltage does not fall outside of 3.3V ±5% under no load and full load. Repeat this with the lowest battery voltage for proper operation, 3.45V. Now lower the input voltage and verify that the output goes to 0V somewhere between 3.45V and 3.40V.
• The remote battery must be able to be recharged via 5V USB input within 2 hours, automatically stopping charging when battery is at fully capacity. The remote battery must be able to last at minimum 5 hours.	• Start with battery discharged to the point where it reads 3.4V. Apply 5V input via power supply to input of BMS. Monitor current delivered to battery w/ multimeter and monitor battery voltage. Verify that at the end of charge cycle, battery voltage is 4.2V and no current is being delivered to the battery. Record time of charge cycle and verify it is less than 2 hours. • Apply 50 mA load and record the time it takes for the output voltage to fall to 3.4V. Ensure this is greater than 6 hours.

2.4 Hardware Design

2.4.1 Operating Voltage & Regulation

We will need to provide regulated voltage to the microcontroller (ATmega328p) and transceiver (nRF24L01) in the remote from the lithium-ion battery output. According to the nRF24L01 datasheet [3] as seen with Figure 6, even though the nRF24L01 is 5V input signal tolerant, it will still requires a supply between 2.7V and 3.3V.

According to the ATmega328p datasheet [4] as seen with Figure 7, the ATmega328p can operate

Operating conditions	Minimum	Maximum	Units
Supply voltages			
V_{DD}	-0.3	3.6	V
V_{SS}		0	V
Input voltage			
V_I	-0.3	5.25	V

Figure 6: nRF24L01 Range of Acceptable Supply and Input Signal Voltages from Datasheet

at a supply voltage between 2.7V to 5.5V. With 3.3V, the fastest clock speed we would be able to run it at would be 8MHz. With 5V, the fastest clock speed we would be able to run it at would be 16MHz.

- Speed grade:
  - 0 to 8MHz at 2.7 to 5.5V (automotive temperature range: –40°C to +125°C)
  - 0 to 16MHz at 4.5 to 5.5V (automotive temperature range: –40°C to +125°C)

Figure 7: ATmega328p Speed Grade

With this given information, it would make the most sense to operate it at 3.3V and operate the microcontroller at 3.3V with 3.3V logic for our remote control. This is for multiple reasons; first of all, the nRF24L01 requires a 3.3V supply even with 5 input signals, so even if we wanted to operate our microcontroller at 5V we need to generate a 3.3V and 5V rail from the battery output. Second, running the ATmega328p in the remote at 8MHz will not be an issue as it will only be communicating with the transceiver, nunchuck board & display in addition to computing some simple logic. Third, running the microcontroller at 3.3V will lead to less power consumption, ideal for optimizing battery power. Lastly and most important, stepping the output up to 5V would require a step-up topology such as a buck-boost, and this would add unnecessary complexity to our design. Instead we will use a linear regulator to step down our voltage to 3.3V.

Figure 8 demonstrates the typical discharge curve of a Lithium Ion 3.7V/4.2V battery [5]. For battery health, we will don't want to use the battery when it is below 3.5V. As seen on the curve, operating at a C-rate of 0.2C will still allow us to use 95% of our battery capacity. We are planning to use a 500mAh battery with battery life of 5 hours- our actual C-rate will be 0.1C so the discharge curve will actually be even higher up than 0.2C, allowing for even more than 95% usage of total capacity. From this curve, we can also see that the battery voltage on average around 3.8V, so we can call this our "average" voltage. Knowing this, we can calculate the efficiency of our linear regulator [6] using Eq. 1 to get around 86.8%. For our purposes this is more than good enough, we

would just have to aim for around 40mA average current draw or less accounting for power loss.

$$\eta = \frac{V_{out}}{V_{in}} = \frac{3.3V}{3.8V} = 86.8\% \tag{1}$$

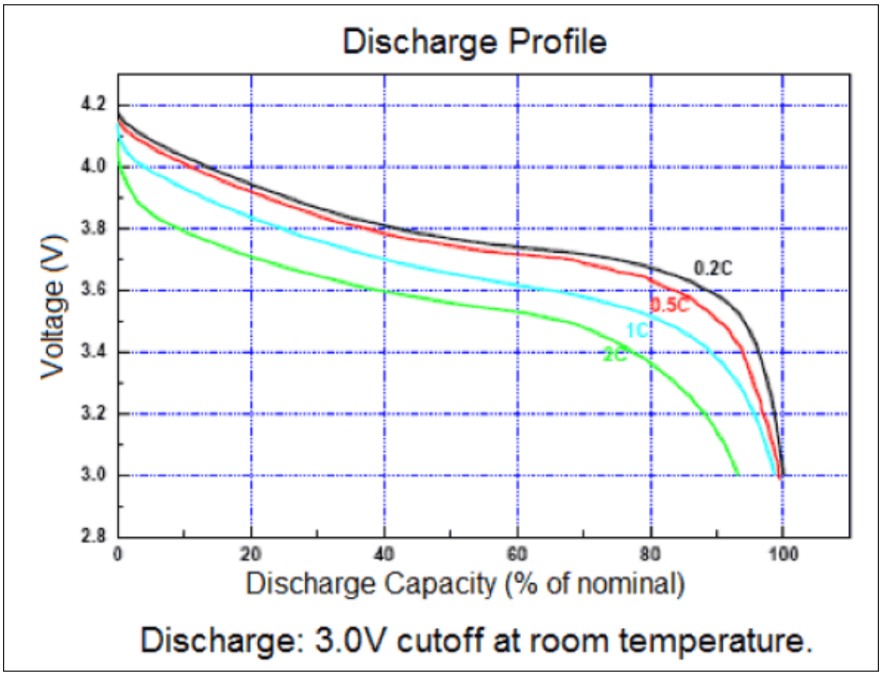

Figure 8: 3.7V/4.2V Lithium Ion Battery Discharge Curve

Since we want our lowest input voltage to be 3.5V when regulating to 3.3V, we will need to use an LDO with a dropout voltage of less than 200mV. The IC we found to fit our needs is the 3.3V version of the LP3985. It is SOT23-5 package size so it will be easy to work with, and there are no external parts required besides a ceramic capacitor on the input and output. According to the datasheet [7], the LP3985 has a 100-mV maximum dropout with a 150mA load, which meets our requirements of 200mV dropout voltage and max current of 50mA.

