Backpack E-Skate

Senior Design II - Group F

Spring 2020 - Fall 2020

Joshua Andrews - Computer Engineering

Danner De La Rosa - Computer Engineering

University of Central Florida

Table of Contents

1. Executive Summary

In recent years, many new short-to-medium-distance transportation devices have become available such as scooters, electric scooters, bicycles, electric bicycles, skateboards, electric skateboards, and hoverboards. These are all great ways to go to the neighborhood's park or to go from class to class while in school; they save time and energy, especially since some classes may be just 10 minutes apart and very far from each other. However, these devices are controlled by humans and rely on human instinct. In addition, they required an inconvenient amount of space. Imagine if all the students in a class were to ride skateboards to class; students leave their skateboards against the wall, which can make 30 to 200 skateboards a very messy process to pick up one by one.

This is a proposal for a skateboard that possesses technology which is competitive with electric skateboards on the market today. This means that its motor and battery life will follow the average standards for its class. In addition, its body will be able to fold for storage purposes, and depending on the available time, this process may be automated to reduce human effort. The board may be aerodynamic and aesthetically pleasant and exciting to look at. LEDs are a huge attraction and luxury, and so we may implement LEDs for visual purposes and functionality, such as nighttime safety.

2. Project Description

As technology advances, man's objective remains to facilitate the things that must be done. This applies to transportation as well and has been seen through history, starting with boats, then the invention of the wheel, and later planes in 1903. Nowadays, it is possible to see the use of transportation, particularly vehicles, almost everywhere. Initially, vehicles such as skates, skateboards and bicycles, required to be powered by human movement, but technology has allowed them to expand on batteries and motors capable of powering smaller mediums of transportation while carrying heavier weights.

When it comes to motorized skateboards, there is a great variety of them on the market today; their features include hub or belt motors, wheels for different types of terrain, LEDs, remote controllers, etc.

2.1 Requirements and Specifications

Before we work on the design of the Backpack E-Skate, we need to first define the requirements and specifications we will use. This includes details about the Interface, Skateboard, Motor, Battery, Sensors, LEDs, and the Bluetooth System. These requirements and specifications can be found in Table 1 below.

Table 1: Requirements and Specifications

Description	Value
Interface	
An application written in java or C# that must control the movement [forward and backwards] of the skateboard.	-
Application must display battery charge and estimated travel distance available.	-

Application must have full control of the LEDs	-
Application must have full control of the bluetooth sound system	-
Skateboard	
The board must be able to fold in half.	-
The board must be able to turn based on the riders lateral inclination.	-
Motor	
The motor must perform well with any rider under a certain weight.	250lbs
Motor top speed	15 - 20 miles
Battery	
The battery’s charge must last enough to travel a certain distance	12 miles
Sensors	

Proximity sensor(s)	5 - 10ft
LEDs	
RGB display(s)	-
Frontal and rear LEDs that provide visibility in the dark	At least 25ft
Bluetooth Sound	
Proximity distance warning sound	5 - 10ft
Entertainment audio	-

2.2 Administrative Organization

In order to make the workload fair and efficient, we decided to assign some of the subsystems as responsibilities for each group member. We are both able to assist each other with these tasks, but this will give each person something to know to work on. Table 2 below describes the breakdown of responsibilities.

Table 2: Subsystems Responsibilities Overview

Subsystems and Responsibilities	
Control	Danner De La Rosa

Power	Josh Andrews
Mechanical	Josh Andrews
Sensor	Danner De La Rosa
LED	Danner De La Rosa
Sound	Josh Andrews

Let us define what these subsystems include

Control: Subsystem responsible for controlling all other subsystems through the mobile app.

Power: Subsystem responsible for storing and distributing power to each component that requires it, such as the motor; revolves around the battery and its circuitry.

Mechanical: Subsystem responsible for the movement of the skateboard by its motor.

Sensor: Subsystem that involves any sensors that will be integrated, such as the proximity sensors.

LED: Subsystem that involves any LEDs that will be integrated.

Sound: Subsystem that involves any sounds, such as warning sounds, that will be integrated.

2.2.1 Diagrams

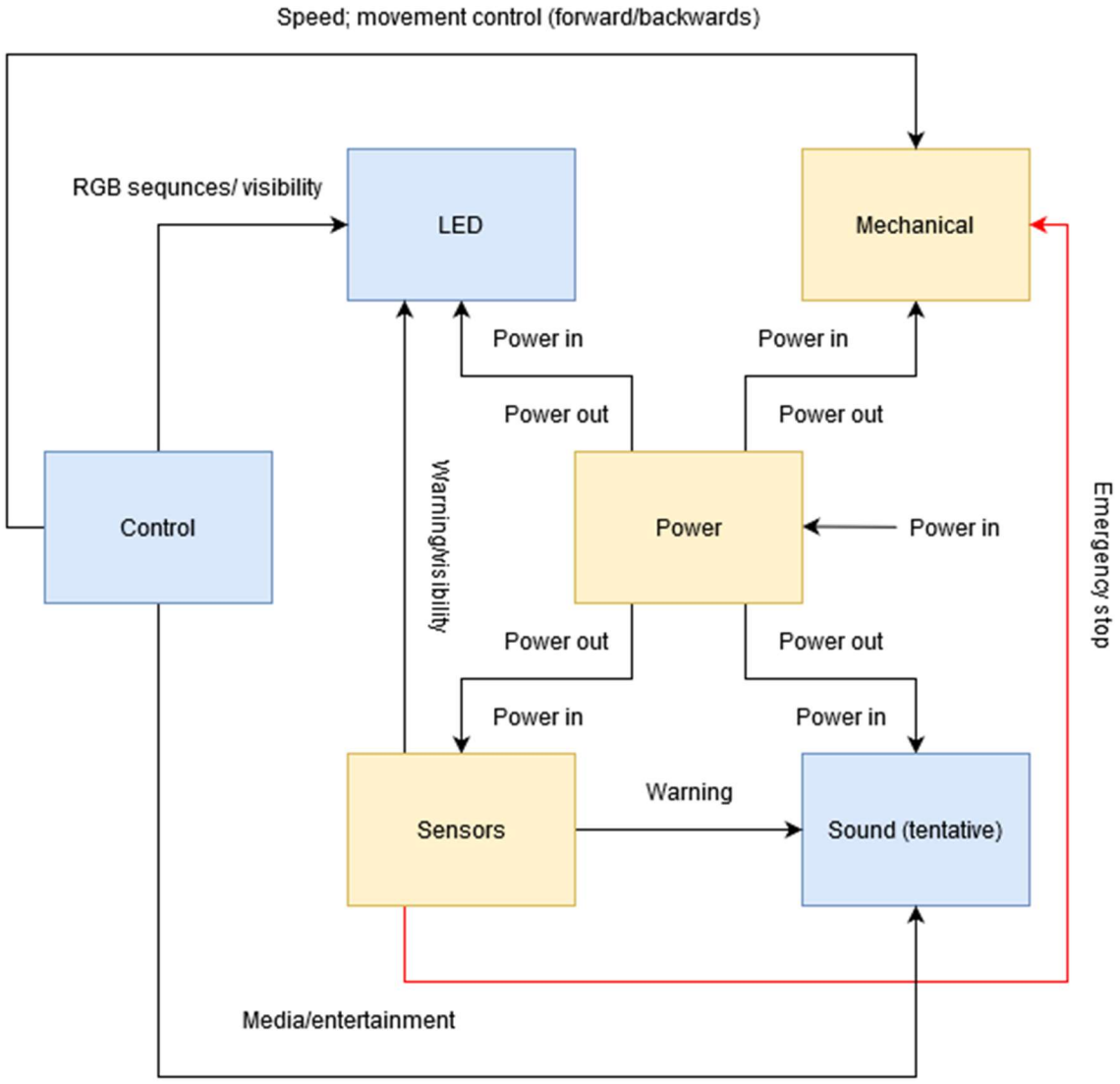


Figure 1: System design logic and responsibilities (refer to Table 2).

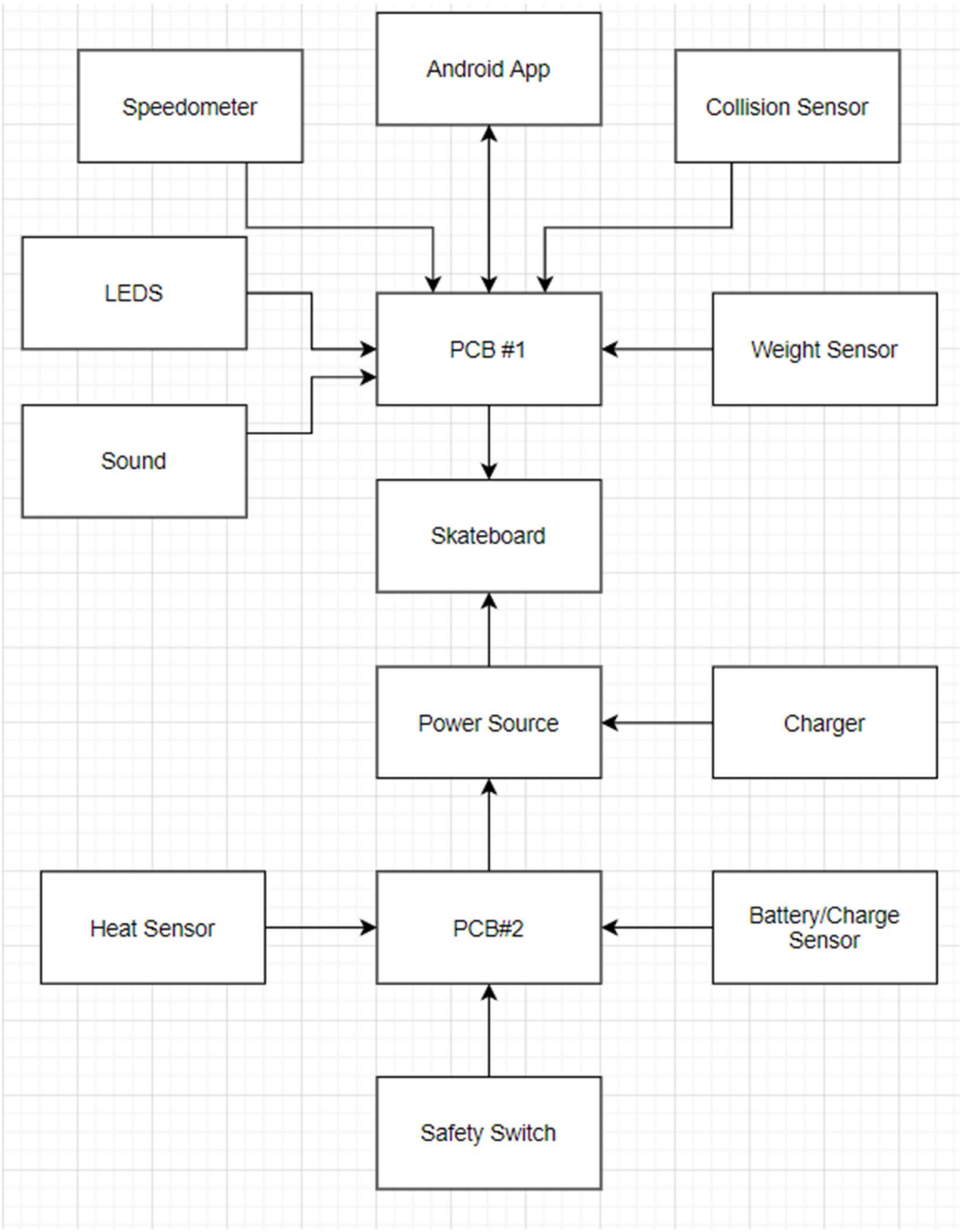


Figure 2: System design with details

Figures 1 and 2 show what we expect to need in the design of our project as of now. Figure 1 is more intended to show the responsibilities of each person. On the other hand, Figure 2 shows an estimate of the specific aspects needed within each module of the project shown in Figure 1. The “System Design Responsibilities” diagram is more permanent, the “System Design with Detail” diagram is more modifiable as we determine what components are more important to implement.

Table 3: Budget and Financing

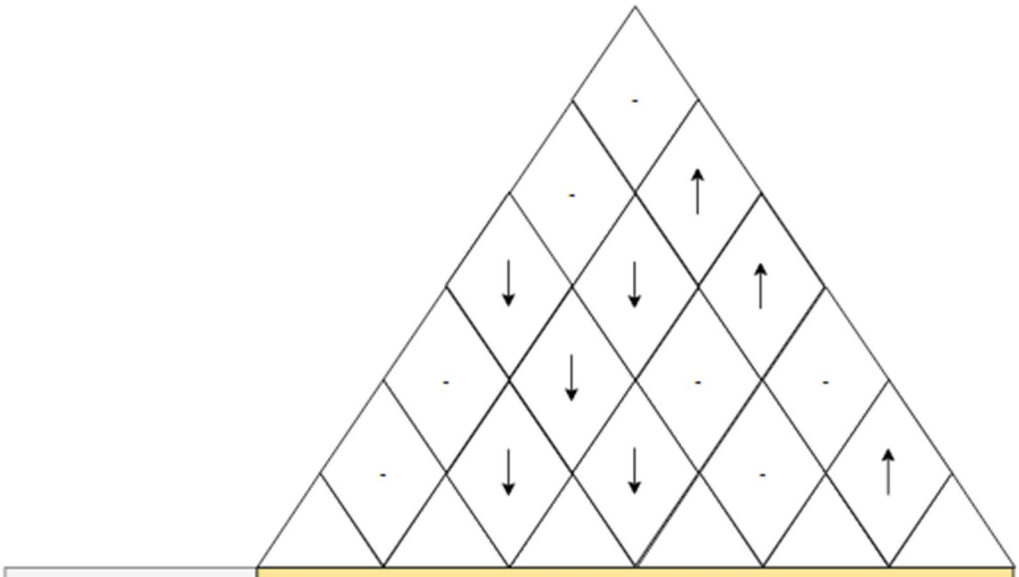
Potential Parts	Estimated Cost
Battery	$50
Microcontroller and Sensors	$30-60
Skateboard	$30-40
Motors	$20
LEDs	$2
Handheld Bluetooth Remote	$10
Beeper	$2
TOTAL	$144-184

Table 4: Senior Design 1 Milestones

Milestone	Due Date
Research components	January 2020
Code structure	
PCB design	
Finalize the list of components to be used	
Firmware development	
Order main components	
Install motor and battery	
App development	
Functionality testing	

Table 5: Senior Design I Milestones

Milestone	Due Date
LEDs installation	TBD
Sensors installation	TBD
Program LEDs and Sensors in the app	TBD
Functionality testing	TBD
Install bluetooth audio system	TBD
Program audio system in the app	TBD
Functionality testing	TBD
Final corrections	TBD



		Engineering Requirements					
		Foldable	Battery Life	Top Speed	Maximum weight	Peripherals	DEV Cost
		+	+	+	+	+	-
Safe	+	-	-	↓↓↓	-	↑↑↑	↑
Manageable	+	↑↑↑	↑↑↑	↑↑↑	↓↓↓	↓	↓
Efficient	+	-	↑↑↑	↑	↓↓↓	↓	↑↑
Powerful	+	-	↓↓↓	↑↑↑	↓↓↓	-	↑↑↑
User Friendly	+	↑↑↑	-	-	-	↓↓	↑
Low Cost	-	-	↑↑↑	↑	↑↑↑	↑↑↑	↑↑↑
Target of Engineering Requirements		2 parts	12 miles @ max weight	20 mph	250lbs	LEDs Speakers Sensors	<$200

Figure 3: House of Quality

3. Research and Part Selection

3.1 Control Systems

For this electric skateboard, one of the most important aspects is controlling the board. This includes the device on the skateboard controlling the motor speed and any peripherals. This device would be some form of microcontroller or programmable board. There also needs to be a way for the rider to drive the skateboard via remote control to wirelessly communicate with the skateboard's hardware.

3.1.1 Selecting the Microcontroller

When selecting the microcontroller, we will be using, we need to consider our needs. Since we want the skateboard to have the furthest range of use possible, it is important that most of the power is used for the motors rather than draining the battery through anything else. Therefore, the microcontroller should be low power. Because of the desire to make the skateboard foldable, it is important for the microcontroller to be small enough that it can fit underneath the skateboard, along with the battery and other necessary components. Lastly, the code that will be running on the microcontroller should not be too complicated. This should help reduce the cost requirement because it does not need to have a very fast processor. With these considerations, there are still many options available. A couple we have are the MSP430G2553 Microcontroller from Texas Instruments and the ARDUINO UNO R3.

The M430G2553 microcontroller is the first option. One key feature is that Texas Instruments has free development software for it, which is Code Composer Studio. This allows all of the code for the microcontroller to be written in C, which provides the benefit of using a familiar programming language for our group. The board advertises ultra-low power power consumption. At full use of the clocks and peripherals, the maximum power is estimated to be in the order of 100's of μW, although this can be brought down to less that 1 μW with efficient use of low-power modes when designing the code. This board costs $16.98 when ordered from Texas Instruments and the price includes shipping. However, we already have one of these microcontrollers because it was required for a previous engineering course, and therefore we would lower the development cost for the project by not needing to buy it.

The other option is the ARDUINO UNO R3. This board also has free development software. Our group members do not have experience developing with ARDUINO, but it turns out the programming language used is just C++. With that fact along with numerous tutorials online, the inexperience with this platform should not be an issue. The ARDUINO UNO R3 is available for $13.98 on Amazon with free shipping. It is also available for $8.90 on eBay from a seller in the United States. The ARDUINO can be considered low power when compared to many devices, but it is estimated to be a bit larger than the MSP430. However, this depends on many things, including what peripherals are enabled, what I/O pins are being used, the speed the internal clock(s) are set to operate at, and the efficiency of the code running on the board.

Comparison Summary M430G2553 vs ARDUINO UNO R3

Advantages, Disadvantages, Neutral

	M430G2553	ARDUINO UNO R3
Flash Memory	16KB	32KB
RAM	512B	2KB
I/O Pins	16	14
Max Power Consumption	100’s µW	100’s mW
Language Expertise	More experience	No experience
Cost	$0 - $16.98	$8.90 - $13.98

3.1.2 Selecting the Wireless Communication Method

Before we can decide on the type of remote to use, we need to determine what kind of wireless communication protocol will be used. The most accessible options are Infrared, RF, and Bluetooth.

Infrared communication can also be used. This technology has seen many applications with things such as intrusion detection for homes and remotes for televisions, DVD players, and VCRs. To use this, an IR receiver module would be required. There are many types for this, and they are all fairly cheap. For example, one that was found compatible with the ARDUINO can be acquired for $6 and even includes the remote that the rider could use. The same module could also be used for the MSP430, but the ARDUINO has more robust support libraries for this technology. Something important to note about IR communication is that infrared operates at a much higher frequency than the other options listed. This has the benefit allowing more data to be communicated between the transmitter and receiver. However, this also means that it has the shortest wavelength, meaning the IR communication is more easily blocked by physical objects than RF or Bluetooth communication. It would be a safety issue if the rider was not accurately aiming the IR remote at the receiver, and thus the rider's controls would not be effective, or if the rider had to focus on aiming the remote accurately, which would cause the rider to be distracted.

