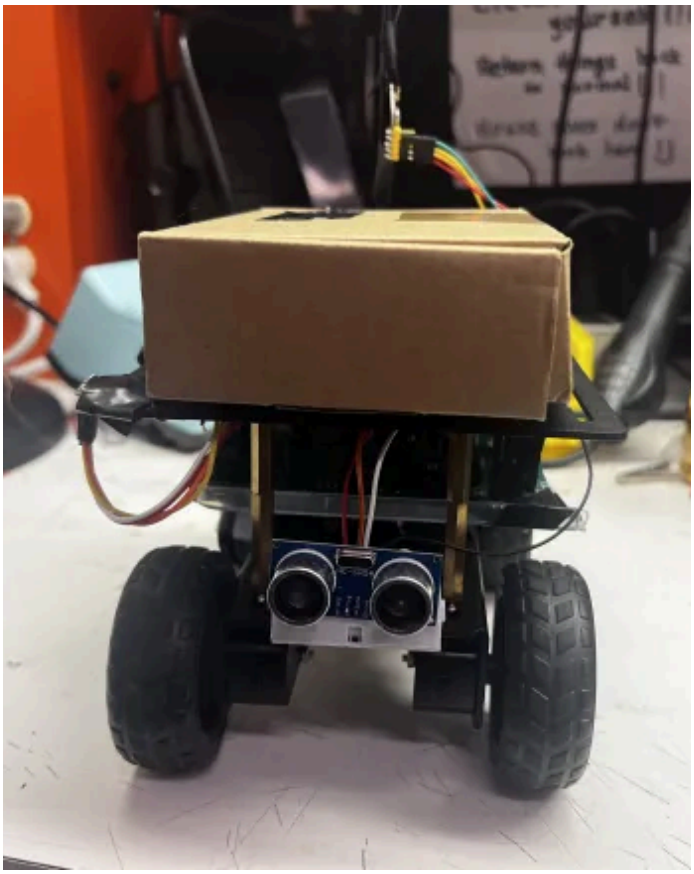
Deliver Me Something



University of Central Florida
Department of Electrical & Computer Engineering

Group 19:

Jacob Raya, Electrical Engineering
Reece Segui, Computer Engineering
Luis Figueroa, Computer Engineering
Connor Hatfield, Computer Engineering

Reviewers:

Dr. George Atia
Dr. Zakhia Abichar

Table of Contents

Section 1: Executive Summary

The world of autonomous vehicles is constantly expanding as the development of technology allows for greater interest, focus, and thereby progression of the topic. In addition, the increased emphasis on prioritizing convenience in the day to day lifestyle of people allows for autonomy to be more directed towards assisting certain areas by limiting the need for human interaction in products. A simple example of this is the self-serving kiosks in grocery stores and restaurants and a more complex application, one that this report defines, is the utilization of autonomous vehicles in delivering products to a company's customers. This can be seen with companies like Amazon, Walmart, UPS, and more adopting flying drones named "unmanned aerial vehicles" or UAVs. The change in direction towards automating delivery processes can also be seen in the semi-truck industry with unique innovations like the Autonomous Relay Convoy from Locomotion in which a human driver operates a semi-truck acting as the lead while another semi-truck follows behind autonomously. This allows the driver of the follower truck to sleep and stay within the allotted driving hours while also continuing to work towards deliveries. This combination of humans and automated vehicles working in tandem is the area in which this report aims to expand upon.

This report details the thought process, explanations, development, and overall completion of our autonomous delivery solution. By researching this area of technology and forming an answer, our Senior Design team plans to introduce an idea that may not revolutionize the engineering of autonomous robots, but offers a realistic option that is more easily adoptable in the realm of servicing customers. This project presents an autonomous, GPS tracked Delivery Bot that travels short distances to complete deliveries of small items to reduce the need for interrupting a person's workflow or increase the efficiency of dispersing products to customers. Essentially, we would like to utilize the technology surrounding automated processes in order to provide a convenient option to an event that people often encounter. This relevance of this project is clear as automation is becoming increasingly popular as shown by the examples given earlier.

The beginning of this report will further explain the motivations, goals, and objectives of our team in performing this research. This also includes the planned workflow and specifications for the successful completion of our Delivery Bot. Then, the report documents the research of each component that eventually determined the route in which the team decided to take in constructing the project. Afterwards, design constraints are verified to ensure the Delivery Bot operates on the correct standards and safety issues are addressed. The following section details the specific hardware and software designs that have been created. Lastly, the budgeting and planning of the project is explained, showing both the expected and actual values of the parts of the robot as well as the time frames for researching, testing, etc.

Section 2: Project Description

This section will detail the project as a whole as well as the motivations that the group has to go forward with it. It will provide a detailed yet succinct overview of the entire project including a quick overview of specifications of the project, any requirements or constraints that were placed on the project, and visual aids that would help one achieve a high-level understanding of the project.

2.1 Project Motivations and Goals

Motivations and goals are always important for any task or project. They are what drive the people working on the project to actually put time into the project to have it go as smoothly as possible. The motivations for a project can be as simple as “it is fun” or it can be an assignment from a higher up employee or a combination of both. Goals are very similar as they can be assigned by either the people working on the project or administrators who are overlooking the entire process. Goals are a way to help measure progress and see if certain parts of the project are being worked on when they are supposed to be. It helps assure that the general timeline of a project is being followed.

2.1.1 Motivations

Overall, the main motivation of this project is to explore and expand upon an ever-growing market of delivery services and automated technology. We plan to create a product that improves the convenience for users of these types of services in addition to the companies that provide these services to their customers. And of course, we would like to learn more about the current and future aspects of our respective fields and hope to apply what we learn in the process of this project in our future careers.

As the popularity of platforms such as Amazon, Uber Eats, and DoorDash continue to rise with people realizing the conveniences of these types of services, more innovation in the methods of delivery should be explored. Some technology that is already being researched are the delivery drones from Amazon and even automated semi-trucks which both have the goal of delivering a user’s parcels in an effective, safe, and quick manner, mostly in a way that requires less human interaction in the process. These technologies also increase the convenience of these services even more than they already appear to be. This is where our project attempts to expand upon.

2.1.2 Goals

Our objective in this project is to create a grounded, autonomous delivery solution that is able to transport small items to and from locations using GPS. It would be meant for deliveries such as food items and small objects in distances

close to the user (2 miles). This allows businesses like fast food restaurants, grocery stores, and local package sorting facilities to use these robots to offer quicker deliveries with no need for a human driver. Some questions may arise from deciding on a grounded bot rather than a drone similar to those made from Amazon and Walmart. Our reasoning for this decision is to avoid restrictions and legalities on drone flight and also prevent safety hazards while testing our designs. This delivery method also suits the types of parcels we are planning to focus on and in addition, should also be much easier to work with than a drone in terms of operation and design.