While we have chosen to run the ATmega328p at 3.3V on the remote, we have decided to run the one on the board at 5V First of all, the ATmega328p on the board can benefit from doubling clock speed to 16 Mhz as it will be doing very time-critical operations such as RPM sensing and sending motor control commands. The ESC already provides regulated 3.3V (for the nRF24L01) and 5V output (all other components), and all components are already 5V logic tolerant so there is no need for any level shifters.

2.4.2 Remote Battery Management System

Since we are using a lithium ion battery in the remote, we will need to make sure we charge it safely, and for this reason, we will be using a battery management system IC. The component we have chosen is the LTC1730, specifically in SOIC8 package size so it is easy to work with. This IC has the basics of charge control based on a programmable voltage cutoff (we will be using 4.2V), but it also has extra safety features mentioned in the datasheet [8] such as battery temperature sensing and maximum current limiting. Additionally, it also has some features that would be useful to the end-user such as the ability to add an LED to show whether a battery is charging, approaching end-of-charge, or not charging. We will be using typical applications circuit as seen in Figure 9

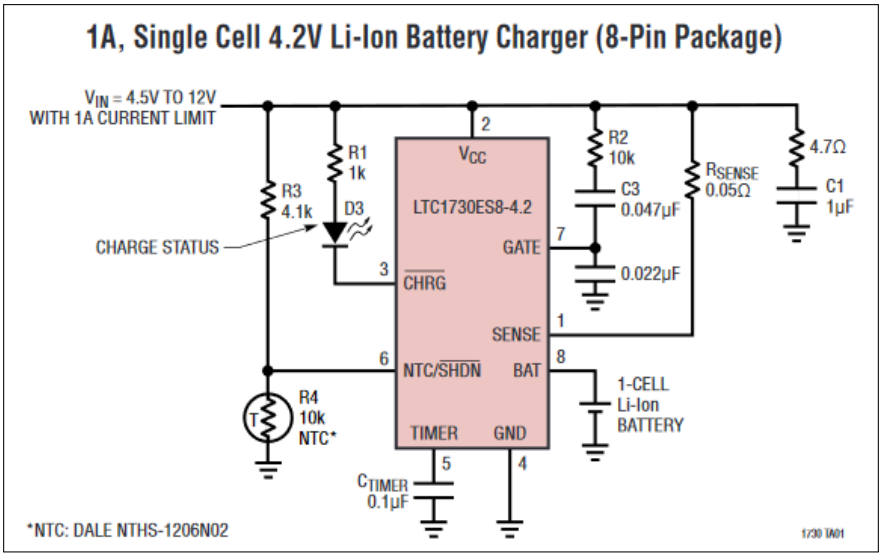

Figure 9: Typical LTC1730 Lithium Ion Battery Charger Application

2.5 Software Design

2.5.1 Motor Control, including Wheel Slip Detection & Correction

The core component of our project is the software decision-making of our board microcontroller. It is ultimately responsible for ensuring that the electronic speed controllers are governed by our safety features or by the user throttle, depending on the external circumstances. To accomplish this, our board microcontroller will take the following as inputs into its algorithm: transceiver input - dividing into throttle input and dead-man switch input, weight sensing circuitry, front wheel RPM sensing algorithm (discussed in a section 2.5.3), and back wheel RPM reports from the electronic

speed controllers. Our algorithm will then take these inputs and choose the appropriate state of operation for the given scenario. Figure 10 shows a flowchart of our algorithm.

The possible states of operation are as follows:

- START: Command the electronic speed controllers as if continuing from the most recent state of operation. If there is no previous state of operation, the default state of operation is COAST.

- STOP: Command the electronic speed controllers to bring the electric skateboard to halt. This is a viable option only if the user is not detected on top of the board, to prevent the user from being jolted off of the board unexpectedly.

- COAST: Command no acceleration or deceleration throttle to the electronic speed controllers. This is generally the safest option if a safety feature is triggered while the user is detected on top of the board. As a result the electric longboard will coast with the inherent resistance of the belt drive motors.

- WALK: Command the electronic speed controllers as the user sets the throttle, but accelerate no faster than 5 miles per hour in either the forward or reverse direction. This state of operation is exclusively for when the user is actively commanding the board to move without actively riding it. This feature is designed for the convenience of the user to change the position of the board at low speeds, without posing any reasonable safety risks.

- RUN: Command the electronic speed controllers as the user sets the throttle. This is the normal operating state of the electric longboard.

- RE-TRACTION: Command the electronic speed controllers as the user sets the throttle, but under the manipulation of our wheel-slip correction algorithm. This state of operation is selected whenever wheel-slip has been detected, and attempts to regain traction of the failing wheel(s). We plan our wheel-slip correction algorithm to set the failing motor(s) RPM to the same value of its corresponding front wheel. Once the wheel RPMs have matched each other, within the threshold outlined in Figure 10, we return to the RUN state of operation unless another safety feature takes effect.