Another option is using RF, which is Radio Frequency communication. This technology is traditionally popular with toys like R/C cars and planes, meaning that applying it to an electric skateboard is similar to known applications of RF communication. Compared to infrared, RF communication operates at a lower frequency, meaning that not as much data can be transferred and that commands between the transmitter and receiver must be simpler. However, the longer wavelength means that RF is more immune to interference and does not require line-of-sight like IR. RF can have long operating distances of 100 meters or more, but this factor is not important to this project in most situations since the person using the skateboard would be controlling it while they are riding. An RF module can be obtained for under $5 on EBay.com.

Lastly, we have the option of using Bluetooth communication. As it turns out, Bluetooth is actually a form of RF that operates at the high end of radio frequencies. This provides a middle ground between IR and RF communications, allowing moderate data transfer capacity while ensuring less obstruction by physical objects. If we use Bluetooth, this will require a Bluetooth module to be

added to the circuitry of the skateboard. Luckily, the same one will work for either the MSP430G2553 board or the ARDUINO UNO R3 board. For this, the HC-05 Bluetooth Module would be the best option. This is available for $8.99 on Amazon.com and it includes the cables necessary to connect with either of the microcontrollers. The same module is also available cheaper from different vendors on Ebay.com, but we would need to consider potentially long shipping times due to some vendors originating in China.

3.1.3 Selecting the Remote Control

The choice of remote control is perhaps the least restricted category within the control subsystem. The options depend on whether we choose IR communication, RF, or Bluetooth.

When it comes to infrared compatible remotes, the options are cheap and diverse. For example, an old television remote can be repurposed as a skateboard remote. The IR module mentioned in section 4.1.1 even included a generic remote, so any IR remote can be considered free.

There are also many Radio Frequency television remotes that can be acquired similarly. In addition, the remote control of an R/C car or plane could be used. The difficulty with repurposing an RF remote control, however, is that the spectrum of radio frequencies is very broad. R/C hobbyists often use their cars or planes together, and so each one must operate on a different frequency to avoid controlling each other's toy. This means that the RF receiver module would need to be sought so that it has the same operating frequency as the controller. The other wireless communication options do not have this issue because their data capacities are large enough to differentiate between devices operating on the same frequency. It is also possible to purchase a pair of RF transmitter and receiver modules to develop a custom remote control. This would cost less than $5 for the modules. However, this option would be challenging for us due to the level of hardware engineering required to build such a device.

Lastly, we have the option to use Bluetooth. Any wireless video game controller, like an old Wiimote or a wireless XBox controller, could be used for this. However, it turns out that any of the remote controls previously mentioned are unnecessary. In addition, a rider likely would not want to need to carry around something extra like a remote in order to ride their skateboard. Luckily, any potential rider should

already have something they can use, which is their smartphone. Every smartphone on the market is Bluetooth-enabled. Instead of investing in the cost of buying a remote control, we would be sacrificing the time required to develop a mobile application. This is a good compromise because a mobile application would be fully customizable so that only buttons necessary for driving the skateboard would be present. Any repurposed television remote or video game controller would have excess buttons that the rider would not need to use, resulting in confusion and an overall poor user experience. Also, the user would not have to keep track of an extra device. Our group has moderate prior experience with developing Android applications, so that is most likely what we would do if we chose this route. It is also the cheapest option because the development software, Android Studio, is free and it is free to test the application on our own Android smartphones.

3.2 Power System

The power system is responsible for the delivery of power to all the power-dependent subsystems from the battery through circuitry. The design specifications state that there will be an LED system, a sound system, and a sensor system. In addition, the power system will deliver power to the motor as well.

3.2.1 Battery

The battery selection is of great importance in the overall performance of the backpack e-skate. The battery directly affects the system's performance through the charge rate, discharge rate, and weight. It must be selected so that it powers the motor, as well as any other subsystem that depends on it, such as the LEDs and the sensors. It must also allow the rider to travel a certain total distance, which may vary depending on the rider's weight.

Sealed Lead Acid Battery

This battery is heavy and quite outdated, but it has a low manufacturing cost, which makes its retail price very cheap. Because of its weight, it is not an optimal choice for electric skateboards, although cheap ones do use this type. It has moderate energy density, as well as moderate rate of discharge. It is considered an environmental hazard due to its lead-based chemistry. It is similar to car batteries.

Alkaline Battery

Another battery which is not suitable for this particular project, alkaline batteries are not rechargeable, which makes the skateboard impossible to use every time the battery runs out. These are more practical for lower-consumption devices, such as remotes, calculators and toys.

Nickel Cadmium Battery (NiCd)

This battery type is rechargeable, but its use of cadmium makes it environmentally unsafe; cadmium is also hard to recycle. The battery itself is inexpensive and withstands high discharge rates with almost no capacity loss. It is not practical for the project due to its heavy weight.

Nickel-Metal Hybrid Battery (NiMH)

This chemistry is used on a typical AA rechargeable battery. It is not expensive and performs better than alkaline batteries in devices that drain charge at faster rates. It has both a fast charge and discharge rates. Its problem is the limited charge cycles before the battery no longer works properly; also, it is not cost-efficient.

Lithium Battery

The lithium batteries have a high number of charge cycles and deliver a lot of charge in compact weight. There are two popular types of these available, the lithium polymer batteries (LiPo) and the Lithium-Ion batteries (Li-Ion). LiPo batteries are inexpensive and provide good performance. However, in comparison to the Li-Ion batteries, its charge cycles are quite low, and its charging quality varies. Li-Ion batteries are well balanced; they allow thousands of charge cycles and are not much heavier nor bulkier. They are very expensive but offer great energy density and low discharge rate. These are used in laptops, digital cameras and cellphones.

Advantages, Disadvantages, Neutral

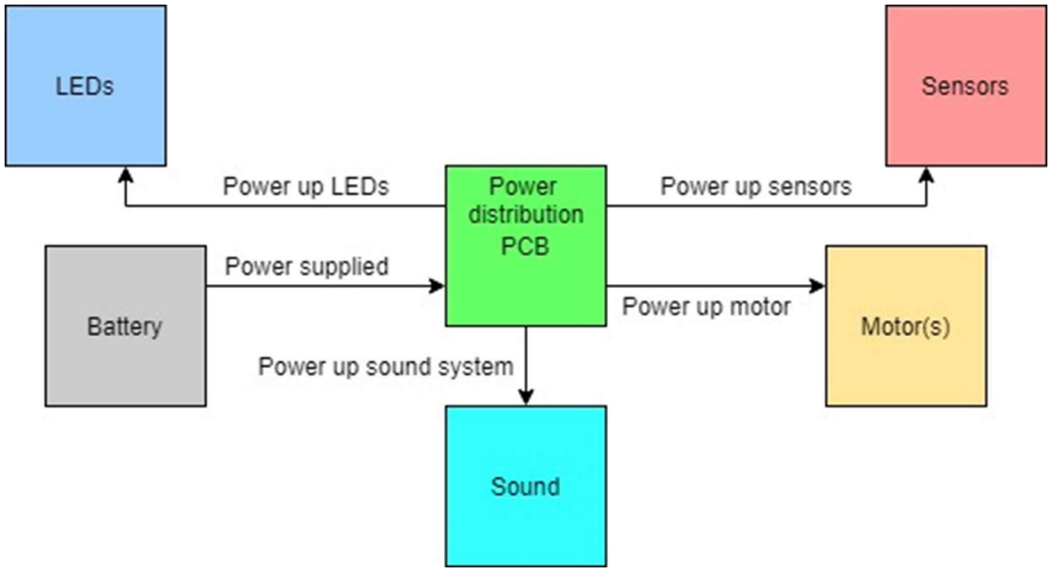
Factor	Lead - Acid	Alkaline	NiCd	NiMH	LiPo	Li-Ion
Anode	Lead	Zinc	Cadmium	Metal Hydride	Aluminum	Graphite
Cathode	Lead Dioxide	Manganese (IV) Oxide	Nickel Oxide Hydroxide	Nickel Oxide Hydroxide	Copper	Lithium Oxide
Rechargeable	Yes	No	Yes	Yes	Yes	Yes
Voltage (100%)	2.23 - 2.32	1.6	1.2	1.2	3.7	4.2
Energy Density (Wh/Kg)	30 - 40	85 - 190	30	100	250 - 730	250 - 693
Power (W/kg)	180	50	150 - 200	250 - 1000	245-430	250 - 340
Unit Cost ($)	25	5	50	60	100	100
Charge Time (hr)	8 - 16	n/a	1	2 - 4	2 - 4	2 - 4

Life (cycles)	200 - 300	50	1500	300 - 500	300 - 500	500 - 1000
Temperature	-20 to 60	-18 to 55	0 to 65	-20 to 60	0 to 60	-20 to 60
Weight	Heavy	Light	Heavy	Middle	Lightest	Light

Comparison of the Different Battery Types

3.2.2 PCB

The Printed Circuit Board (PCB) behaves like any circuit board in terms of electricity. It is responsible for taking in power from the battery and distributing it to any components of the system that require power to work. PCBs provide many advantages over breadboards and any other circuitry build style; they increase the circuit's life span and its portability. In addition, manufacturing PCBs can be done with ease and allow high complexity levels and low electrical noise. Circuit duplication can also be done easily and allows more testing and experimenting and troubleshooting. The diagram illustration below provides more understanding of the subsystems connected through the PCB.



To design the necessary PBC, a simulation software will be used; its name is PCB Creator, by BayArea Circuits. Simulation will allow a very reasonable prebuild testing period, in which the circuit design is tested to make sure the currents and voltages through each node behaves as expected, reducing the need to print PCBs unnecessarily. There is also another simulation software option from Advanced Circuits, which is free if there were any complications with the first option. With more options, this process will not be delayed by complications and/or software interface unfriendliness.

Once the design has been determined, Advanced Circuits also provides PCB printing services, with a variety of options from ultra-low cost to $66 per layer as well as student discounts. In addition, first time customers also receive a 50% off offer on their first order. Another option for PCB printing is RushPCB, which offers reasonable prices. When the moment to print comes, multiple options will be tested, and the lower priced option will be used.

3.3 Mechanical Subsystem

This is the subsystem responsible for the movement of the skateboard by its motor. This subsystem also includes some physical aspects of the board, such as the way it will fold. Because of the desire for a foldable design, we will not only need a hinge, but also a locking mechanism to ensure the skateboard does not fold up while being ridden. We will also likely need to split up the electrical components so

that they fit on the two halves of the board. This will require careful thought and planning.

3.3.1 Electric Motor

Due to small light-weight design, electric skateboards on the market mostly use DC brushless out-runner motors; this maximizes energy, which is pivotal when it is limited to the capabilities of a battery. There are two conventional types of motor that can be used on the implementation of this project; these two types of motors consist of belt-driven motors and hub motors. As suggested by the name, belt motors use a belt that is connected to a pulley in order to spin the wheels. Hub motors, however, are built into the wheels themselves.

3.3.1.1 Belt Motors

A belt is made from a material capable of bending easily without breaking; it is the cheapest choice when it comes to the transmission of power between pulleys or shafts, which are not required to be at the same height. A belt works by connecting two pulleys together, one on the motor and one on the axle [intended to rotate], by looping around them, transferring mechanical energy from the motor to its pulley to the pulley on the axle, which then causes the axle to rotate. A typical characteristic of belt-pulley systems is the ability to change the speed by moving the belt to different-sized gears. This system runs smoothly and quietly, while providing shock absorption for the motor, loads and bearings. In addition, it prevents the system from experiencing jams and requires no lubrication.

The power transmission of the belt-pulley system is defined by the following equation:

$$P = (T1 - T2)\, v$$

Where T stands for the tension in each side of the belt and v stands for the velocity at which the belt moves.

Tension is a variable that directly affects the power transmission. A loose belt consists of lower tension and will slip, failing to rotate the pulleys; a very tight belt will increase the tension, and therefore the stress on the belt and pulleys, lowering durability. It should be regulated based on a specific system and factors such as pulley size, distance, and speed.

The tensions of the belt are related by the equation:

$$T1/T2 = e^{\mu s \beta}$$

Where μ stands for the coefficient of friction and β stands for the angle in radians at which the belt contacts the pulley.

Figure 3.3.1.1.1: Wheels powered by belt motors

3.3.1.2 Hub Motors

Hub motor systems are based on the electromagnetic field provided to the winding of the motor, which causes the exterior of the motor to rotate by following the field; for this type of motors, the exterior is the wheel itself, which automatically makes it rotate. Hub motors is an emerging technology, and two different configurations

are used frequently, inner-rotation and outer-rotation. In inner-rotation motors, the rotor is found inside the stator, which is the traditional way of motor design; outer-rotators are the exact opposite. Like other electric motors, hub motors have great torque at the start, which make them perfect in the automobile industry, since automobiles use the most torque at the start as well.

The ability to have a motor at each wheel allows more stability control; this is attained by the resulting increase in acceleration and braking. In addition, the hub system requires less components, such as axles, gears, pulleys and belts, significantly reducing weight, which is advantageous for performance and manageability. However, it is true that handling and ride decrease with hub motors due to the unsprung weight at the wheels, but companies are currently addressing this problem.

Figure 3.3.1.2.1: Wheels powered by hub motors

3.3.1.3 Belt Motors vs. Hub Motors

Advantages, Disadvantages, Neutral

Factor	Belt Motors	Hub Motors
Technology Age	20+ years	2+ years
Maintenance	Unprotected moving parts	Moving parts protected by the wheel's hub
Foot-Power Option	Excessive friction created by the belt makes this option difficult	Very little friction provides an experience like a regular skateboard
Acceleration	Pulleys' system increases torque, and, therefore, acceleration	1 motor per wheel could allow competitive acceleration
Breaking	A small disadvantage on braking control	Great braking control
Incline	Better for climbing hills due to better torque	Varies based on choice and number of motors
Wheels	Most wheels allow pulleys, which increases design and aesthetic possibilities	Only hub motor wheels, a limiting factor for design, look, and ride.

Heat Dissipation	Less heat dissipation due to the separation of the motor and the wheels	Much more heat dissipation and possible safety impact
Durability	Increased durability due to less heat dissipation	Heat dissipation and vibrations shorten durability
Noise	Slight noise by the belt running with the gears	Enclosed motors make less noise for almost every manufacturer
Weight	More weight caused by pulleys, gears and a bigger motor	4 [smaller] motors result in less weight
Portability	Depending on the number of folds allowed, portability may decrease	High portability regardless of the number of folds
Weatherproof	Exposure to dust and moisture	Less vulnerable to dust and moisture
Efficiency	Less efficiency due to the gears, pulleys and belt.	The motor is close to 100% efficiency since all its power goes to the wheel
Ride	Wheels ride smooth and have more suspension	Stiffer overall drive

Figure 3.3.1.1: Comparison between Belt Motors and Hub Motors.

3.3.2 Selecting the Wheels

This section will show the research process to determine the type of wheels that will be used for our electric skateboard. The board will primarily be used for cruising, such as when a student is riding between classes, and it is not intended for doing tricks. With this in mind, longboard wheels seem like the obvious choice, but there are still many aspects to decide within that category.

The first thing to look at is the size of the wheels. The tiny wheels on a traditional skateboard are notorious for getting stuck or tripped up by even small rocks or cracks in the pavement being ridden on. With that in mind, longboard wheels are typically between 64mm and 80mm in diameter. Smaller wheels have a faster acceleration and lower top speed while larger wheels have slower acceleration with higher top speed. Larger diameter wheels are also better at ignoring terrain and riding smoothly. In addition, for our purposes, the larger wheels have the benefit of providing more ground clearance, useful considering the electronics that will be taking up space underneath the board. The main downside of choosing larger wheels is whether they will fit on the board. Wheels beyond a certain diameter, about 65mm, either need taller trucks than normal or cut outs on the deck for the wheels so that they do not touch the underside of the board.

Next is the contact patch. This refers to the surface of the wheel that contacts the ground. Wider wheels provide better grip and improve stability, especially when riding on hills. They are also better for the overall life expectancy of the wheels and the speed that can be achieved because the weight of the rider and the board are distributed over a larger surface area, reducing the structural stress experienced where the inside of the wheel contacts the trucks. A downside of wider longboard wheels is a more abrupt braking experience when sliding. However, since the motors of our electric skateboard will be controlling the braking, this should not be an issue. The contact patch is also affected by the shape of the wheel. This is because wheels with rounded or bevelled edges have less of their width actually contacting the ground. Wheels with square edges, on the other hand, make ground contact over the full width of the wheel, and have the benefit of full traction possible.

Another consideration is the hardness of the wheel. On skateboards and longboards, this is measured on a scale called the Durometer A Scale. It goes from 1a, which is extremely soft, to 100a, which is extremely hard. Longboards usually fall into the range of 75a to 90a while skateboard wheels tend to be above 95a.

Soft wheels have good grip and shock absorption, which allows the rider to cruise smoothly even over rough surfaces, ignoring bumps and cracks on the riding surface. Harder wheels are faster with less grip, but hard longboard wheels still have decent grip since they are not as hard as skateboard wheels.

The last consideration for longboard wheels is the style of the core. This is important because a good core is responsible for dissipating the heat caused by friction in the bearings of the wheel. A good core is generally plastic, although some can be made from 100% polyurethane or a compound of the two. In addition to material composition is the placement of the wheel core, which can be backset, centerset, or sideset. Backset wheel cores are aligned with the inner edge of the wheel. These are stiffer toward the inner edge of the wheel and are more flexible toward the outer edge, making sliding the easiest. However, this results in deformation due to uneven wear over the width of the wheel. There are also centerset wheels, which just means the core is centered on the wheel. These have the best grip and are good for cruising. Since centerset wheels are symmetrical, they can also be flipped the other way to promote even wear. Lastly are sideset cores, which are placed somewhere between centerset and backset, allowing a balance in the qualities from each.

3.3.3 Selecting the Folding Method

A form of hinge would be necessary to complete the fold. When the skateboard is closed, the two halves of the board should be touching along the top surface of the deck, with the wheels being opposite each other. If the skateboard were folded in the other direction, the electrical components could be crushed between the two halves of the board. One or a pair of heavy-duty door hinges would most likely be tough enough for the task, but something will need to be done to ensure the skateboard does not fold beyond what is allowed. This can be done by making the skateboard thicker underneath where the hinge would be to dissipate the tension caused by pinching the board.