With regards to our design in particular, we designed a four-wheeled, autonomous delivery robot that implements GPS tracking and security measures to ensure that packages cannot be stolen or damaged while on its route. Considering our bot is meant to be autonomous, safety during transport is a high priority. This means solutions regarding avoiding pedestrians and safely crossing streets with traffic. Some additional features could include some sort of docking station for charging and protection from natural elements like rain and heat/cold. Also, there will likely be a locking system in the form of an automated deadbolt that locks during travel and opens once it reaches its location, or a keypad that users can enter a code to unlock the parcel container.

Limitations on the scope of our project have already been established to maintain safety and testability while also ensuring the time frame will allow for the full completion of our product. Some of these limitations include having a distance of less than 2 miles, as stated previously, a maximum speed of 2 miles per hour to allow safe travel in locations where pedestrians may be present, and a maximum weight of 20 pounds to reduce costs of components such as motors. In addition to these limitations, we have assumptions that most of the mechanical components are bought rather than built, as our fields of study are in Computer and Electrical Engineering. Therefore, more focus is applied to coding the operation and facilitating the assembly of the vehicle.

2.2 Project Objectives

The definitive objective of this project is to create an autonomous delivery system that is able to operate with minimal human interaction and deliver parcels to its users within a short distance in a reasonable amount of time. The use cases for our design include functions where the delivery bot only has to travel within a certain area such as a college campus, section of a neighborhood, or within a company’s premises, among many others. An example of these use cases could include a student ordering an energy drink or food from a convenience store to their current location in the library. In this case, the owner of the delivery bot would place their order within the payload carrier and send the bot to the user’s GPS location. Once the payload is received, the delivery bot returns to its original location where it will be ready for another delivery. As stated previously, it could

also be used in the case of a mail service where instead of going door to door to each house, a mail carrier can go to the center of a section of neighborhood and send several of the delivery bots to various addresses where the payload can be picked up individually. This would reduce delivery times and increase the efficiency of the delivery process by completing deliveries in parallel rather than in a series.

An additional benefit of utilizing this method of autonomous delivery rather than the more futuristic solutions like flying drones and autonomous semi-trucks, is that our project is much more easily applied to current problems. With solutions as drastic as the ones just mentioned, developers must be mindful of catastrophic failures that could put the users' products in danger and even more importantly, the lives of people in the surrounding area in danger. With our small product design, the safety of pedestrians is considered but the possibility of extreme injury of pedestrians in the area is very unlikely. This differs from the considerations of companies attempting to utilize autonomous trucks, for example. The risk of injury is very high and the unreliability of situations that the product could be put through is also a consideration to be aware of. In other words, another objective of this project is to lower the risks of autonomous delivery by operating at a smaller scale to hopefully bring the world of autonomous delivery into greater use. This project is not attempting to revolutionize autonomous delivery, but instead offer a realistic stepping stone to the acceptance of the more futuristic and exciting designs.

2.3 Requirements Specifications

Below are the requirement specifications used for the project. These specifications indicate successful operation of our design and will be actively reviewed during the construction of the delivery bot to determine the fulfillment of the project goals and motivations.

Specifications Table	
Travel Radius/Automation	~2 miles
Destination Recognition	Stop when intended destination is reached
Maximum Speed	1 mile/hr

Payload Weight	≤ 5 pounds
Total Weight	≤ 20 pounds
Security Measures	Automated locking mechanism
Cost	≤ $375
Dimensions	1 ft x 1 ft x 0.85 ft
Power Use	< 60 watts
System Location	GPS
Weather Protection	IP44 equivalent
Implementation Time	~10 weeks
Self-Orientation	Autonomously orient itself towards the goal destination
Obstacle Avoidance	Autonomously avoid any hazards it may come into contact with

2.4 House of Quality Analysis

The house of quality analysis examines the requirement specifications and divides them into engineering requirements and user requirements and measures them against each other to collect the positive and negative correlations, it also specifies the targets for the engineering requirements which are directly taken from the specifications table. In this analysis it also includes the polarity of each requirement and the polarity of the requirements against each other.