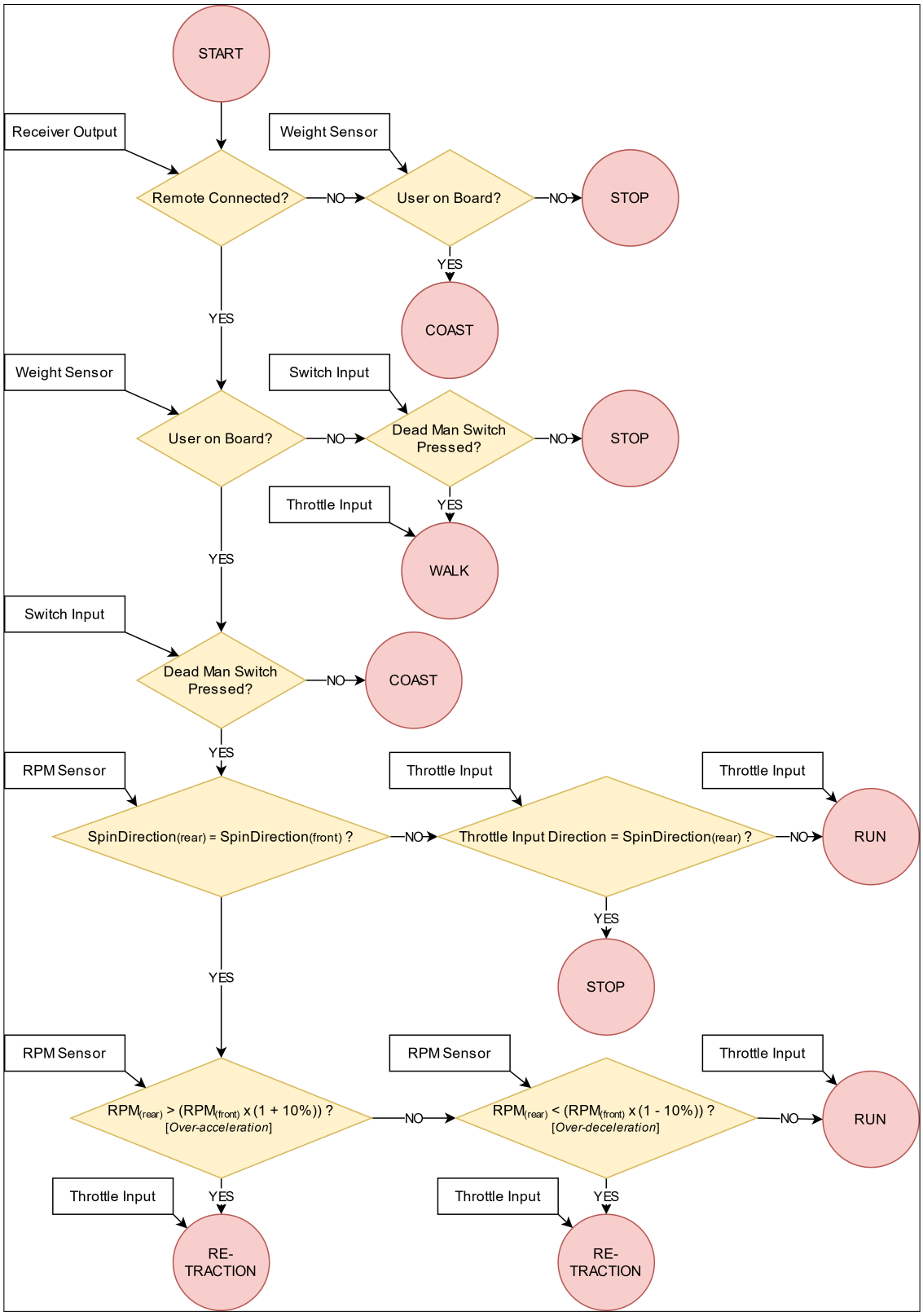

Figure 10: Wheel Slip Detection & Correction Flowchart

Our board microcontroller determines whether the remote is connected as described in section 2.5.3. For calculating whether the user is on the board, our board microcontroller will read the voltage of the weight sensor resistor divider circuit. Further, the microcontroller will read the last transmission for the dead-man switch value from the remote. Our microcontroller will then measure wheel rotation direction and wheel rotational speed either from the electronic speed controllers (for the powered wheels) or as described in section 2.5.2 (for the unpowered wheels).

2.5.2 RPM Sensing

For our RPM sensing, we will be using "latching" Hall effect sensors. The way that latching Hall effect sensors work is that when it detects the presence of a "positive" magnetic field from a magnet with south polarity, the output will go HIGH. This output will stay HIGH until it detects a "negative" magnetic field from a magnet with north polarity, in which the output will go LOW. Because of this behaviour, our ring of magnets will be alternating north-south polarity, as seen in Figures 12 and 14 with red representing a positive magnetic and blue representing a negative magnet.

Figure 12 demonstrates the behaviour of clockwise rotation: initially both Hall sensors are latched LOW since they both passed by a negative magnet, marked #1. After rotating, Hall A will detect the presence of the positive magnet marked #2, and the Hall A output will go HIGH. After rotating a bit more, Hall B will detect the presence of the positive magnet marked #2, and the Hall B output will go HIGH.