Due to space management underneath the skateboard, the electrical components may need to be split in order to fit onto each half of the board. This is important because some components, especially the batteries, are likely to be large as far as volume goes. Braided heat shrink can be used to stylishly protect the wires that would connect the two halves.

Finally, the whole purpose of making a foldable skateboard is to make it compact for easy transport. Therefore, when the board is folded up, it would be a good feature for the two ends of the skateboard to meet with a locking mechanism. If possible, this feature could double as a comfortable handle.

3.4 Battery and Motor: How they fit together.

If there are any two components that must work together to achieve performance standards, they are the battery and motor. It is simple; the battery powers the device, while the motor consumes the most power. As a result, it is of utmost importance that the battery can support the motor, along with the rest of the components of this system. This means that, given the power consumption of the motor, and leaving some wiggle room for the consumption of the other components, the battery must fulfill the design standards. These refer to the travel distance range and battery hours, which are highly affected by factors like the rider's weight and terrain, because of friction. In addition, it is important to keep in mind that the power consumption of the motor is proportional to the motor's speed and will ultimately affect battery life and travel distance as well.

The manufacturer should not limit the customer's experience by setting boundaries, aside from critical ones, on how to use a product; it is the manufacturer's task to provide a product that is functional and versatile. For example, if one buys a skateboard, they should decide whether they ride on the sidewalk, leveled ground, elevations, asfalt, etcetera; however, it is critical that one does not ride on medium to high grass if specified on the user's manual, as the wheels may get stuck, causing the motor to overwork and possibly preventing the skateboard from moving. Although terrain matters for performance, customers pay for a particular standard, meaning they should not have to worry about technicalities. The manufacturers worry about the standard features and capabilities; the customer decides which product standards satisfy their needs and wants.

A list of popular brands of skateboards was gathered, along with information of the capable travel distance and battery life. This is shown in Table 3.4.1 and is important because a product below customers' standards is likely to fail on the market. The e-skate must fulfill similar average travel distance and battery life capabilities to these available products.

Electric Skateboard Model	Range (Miles)	Battery Size (Watt Hours)
Acton Blink Qu4tro	22 mi	240 Wh
Aeboard AT2	13 – 20 mi	288 – 576 Wh
Aeboard GT	17 mi	504 Wh
Backfire Galaxy G2t	15 mi	216 Wh
Backfire Ranger X1	21 mi	504 Wh
Bajaboard G4	24 mi	900 Wh
Bajaboard G4X	21 mi	900 Wh
Bajaboard Pantera	28 mi	1100 Wh
Bajaboard S2	37 mi	1000 Wh
Bioboard Thorium X	50 mi	704 Wh
Boosted Board Mini S	7 mi	99 Wh
Boosted Mini X	14 mi	199 Wh
Boosted Plus	14 mi	199 Wh
Boosted Stealth	14 mi	199 Wh
Enertion Raptor 2	25 mi	432 Wh
Evolve GTR (new)	31 mi	504 Wh
Evolve GTX	31 mi	360 Wh
Evolve GT	19-31 mi	234 – 360 Wh

Exway X1 Pro	16 mi	193 Wh
Inboard M1	7 mi	97 Wh
Kaly NYC XL 2.0	24 mi	700 Wh
Kaly NYC XL 40	42 mi	1300 Wh
Kaly NYC XL 50+	55 mi	1732 Wh
Lacroix DSS50+	37.5 mi	786 Wh
Lacroix Jaws	24 mi	726 Wh
Lacroix Nazaré Lonestar	60 mi	2178 Wh
Lacroix Nazaré	37 mi	1089 Wh
Meepo NLS	14 – 17 mi	180 – 288 Wh
Meepo NLS Pro	20 mi	336 Wh
Meepo City Rider	16 mi	504 Wh
Meepo V3	11 – 20 mi	144 – 288 Wh
MetroboardX	30 mi	576 Wh
OneWheel + XR	15 mi	324 Wh
OneWheel Pint	7 mi	Not Released
Ownboard GT/AT	19 mi	504 Wh
Ownboard W2	15 mi	324 Wh
Ownboard W1S	12 – 19 mi	144 – 292 Wh

Riptide Boards	7- 14 mi	97 – 199 Wh
Starkbaord	12 mi	208 Wh
Teamgee Boards	8 – 11 mi	126 Wh
Verreal V1S	9 – 19 mi	157 – 302 Wh
Vestar Black Hawk	32 mi	630 Wh
Vestar NightFury	30 mi	504 Wh
WowGo 3	13 mi	216 Wh
WowGo 2S	10 – 22 mi	144 – 306 Wh

Table 3.4.1: A list of electric skateboard models, their travel distance and battery life.

3.5 Speed, Light, and Motion Sensors

The typical electric skateboard usually does not have sensors, but the backpack e-skate promises to have as much technology available as possible while keeping the project build affordable for the team. Sensors feel like the right move, as they provide functionalities that otherwise could not be implemented. In fact, built-in sensors could allow anything the team chooses to integrate, and the only constraint as to what will be implemented are time and funds.

A few sensor types have been chosen for this system; these include, but are not limited to a speed sensor, a light sensor, and motion sensors. The speed sensor has multiple uses, such as measuring the traveling and maximum speeds, which determine the overall power of the system created. Moreover, this sensor will be used to provide safety for both riders and pedestrians. A particular thought comes to mind about the possibility of measuring traffic through the skateboard's app, much like google maps (and perhaps using google maps) in order to determine how safe is the speed at which the rider is traveling. The light sensor will be responsible for detecting the time of the day through captured light in order to turn on/off the night visibility light, also capable of being turned on and off manually. Finally, the motion sensor(s) will be placed strategically in order to detect movement at close, and possibly unsafe, distances at which the rider may not be able to react in time. At this point, a useful function of the skateboard would be to reduce speed and possibly stop for each appropriate scenario.

There are several options available for each type of sensor. In order to choose effectively a table will be constructed for each type, showing the advantages and disadvantages for each one for important aspects, such as size, price and effective distance.

3.5.1 Reed Switch Sensors

A reed switch consists of an electrical field that works through an applied magnetic field; reed switch sensors utilize moving parts. These include a couple of wires which are normally open and do not make electrical contact. If the magnet is positioned near the switch, the contacts will close, and if it is pulled away from the switch, the contacts will open. The magnetic axis of the magnet must be parallel to the sensor.

The spec sheet of the sensor indicates the strength of the magnetic field required to activate the switch, usually around 10 to 100 gauss. It is safe to approximate a gauss to 1 Ampere Turn (AT) if the spec sheet gave the data with the latter unit. This information is useful to determine the size or strength of the magnet.

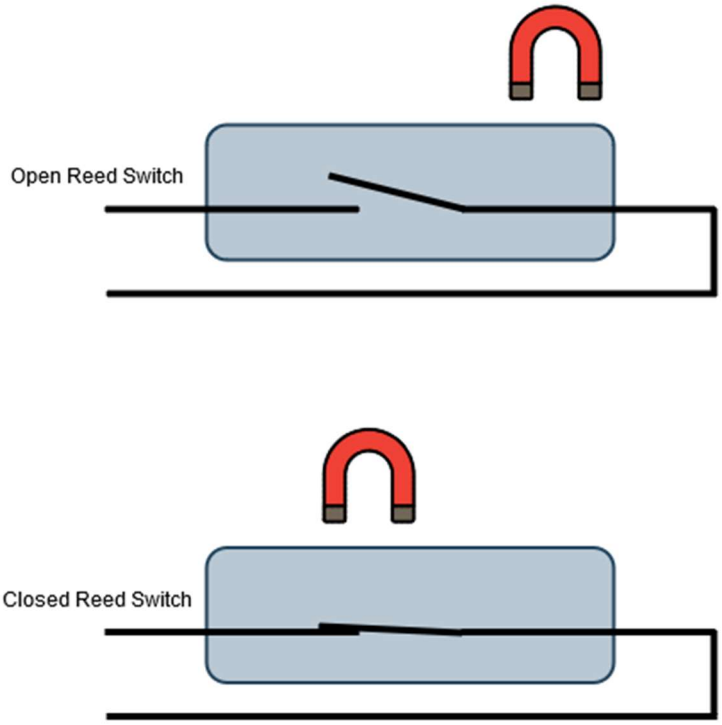


Figure 3.5.1.1.1: Reed switch stages

3.5.2 Hall Effect Sensors

Hall Effect sensors vary the output voltage as a reaction to magnetic field changes. With these, it is possible to perform the same action of reed switch sensors, but without the requirement of moving parts. Hall effect sensors are commonly used for digital applications.

Unlike reed switches, the orientation of the magnet is perpendicular to the sensor. Sensor documentation usually instructs the south pole of the magnet to point to a specific spot of the sensor, but it is recommended to verify. If the magnet is not properly placed, this may cause the sensor to not activate.

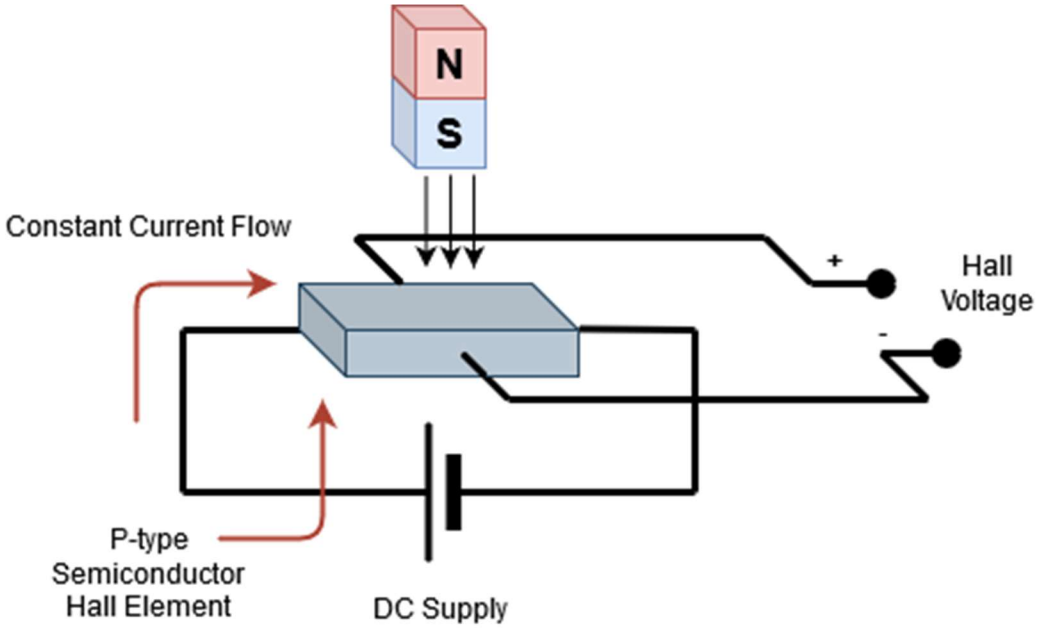


Figure 3.5.1.1.2: Demonstrates how hall effect works

The table below presents a few models that could be used for the e-skate.

Advantages, Disadvantages, Neutral

Model	Price (USD)	Size (in)	Type	Output
MP201901	4.18	1.125x0.75x0.25	Reed Switch	SPST-NO
MK03-1A66B-500W	1.88	1.00x0.23π	Reed Switch	SPST-NO
ATS668LSMTN-T	5.25	0.92x0.3x0.3	Hall Effect	Digital

MK04-1A66B-500W	1.91	0.55x0.91x0.23	Reed Switch	SPST-NO

Table 3.5.1: A table that compares different speed sensors

3.5.3 Selecting Speed Sensor

Using this knowledge, it can be determined which type of sensor fits the project best. Ideally, speed would be displayed on a small LCD and/or through a mobile application. The importance of a built-in LCD screen is mostly safety-oriented, as it is easier to look down for an instant rather than navigate through a phone app to find this information. This means that Hall Effect sensors are a better choice; the model ATS668LSMTN-T in particular is quite small, outputs digitally, working great for LCD display, and is not significantly pricier. In addition, this selection is the best considering that Hall Effect sensors can do the same thing as a Reed Switch sensor without the need for moving parts, which was previously stated.

Going back on the safety aspect, the LCD screen can be used to track other important metrics, such as sensor-captured information; these metrics can be displayed one by one at the press of a button.

3.5.4 Light Sensors

Ambient light sensors measure the intensity of visible light. They operate in a similar way to the human eye, rejecting a considerable amount of infrared. This allows light sensors to measure ambient light properly. Moreover, they measure light in continuous or single-shot modes, using features such as control and interrupt to function autonomously. Once the target ambient light is encountered, the sensor triggers wake-up events that are usually reported by an interrupt pin, converting light intensity to a digital signal output through I2C.

There are many uses for light sensors on technology today. For example, a car can turn its headlights off if the sun sets or if the car enters an area with poor illumination, such as a parking garage or a tunnel. Smartphone screens also dim and brighten according to the ambient light captured by a light sensor, allowing the

user to see at the appropriate brightness while adjusting battery power consumption.

Figure 3.5.4.1: An example of a light sensor for an automobile, located behind the rearview mirror and against the windshield.

The e-skate project will use a light sensor to automatically turn on its headlight. It is likely that the user will have an option to program the RGB LEDs to turn on automatically as well; this could ultimately end up being a default setting as it increases safety for everyone by alerting people around the area, leaving the user a possibility to toggle this option off instead. As briefly stated before, there is a possibility of the e-skate containing an informational LCD screen; if this option makes it to production, the light sensor will control its brightness to maximize power efficiency.

Advantages, Disadvantages, Neutral

Model	Price (USD)	Size (in)	Wavelength (nm)

OPT3001DNPR	1.26	0.08x0.08	550
LTR-329ALS-01	0.40	0.08x0.08	-
LTR-303ALS-01	0.40	0.08x0.08	-
APDS-9005-020	0.52	0.02x0.06x0.06	500
APDS-9306-065	0.53	0.08x0.08x0.03	560
Photoresistor	0.95	0.2 diameter	520

Table 3.5.2: A table that compares different light sensors

3.5.5 Selecting Light Sensor

According to table 3.4.2, there are no true disadvantages between the featured products. Although APDS-9306-065 seems like a clear winner, its datasheet says it is best suitable for operations behind dark glass, as it has high sensitivity in low flux conditions. The datasheet for model APDS-9005-020 claims that this sensor has low sensitivity variations across various light sources, but its applications are mostly indoors and unrelated to an extent. At last, there is a slight price disadvantage for model OPT3001DNPR, but this product satisfies all the requirements of the light sensor and its datasheet says that it is suitable for [outdoor] lighting control systems, making it the right selection for this project.

3.5.6 Motion Sensors

Motion sensors are responsible for detecting specific moving objects for a certain range of distances specified by datasheets. There are two basic types of motion sensors, active and passive. Active motion sensors, or radar-based motion detectors, emit ultrasonic sound waves, while the passive ones detect infrared waves.

3.5.6.1 Active Motion Sensors

Active motion sensors radiate ultrasonic waves; these waves repeatedly reflect back on the sensor. If this signal is interrupted by any kind of interference, the sensor triggers and sends a signal to cause an action.

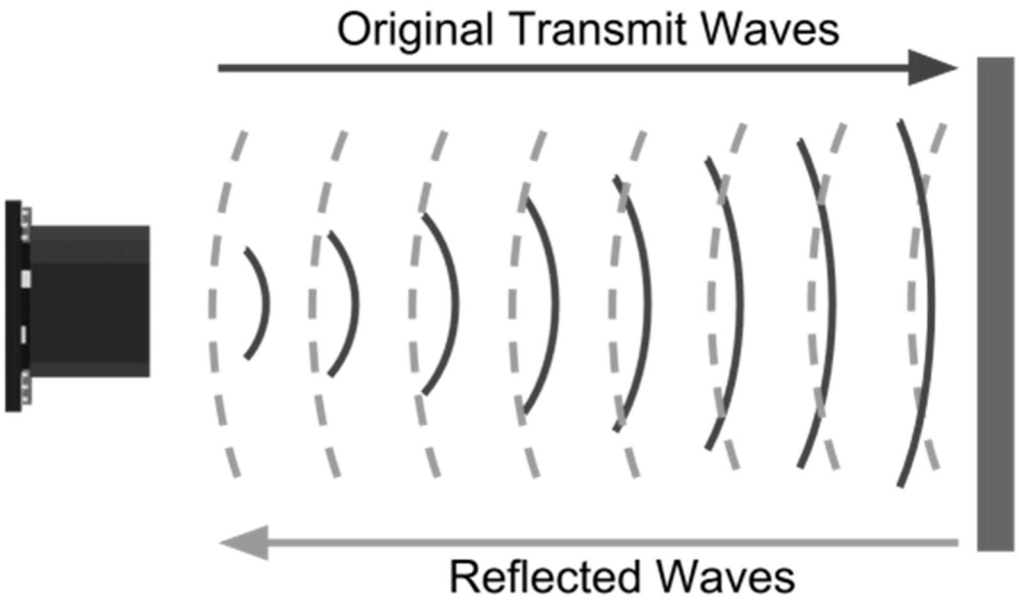


Figure 3.4.7.1: Active motion sensors emit a wave that is reflected back; objects are detected when the reflected wave is disrupted.

3.5.6.2 Passive Motion Sensors

Passive infrared motion sensors (PIR) detect infrared waves emitted by living beings in the form of heat. A sudden spike in infrared energy triggers an alarm, but the sensor does not detect small infrared variations. This type is most commonly used in residential security systems.

Passive motion sensors can also employ a technique that uses laser beams and light sensors. The light is targeted toward the light sensor and awaits an interference, at which point, the sensor registers the change in light level and triggers an alarm in response.

As given by the above information, the e-skate cannot use passive motion-sensor systems as their use cases do not apply to this project. Laser-motion-sensor systems and infrared motion sensors are for stationary uses, and the skateboard is expected to move constantly, exposing the sensor(s) to higher infrared energy fluctuations.

On the other hand, active motion sensors are the right choice because they will be able to catch signal interruptions due to objects or people within sensing range. This is not affected by the moving skateboard. Table 3.4.3 shows a variety of active motion sensor choices that will fit nicely into the skateboard build.

Advantages, Disadvantages, Neutral

Model	Price (USD)	Size (in)	Distance (ft)
IRA-S210ST01	3.12	0.19 x 0.36π	9.84
EKMC1601111	12.57	0.374 x 0.433	16.40
EKMB1393111K	12.62	0.677 x 0.587π	7.22
28032	12.99	1.42 x 1.42 x 0.79	30
HC-SR501	$1.99	1.18 x 0.98 x 0.79	<23

Table 3.4.7.3.1: A table that compares different motion sensors

3.5.7 Selecting Motion Sensor

The diagrams below show two possible builds with the sensors in table 3.4.7.3.1.