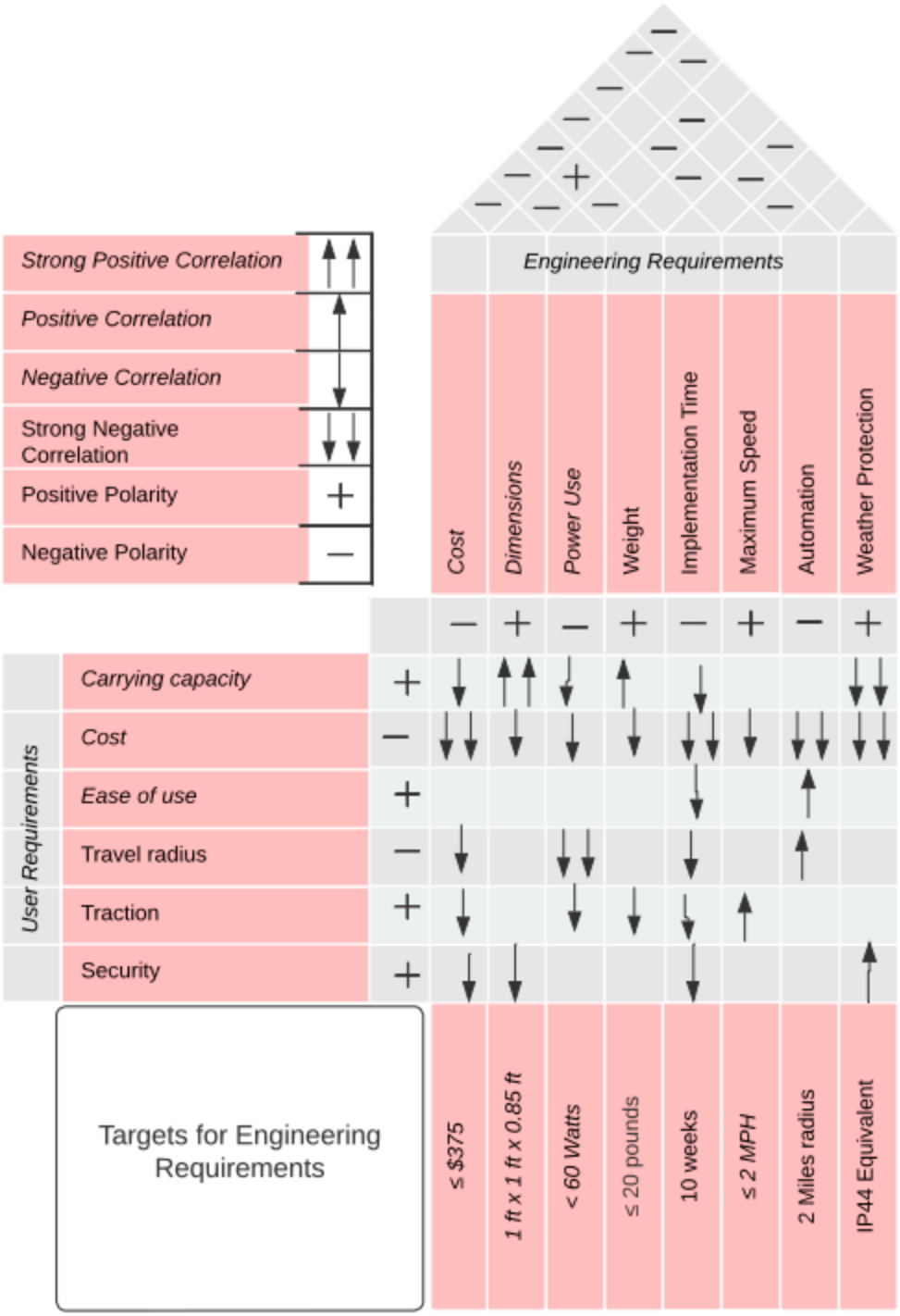


Figure 2.1 House of Quality for Land Delivery Bot

2.5 Block Diagrams

Block diagrams are a very useful visual aid. They help show the inner workings of a system without going into too much detail as to overwhelm the reader. They are also useful for the people working on the project itself as a reminder as to where each part of the system is supposed to go in relation to the other parts and how each part interacts with each other.

2.5.1 Software Diagram

This section represents the coding aspect of the project. It shows the planned function of the robot during operation and the simple functions that are to be implemented along with the logic to be used while traveling to a destination. Essentially, the process is to begin travel toward the destination. It constantly checks the location utilizing the GPS system. If it has not reached the destination it will continue along the route. The other aspect of the software is that there is an object detection along with the GPS locational data. This is used to determine actions based on objects in the delivery bot's path.

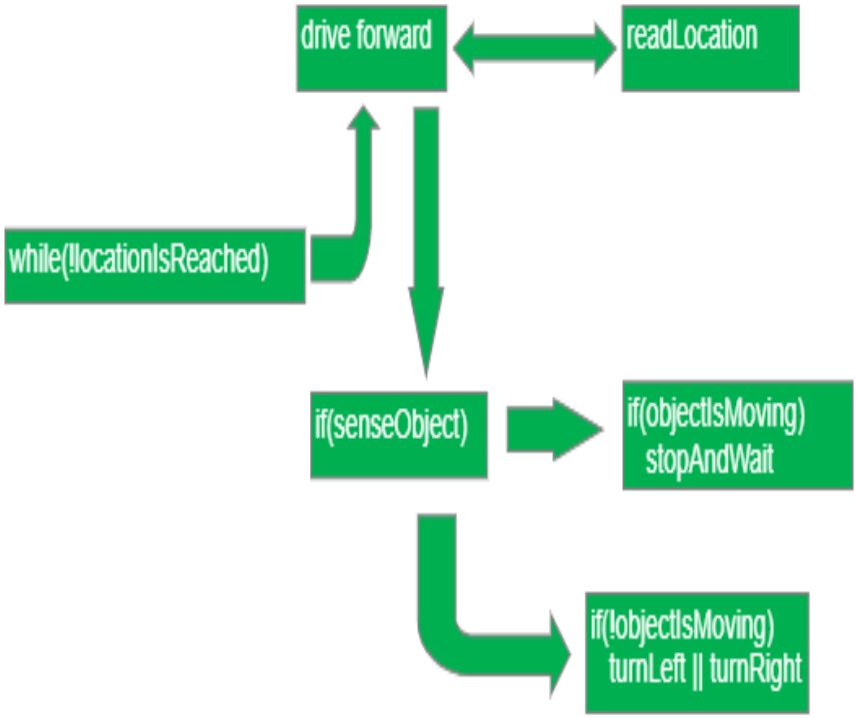
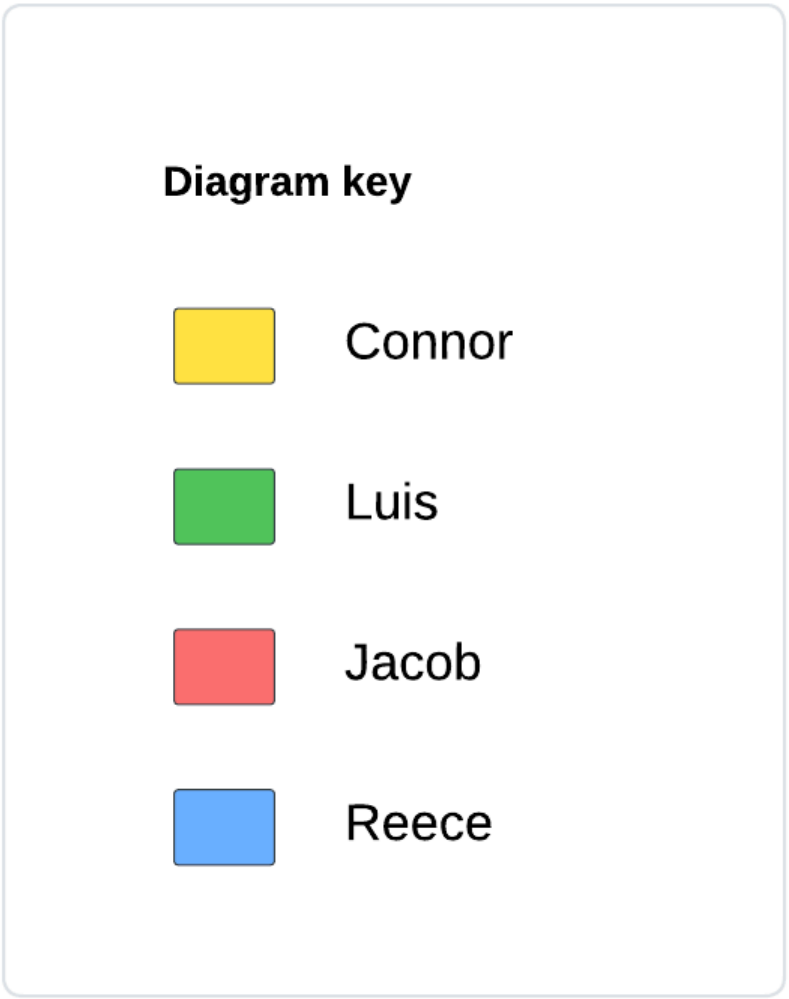


Figure 2.2 Program Flow Chart

2.5.2 Hardware Diagram

Similarly to the previous diagram, the one depicted below represents the interactions of the hardware components in our system. These aspects mark the most important components of our system's operation and have the most impact on the success of our project. For this reason, the power supply, sensors, GPS, and motors are all listed. These are all controlled by utilizing an Arduino microcontroller and communication is carried out utilizing an I2C method (these can be seen in more detail within the corresponding areas in sections 4 and 5). These aspects were also the first to be tested in the prototyping section of the project.