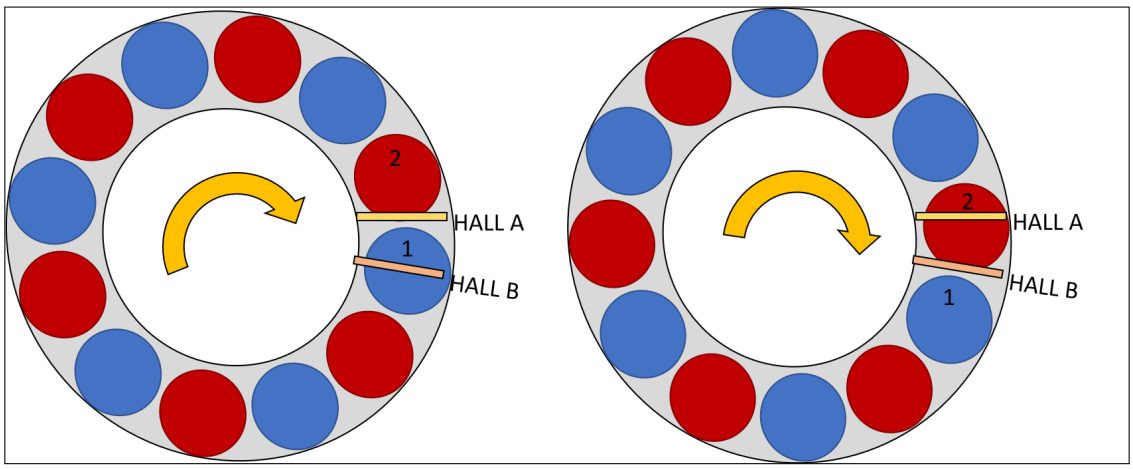

Figure 11: Visual Demonstration of Clockwise Rotation

The resulting output waveforms are seen in Figure 13: the Hall A output is leading the Hall B output by a 1/4 of a period. Note that in our case since we have 12 magnets, a period of the output waveforms corresponds to a 60 degree wheel rotation. Hall sensors A and B have specifically for this reason been placed 15 degrees apart in order for the two signals to be offset by 1/4 of a period. We also notice that on the rising edge of Hall A, Hall B output is LOW. Now let us consider the

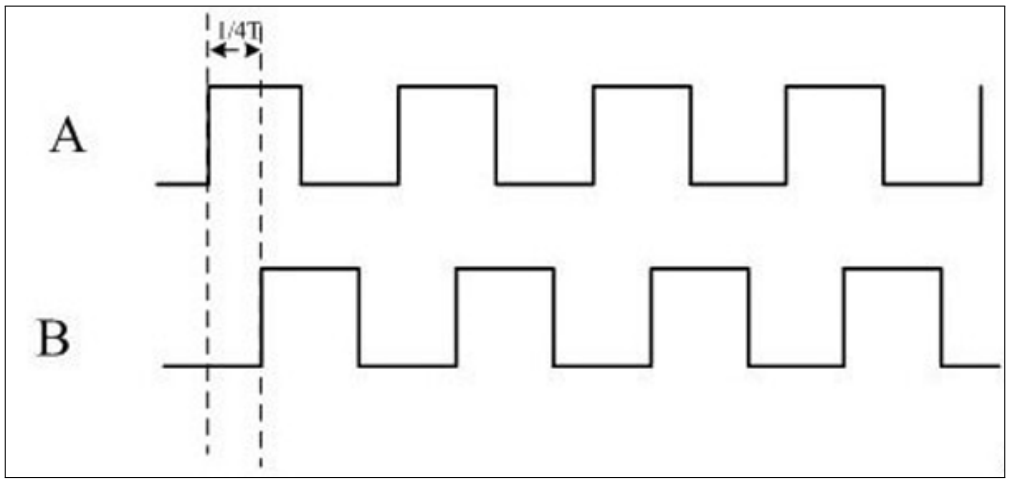

Figure 12: Hall Effect Output Waveforms for Clockwise Rotation

alternate case of counter-clockwise rotation as seen in Figure 14. Initially both Hall sensors are latched LOW since they both passed by a negative magnet, marked #1. After rotating, Hall B will detect the presence of the positive magnet marked #2, and the Hall B output will go HIGH. After rotating a bit more, Hall A will detect the presence of the positive magnet marked #2, and the Hall A output will go HIGH.

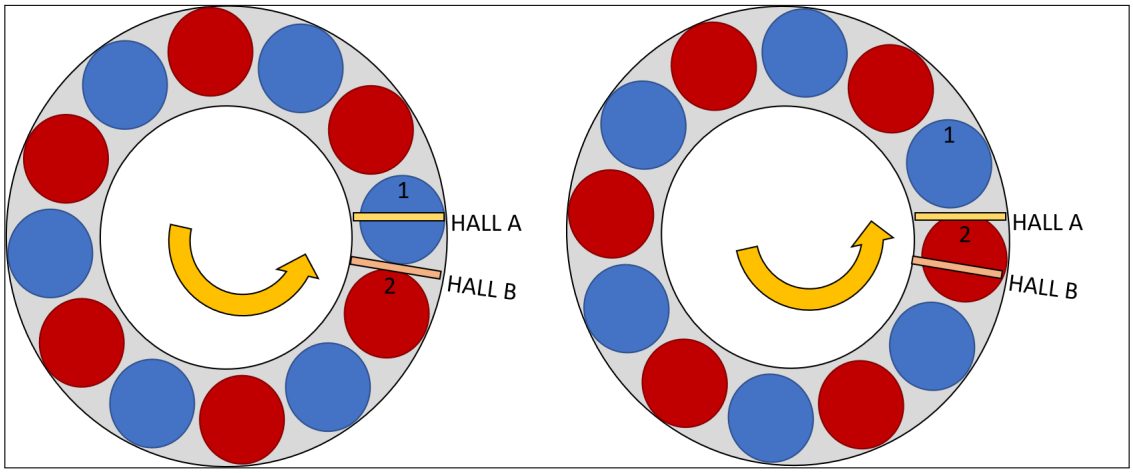

Figure 13: Visual Demonstration of Clockwise Rotation

The resulting output waveforms are seen in Figure 14: the Hall A output is lagging the Hall B output by 1/4 of a period. We also notice that on the rising edge of Hall A, Hall B output is HIGH. Note that while we should expect output waveforms of 50% duty cycle, the reality is that the output waveform might not be HIGH for the exact same duration that it is LOW. This is because of the built-in hysteresis in the Hall effect sensors. The magnetic field strength threshold for detecting a positive magnetic field can be slightly higher or lower than the magnetic field strength threshold for detecting a negative magnetic field.