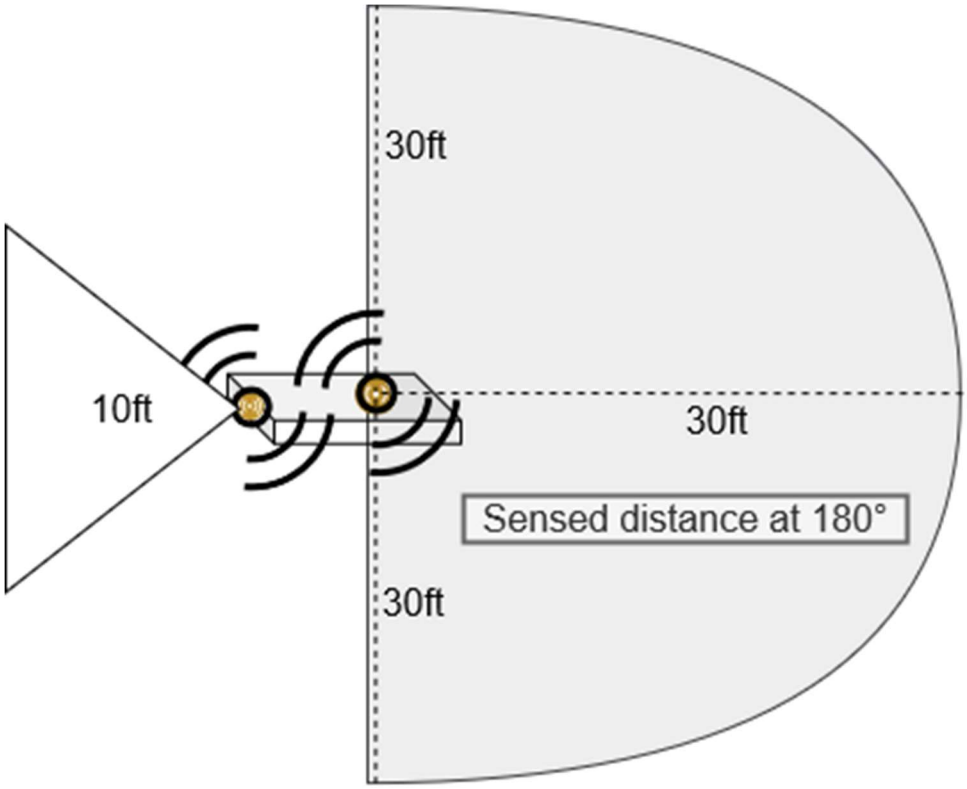


Figure 3.4.8.1: E-skate sensing +20ft in every direction but back (10ft)

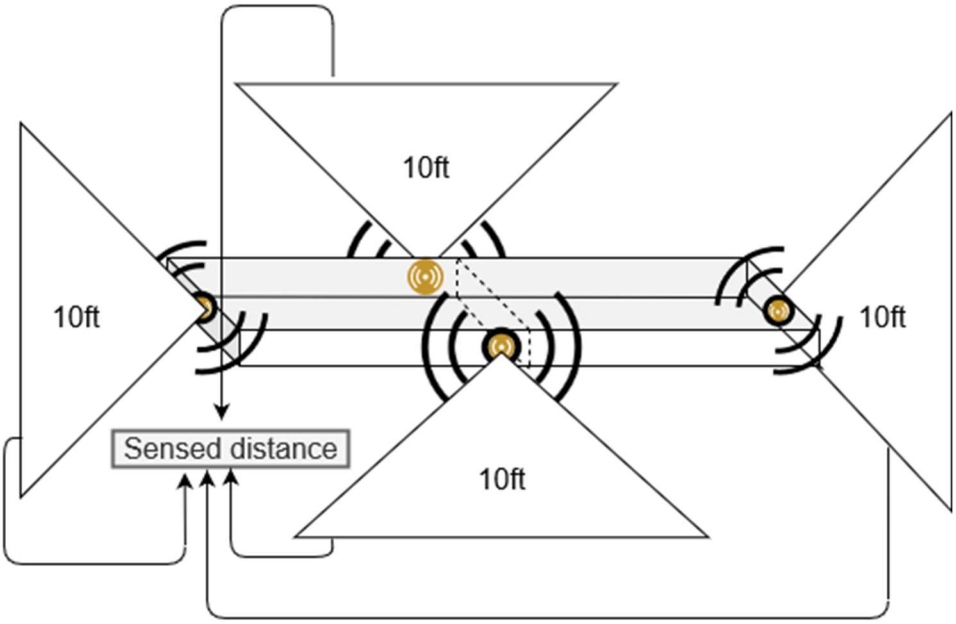


Figure 3.4.8.2: E-skate sensing 10ft in each direction

The diagrams above are a clear, visual explanation of the simplest, most effective, cost-efficient setups. By utilizing a single 28032 sensor, each side in a semicircular area will cover an area of 30ft - x, where x is the distance between the sensor and the edge of the e-skate. The back will cover the same 10ft backwards for reverse drive awareness.

The setup on Figure 3.4.8.1 has a clear advantage on distance sensed per dollar, but model 28032 is over double the size of the next biggest one. However, at dimensions of 1.42 x 1.42 x 0.79, it will not present difficulties for the rider. Ideally, this sensor would be mounted below the board to keep the top smooth, but it could also go at the front end on the top side, allowing the same distance sensing capabilities laterally and forward than under. One clear problem with sensing as far as 30 feet forward and laterally is the possibility of the e-skate stopping itself frequently; although this does not have to be the case, 30 feet sensors may just be too much for this project and not worth the additional expenses.

The setup in Figure 3.4.8.2 is a better approach; it saves 20% on cost, covers all the designated areas, and the IRA-S210ST01 sensors are much smaller. Their cylindrical shapes targeting a direct, forward path allow mounting below as well as the sensors to be pointed toward a specific area. It is also the safest design structure as far as protecting the board from user interference through movement (as all the sensors could be mounted under rather than risking having to mount a bigger sensor at the top). Finally, 10 feet is a reasonably close distance for the board to gently autostop if the rider was unaware of someone or something ahead.

3.6 LED Subsystem

3.6.1 LED Research

In recent years, almost any electric product contains LEDs; LEDs significantly boost the visual appeal and provide aesthetic customization. For the same reasons, the e-skate that will be manufactured must contain an LEDs subsystem considering the demand for it on the market today. They come in two distinct setups, single LED bulbs and LED strips, and their top-selling characteristics include brightness and durability.

Single bulbs can be used by themselves or as a group. Their objective can vary from visual effects to area illumination and visibility support, especially during nighttime. They are common on other small-distance transport electronics, such as the popular hoverboards. In addition, various automobile headlight bulbs and even flashlights are manufactured today using this setup, and some have been tested to be more durable and brighter than previous methods, making it a very dependable choice for any sort of nighttime headlight design on the e-skate.

Figure 3.5.1.1: LED bulb

Figure 3.5.1.2: A synchronized group LED bulbs

LED strips can be used to illuminate and decorate large areas and they have an aesthetic advantage over single bulbs as each strip, containing hundreds or thousands of LEDs, is considered a single unit and can have pre-programmed visual patterns that cannot be easily implemented with single bulbs. LED strips are very common in the house, usually around the crown molding and under or behind furniture. They are also popularly sold separately with an adhesive band to stick it just about anywhere. It is a good option to give the e-skate presence, as it can be installed on the bottom, and gives the surrounding crowd awareness of the approaching vehicle. The e-skate was planned out to be controlled through a mobile app; some manufactures provide smartphone-controlled LED strips.

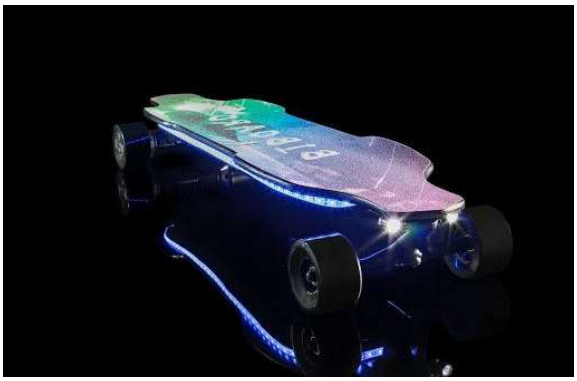


Figure 3.5.1.3: LED strip Figure 3.5.1.4: Similar and related LED strip application

3.6.2 Selecting the LEDs

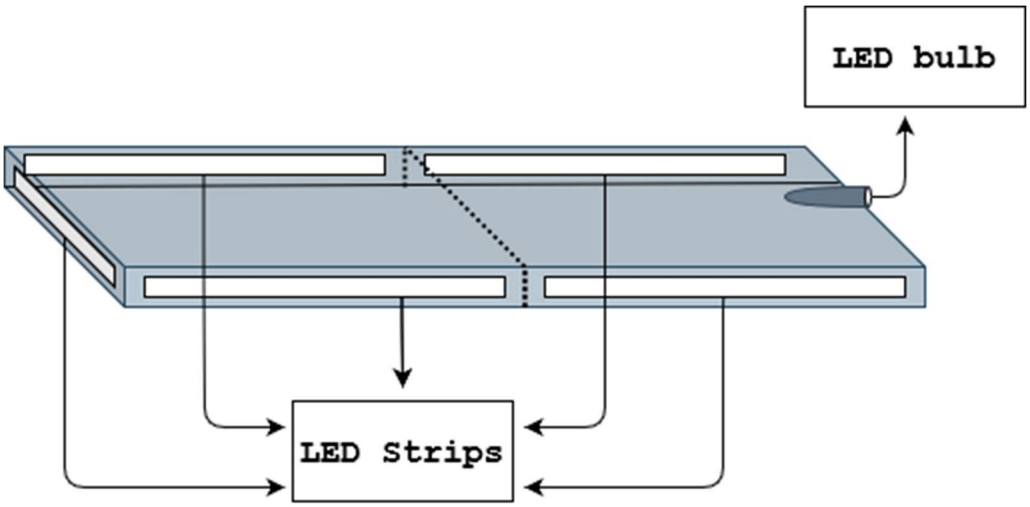


Figure 3.5.1.5: Diagram that shows desired LED strips and bulbs

The diagram above shows the LED setup. The LED strips at the sides and back provide awareness to bystanders during nighttime as well as modern aesthetics that fit well into the current trend. With that in mind, these must be RGB and must maximize value by providing the most amount of brightness available at the budget established.

The LED bulb on the front does not need to be RGB, although it is a good feature for aesthetics. The main functionality of this bulb is visibility and must cover a certain distance forward. It is possible to add lenses that enhance this distance as well. If there is an RGB bulb that can fulfil these requirements, then it will be likely used. Otherwise, bright white is the obvious choice for this case. The tables in the next page cover tentative LED choices on the market explored by the team.

Product	LEDs/meter	Luminous Efficacy (lm/W)	Price (USD)

ZFS-105000-24RGB	60	18	120.40
AB-FA01206-19700-XA2	60	90	0.76
AB-FA02408-19712-8A1	60	24	53.03
12V-SB-RGB-5M	60	-	2.19

Table 3.5.1.1: A table that shows LED strip choices

Product	Illumination (lm)	Power Consumption (W)	Size (in)	Price (USD)
1157-24SMD5050-30K	350	3	1.77x0.7π	3.43
BA15DS036CA-30VDC-27K-AI	400	4	2.1x0.8π	5.33
G4-49SMD5050-30K	600	6	1.5x0.98π	5.07
PLT-11697	500	5	2.5x0.78π	2.90

Table 3.5.1.2: A table that shows LED bulb choices

3.6.3 Motion Sensor to LEDs

An outstanding feature supported by the hardware components and that can simply be programmed is warning signals through LEDs. This is possible through a motion sensor; if the motion sensor located at the front detects a person or object ahead, it can trigger the LED strips to flash in red, as a visual warning signaling a sudden lateral move or stop is necessary to avoid crashing. Another warning color sequence would be an alternating flash in red and blue, much like a police car. Because the latter sequence is slightly more complicated to code, it will be the one used for warning purposes, demonstrating proficiency and dominance on the subject.

3.7 Sound System

3.7.1 Sound System Research

When it comes to skateboards, especially electric ones, sound systems are not necessarily rare. Other vehicles, usually for medium to long distances, have standardized sound systems as a means of entertainment, and it is important to include one on the e-skate as well if the hardware facilitates it. People should have that option made available, whether it is to listen to that song stuck on their head on a ride to the park, or under a tree at school, while waiting for a ride home. There are different types of sound systems, but the most important categories are wired or wireless, sound quality, size and power consumption.

To select the right kind of sound system, it is important to analyze the situation. For example, it is important to realize that a wired sound system would be inconvenient; connecting a wire to the phone would leave the phone loose and, therefore, insecure. Furthermore, the user would have to leave the phone on the skateboard while riding, preventing media control always. Because the rider is almost always skating when the e-skate is being used, there is a possibility to spend less on sound quality if audio is clear. Lastly, the audio system must be small and rigid to fit well and withstand uneven paths.

Unfortunately, it was not possible to implement this system, mostly because the team had two integrants only. It was necessary to focus on other things

4. Design Standards

In this section, we will discuss the industry standards that will shape our design. Whether it is a commonly-used networking protocol or a standard connector size for a cable, it is important to comply with industry standards because it affects compatibility and it is usually safe or optimal.

4.1 Battery Standard

The battery we chose has type F1 terminals, which is a standard size for that type of battery. Therefore, it is important to use type F1 female connectors on the E-Skate instead of simply soldering directly to the battery terminals. That way, new batteries can easily be installed in the future if the old ones become unreliable. Similarly, it is important to use a standard AC to DC power adapter. We would not be making a custom adapter, so that means we will need to make sure to use a common size for the connector installed in the E-Skate. IEC 60130-10:1971 is a standard for DC power connectors. It defines a Type A DC Power Connector as one with a 5.5mm outer diameter and a 2.1mm inner diameter. This is the size of the charger we chose in section 5.2.1, so we will make sure to choose a female DC power connector with the same specification.

4.2 Bluetooth and Android Standards

Because of the method of implementation, it makes sense to talk about the standards for Bluetooth and Android together. Bluetooth will be used to make communication between the E-Skate and the Android app. The default Bluetooth libraries for Android are a bit overpowered for our needs considering they can handle media streaming and we are just sending simple commands to a microcontroller. Luckily, Android 4.3 and newer versions support Bluetooth Low Energy (BLE), which is better optimized for pairing with devices with low power consumption. The documentation from the Android Developers website lists proximity sensors, heart rate monitors, and fitness devices as examples, but we expect it would be a good option considering the microcontroller used in the E-Skate. For our application, we decided to use Android 5.0 (Lollipop), which is API level 21. This version of Android will allow the application to be compatible with approximately 94.1% of devices.

5. Design

5.1 Control System

The control system will allow the rider to control the Backpack E-Skate with the Android application on their phone. The rider will be able to press buttons on the application on their phone, which will send a Bluetooth signal to the control PCB on the skateboard through a Bluetooth module interfacing with the microcontroller. The control system will also handle the logic needed to operate any sensors or peripherals, such as determining when to turn on the LED fixtures.

5.1.1 MSP430G2553 Microcontroller

The microcontroller we selected for the Control System is the M430G2553 from Texas Instruments. It has 16 kilobytes of Flash memory, 512 bytes of RAM, and two 16-bit timers. This should be enough hardware support to handle control of all the other subsystems. For development purposes, we will be using the MSP430 Value Line LaunchPad Development Tool as a development board and we will program it with Code Composer Studio, which has programming libraries for all of the microcontrollers made by Texas Instruments. After the program is developed and tested, the M430G2553 microcontroller can be removed from the LaunchPad development board and dropped into a printed circuit board.

This microcontroller is an excellent choice because it fits our needs. The microcontroller is not incredibly powerful by any means, but this just means that the programming will need to be smart. The actual requirements for the program is fairly simple as far as the necessary functions and the peripheral support. As long as we can use good memory management with the program to not exceed the RAM space, the benefit is that we do not have to use a microcontroller with more hardware and higher power consumption.

The fact that the M430G2553 microcontroller is minimalistic means that it is easy to make it compatible with simple peripherals. For example, by setting a logical high output to one of the pins, the pin can be connected to toggle an LED fixture. Similarly, motion and light sensors can be connected to input pins on the microcontroller and configured with a custom electronic circuit to define the logic voltage based on the readings from the sensors.

5.1.2 HC-05 Bluetooth 2.0 Module

We chose the HC-05 Classic Bluetooth 2.0 Serial Wireless Bluetooth Module from DSD TECH. The module defaults to slave mode, which is what it will be mostly used for, but it can also act in master mode. It has 2 megabytes of memory and a default sampling rate of 9600 baud, which can be changed. The module has six pins, which are the **Key/EN**, **VCC**, **GND**, **TXD**, **RXD**, and **State**.

The **Key/EN** pin is used to set the module to AT commands mode. When this pin is high, the module operates in command mode, and it operates in data mode otherwise. The default baud rate of the module for command mode and data mode are 39400 and 9600, respectively. Data mode is the normal mode for exchanging data between it and another device. The command mode uses AT commands through the module serial port (USART) that can change the settings for the device, such as renaming the module to something like “E_Skate” or changing the password to connect to it.

The **VCC** pin is the voltage input for the module, which can be either 3.3 V or 5 V. This voltage is relative to the **GND** pin, which is the ground voltage input. It is best to provide 3.3 V across these pins because it will allow for lower power consumption. The Bluetooth module itself has an onboard 5 V to 3.3 V regulator, so anything above a 3.3 V input is a waste.

The **TXD** pin is used to transmit serial data. It is an output, which will connect to the serial input pin on the M430G2553 microcontroller. This path will be used to send commands from the Android application to the skateboard, such as accelerating forward or reversing. The **RXD** pin is used to receive serial data. It is an input, which will connect to the serial output pin on the M430G2553 microcontroller. This path will allow the skateboard to send feedback to the mobile application. It is important to note that the microcontroller is capable of handling the 3.3 V from the **TXD** pin on the Bluetooth module. However, the microcontroller will send logic high signals of 5 V to the **RXD** pin on the Bluetooth module, so we need to shift the transmit voltage level.

The **State** pin simply reads if the module is connected or not. This can be used as a status indicator. The Bluetooth module also has a red LED, which gives a visual indication of the connection status. The LED blinks rapidly before connecting to a device and this slows down to pulse twice every two seconds when the module is

successfully paired with a device. If we can place this Bluetooth module strategically on the skateboard so that the rider can see when their Android smartphone is connected properly, it will eliminate the need to use the **State** pin for alternative indication.

5.1.3 Android Application

The smartphone application to control the skateboard will be developed in Android Studio. There are robust mobile Java libraries available for Bluetooth support, so this is a good option. To make the Android application work well with the microcontroller in the skateboard, which is relatively much lower tech than the smartphone, the commands transmitted by the application will be single ASCII characters, most likely letters of the English alphabet. By doing this, the package size of each command will only be one byte, allowing for the simplest possible transmission between the devices and the fastest possible processing by the microcontroller. Otherwise, there could be issues with the microcontroller's RAM being completely filled with commands, causing overflow errors and undefined behaviors.