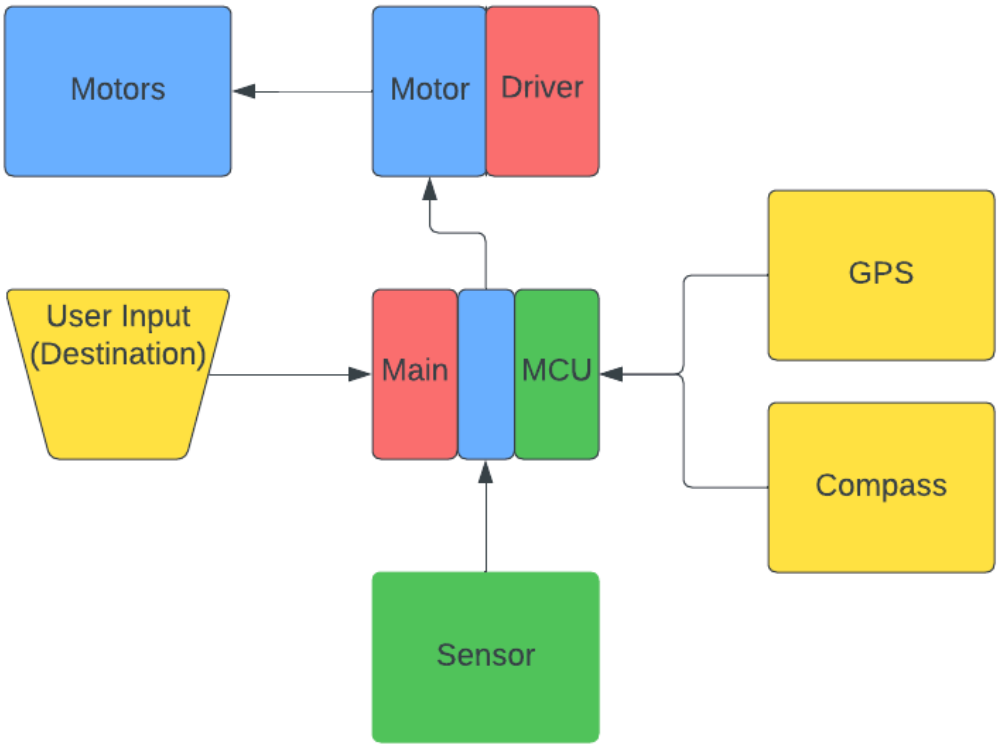


Figure 2.3 Component Flow Chart

Section 3: Research and Part Selection

Section 3 focuses on the research and the selection of the most fitting parts to be used on the project after comparing them to different parts of the same category and studying how they could be implemented on the project. The areas that will be explored further are the type of sensors, the power source, the microcontrollers that will be used for the parts, GPS system, PCB, the chassis to hold the parts, and the motor.

3.1 Sensors

Sensors are a key component that must be incorporated on the delivery bot, this will play a very important role in letting it be autonomous, so it can navigate around the routes successfully without bumping into objects, being able to turn on the appropriate moments, stay on the correct path without getting stuck and such. The sensors work in a similar way as how an autonomous car's sensor works but in a smaller and more basic way, the sensors are not as sophisticated as an autonomous car as the safety of pedestrians and other drivers is not a big risk and our vehicle will not be going at high speeds.

3.1.1 Types of Sensors

We explored different kinds of sensors and determined which one is the most cost-effective and functioning solution. A solution is a sensor that can send out a signal and record if the signal bounces back from an object to determine if there is something to be avoided.

3.1.2 Ultrasonic Sensors

Ultrasonic sensors work by sending out a soundwave at a very high frequency and then, if this signal stumbles upon an obstacle, it bounces back to the sensor to find out how far it is from the sensor. They do not need to make physical contact with the obstacle which is an advantage. The major advantage of ultrasonic sensors is their price. They are on the lower end and are very affordable, another advantage is that they are less likely to be interfered with by conditions like smoke, mist, or darkness.

Some disadvantages of this type of sensor are that some other sensors could have a faster response time compared to the time it takes the ultrasonic sensor to get the signal back and the surface in which the bot is traversing could affect the accuracy of the sensor as well, because if the surface has the capacity to absorb acoustic signals it could reduce the accuracy of the sensor. A possible solution to some of the constraints for this sensor is to use multiple sensors and position them in an effective way to avoid interference.

Figure 3.1 Sensor

3.1.3 Infrared Sensors

Infrared sensors work by emitting a light wave and like the ultrasonic sensor it waits for the light wave to reflect back from the object and then estimate the distance from the object to the sensor. They also don't need to make physical contact with the obstacles. A big advantage of these sensors is the size. They are considerably small and light so it helps with the dimensions aspect of the bot, they are not very expensive. It doesn't matter if the surface they are on absorbs the acoustic waves; they are also good when there are hard visibility conditions like darkness or snow.

On the other hand, they have a big disadvantage which is the range, it is very small compared to other sensors, so it could work as a proximity sensor for short distances but not good for general purposes, another disadvantage is a condition like rain that obstructs the beam of light will reduce the accuracy greatly; overall useful for indoor uses but not very outdoor for outside uses.