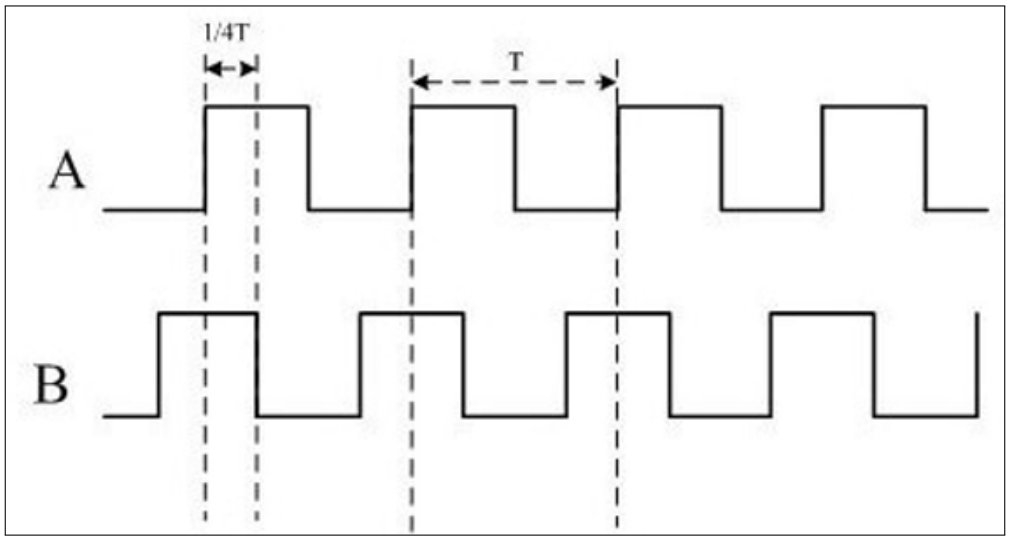

Figure 14: Hall Effect Output Waveforms for Counter-clockwise Rotation

With just two hall sensors per wheel, we can detect both wheel rotation speed and direction. We will use Hall A output as an interrupt signal and Hall B output as a general-purpose input signal. This corresponds to 4 total hall sensor outputs being read, and since the ATmega328p has two external interrupt pins, we will connect Hall A from the left wheel to the first external interrupt pin and Hall A from the right wheel to the second external interrupt pin. Because of the possible asymmetrical time-on period corresponding to different magnetic field thresholds for positive and negative field detection, we will only use have the interrupt for Hall A to trigger on the rising edge. On every rising edge we will record the current time and find the difference between the last recorded time. The difference between these two times corresponds to time it took for the wheel to rotate 60 degrees. With this information, we can calculate the RPM of the wheel, which is more thoroughly discussed in Section 2.7.1.

We can also determine whether the Hall A output signal is leading or lagging: if Hall B output

is LOW on the rising edge of Hall A output, this means that Hall A output is leading Hall B output. If Hall B output is HIGH on the rising edge of Hall A output, this means that Hall A output is lagging Hall B output. On every rising edge of Hall A output, all we need to do is read the state of Hall B, and from there we can determine whether Hall A output is lagging or leading B, and based on this we can determine clockwise or counter-clockwise rotation.

2.5.3 Transceiver Communication

The nRF2401 transceiver on the board will communicate with the nRF2401 transceiver on the board. The nRF2401 is able to transmit a single 4-byte message at a time, and we will dedicate a specified bit field to contain a specific value. We will do this with bit-masking and bit-shifting. The bit breakdown for the message the board transmits to the remote can be seen in Figure 15. Bits (2:0) are include an error message with each bit corresponding to whether or not one of the three errors has occurred. Bit (0) = 1 means wheel slip has occurred, bit (1) = 1 means the user is not on the board, bit (2) = 1 means the board battery is low. Bits (6:3) represent the 4-bit battery level where xF = 100% battery and x0 = 0%. Bits (15:7) are reserved for future possible additions to the error message.

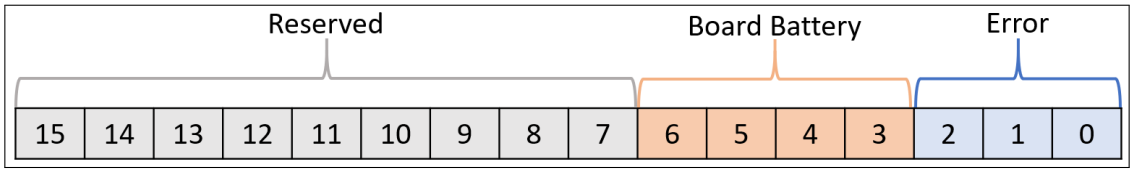

Figure 15: Bit breakdown of message transmitted from board to remote

The bit breakdown for the message the remote transmits to the board can be seen in Figure 17. Bits (7:0) represent the throttle input, where xFF is full forward throttle, x00 is full reverse throttle, and no throttle will be the midpoint value, x10 & x0F. Bit (8) corresponds to the status of the dead man switch. If the switch is pressed Bit (8) =1, else it is 0. Bits (15:9) are reserved for future possible additions to the error message.

In order to determine whether the two modules are connected or not, we will be calculating timeouts. The board and remote will be exchanging information every 100ms. If a transceiver module has not received a message within 1 second, we will assume that the connection has been severed. In this case of disconnection, our board microcontroller will safely command the electronic

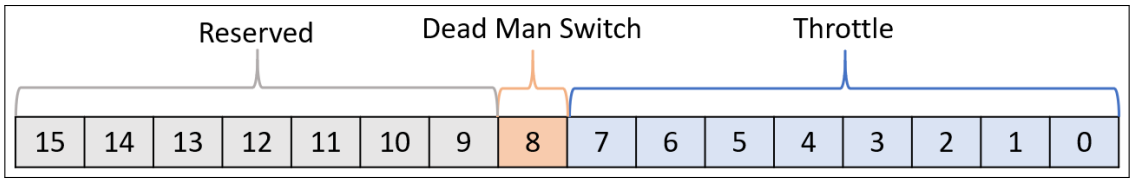

Figure 16: Bit breakdown of message transmitted from remote to board

speed controllers as described in Section 2.5.1.