Here will be described the functionalities that must be present in the Android Application. For starters, the application needs to verify that the smartphone is properly paired with the Bluetooth module in the skateboard. The connection status should appear on the first screen when the application is opened. If the connection is good, the message should display briefly and disappear to show the main screen in the application. If the connection is unsuccessful, an error message should appear on the screen and remain until the error is resolved. If there is a connection error after the application has been running, the error message should interrupt operation of the application immediately so the user can resolve the issue instead of continuing to input commands that do not work. On the main screen of the application, there should be a few buttons available so that the rider can operate the skateboard. Necessary buttons include "Forward," "Backward," and "Stop." The names of these buttons and their specific implementation might change as we discover what is intuitive and user-friendly. Forward would accelerate the skateboard forward, increasing the speed with each press until a safe maximum. If the skateboard is already moving backwards, pressing Forward should slow them down. Similarly, the Backward button should accelerate the skateboard backward, which means slowing it down if it is already moving forward. The Stop button should simply bring the skateboard to a velocity of zero, regardless of the original direction. The stopping time should be balanced so it is gradual enough to be safe, but also quick enough in case the rider is using it to avoid a collision. The Forward, Backward, and Stop buttons should take up most of the screen of the

smartphone so that they are easy for the rider to press the correct one while riding. While not necessary, it would be a nice feature to have a button in the application that can toggle the LED fixtures. This can be a small button in the corner. Optionally, this can even be two buttons, one for aesthetic lights and one for functional headlights. Another good feature in the application would be a battery indicator for the skateboard. The figure below shows a sketch of what the user interface in the Android application might look like.

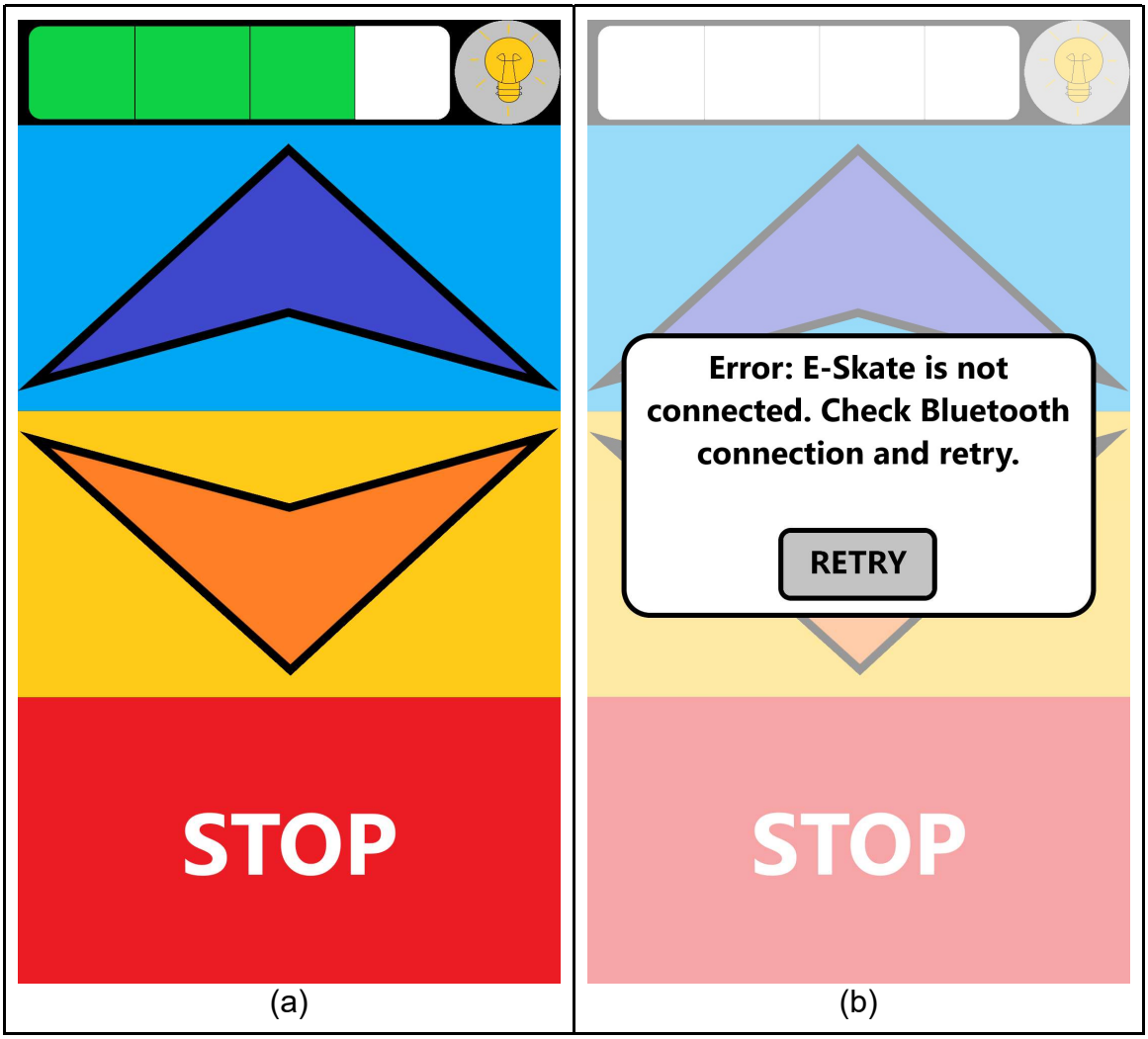


Figure 5.1.3: (a) Represents what the interface of the Android application might look like during normal operation. (b) Shows what the application might look like when there is an issue with the Bluetooth connection between the phone and the skateboard.

Also, if we want to save on our budget and reduce the loads on the microcontrollers in the E-Skate, we can eliminate the need for a separate speed sensor by using the rider's Android phone to calculate the speed in the application. In order to achieve this, most Android devices have an accelerometer and a GPS system, both of which can be used to calculate the speed. As it comes time to develop the application, we will have to see which option is better as far as accuracy and the complexity of the code.

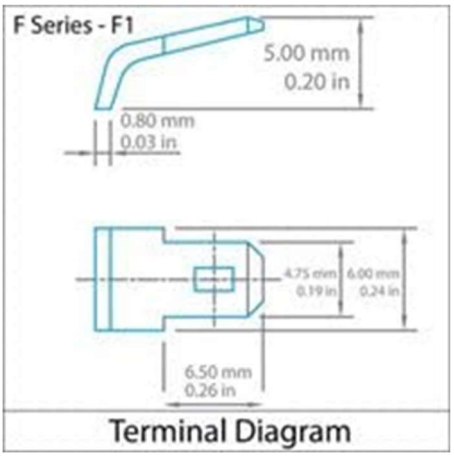
5.2 Power System

The Power System will be responsible for distributing electrical power to all other components of the E-Skate. This includes power for both PCBs, the motors for the wheels, and all sensors and peripherals.

5.2.1 Battery

For the battery we chose the UB1280 12 V battery from Universal Power Group. It is a non-spillable sealed lead-acid battery and it has a capacity of 8 Ah. By combining two of these batteries in parallel, it will have a combined capacity of 16 Ah, which should last for at least 3 hours of operation if the output current is 5 amps or less.. Each battery has a form factor of 5.94in x 3.94in x 2.56in and weighs 4.96 pounds, together weighing just under 10 pounds. This will add a bit of weight to the E-Skate, but it should be tolerable. The figure below shows the battery and a diagram for the terminals.

(a)



	(b)

Figure 5.2.1.1: (a) Shows the battery itself. It is a simple shape, a rectangular prism, so it should easily fit into the build without being awkward. (b) The battery has type F1 terminals with the dimensions shown above. This is useful to install the correct connectors so that the batteries can be replaced in the future if necessary. They do have a 1 Year Warranty.

The battery itself should not drop below 10.50 volts. Otherwise, there is a risk of damaging the battery. Therefore, the documentation for this battery recommends operating between 11.50 volts and 13.25 volts. The charger should have an output of 14.5 V to 15 V at up to 2.4 A. Therefore, the “Maxson 15V 2A Power Supply Charger” should work, which can be found on Amazon for $12. Aside from that, F1 terminal adapters can be found very cheap, which will allow us to easily connect and disconnect the batteries.

Figure 5.2.1.2: This is the AC adapter described above.

5.2.2 M430G2452 Microcontroller

The microcontroller we selected for the power subsystem is the M430G2452 microcontroller from Texas Instruments. The microcontroller has 8 kilobytes of Flash memory, 256 bytes of RAM, and one 16-bit timer. For development purposes, we will be using the MSP430 Value Line LaunchPad Development Tool as a development board and we will program it with Code Composer Studio, which has programming libraries for all of the microcontrollers made by Texas

Instruments. This microcontroller will handle power distribution of the E-Skate. The M430G2553 in the control system will transmit the velocity it calculates to the M430G2452, which will be responsible for actually controlling the speed and direction of the E-Skate by regulating the voltage applied to the motors with the Pulse Wave Modulation function. The Pulse Wave Modulation works by changing a value called the duty cycle, which will affect the RMS voltage applied to the motors. The microcontroller also has an Analog-to-Digital Converter, which will be used in combination with a voltage-divider circuit to estimate the charge of the battery.

5.2.3 BTN7971B High Current PN Half-Bridge

In order to control the E-Skate motors, a half-bridge is needed. The half-bridge can be modelled as a set of four transistors acting as switches. The figure below demonstrates this. If a high logic voltage is applied to Q1 and Q2, and a low logic voltage of 0V is applied to Q3 and Q4, Q1 and Q3 will act as open switches and Q2 and Q4 will act as closed switches. This will cause the motor to spin in one direction. If the voltages for each pair are switched, then Q1 and Q3 will become the closed switches, making the motor spin in the opposite direction. By doing this, we can achieve forward and backward control of the E-Skate.

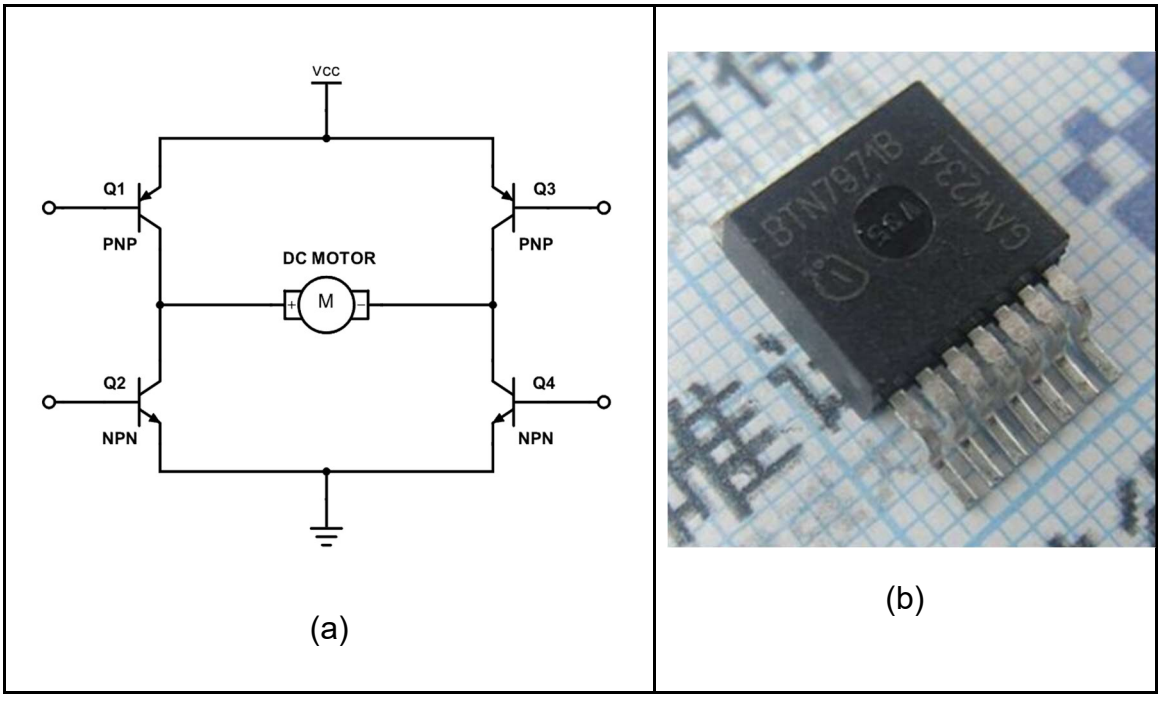


Figure 5.2.3.1: (a) Represents how a half-bridge is implemented with transistors. (b) Shows the actual component we chose.

5.3 Mechanical System

The mechanical system includes everything about the physical structure of the skateboard and what makes it move. The motor could have been discussed in the Power System in section 5.2 since it is electrical, but it is discussed as part of the Mechanical System because it drives the E-Skate by converting electrical energy to kinetic energy.

5.3.1 Selecting the Electric Motor

We have to complete more research in order to find a suitable model of motor. It should be rated for 12 V and have a good power/torque rating to handle driving fast under the weight of a rider. The easiest choice would be to choose a dual hub motor drive kit, which would include the wheels, motors, and trucks already assembled. However, the cheapest option with going this route is a kit for $120. Since we are working with a budget, this might be too much, but we will have to see if this is cheaper than buying other parts separately.

5.3.2 Wheels

We chose 97mm Flywheels for the E-Skate. They are normally $75 for a set of four, but they are currently on sale for $30, which is a great deal considering most longboard wheels go for much more. As the name implies, these wheels are big at 97mm in diameter, which is almost 4 inches. Combined with good trucks, they should provide plenty of ground clearance to account for all of the electronics under the board. They are also fairly wide at 52mm and they have a hardness of 78a on the Durometer A Scale, meaning they should provide a smooth riding experience. We do not have a selection for the trucks yet. It is difficult to find ones that will fit the necessary motor components without buying a very expensive kit, so we will continue to research this.

Figure 5.3.2.1: These are the wheels we chose.

5.3.3 Board Design

For this, unless we can find a good deal on a used board, the best option is that we construct our own deck. By constructing our own board, we can design it with the exact dimensions and shape we want. Later on, if we do go this route, we will provide a schematic diagram here for the board. Longboards are typically about 10 inches wide and at least 30 inches long. The largest electrical components that will be attached under the board are the batteries. Since they are 3.94 inches wide and we have two, it makes sense to place them next to each other with the width of the batteries aligned with the width of the board. We can make the board wider as needed, but 10 inches is enough to have 2 inches of variance. Other than this, we also need to consider the folding mechanism of the E-Skate. In order to make it work, the board will need to be thicker where the two halves meet. Otherwise, a rider standing on the board will apply a lot of pressure to the folding axis and possibly cause the board to snap or fold while ridden.

5.4 Sensors

There are three types of sensors we plan to implement into the design of the E-Skate: speed sensor, light sensor, and motion sensor. We already mentioned in section 5.1.3 that we will be using the Android application to control, measure and display the speed of the E-Skate. However, depending on what we discover during testing, we might find it better to implement a speed sensor in the electronics of

the E-Skate. If we determine it is the better choice, we will use the ATS668LSMTN-T sensor. It is a Hall Effect sensor with a digital output, which would be easy to transmit to an LCD.

For the light sensor, we chose the OPT3001DNPR for Texas Instruments. As expected from TI, the datasheet for this part is very thorough. With the documentation, we were able to determine that it fit our build best despite being more expensive than the other options we researched. Above all, its detection range is very similar to the portion of the electromagnetic spectrum visible to humans, meaning it will not easily be tricked into thinking it is bright at night because it detects infrared radiation. By implementing one of these sensors into the build, we can have the LEDs on the E-Skate be controlled automatically.

For the motion sensor, we chose to go with the IRA-S210ST01 from Murata Electronics. These are cheap at about $3 and provide a detection range of about 10 feet. By using four of these, one facing each direction of the E-Skate, we can achieve a 10-foot radius around the E-Skate to detect pedestrians. If we can use the signals from the sensors to force the E-Skate to slow down, this can be a good safety feature to prevent collisions if the rider or a pedestrian is not paying attention. Other sensors provide longer range, but a longer range can actually be less desirable. If the E-Skate is constantly slowing down because it detects someone 30 feet away, this is a bit excessive and can cause a poor experience for the rider. The 10-foot radius is a better balance to only detect obstacles that are likely to pose a risk of danger.

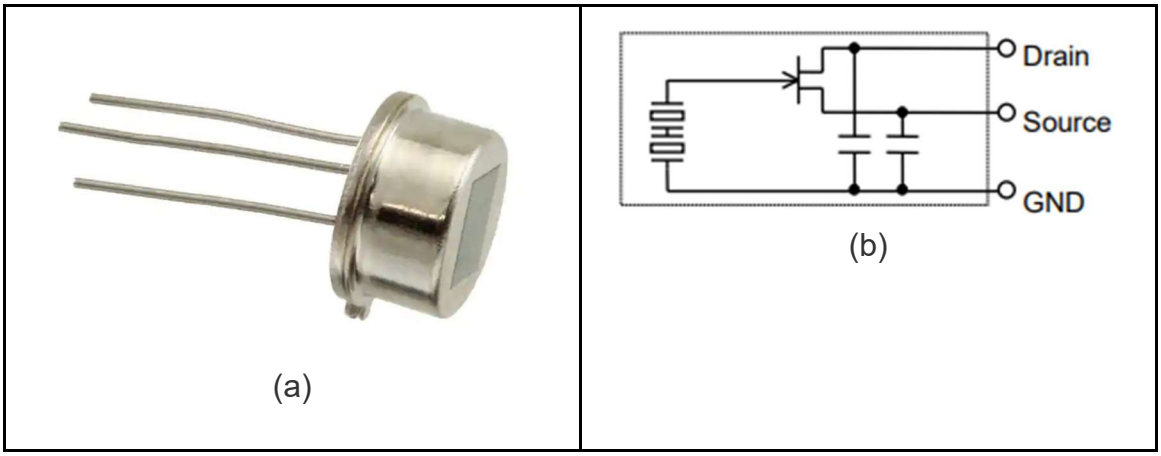


Figure 5.4.1: (a) Shows the actual motion sensor. (b) Shows the internal circuit diagram of the sensor, which is a transistor with infrared ray-receiving electrode.

If we can figure out a good implementation, we are also considering to include a pressure sensor pad as a fourth sensor type. At a minimum, this could improve safety by detecting whether a rider is on the E-Skate or not. This can allow the E-Skate to automatically stop in case the rider bails. At the best case scenario, the pressure sensor can even be used to control the speed of the E-Skate based on how the rider leans forward. This would increase safety even more by eliminating the need for the Android application.