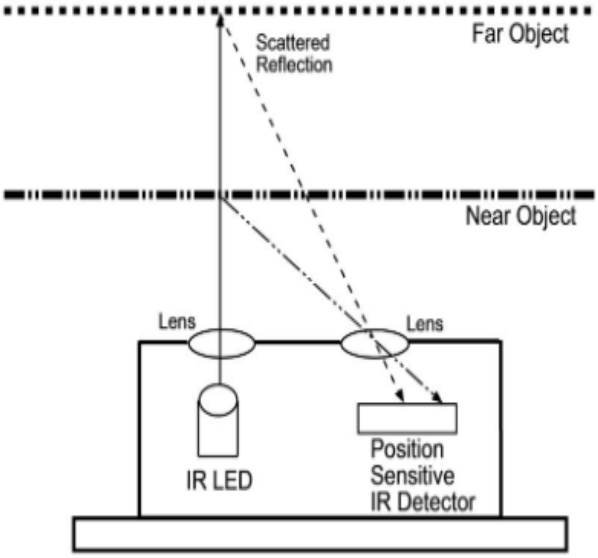


Figure 3.2: IR distance measuring. Source: seedstudio.com

3.1.4 LIDAR Sensors

LIDAR (light detection and ranging) sensors are the type of sensor that is used the most for autonomous driving cars, the way it works is that it uses laser technology by projecting a lot of light points in little time to try and recreate a 3D image and the map the surroundings as it can project in a 360-degree range of view from the center point of the vehicle. This tool gives a great advantage in the range of view of the bot as it maps around the whole bot and is not constricted in a certain narrow field of view it has a lot of room for improvement with AI that can be developed in the future.

Some disadvantages of LIDAR are they are more expensive as they are somewhat newer, they need a considerable amount of power also a higher computational power is needed for LIDAR to process the input around the vehicle, snow and fog can greatly disturb the accuracy of the sensor since is light based, acoustic wave absorption is not a problem for this sensor, the dimensions of these sensor are usually bigger than the ultrasonic and infrared.

3.1.5 Radar Sensors

Radar (radio detection and ranging) sensors function by sending a lot of radio waves in a direction and then it waits for a response of the radio signal to measure the distance to the obstacle, they receive the signal bounce back response faster than ultrasound. This sensor works well in harsh weather conditions like rain, fog, or snow, it could have long range as well as short range, works in different types of terrain and is very good for determining the speed of another object compared to the bot.

Some disadvantages of using radar sensors are that it detects the objects at a lower resolution compared to LIDAR for example, is more expensive and consumes more power than the first two options, the output has a bigger size that has to be computed. Radar sensors are great to complement the vehicle with some other sensors as they can be relied on if it's raining harshly.

3.1.6 Cameras

Cameras are some of the most popular and first ideas used for autonomous vehicles as cameras can be used to mimic a human in the way that it sees the objects and the route, because is one of the oldest tools used it has a lot of support and auxiliary tools to make a better and smoother integration to with the

vehicle, they are also easier to train with AI as the data a camara gathers is easier to work with than other sensors like LIDAR.

On the other hand cameras are very data heavy since it gathers videos the computing processing it needs its very big, harsh weather conditions render them almost useless since is like a human eye that if it cannot see it does not work, if it gets dark it does not work without some sort of support like headlights or something like that, if an object's color is very similar to the background the camera might get confused and not see the object, they are more expensive to implement.

3.1.7 Thermal Camera

Thermal imaging has begun to regain popularity in recent years especially for the use of autonomous vehicles. It has all the advantages of a regular camera in the way it works and the tools developed for regular cameras a lot of times also work for thermal cameras. It also has some extra advantages like it can detect extra hazards that include temperature like fires it can also detect animals and humans even if it is dark.

The main disadvantages are still very similar to the regular camera, which are price, size, data size, computation and energy consumption are very big. It improves the regular camera on the harsh weather conditions and poor lighting uses.

3.1.8 Chosen Sensor: Ultrasonic

The selection process for the sensors is very important for this project as the sensors are what will make the automation successful. One of the most important aspects for this selection was the cost of the sensor and it seemed like ultrasonic sensors had the upper hand over many other sensors that were explored cost wise. The technology behind ultrasonic sensors is not a new technology giving it the upper hand with support and examples that use this type of sensor. This makes the implementation easier and faster to solve any issues that might arise. The size of the sensor is the second to last as the smallest compared to the other options explored making it better for the bot as they can be placed more strategically.

Ultrasonic sensors are not as fast as some of the other options like the infrared or LIDAR but it does have a better range and is a little more reliable when it comes to weather conditions like fog or if it's dark. The sensors do not need to have the fastest response rate because the bot will not go over two miles per

hour for speed. The group has experience with this type of sensor as it was used in a previous class to design a distance measurer, this is an advantage over any other sensor on this list that the group has never worked with before.

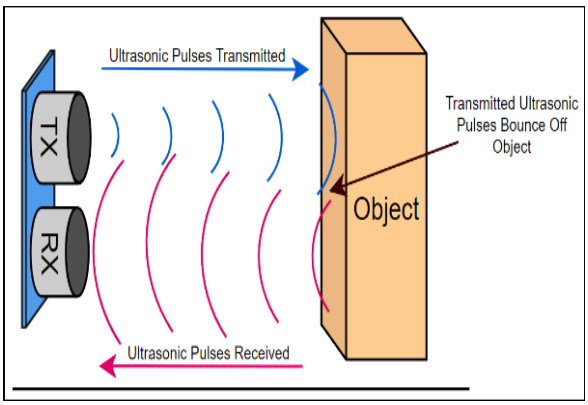


Figure 3.3: Ultrasonic sensor work principle source: packtub.com

The ultrasonic sensor is based on the principle of echolocation, so it works by generating ultrasound transformed from electricity oscillating a charge to piezoelectric crystal in the sensor's transmitter that will generate a sound wave directed at an object. The wave bounces back from the object to the receiver that contains a piezoelectric receiver and through this is converted back into electricity this principle can be observed on figure 4 above.

The way that the ultrasound sensors will work is that they will be connected to a microcontroller that can interpret the time signals into distance. Microcontrollers can do this by using equation 1 below, the echo pulse width is obtained from the difference of acoustic burst and the reflected signal (propagation delay) and it depends on the distance.

$$Distance(cm) = \frac{echo\ pulse\ width(uS)}{58}$$

Equation 3.1 Sensor distance

3.1.9 Table Comparison for Sensors

The sensors are compared below based on their characteristics and how they measure against each other in range, power usage, strengths, weaknesses, cost, signal response time and implementation complexity of the Ultrasonic, infrared, LIDAR, Radar, Camera and thermal camera. Also, the chosen sensor is indicated by being highlighted on the chart.

Sensor			
Ultrasonic	Works well in different lighting conditions like darkness and very bright	Longer range, Less computing power necessary lowering the cost and stress on the processor.	Works better in situations like Fog.
Infrared	Smaller than ultrasonic	Slightly cheaper	Faster signal bounce back time than ultrasonic
LIDAR	360-degree field of view possible for mapping.	Newer technology can be developed further and it will stay relevant longer than ultrasonic.	Not too many units necessary.
Radar	Best for rain situations	Faster signal response than ultrasonic.	Great secondary sensor for specific situations if budget allows
Camera	A lot of support for this technology as it is used for autonomous cars	Works accurately even on surfaces that could absorb the acoustics	Is the one that resembles the human eye the closest in the way it processes images
Thermal Camera	Can identify hazards like fires or obstacles like animals and people based on their temperature	If an object has an erratic temperature it can confuse the camera sensor	Better for cold places as it makes it easier to identify obstacles with higher temperatures.