There is a possibility that data could get corrupted due to various reasons such as other radio frequencies or just noise that exists in our atmosphere. To ensure the data transferred between the remote and the board we will be implementing a form of checksum. This feature is already built into the NRF24L01 and the NRF24 library. The 2 least significant bytes can be used to perform the checksum. Using CRC, cyclic redundancy check, we will be able to ensure that the packets received are not corrupt. Figure 18 clearly indicates the breakdown of data. [9] This not only adds better performance for the board, but also ensures the safety of the driver.

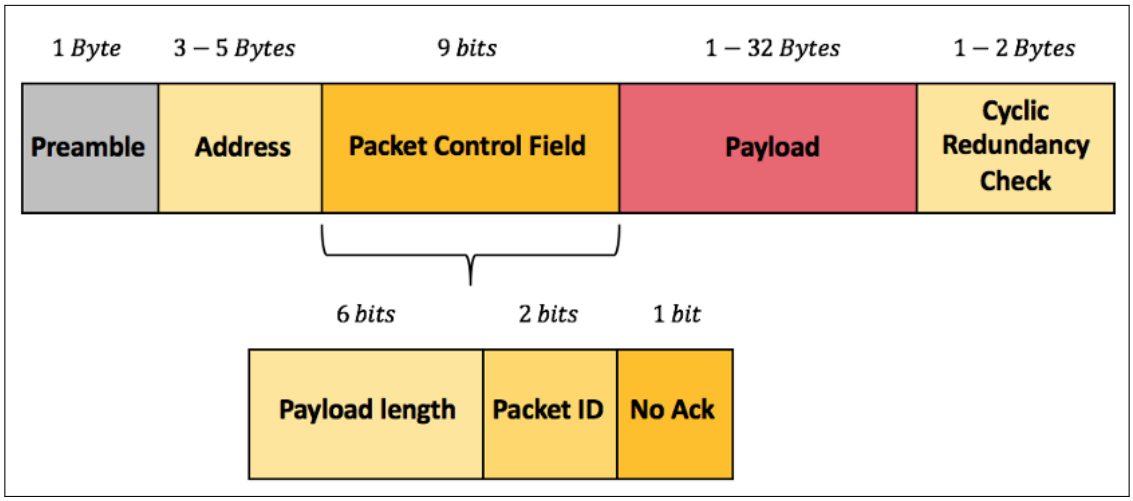

Figure 17: Breakdown of Data in Packet Structure

2.6 Commercial Component Selection

2.6.1 Rear Wheel Motors

For our electric skateboard build, we will be using 6354 brushless DC outrunner motors with integrated Hall Sensors. We chose this specific motor as it had the shortest width, while maintaining the motor power/capability that we felt was necessary for powering our electric longboard. The

integrated Hall Sensors may be used to determine RPM of motor.

2.6.2 Electronic Speed Controller

For our electronic speed controller, we will be using two individual controllers based on Benjamin Vedder's open-source electronic speed controller project. The designs for both the hardware and MCU firmware are open to the public. We chose this for our project, as they are the most common for personal electric skateboard builds, and they offer us the flexibility to modify the firmware to our needs, if necessary.

2.6.3 Board Battery & BMS

For our battery, we chose to custom build our battery using 18560 Samsung 30Q cells in a 12s4p format. We chose this as it was the most economic blend of capacity, discharge rate, and size. Samsung 30Q cells support a capacity of 3Ah, a nominal voltage of 3.6V, a continuous discharge rate of 15A, while being 18mm in diameter and 560mm in length. This means our full pack will support 60A continuous battery current & 518Wh total battery energy, which for an average user user of 18Wh/mile [10] equates to 30 miles of range. Further, our total battery size will be 8.7 inches by 5.5 inches by 2.0 inches, which can fit comfortably on the underside of our longboard deck. Additionally, we chose an LLT Smart BMS to protect our battery, as it comes integrated with a bluetooth module, allowing us to monitor the parallel group voltages and configure its charging/balancing behavior.

2.7 Tolerance Analysis

2.7.1 RPM Sensing Timing

The max speed of our board ($V_{board,\,max}$) is 30 mph, and we also know that our wheel diameter (d_{wheel}) is 85 mm. We can use these two values to determine the max angular frequency as seen below.

$$V_{board} = r\omega \implies \omega = V_{board}/r = (30mph)/(85m/2) = (13.411m/s)/(0.0425m) = 315.6\,rad/s \quad (2)$$

From there we can calculate the max frequency of wheel rotation.

$$f = \omega/2\pi = (315.6 rad/s)/2\pi = 50.229\, hz \tag{3}$$

As mentioned before, a full rotation of the wheel corresponds to 6 rising edges or in other words 6 triggered interrupts. We also need to keep in mind that there are two wheels so this will double the frequency of when an interrupt occurs. Altogether, we can calculate the frequency of an interrupt from the Hall-sensing system.

$$f_{interrupt} = (50hz) \times (6) \times (2) = 0.602\, khz \tag{4}$$

We can find the minimum time between interrupts ($t_{crossing}$) using the equation below.

$$t_{interrupt} = \frac{1}{f_{crossing}} = 1.658\, ms \tag{5}$$

Our microcontroller, the ATmega328, will be running at a clock frequency of 16 Mhz (f_{clock}). Given the sampling frequency calculated, we can determine the number of clock cycles until the next crossing of the Hall sensors. Using the equation below, we find that this will be 26550 clock cycles.

$$n_{clock\,cycles} = f_{clock} \times t_{interrupt} = 16Mhz \times 1.658ms\,\mu s = 26550 \text{ clock cycles} \tag{6}$$

The ISR is triggered by the rising edge of a pin connected to a Hall sensor's output, and all the ISR would have to do is record the current time, find the difference from the last interrupt time, and read the state of the Hall B sensor. Including the overhead when using an interrupt, this time will be negligible.