5.5 LEDs

When deciding on what LEDs to use, we had to choose for two different purposes: one type prioritizes style, and the other prioritizes function.

First, we chose the AB-FA01206-19700-XA2 LED RGB Mod strip from American Bright Optoelectronics Corporation. This one will wrap around the E-Skate, providing a variety of RGB lighting. While the goal of this piece is mainly aesthetic, they also act as a safety feature to make the rider and E-Skate more visible to other people at night. This particular model was chosen for being the most economical option.

Next we chose the PLT-11697 LED, which is a single bulb with high luminosity. It is cheap and bright, so it was an easy choice. This bulb can be combined with a reflective housing in order to act as a headlight for the rider at night, providing a beam of light in front of the E-Skate's path. This will improve safety for the rider since they will be able to see where they are going even at night.

Figure 5.5.1: This image shows the LED bulb we chose with the size included.

5.6 Sound

All things considered, the sound system is a difficult task at the moment. The microcontrollers being used do not have powerful enough hardware to handle playing audio. After the regular program code is uploaded to the M430G2553 microcontroller, there will not be enough space to store audio files locally. Furthermore, the bit-rate required to stream audio through the Bluetooth module would interrupt the communication, possibly delaying some driver commands from transmitting to the E-Skate. As of now, the only way to make audio possible would be to modify an existing Bluetooth speaker and integrate it into the build to be powered by the batteries.

5.7 Design Summary

Here is a list to summarize the parts we chose:

- M430G2553 and M430G2452 microcontrollers
- HC-05 Bluetooth 2.0 Module
- Android application made in Android 5.0
- UB1280 12 V Battery (x2)
- Maxson 15V 2A Power Supply Charger

- BTN7971B High Current PN Half-Bridge
- Motors (TBD)
- Trucks (TBD)
- 97mm Flywheels (x4)
- Longboard deck (or lumber to construct our own)
- MSP430 Launchpad Development Kit
- 6-Pin Dupont Connector Cable
- IRA-S210ST01 motion sensor (x4)
- OPT3001DNPR light sensor
- AB-FA01206-19700-XA2 LED RGB Mod Strip
- PLT-11697 LED Bulb

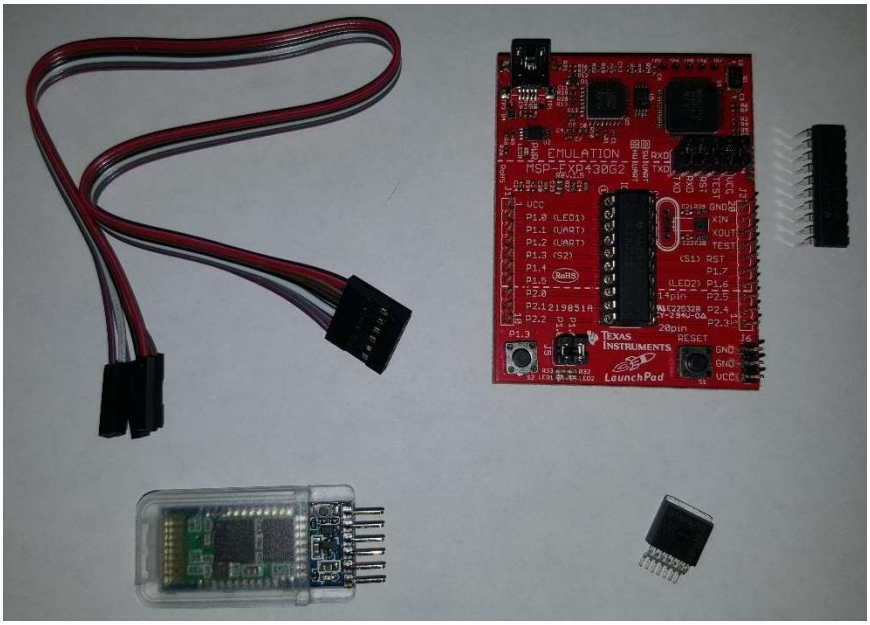


Figure 5.7.1: This figure shows the parts we currently have. This includes the MSP430 Launchpad Development Kit, two microcontrollers (one of which is currently plugged into the development board), the half-bridge, the HC-05 Bluetooth 2.0 Module, and the Dupont connector for the BT module during the development phase.

6. Implementation

As we worked on putting our design together and testing it, we discovered that the design we originally planned would not work as planned. In this section, we will discuss the changes we made to the final implementation of the design and why the final design is better.

After carefully analyzing the implementation idea, the team reached the conclusion that it would be a safety liability to let the sensors interfere with the e-skate speed. This is essentially caused by how skateboards work; the only thing keeping a rider on the board is gravity. Additionally, anyone who does not have enough experience can easily fall, given the need to balance themselves according to momentum and direction. If there is a mechanism that overrides the rider's ability to control the vehicle, it can result in one flying off the board and sustaining terrible injuries. With this in mind, it was determined that this idea was simply too advanced to test on a skateboard.

6.1 Microcontroller and Bluetooth Module

The bluetooth module HC-05 used for the project was first tested using the MSP430G2553 development board; it is easily connected after analyzing its pinout. The STATE and KEY pins were not needed; the VCC and ground pins are directly connected to a VCC and ground pin on the microcontroller. Pins TXD and RXD connect to the UART pins 1.1 and 1.2 respectively.

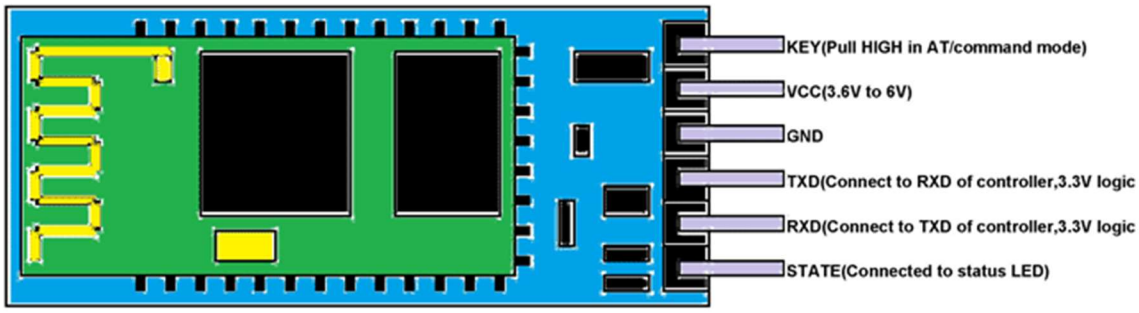


Figure 6.1.1: Bluetooth module HC-05 pinout.

After successfully connecting the module, we can simply use the polling technique to check when any incoming information is available. These are the serial functions that were necessary to implement this code.

Function	Action
Serial.begin();	Enables serial communication
Serial.end();	Disables serial communication
Serial.read();	Reads incoming serial data
Serial.available()	To get number of bytes available

Table 6.1.1: A table that shows the serial functions necessary to implement the Bluetooth section of the code.

The polling technique works with Serial.available() inside the infinite loop that constitutes the main function of the code. When the function returns anything greater than zero, the bluetooth module has sent serial data to the microcontroller. Then the function Serial.read() can be used to identify the data received and save it to a character variable. From here, the code can include a variety of blocks that check the character to execute each particular block when needed.

When we removed the MSP430 microcontroller from the development board, we discovered an issue with the voltage levels between the MSP430. Originally, we attached the MSP430 to an LM317 regulator adjusted to 3.3V. The Bluetooth module could not be connected to this regulator because it needs at least 3.6V. However, when we attached the Bluetooth module to a higher voltage separately, there was trouble with the MSP430 and the module communicating since they had different supply voltages. The solution was to adjust the LM317 to an output of 3.75V and attach both the MSP430 and the Bluetooth module to this voltage. The circuit design for this voltage regulator can be seen below.

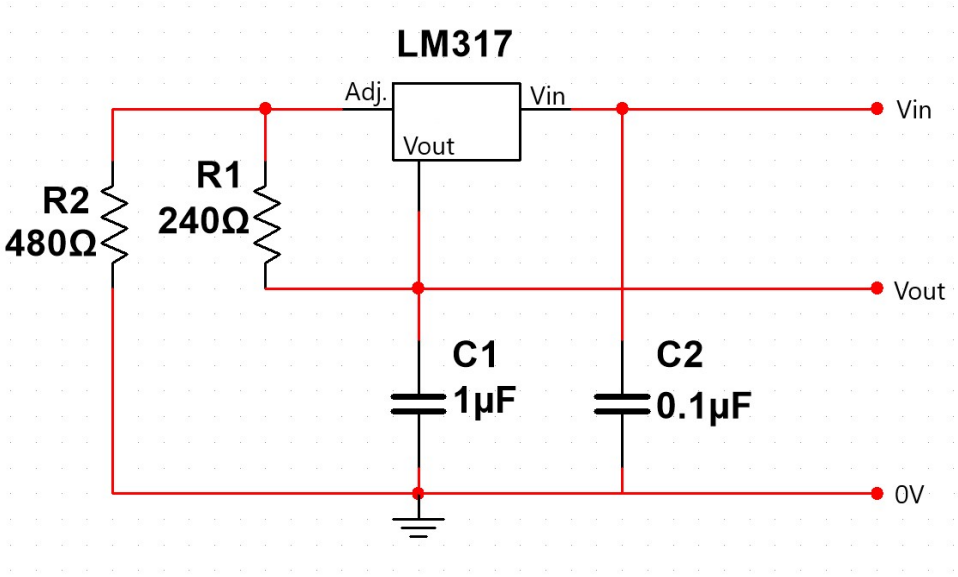


Figure 6.1.2: This is the design of the circuit to regulate the voltage for the MSP430 and the Bluetooth module. Vin can be any value, but in this case it connects to the 13V output of the ESC. With the given resistor values, the Vout is adjusted to 3.75V, which is a suitable Vcc value for the MSP430 and the Bluetooth module. In fact, the 3.75V output was also suitable for the motion sensor, which we discuss in section 6.6.1.

6.2 Android Application

The android application needs to send characters to the microcontroller; the microcontroller will use the code to execute the right action. For applications as such, the MIT App Invertor 2 is a convenient tool that allows applications as such to be developed. This tool is divided into two sections, graphic user interface (GUI) development and backend and execution commands in the form of blocks.

Using the GUI development section, the team generated a list picker, a label, and 10 buttons, along with a background; all these things were organized through horizontal and vertical arrangements found in the layout tab. A descriptive image was chosen for all the buttons and the list picker in order to enhance the appearance of the application. The picture that follows shows the GUI.

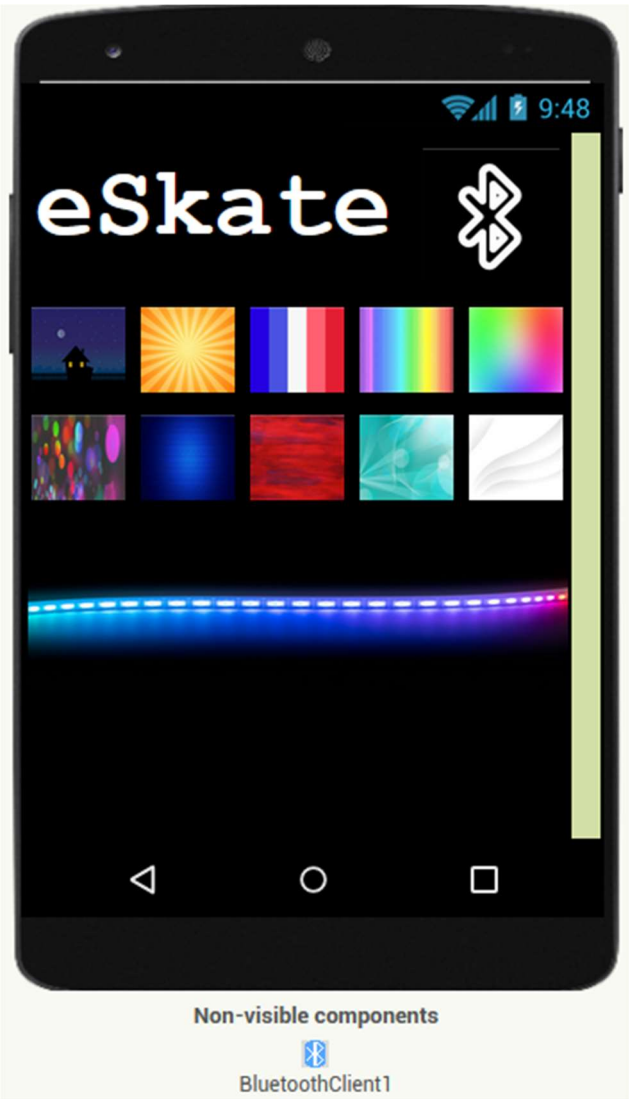


Figure 6.2.1: The mobile application's GUI

Pressing a button triggers the Bluetooth module to send data according to the blocks. For example, the blocks in Figure 6.2.2 indicate that when button 0 is clicked, the Bluetooth client is called, and the character 0 is sent through. Likewise, the blocks indicate that before picking from the list picker, its elements will be set to the addresses and names available to the Bluetooth client; after an address selection has been made, its corresponding Bluetooth module will be connected. The entire set of code blocks can be seen in Figure 6.2.2 shown in the next page.

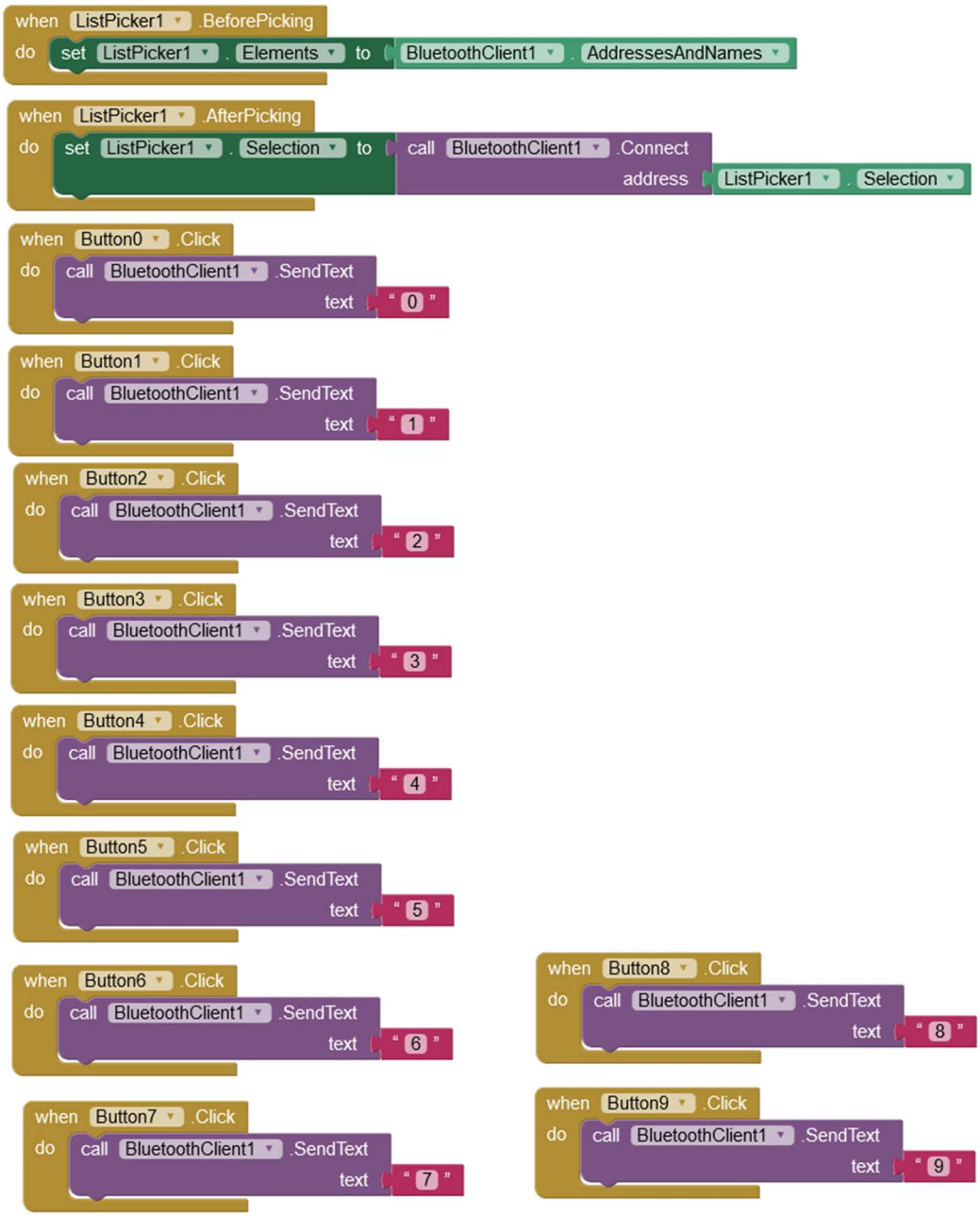


Figure 6.2.2: The mobile application's backend blocks of code

6.3 Hub Motor Kit

For the motor and wheel kit, we chose the “75mm 350W Brushless Hub Motor Kit” from Amazon. This model is single drive and is affordable. We originally bought a more expensive dual hub motor kit from EBay, but this arrived damaged due to poor packaging by the EBay seller. To account for the hit to the budget, we went with a cheaper replacement. This kit still performs well, achieving a top speed of about 16 miles per hour, which matches our target. This kit also has the benefit of using brushless motors, which has energy efficiency advantages over regular DC motors. In typical “brushed” DC motors, the motors are spun internally using electromagnetic brushes to create a magnetic field. However, these waste energy because the brushes create a lot of friction inside the motor. Brushless motors work instead by using an electronic speed controller (ESC), which externally generates the magnetic field by cycling voltage periodically through three input wires, usually blue, green, and yellow.

Figure 6.3.1: This is the hub motor kit we chose.

6.4 Electronic Speed Controller

In the original design, we planned to use a PN Half-Bridge to control the speed of the motors by regulating the voltage. This would have worked with regular DC motors, but the brushless motors require an ESC to generate the magnetic field inside the motors. To match with the motor kit we chose, we went with a 24v-36v Single Drive ESC. This one in particular was rated for up to 650W, which gave us plenty of room to safely operate our 350W motor kit. This electronic speed controller in particular was convenient because it came with a battery indicator, power button, and even a remote. The handheld remote that came with the ESC

had a simple and intuitive design to drive the E-Skate. We started to think that using the Android application to drive the E-Skate would be a dangerous distraction, so the fact that this ESC came with a remote allowed us to focus on just controlling aesthetics with the app, namely the LED functions.

Analysis of the ESC's circuit board allowed us to find a set of pins that output 13V when powered on. By making solder connections between these pins and our circuits, we were able to power the rest of our components, including the LEDs, sensors, Bluetooth module, and the microcontroller.