Table 3.1: Ultrasonic vs infrared vs LIDAR vs Radar vs cameras

3.2 Power Source

The delivery bot needs to have a power source otherwise it will not move or do any task, making this one of the most important components of the whole system it needs to power the motor, the sensors, the GPS system, the processor, the

microcontroller. Because the bot needs to move around a radius of at least two miles to make the deliveries by itself, a wireless independent power supply is needed for it to keep moving after leaving the home base. The power supply design is a very important step because a flawed design that delivers too much power or too little power to a system could be harmful to the components. Some key aspects for this design are the main power source like a battery, voltage regulator to supply the components and heat dissipation to avoid frying the components in the system.

3.2.1 Battery

There is a really wide selection of batteries that can be used for projects, for this specific project, the ideal functioning time of fully operation is at least one hour. To get this information batteries will have to be compared to get one with the characteristics of a good continuous discharge rate, not too big in size and weight to make space for other components, preferably not too high temperature to avoid overheating on the battery; if the battery overheats and fails the whole system will fail and that is very bad news if it happens.

There are two main types of batteries used for this type of systems like RC cars, battle bots, etc. These are Lithium-ion (Li-ion) and Lithium Polymer (LiPo) batteries. Lithium-ion batteries are the older of the two options this makes it have more support and to be less expensive but also a little less effective than the newer option, they consist of a positive and a negative electrode, the chemical liquid in between and the power and discharge regulator they can handle many discharge/charge cycles but they have a very small possibility of catching fire and exploiting like it happened with cell-phones for a period of time.

Lithium polymer batteries are build the same way as lithium-ion but instead of the liquid chemical they use solid-like chemical polymer, they are newer which makes them more pricey than the previous option but they are also more safe, they will not leak liquid electrolyte and explode, they can be more flexible in size compared to lithium-ion so its better for component arrangement, it has a great lifespan making it more reliable, it is very lightweight so it would influence the payload of the bot and power consumption in a positive way, it has a high capacity to hold in more power. Below is a table comparing the two types of batteries.

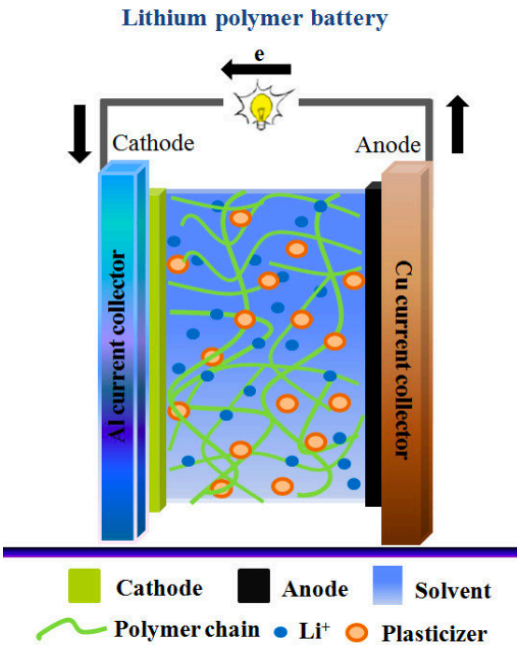


Figure 3.4: LiPo battery

	Lithium-Ion	**Lithium Polymer (selected)**
Cost	Cheaper	More expensive
Dimensions	Bigger in size	Smaller, more flexible
Safety	Small chance of exploding	Should not have an explosion concern
Lifespan	Charging capacity gets smaller overtime	Better charging capacity holding overtime (≅1000 times more effective)
Weight	Heavier	Lighter
Nominal voltage (V)	Lower; could reach 7.4	Higher; could reach 7.6

Recharge cycles	More recharges; 400 to 500	Less recharges; 300 to 400
Energy density (WH/KG)	Higher density; 100 to 250	Lower density; 130 to 200
Operational temperature	Can reach higher temperatures	Can be kept under 50℃

Table 3.2: Lithium-Ion vs Lithium Polymer batteries

This project uses a Lithium Polymer battery based on the given characteristics compared to a Lithium-Ion battery.

3.2.2 Voltage Regulator

The voltage regulator is another key component of the power supply system, this is what will supply the components with the correct amount of power so they don't get destroyed by excessive voltage or if they get too little voltage they will just not have enough to work. Most electronic systems run on 5V or 3.3V DC, in this case every component has a different input voltage needed hence the need of a voltage regulator.

There are two main types of voltage regulators: the switching and the linear. The switching specializes in transferring the most of the input power to the output trying to be as efficient as it can, without dissipating too much power by being this efficient and not dissipating too much power they do not produce a lot of heat but this means that it cannot dissipate too much power and it will probably need some extra components like capacitors, feedback resistors, inductors, etc. The switching voltage regulator's active pass element acts as a switch so it either stays in on or off state, by adjusting the time the switch stays in these states the output voltage gets regulated to a constant value.

The linear voltage regulator is the original kind of voltage regulators, making them have a very simple and straightforward implementation, because they are the older technology, they are cheaper compared to switching voltage regulators. They tend to dissipate a significant amount of power as heat, we need this in this case as the output voltage we are trying to get is DC voltage of 3.3V and 5V coming from a 7-8V power source. There are many types of linear voltage regulators, all we need for this project is an LDO (low-dropout regulator).

	Switching Voltage Regulator	**Linear Voltage Regulator**
Price	More expensive	Cheaper
Dissipation Capabilities	Less dissipation	More Dissipation
Implementation Complexity	More complex design	Simple design
Operating Temperature	Will stay at a lower temp	Will raise the temperature due to the power dissipation being transformed into heat

Table 3.3: Switching vs Linear voltage regulator

The LDO needed for this project are two: one chip for the 3.3V output and one chip for the 5V, the choice of an LDO should work fine, they are small in size and very effective dissipating the power and dropping to a smaller DC voltage. LDOs consist of open collector topology, in this topology the transistor can be easily driven to saturation letting it easily drop the voltage because the DC is limited to the saturation limit of the transistor.

The Quiescent current is another characteristic of the LDOs also known as the ground current, this accounts for the difference between input and output currents, when the LDO is idle it will draw a small amount of Quiescent current to keep the circuitry just in case.