2.8 Cost Analysis

The total cost for parts as seen below in Figure 18 before shipping is $945.00. 5% shipping cost adds another $47.25 and 10% sales tax adds another $94.50. We can expect a salary of $40/hr×2.5 hr×60 = $6000 per team member. We need to multiply this amount with the number of team members, $6000× 3 = $18,000 in labor cost. This comes out to be a total cost of $19,086.84.

Description	Manufacturer	Quantity	Extended Price	Link
IC BATT CHG LI-ION 1CELL 8SOIC	Analog Devices Inc.	3	$14.82	Link
IC REG LINEAR 3.3V 150MA SOT23-5	Texas Instruments	3	$2.04	Link
THERMISTOR NTC 10KOHM 3977K BEAD	Vishay Beyschlag/Draloric/BC Components	3	$2.31	Link
CONN HEADER VERT 2POS 2MM	JST Sales America Inc.	3	$0.51	Link
BATTERY LITHIUM 3.7V 500MAH	Adafruit Industries LLC	1	$7.95	Link
PROTO BOARD ADAPTER SMT SOT-23-5	Capital Advanced Technologies	3	$7.11	Link
SOCKET ADAPTER SOIC TO 8DIP	Aries Electronics	3	$7.68	Link
CAP CER 1000PF 50V X7R RADIAL	Vishay Beyschlag/Draloric/BC Components	1	$0.20	Link
CAP CER 22PF 200V NP0 RADIAL	KEMET	10	$1.39	Link
CAP CER 47PF 50V C0G/NP0 RADIAL	Vishay Beyschlag/Draloric/BC Components	1	$0.24	Link
CAP CER 100PF 100V X7R RADIAL	Vishay Beyschlag/Draloric/BC Components	1	$0.21	Link
RES 4.12K OHM 1/4W 1% AXIAL	Yageo	1	$0.10	Link
RES 1K OHM 1/4W 1% AXIAL	Yageo	5	$0.50	Link
RES 10K OHM 1/4W 1% AXIAL	Stackpole Electronics Inc	5	$0.50	Link
RES 0.05 OHM 5W 5% RADIAL	Stackpole Electronics Inc	1	$0.76	Link
10PC DISCRETE 0805 TO 300MIL TH	Chip Quik Inc.	1	$4.90	Link
RES 4.7K OHM 1/4W 1% AXIAL	Stackpole Electronics Inc	1	$0.10	Link
IC MCU 8BIT 32KB FLASH 28DIP	Microchip Technology	4	$10.56	Link
CRYSTAL 16.0000MHZ 20PF TH	ECS Inc.	2	$1.38	Link
CRYSTAL 8.0000MHZ 18PF TH	CTS-Frequency Controls	2	$0.72	Link
Flipsky Upgrade 6354 190KV Brushless Motor	Flipsky	2	$166	Link
Complete 36T Kegel Pulley System	M Boards	2	$80	Link
TORQUE6 ESC	Torque Boards	1	$135	Link
ChiBatterySystems Custom 18650 Battery Packs	ChiBatterySystems	1	$270	Link
MBoards Extended Trucks	MBoards	1	$50	Link
Caliber II 63mm Motor Mount	Boardnamics	2	$50	Link
Caguama, 85mm	Loaded	1	$68	Link
Loaded Jehunion (JEHU) V2 Bearings	Loaded	1	$28	Link
Vanguard Deck	Loaded	1	$0.00	Link
5mm LED Diodes Kit	MCIGICM	1	$4.99	Link
Buzzer 5V	Adafruit	2	$1.90	Link
HATCHBOX PLA 3D Printer Filament	HATCHBOX	1	$22.99	Link
Rare Earth Magnets, 10 Pc	Harbor freight	2	$5.58	Link
Hall Effect Sensor	Esooho	1	$7.30	Link
Velostat	Adafruit	1	$5	Link
Copper Foil Tape (1inch)	LOVIMAG	2	$22	Link

Figure 18: Itemized list of Components and Costs

2.9 Schedule

Week	Task	Person
February 21st- February 28th	Order parts for prototyping	Everyone
	Start board assembly	Alex
	Front wheel sensing and force sensor prototype	Pat
	Research on transceiver communication	Pouya
February 28th - March 7th	Continue board assembly	Alex
	Print first versions of 3d printed prototypes to secure electronic components.	Pat
	Establish transceiver communication, encoding messages	Pouya
	Make BOM and order parts from digikey	Everyone
March 7th - March 14th	Finalize 3D prints	Pat
	Gather data on timing w/ oscilloscope, figure out how to efficiently read/write to registers, hall sensor circuitry (no-module)	
	Finish two-way transceiver communication	Pouya
	Integrate transceiver communication w/ hall sensing Prototype microcontroller ATmega328 @ 3.3V and 5V	Pat & Pouya
	Start PCB Design	Everyone
	Finish board assembly, finish BMS assembly	Alex
March 14th - March 21st	Finish PCB Design & Pass Audit	Everyone
	Finalize microcontroller ATmega328 @ 3.3V and 5V prototype	Pouya
	Integrate weight sensing, prototype remote w/ circuitry inside, battery life testing	Patrick
	Establish Communication with one vesc	Alex
	PCB ORDER MARCH 18TH	Everyone
March 22th - March 28th	Establish Communication with two vescs	Alex
	Finalize remote enclosure/ 3d printing assembly	Patrick
	Revisions to PCB Design	Pouya
March 28th - April 4th	Integration of vesc communication w/ existing integrated parts	Pouya & Pat
	Revisions to PCB Design	Alex
	PCB ORDER MARCH 26TH	Everyone
April 4th - April 11th	Finalize Assembly	Pat & Pouya
	Revisions to PCB Design	Alex
	Integration Tests	Pat & Pouya
	PCB ORDER APRIL 5TH	Everyone
April 11th - April 18th	Fix Existing Minor Bugs	Everyone
April 19th	Demo	Everyone