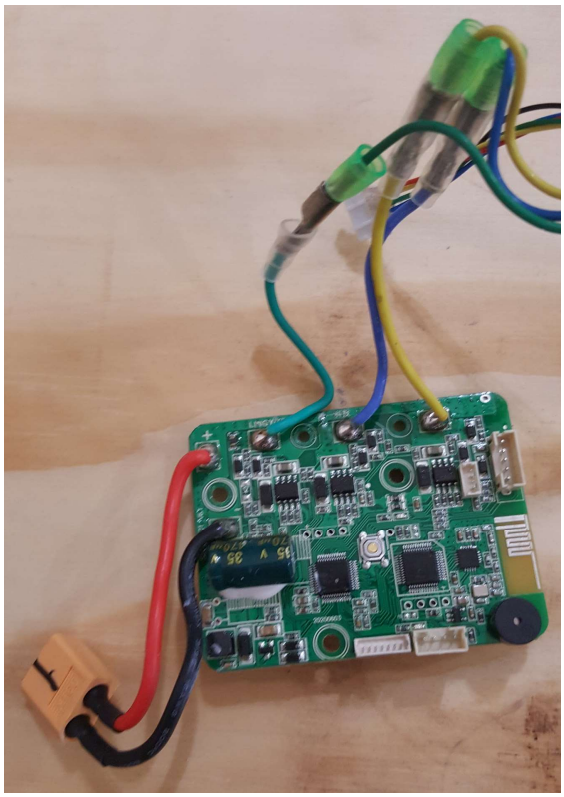


Figure 6.4.1: This shows the Electronic Speed Controller we chose.

6.5 Battery

When going over our original design, we realized that our battery choice was unrealistic. First of all, using two lead-acid batteries would be very heavy. Aside from making the E-Skate less portable, it would also decrease the travel range because extra energy would be wasted to transport the extra weight of the

batteries themselves. Second of all, we chose 12V batteries, which is low. Anything other than the lowest-end electric skateboard motors have input requirements of at least 24V. With these things in mind, we opted for a 7S2P Lithium-ion battery. The code stands for “7-series 2-parallel,” indicating the battery is made of seven Lithium cells connected in series to provide the needed voltage, and then two of these sets are attached in parallel to double the capacity. This battery has a nominal voltage of 25.2V and goes up to 29.4V on full charge, making it perfect to go with our motor kit and ESC.

Figure 6.5.1: This shows the battery we chose.

6.6 Sensors

6.6.1 Motion Sensor

Additionally to the main sensor setup designs that we had researched, there was a simpler, and still efficient way of implementing motion sensors on the project. Considering that the skateboard will primarily move forward, a PIR motion sensor mounted under the skateboard at the front would detect movement ahead, but also in a semicircular area once a dome has been mounted on the sensor. This is ultimately the design procedure we followed, given that it would not have served much of a purpose to install sensors on the sides and/or the back; those

installments would work better for cars and other bigger vehicles, with closed cabins that limit visibility.

Figure 6.6.1.1: PIR Motion Sensor

Figure 6.6.1.2: PIR Motion Sensor

The sensor selected has been highlighted in blue for convenience in Table 3.4.7.3.1, the table of tentative selections researched months back. The selection included a circuit board and a dome, facilitating the sensor's installation for our team, which has no electrical engineer. This selection was also ideal considering that a bare sensor alone could be priced much higher; it also takes away the need to buy components and put them together in a circuit on the PCB, minimizing work, while maximizing cost efficiency and time. One other great attribute of this selection is the dual potentiometer system installed on its board; it allows one to control the sensitivity and delay. Sensitivity refers to how much movement is required to trigger the sensor pin, while delay refers to how soon after a trigger the sensor is ready for the next trigger. The figure below shows the motion sensor selection, along with its pinout.

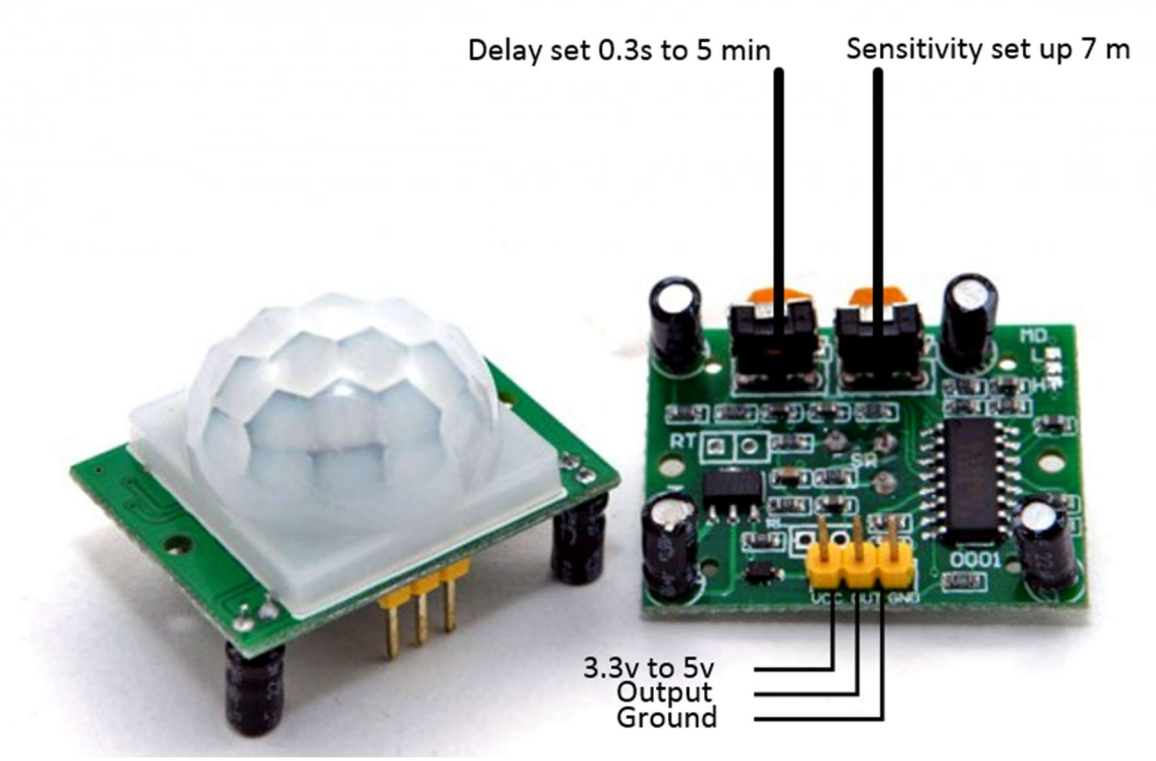


Figure 6.6.1.3: PIR Motion Sensor covered by its dome on the left; circuit board, including both potentiometers on the right.

6.6.2 Light Sensor

Unfortunately, we quickly discovered upon the arrival of our selected light sensor, model OPT3001DNPR, that this latter model is very small, and its pins are absolutely impossible to solder onto without the proper equipment, which must include very thin tips to prevent the pin nodes from merging. We speculate that it is produced for smaller objects, such as phones.

This dilemma took the team back to the research phase to look for project examples that used light sensors in similar environments. Then we stumbled upon the basic light sensor, a photoresistor, which we added its information to table 3.5.2 and highlighted it blue in a more recent revision of this document. The photoresistor is just as good as any other sensor listed in the table, because it can also be read analogically or digitally using a microcontroller. Analog read allows our team to dim or brighten LEDs without turning them completely on or off; this

functionality is not necessary though. Instead, we will read the photoresistor's output and decide at which brightness the LEDs will turn on or off. For example, the value that we decided to use was 50, which was good to perform our tests. When the environment's brightness dips below 50, the lights turn on automatically; similarly, the lights will turn off if the environment's brightness hits 50 or above.

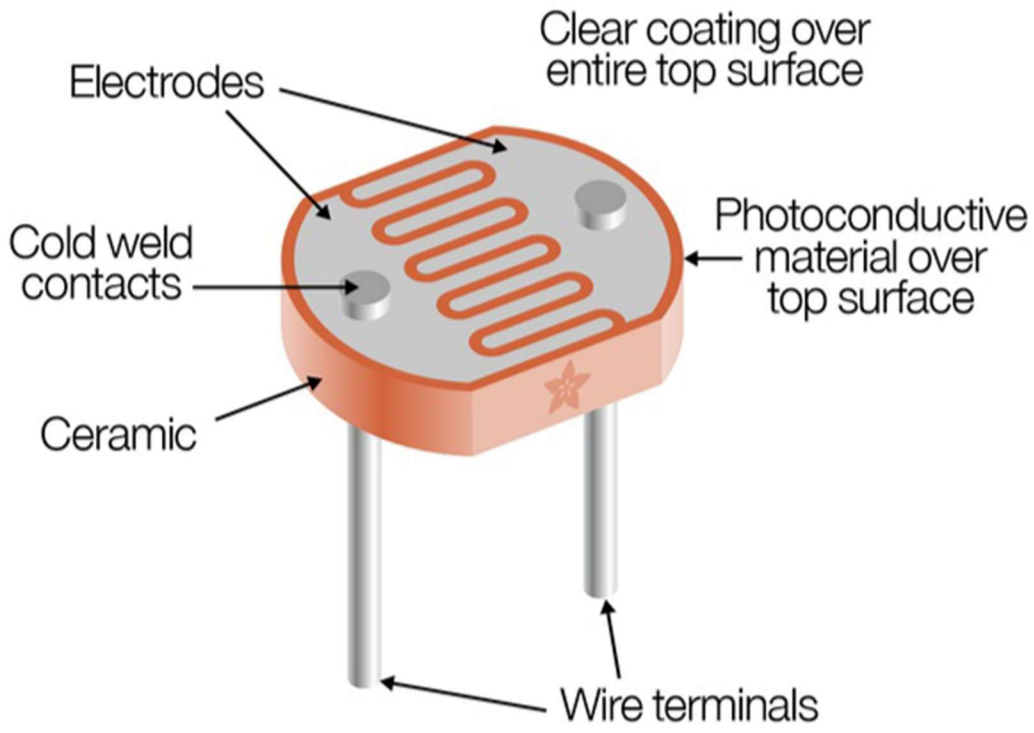


Figure 3.5.4.2: The components of a photoresistor.

6.7 LEDs

Once the LED strip arrived, we were surprised to find out that we had bought a grand total of 12 LEDs, four strips of 3 LEDs each, at a price of about $8 to $12. This was a loss, since a whole roll of LEDs could have been bought from Amazon for about $20. We proceeded to test the LEDs received on the breadboard to make sure we understood how the circuit works with a microcontroller.

In order for a microcontroller to turn a 12V LED strip on or off, it requires the use of transistors; transistors behave like switches controlled by the microcontroller. First, the strip is connected from the 12V pin to the 12V source, which is a circuit

that regulates the voltage of the battery, 25.2V. An LED strip has three different grounds, one for each color, which share the voltage in parallel; each ground is connected to a collector pin of a TIP 120 transistor, which were purchased from Skycraft at $1.50. The base of each transistor is connected to an output pin from the microcontroller. Passing enough current through this pin turns the transistor on. Oppositely, removing the current through the base will cause the transistor to turn off. At the emitter pin of each transistor, the ground is connected; when the transistor switches on, its LEDs connect to the ground, thus completing the circuit, and turning on the LEDs of the color designated to that transistor. Please refer to the schematics in Figure 6.7.2.

These schematics in Figure 6.7.2 include the headlight, which was implemented the same way in regard to the circuit. Figure 6.7.4 shows the location of the headlight and how it was made. The idea originated from the LED bars that have been appearing on trucks in the last couple of years; they are annoyingly bright and provide great visibility. During a trip to skycraft, a team member stumbled upon 2x3 LED warm white panels. We used three of these panels connected in parallel, all hooked to the same voltage source as the LED strip. When the headlight was finally ready, it was mounted with hot plastic, just like the LED strip. This can be seen in the section “Physical Construction”.

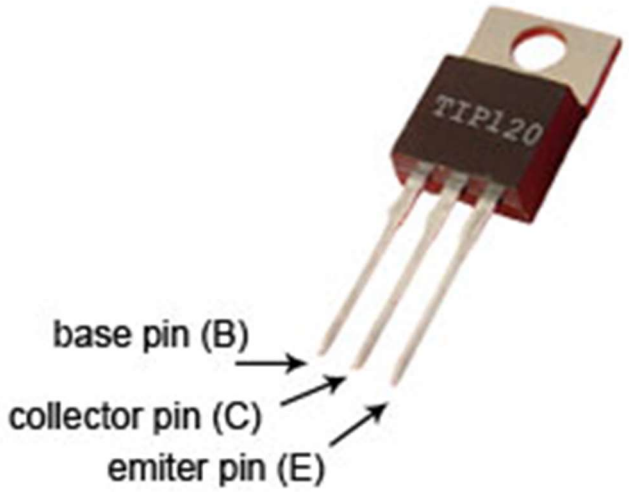


Figure 6.7.1: TIP120 transistor pinout.

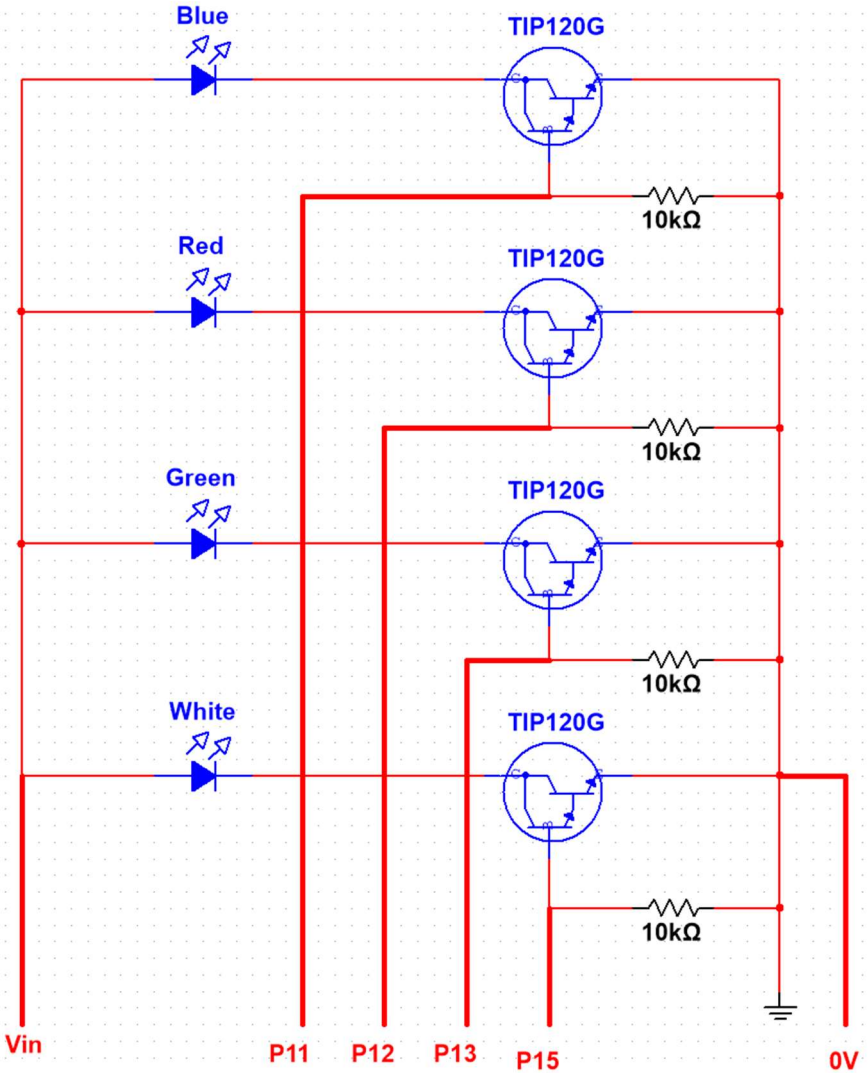


Figure 6.7.2: Circuit design for controlling all three colors of LEDs, one color per transistor and output pin.

After all the testing was done, we proceeded to get the LEDs. Around that time, a team member found a long LED strip sitting at his house; it was broken when removed from the desk on which it had been glued to, but that only constituted a loss of 3 LEDs in total. Then the time came to mount the LEDs, and we quickly discovered that working with an LED strip as a whole is an absolute nightmare, given that the E-Skate is foldable. Figure 6.7.4 is a more accurate representation of what the LED mounting diagram looks like in reality. Mounting the LEDs had another major complication; the weird bending of the strip caused the strip to start breaking in bad places, causing some colors not to work. Although we tried to take off the black vinyl, the PCB started breaking, and the team ultimately decided to cut the pieces out and resolder. Resoldering was also complicated; the soldering

compound had a hard time sticking to the PCB, but ultimately, it ended up sticking when a thinner soldering tip was used.

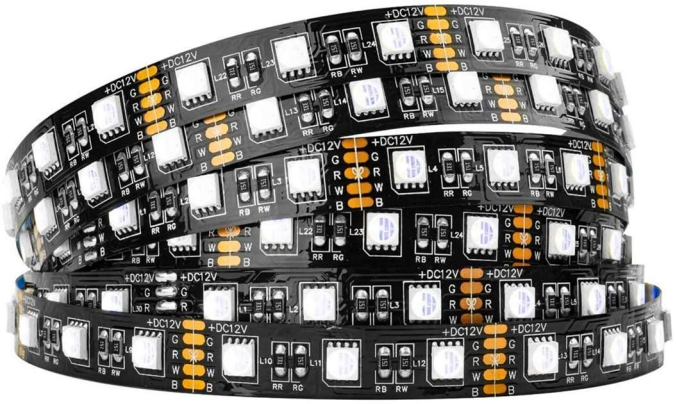


Figure 6.7.3: The LED strip used for the final project.