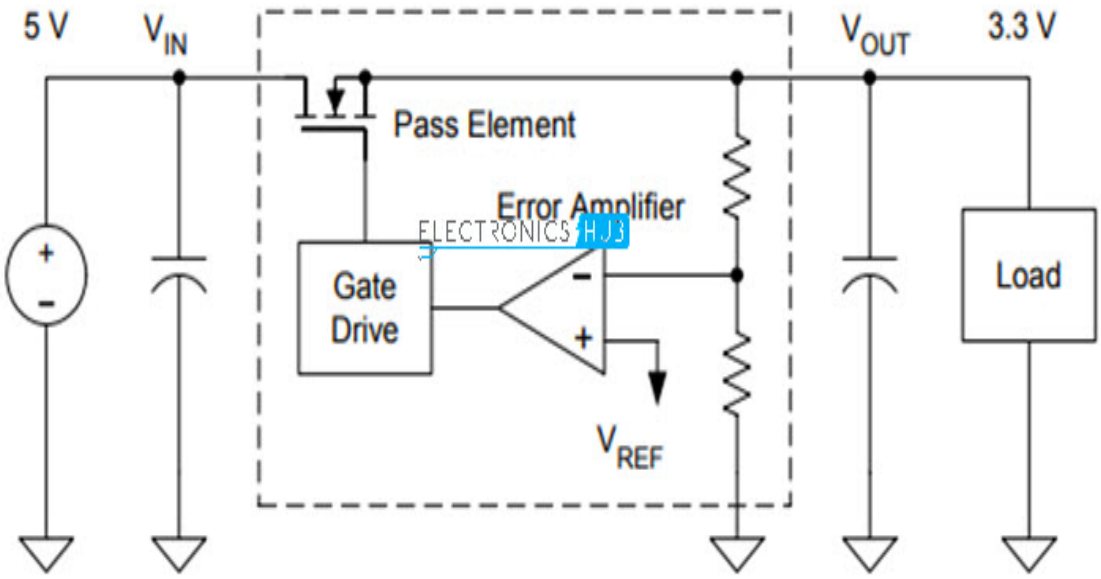


Figure 3.5: Example of a Low dropout voltage schematic for a 3.3V source: www.electronicshub.org

For the 3.3V not too much current will be drawn, so no more than 1 amp will be acceptable, for the 5V the max current output will have more importance, so we will need a higher rated maximum output current to make sure the regulator is not working at its maximum capacity because it could heat up. Another 5V voltage regulator is needed for the car motor. Because the terminal voltage might be variable, that has to be taken into account so the most amount of potential energy can be used during the battery's lifespan, voltage drops and voltage tolerance have to be taken into account.

The voltage regulators were monitored to supervise for failures so for example, if there is a system failure in one of the sensors it could send an alert to the user to prevent the delivery bot from getting lost in an unknown environment.

3.2.3 Heat Dissipation

A big challenge for the project is that it is meant to be outdoors for the deliveries, so it will have to withstand the elements in specific to look in this case the Florida sun. The sun can cause some serious heating of the bot and its components plus the power drawn over time will heat the system's components. Every component has a temperature threshold that can reach without causing problems to them. If the sun and the heat created by the power conversion are not handled properly, the heat in the bot will go above the thresholds.

A cooling method is implemented to keep the temperatures under the thresholds for the included components. The implementation involves using enough ventilation and laminar airflow to keep the system and the components ventilated when moving plus the addition of heatsinks to the voltage regulators to safeguard the battery from overheating and avoid losing the battery and power on the robot.

3.3 Microcontroller

The land delivery bot uses a microcontroller unit for processing the data gathered by the sensors and some of its other components to complete the tasks that will make the bot be a successful autonomous delivery system. The same microcontroller is used for the sensors and other components to make the implementation and development of the project simpler and faster, this decision will also help to the maintenance of the bot because after the first microcontroller is confirmed to have worked, the team will know how it works and it will make the implementation and troubleshooting easier.

The ideal microcontroller should not be too big or heavy and preferably it should not consume too much power because if it is too powerful and not all the functions are used it will be a waste of battery power that could be used somewhere else. Ideally the microcontroller should come with a development board so extensive testing of the functions can be done before assembling the components into the delivery bot, this can be done with the microcontroller's development kit, the components, and a breadboard.

3.3.1 MSP430G2553

This is one of the first if not the first microcontroller development kit a student uses for school, is very low cost and everybody on the team has worked with it before. The MSP430G2553 has 16 KB of flash memory and 512B of RAM; it runs on a 16-bit RISC architecture. It works with the program Code Composer Studio developed by Texas Instruments and it has a USB access that can be connected to CCS and be easily programmed. One drawback is that the MSP430G2553 has only 20 pins and very low power.

3.3.2 MSP430FR6989

The other obvious option to consider is the MSP430FR6989, because the team has also already worked with this microcontroller in two previous opportunities and are familiar with it; it is a more functional version of the previous MSP microcontroller. It has better security features, it is still low power consumption which is great for the purposes of power saving, it includes a 16MHz clock, AES-256 encryption and decryption module, temperature sensor on the chip, it

also supports USB communication with CCS making it easy and familiar to program and debug. The memory consists of 128KB flash memory, 2KB of RAM and 8KB of FRAM. The microcontroller has 32 GPIO pins.

3.3.3 ATmega328p

The ATmega328p comes with the Arduino Uno development kit which makes prototyping and designing easier as it includes USB to connect to the code editor and debugger. This is a widely used microcontroller which is very good because a lot of resources are available online for learning the basics of it and to solve any issues that may arise. It is very versatile and is not too expensive.

It has a clock of 20MHz, can use voltages from 1.8V to 5.5V, it has 6 analog to digital converters which are important to interpret signals, 2 pulse width modulation channels. The memory is 32 KB of flash memory, 2KB of SRAM and 1 KB of EEPROM. The microcontroller has 23 GPIO pins.

3.3.4 ATmega2560

The ATmega2560 comes with the Arduino Mega 2560 development kit, which is designed to work with a USB interface to use the code editor and debugger for convenient prototyping and debugging. The main advantage of this microcontroller is the number of pins with 54 GPIO pins and 16 analog input pins; this is good for connecting all the sensors. It has power saving modes for when it is idle to save battery life. It has 16-bit timers and for memory it has 256KB flash memory, 8KB SRAM, 4KB EEPROM.

3.3.5 Broadcom 2835

The Broadcom 2835 comes with the very popular Raspberry Pi “single-board computer” but after further investigation it was discovered that it is not too much of a microcontroller but more of a “system on a chip” this makes sense as that is the whole mission of the raspberry pie, but the Broadcom even has its own processor and GPU, because it is not a microcontroller itself it will not be used or investigated further.