Figure 19: Schedule for Project Progression

2.10 Risk Analysis

The nature of riding electric skateboards is inherently dangerous, and requires a skilled, attentive operator at high speeds. Nearly every subsystem of the electric longboard is integral to upholding the safety of the user. When focusing specifically on components which we aim to build ourselves, the most important among them is the main microcontroller. It is ultimately responsible for directing the throttle of the electronic speed controllers, which in turn direct the powerful motors. If our microcontroller fails to perform this duty properly, control of the longboard is lost, potentially leading to serious bodily harm to users or pedestrians. To reduce this risk of injury, we will incorporate the following safety principles into our design. Should the wireless connection to the remote control fail, our microcontroller must disengage the motors entirely - not braking if the user is still riding, as this can upset the balance of the user. Because we are using belt-driven motors, our longboard will coast with high resistance, slowing down gradually over a small period of time. Furthermore, should our pressure sensors fail, a similar situation as the one just described will occur. The user will maintain his or her momentum with constant resistance, and can still use the small amount of motor power if the need arises. Finally, should our wheel revolution sensors fail, our main microcontroller will apply the wheel-slip mitigation protocol that reduces motor power over time. In turn, the user will be put again in the aforementioned scenario of coasting with resistance. In short, we will attempt to mitigate any component failures to the best of our ability by utilizing the inherent resistance of our belt-driven motors.

3 Ethics and Safety

Our project aims to improve safety features on consumer electric longboards by targeting two dangerous situations that are possible with general electric longboard designs. With this project we are increasing the effectiveness of safety standards, which correlates to IEEE's Code of Ethics Section I.1, which is to "to hold paramount the safety, health, and welfare of the public" [11]. Skateboards and longboards are known to be dangerous, as users are not secured to the vehicle. Should a user lose his or her balance, they automatically risk injury to themselves or others. Our project aims to mitigate some of the hazards present in current electric longboard designs; however, we will not completely eliminate the risks involved. To do so would be to eliminate the longboarding part of the experience entirely. With regarding the effectiveness of the mitigation we will not falsely claim that our that our system cannot prevent all accidents, which falls under IEEE's Code of Ethics Section I.5 by being "honest and realistic in stating claims or estimates based on available data" [11].

Additionally, there are concerns present with the use of Li-Ion batteries. Lithium-Ion batteries pose a fire hazard, especially if the battery is poorly maintained or is in bad health. Fortunately, with our smart BMS, we will be able to precisely monitor our battery pack's health. We aim to follow guidelines to regulate charging and discharging of the battery to ensure the battery's health and users safety. This can prevent the fault of overcharging, which "can lead to more severe faults, such as accelerated degradation and thermal runaway" [12].

By undertaking this project, we will bring awareness to the discoveries we have made in regards to improving the safety of longboarding "to improve the understanding by individuals and society of the capabilities and societal implications of conventional and emerging technologies, including intelligent systems" as in IEEE's Code of Ethics Section I.2 [11].

References

[1] T. Conneally, "The motorized skateboard renaissance," 2014. [Online]. Available: https://www.forbes.com/sites/timconneally/2014/09/02/the-motorized-skateboard-renaissance/?sh=1a04e4055099

[2] P. Hertach, A. Uhr, S. Niemann, and M. Cavegn, "Characteristics of single-vehicle crashes with e-bikes in switzerland," *Accident Analysis & Prevention*, vol. 117, pp. 232–238, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S000145751830174X

[3] *nRF24L01 Single Chip 2.4GHz Transceiver*, Nordic Semiconductor, 2007, rev. 2.0. [Online]. Available: https://www.mouser.com/datasheet/2/297/nRF24L01_Product_Specification_v2_0-9199.pdf

[4] *8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash*, Atmel, 2015, rev. D. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

[5] L. Ada, *Li-Ion & LiPoly Batteries*, Adafruit, 2012. [Online]. Available: https://learn.adafruit.com/li-ion-and-lipoly-batteries/voltages

[6] H. J. Zhang, *Basic Concepts of Linear Regulator and Switching Mode Power Supplies*, Analog Devices, 2013, AN-140. [Online]. Available: https://www.analog.com/media/en/technical-documentation/app-notes/an140.pdf

[7] *LP3985 Micropower, 150-mA Low-Noise Ultra-Low-Dropout CMOS Voltage Regulator*, Texas Instruments, 2015, rev. SNVS087AE. [Online]. Available: https://www.ti.com/lit/ds/symlink/lp3985.pdf?ts=1614781353213&ref_url=https%253A%252F%252Fwww.google.com%252F

[8] *Lithium-Ion Battery Pulse Charger with Overcurrent Protection*, Linear Technology, 2001. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/1730fs.pdf

[9] B. Fraser, *Arduino NRF24L01+ Communications*, Medium, 2017. [Online]. Available: https://medium.com/@benjamindavidfraser/arduino-nrf24l01-communications-947e1acb33fb

[10] *esk8 Calc*, esk8. [Online]. Available: https://calc.esk8.news/#/0

[11] *IEEE Code of Ethics*, IEEE, 2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html

[12] L. Kong, C. Li, J. Jiang, and M. Pecht, "Li-ion battery fire hazards and safety strategies," *Energies*, vol. 11, p. 2191, 08 2018. [Online]. Available: https://www.mdpi.com/1996-1073/11/9/2191