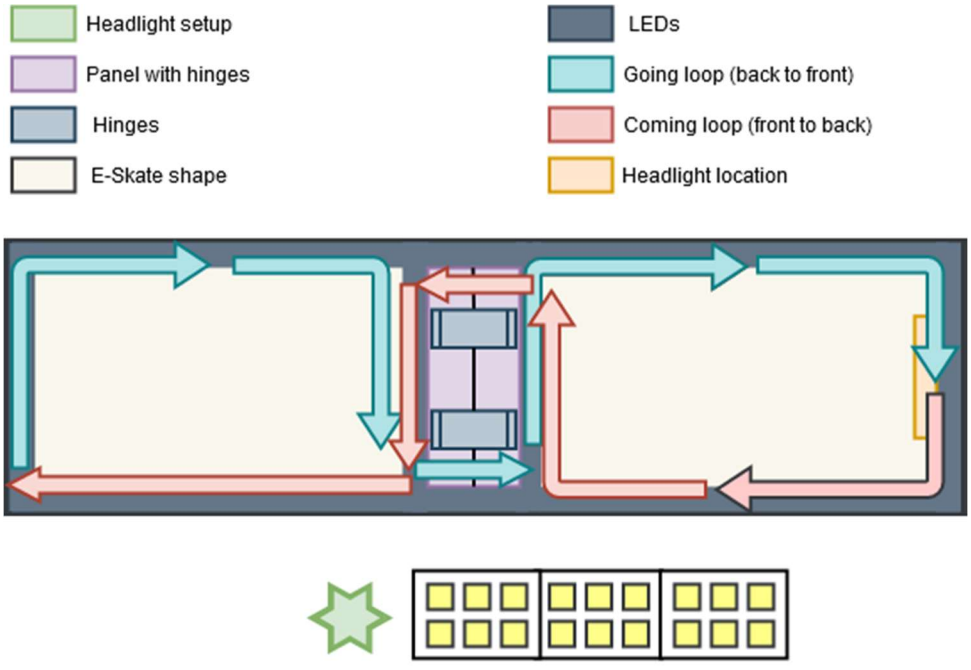


Figure 6.7.4: Accurate LED system setup based on actual build

6.8 Physical Construction

This section is concerned with the design of the deck itself, including the construction and the folding mechanism. The deck in total is 40 inches long, 10 inches wide, and ½ inch thick. These proportions can be seen in Figure 6.8.1 below. This was achieved by double layering pieces of ¼ inch plywood. Cutting pieces 20 inches long gives us the two sections that fold on top of each other. Figure 6.8.2 below shows the bottom of the E-Skate, so the layout of components and the space management can be seen. Notice how the front wheels (left side of picture) are placed back several inches. This gives the space for the E-Skate to fold without the sets of wheels colliding. We show what the pivot point of the board looks like in Figure 6.8.3. Notice the durable steel hinges and the placement of the wires such that they sit below the surface level of the hinges. This allows the E-Skate to fold without pinching the wires, which could damage them. Finally, see Figure 6.8.4, which shows the shape of the folded E-Skate. Notice the reinforcement at the pivot of the E-Skate (right side of picture). This helps distribute the tension force caused by the rider's weight on the middle of the board, preventing it from snapping at the hinges or folding in the wrong direction. They also serve to limit the range of motion when folded completely. Otherwise, if the hinges were flush with the bottom of the deck, the folded shape would form a sharp angle that can crush the electrical components with enough pressure.

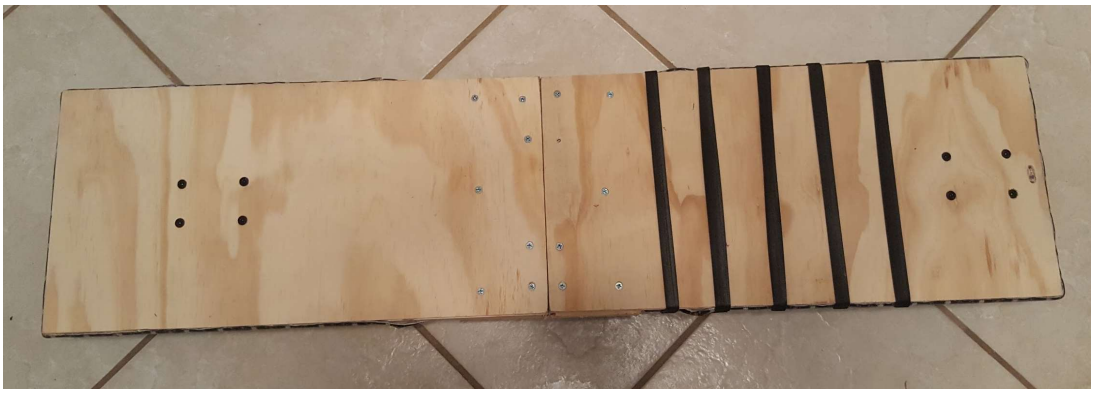


Figure 6.8.1: Top View of the E-Skate

Figure 6.8.2: Bottom view of the E-Skate.

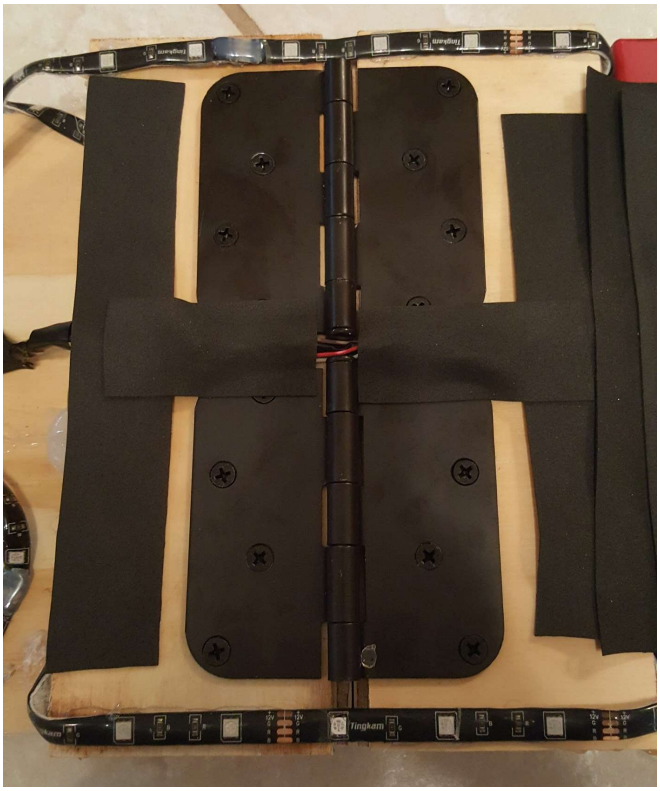


Figure 6.8.3: View of the folding mechanism.

Figure 6.8.4: View of the E-Skate in its folded form.

6.9 Budget

This section simply shows the costs associated with the Backpack E-Skate project. Only the parts that made it to the final design are shown. Anything we bought that we did not end up using are not included. For example, the original dual hub motor kit we bought is not included in the total cost because we ended up using a different model.

	Component	Cost
1.	Battery and Charger	$40
2.	MSP430G2553 w/ dev. board	FREE
3.	HC-05 Bluetooth 2.0 Module	$6.99
4.	Wood and Hardware (includes screws, hinges, etc.)	$45.62
5.	PCB (includes costs of various circuit components used)	$53.62
6.	Sensors (PIR and photoresistor)	$2.48
7.	Electronic Speed Controller	$50
8.	Single Hub Motor Drive Kit	$85
9.	RGB LED Strip	FREE
10.	White LED modules	$7.99
11.	Total	$291.70

Appendix I: Source Code

Source Code for MSP430G2553

```
// Sensor pins
int Lsens = 5, Msens = 6;

// Light pins
int blue = 11, red = 12, green = 13, white = 15;

// Sensor variables
int Lval = 0, Mval = 0, Lflag = 1;

// Bluetooth variables
int Bin, seqNum;

// Fade speed for color sequences
#define FADESPEED 5

// Isolated function
// Turn Headlight ON/OFF
void headLights(int x)
{
	// Headlights ON
	if (x == 1)
	{
	analogWrite(white, 255);
	}

	// Headlights OFF
	else if (x == 0)
	{
	analogWrite(white, 0);
	}
}

// Isolated function
// Police WARNING!!!
void warning(int x)
{
	if (x == 911)
	{
	// Everything else OFF
	seqOFF();

	analogWrite(red, 255);  // turn LED ON
	delay(100);
	analogWrite(red, 0);
	analogWrite(blue, 255);
	delay(100);
	analogWrite(blue, 0);
	analogWrite(red, 255);  // turn LED ON
```

```
            delay(100);
            analogWrite(red, 0);
            analogWrite(blue, 255);
            delay(100);
            analogWrite(blue, 0);
            analogWrite(red, 255);  // turn LED ON
            delay(100);
            analogWrite(red, 0);
            analogWrite(blue, 255);
            delay(100);

            seqOFF();
            }
}

// Isolated function
// All OFF
void seqOFF()
{
            analogWrite(red, 0);
            analogWrite(green, 0);
            analogWrite(blue, 0);
            headLights(0);
}

// Includes OFF, headlights, 911
// and color sequences 3 to 9
void ColorSeq(char code)
{
            int brightness = 0;
            int fadeAmount = 1;
    int wait = 10;

            if  (code == '0')
            {
            analogWrite(red, 0);
            analogWrite(green, 0);
            analogWrite(blue, 0);
            headLights(0);
            }

            // Headlight ON
            if (code == '1')
            {
            analogWrite(white, 255);
            }

            if (code == '0')
            {
            analogWrite(red, 0);
            analogWrite(green, 0);
            analogWrite(blue, 0);
            headLights(0);
```

```
}

// Police Sequence
else if (code == '2')
{
analogWrite(red, 255);  // turn LED ON
delay(100);
analogWrite(red, 0);
analogWrite(blue, 255);
delay(100);
analogWrite(blue, 0);
analogWrite(red, 255);  // turn LED ON
delay(100);
analogWrite(red, 0);
analogWrite(blue, 255);
delay(100);
analogWrite(blue, 0);
analogWrite(red, 255);  // turn LED ON
delay(100);
analogWrite(red, 0);
analogWrite(blue, 255);
delay(100);
}

// Color Switch Sequence
else if (code == '3')
{
analogWrite(blue, 255); // Blue
delay(1000);
analogWrite(red, 128); // Violet
delay(1000);
analogWrite(red, 255); // Magenta
delay(1000);
//analogWrite(blue, 128); // Raspberry
//delay(1000);
analogWrite(blue, 0); // Red
delay(1000);
analogWrite(green, 128); // Orange
delay(1000);
analogWrite(green, 255); // Yellow
delay(1000);
analogWrite(red, 128); // Spring Green
delay(1000);
analogWrite(red, 0); // Green
delay(1000);
//analogWrite(blue, 128); // Turquoise
//delay(1000);
analogWrite(blue, 255); // Cyan
delay(1000);
analogWrite(green, 128); // Ocean
delay(1000);
analogWrite(green, 0); // Blue
}
```

```
        // Color Swirl Sequence
        else if (code == '4')
        {
        int r, g, b;

            // fade from blue to violet
        for (r = 0; r < 256; r++) {
        analogWrite(red, r);
        delay(FADESPEED);
        }
        // fade from violet to red
        for (b = 255; b > 0; b--) {
        analogWrite(blue, b);
       delay(FADESPEED);
        }
        // fade from red to yellow

        for (g = 0; g < 256; g++) {
        analogWrite(green, g);
        delay(FADESPEED);
        }
        // fade from yellow to green
        for (r = 255; r > 0; r--) {
        analogWrite(red, r);
        delay(FADESPEED);
        }
        // fade from green to teal
        for (b = 0; b < 256; b++) {
        analogWrite(blue, b);
        delay(FADESPEED);
        }
        // fade from teal to blue
        for (g = 255; g > 0; g--) {
        analogWrite(green, g);
        delay(FADESPEED);
        }
        }

        // Color Flash Sequence!!!!!
        else if (code == '5')
{
        analogWrite(blue, 255); // Blue
        delay(200);
        analogWrite(blue, 0);
        delay(200);
        analogWrite(blue, 255); // Violet
        analogWrite(red, 128);
        delay(200);
        analogWrite(blue, 0);
        analogWrite(red, 0);
        delay(200);
        analogWrite(blue, 255); // Magenta
```

```
analogWrite(red, 255);
delay(200);
analogWrite(blue, 0);
analogWrite(red, 0);
delay(200);
   analogWrite(red, 255); // Raspberry
analogWrite(blue, 128);
delay(200);
analogWrite(red, 0);
analogWrite(blue, 0);
delay(200);
analogWrite(red, 255); // Red
delay(200);
analogWrite(red, 0);
delay(200);
analogWrite(red, 255); // Orange
analogWrite(green, 128);
delay(200);
analogWrite(red, 0); // Orange
analogWrite(green, 0);
delay(200);
analogWrite(red, 255); // Yellow
analogWrite(green, 255);
delay(200);
analogWrite(red, 0);
analogWrite(green, 0);
delay(200);
analogWrite(green, 255); // Spring Green
analogWrite(red, 128);
delay(200);
analogWrite(green, 0);
analogWrite(red, 0);
delay(200);
analogWrite(green, 255); // Green
delay(200);
analogWrite(green, 0);
delay(200);
analogWrite(green, 255); // Turquoise
analogWrite(blue, 128);
delay(200);
analogWrite(green, 0);
analogWrite(blue, 0);
delay(200);
analogWrite(green, 255); // Cyan
analogWrite(blue, 255);
delay(200);
analogWrite(green, 0);
analogWrite(blue, 0);
delay(200);
analogWrite(blue, 255);
analogWrite(green, 128); // Ocean
delay(200);
analogWrite(green, 0);
```

```
        analogWrite(blue, 0);
        delay(200);
}

        // Flash Green Sequence
        else if (code == '6')
        {
        for (brightness = 0; brightness <= 255; brightness++)
        {
        analogWrite(green, brightness);
        delay(wait);
        }

        for (brightness = 255; brightness >= 0; brightness--)
        {
        analogWrite(green, brightness);
        delay(wait);
        }
        }

        // Breathe Red Sequence
        else if (code == '7')
        {
        for (brightness = 0; brightness <= 255; brightness++)
        {
          analogWrite(red, brightness);
        delay(wait);
        }

        for (brightness = 255; brightness >= 0; brightness--)
        {
        analogWrite(red, brightness);
        delay(wait);
        }
        }

        // Flash Cyan Sequence
        else if (code == '8')
        {
        analogWrite(blue, 255);
        analogWrite(green, 255);
        delay(200);

        analogWrite(blue, 0);
        analogWrite(green, 0);
        delay(200);
        }

        // Flash White Sequence
        else if (code == '9')
        {
        analogWrite(blue, 255);
        analogWrite(green, 255);
```

```
analogWrite(red, 255);
delay(200);

analogWrite(blue, 0);
analogWrite(green, 0);
analogWrite(red, 0);
delay(200);
}

// Always ON one-color seq's by initials
else if (code == 'r')
{
analogWrite(blue, 0);
analogWrite(green, 0);
analogWrite(red, 255);
}

else if (code == 'b')
{
analogWrite(blue, 255);
analogWrite(green, 0);
analogWrite(red, 0);
}

else if (code == 'g')
{
analogWrite(blue, 0);
analogWrite(green, 255);
analogWrite(red, 0);
}

else if (code == 'c')
{
analogWrite(blue, 255);
analogWrite(green, 255);
analogWrite(red, 0);
}

else if (code == 'm')
{
analogWrite(blue, 255);
analogWrite(green, 0);
analogWrite(red, 255);
}

else if (code == 'y')
{
analogWrite(blue, 0);
analogWrite(green, 255);
analogWrite(red, 255);
}

else if (code == 'v')
```

```
        {
        analogWrite(blue, 255);
        analogWrite(green, 0);
        analogWrite(red, 128);
        }

        else if (code == 'a')
        {
        analogWrite(blue, 128);
        analogWrite(green, 255);
        analogWrite(red, 0);
        }

        else if (code == 'n')
        {
        analogWrite(blue, 0);
        analogWrite(green, 255);
        analogWrite(red, 128);
        }

        else if (code == 'o')
        {
        analogWrite(blue, 0);
        analogWrite(green, 128);
        analogWrite(red, 255);
        }

        else if (code == 'e')
        {
        analogWrite(blue, 255);
        analogWrite(green, 128);
        analogWrite(red, 0);
        }

        else if (code == 'w')
        {
        analogWrite(blue, 255);
        analogWrite(green, 255);
        analogWrite(red, 255);
        }

}

void setup() {
        Serial.begin(9600);
        pinMode(red, OUTPUT);
        pinMode(green, OUTPUT);
        pinMode(blue, OUTPUT);
        pinMode(Lsens, INPUT);
        pinMode(Msens, INPUT);
}

void loop() {
```

```
// read the value from the light sensor:
Lval = analogRead(Lsens);

// Read value from the PIR motion sensor
Mval = digitalRead(Msens);

// Check motion sensor trigger
        if (Mval == HIGH)
        warning(911);

 // Check light sensor trigger
        if (Lval < 60)
        {
        if (Lflag == 1)
        {
        ColorSeq('w');
        headLights(1);
        }
        }

        else if (Lval >= 60)
        {
        seqOFF();
        }

        ////////////////////////////////////////
        // Bluetooth Color Sequence Activation //
        ////////////////////////////////////////
        if (Serial.available() > 1)
        {
        // Turn Light Sensor Flag OFF
        Lflag = 0;

        // Turn OFF previous Sequence
        seqOFF();

        // Obtain the bluetooth signal
        Bin = Serial.read();

        // Avoid a consecutive read that
        // stops the color sequence loop
        Serial.end();
        }

        ////////////////////////////////
        // EXECUTE A BLOCK PER INPUT //
        ////////////////////////////////

        // OFF SEQUENCE!!
        if (Bin == '0')
        {
        Lflag = 1;
```

```
ColorSeq('0');
}

else if (Bin == '1')
{
// seqOFF();
ColorSeq('1');
}

else if (Bin == '2')
{
//seqOFF();
ColorSeq('2');
}

else if (Bin == '3')
{
//seqOFF();
ColorSeq('3');
}

else if (Bin == '4')
{
//seqOFF();
ColorSeq('4');
}

else if (Bin == '5')
{
//seqOFF();
ColorSeq('5');
}

else if (Bin == '6')
{
//seqOFF();
ColorSeq('6');
}

else if (Bin == '7')
{
//seqOFF();
ColorSeq('7');
}

else if (Bin == '8')
{
//seqOFF();
ColorSeq('8');
}

else if (Bin == '9')
{
```

```
//seqOFF();
ColorSeq('9');
}

else if (Bin == 'b')
{
//seqOFF();
ColorSeq('b');
}

else if (Bin == 'r')
{
//seqOFF();
ColorSeq('r');
}

else if (Bin == 'g')
{
//seqOFF();
ColorSeq('g');
}

else if (Bin == 'v')
{
//seqOFF();
ColorSeq('v');
}

else if (Bin == 'o')
{
//seqOFF();
ColorSeq('o');
}

else if (Bin == 'c')
{
//seqOFF();
ColorSeq('c');
}

else if (Bin == 'm')
{
//seqOFF();
ColorSeq('m');
}

else if (Bin == 'y')
{
//seqOFF();
ColorSeq('y');
}

else if (Bin == 'a')
```

```
        {
        //seqOFF();
        ColorSeq('a');
        }

        else if (Bin == 'n')
        {
        //seqOFF();
        ColorSeq('n');
        }

        else if (Bin == 'e')
        {
        //seqOFF();
        ColorSeq('e');
        }

        else if (Bin == 'w')
        {
        //seqOFF();
        ColorSeq('w');
        }
}
```