3.3.6 STM32F302VCT6

The STM32F302 is designed for low power applications like it would be ideal for this project, this microcontroller supports advance analog and digital signal processing, this feature makes this microcontroller ideal for the tasks our bot needs to perform like controlling the motor for movement, audio conversion for

the ultrasound sensor's signals, it has good security like hardware CRC calculator and a tamper detection module.

Based on the ARM Cortex-M4, it has 256 KB of flash memory, 40 KB of SRAM and 8KB of EEPROM. This microcontroller can be found on the STM32F3DISCOVERY development board, is a low-cost board that also comes with an ST-Link/V2 for programming and debugging the board.

3.3.7 Comparisons

A total of 5 microcontrollers were deeply researched to figure out which microcontroller is the best suited for this project and that can be used for every task needed without having to use different microcontrollers for different components but instead just using the same one multiple times. This comparison will be made based on the specs researched; the specs are: Price, number of pins, resource availability, performance, memory, efficiency, power consumption and reference material availability.

All these microcontrollers seem to be almost even in availability and reference material available online, the only ones at an advantage in this aspect are the MSPs chips that have been used in previous projects by team members but the other have plenty of documentation and resources that makes that familiarity not a deciding factor.

3.3.8 Price

The price of the microcontroller and development board is an important aspect to take into consideration, because the budget for the project is trying to not surpass an amount, the developers of this project is a group of college students after all, so ideally the price of the components should not go past $8 unit price. By keeping the price of the components low also if the delivery bot is a success that can be sold, by keeping the manufacturing price low, the final product can be sold at a competitive price.

Unit	Price	Total (10 units)
MSP430G2553	$3.60	36
MSP430FR6989	$9.50	95

ATmega328p	$4.00	40
ATmega2560	$9.00	90
STM32F302VCT6	$5.00	50

Table 3.4: Microcontroller price comparison

In this category the better priced microcontrollers are the MSP430G2553 and the ATmega328p by a range of almost half the price of its competitors. The price difference adds up when the need of buying more than one component arises because it gets multiplied by 10 for the whole system.

3.3.9 Number of Pins

The number of pins is important for the microcontroller selection as if it has too low of a pin count it would be impossible to attached it to all the corresponding components that will need to control or gather data from and because the same microcontroller needs to be used for every component that needs one this is important. Is better to have some extra pins and not use them than needing some extra pins and not having them.

Pins	GPIO pins	Total pins
MSP430G2553	16	20
MSP430FR6989	74	80
ATmega328p	23	28
ATmega2560	56	100
STM32F302VCT6	51	100

Table 3.5: Microcontroller pin count comparison

For this metric the best characteristic is the GPIO pins and the best microcontrollers that shine in this category are MSP430FR6989, ATmega2560 and STM32F302VCT6.

3.3.10 Memory

Memory is crucial for choosing the correct microcontroller as this will dictate how fast data can be processed and how much data can be store for processing or buffering inside the microcontroller without having to go all the way to the processor, ideally the most amount without being overkill so it doesn't kill the battery when idle or during use is the best choice.

Memory	Flash	SRAM	EEPROM
MSP430G2553	16KB	512 bytes	512 bytes
MSP430FR6989	128KB	2KB	2KB
ATmega328p	32KB	2KB	1KB
ATmega2560	256KB	8KB	4KB
STM32F302VCT6	256KB	40KB	8KB

Table 3.6: Microcontroller memory comparison

There are some good memory options among these microcontrollers, the one with the most memory overall is the STM32F302VCT6 but it seems like overkill not all that memory is used for the purposes of this project the best options seem to be the ones right in the middle the MSP430FR6989 and ATmega328p.

3.3.11 Power Consumption

Power consumption will play a big role with the microcontroller because multiple of them are being used so if all of them are drawing a lot of power from the battery then little power will be left for the rest of the components is about efficiency and not using more than needed even when idle.

	Power Consumption (active)	Power Consumption (power - down)	Power Consumption (standby)	Lowest Operating Voltage
MSP430G2553	320μW	0.1μW	0.1μW	1.8V

MSP430FR6989	143µW	1.1µW	0.5µW	1.8V
ATmega328p	6.2mW	0.1mW	0.5 µW	1.8V
ATmega2560	22mW	0.1mW	0.5µW	1.8V
STM32F302VCT6	12.4mW	1.5µW	0.28µW	2.0V

Table 3.7: Microcontroller power consumption comparison

In this case the ideal microcontroller is the one that consumes the least power in all modes but it is still able to perform the needed tasks. The best in this category are the MSPs microcontrollers and the ATmega328p.

3.3.12 Chosen Microcontroller: ATmega328p

The chosen microcontroller was the ATmega328p that was at the top of almost every category compared in above tables it is at the tops of power consumption, is right at the middle in memory uses which is good because no overkill is needed, it has the second to best price and it comes on the Arduino uno development kit that has a lot of online support and it has the characteristics of working efficiently with sensors and motors which are the basics of this project.

unit	Price (10 units)	N° of GPIO pins	Flash memory	SRAM memory	EEPROM memory	Power consumption (active)
MSP430G2553	$36	16	16KB	512 bytes	512 bytes	320µW
MSP430FR6989	$95	74	128KB	2KB	2KB	143µW
ATmega328p	$40	23	32KB	2KB	1KB	6.2mW
ATmega2560	$90	56	256KB	8KB	4KB	22mW
STM32F302VCT6	$50	51	256KB	40KB	8KB	12.4mW

Table 3.8: Microcontroller selection summary table

3.4 GPS

This project makes use of a GPS device to help the delivery drone effectively navigate its surroundings. The device is connected to the main microcontroller and constantly feeds in location data.

3.4.1 Background

A GPS, or Global Positioning System, is a device that makes use of the system of satellites created by the world governments allowing it to pinpoint the device's location on the planet. GPS' are used not just to determine the current location, but to also determine a path to another location which may be a desired destination. They are used all around the world in a variety of different applications and one of the more commonly seen ones is delivery. GPS devices are very effective for helping to deliver packages to a location because they can allow the sender to find a path to the receiver. This is where our project comes in. Our delivery drone intends to make use of GPS to find the physical location where it needs to deliver the package and to create the route that it needs to follow to successfully deliver that package.

A GPS device makes use of the 31 satellites that are set up in the Earth's orbit in order to receive information about its current location. The device is set up to receive the signals that these satellites are constantly sending. These signals are encrypted and contain information about the location and time. Once signals from at least three satellites have been received, the device performs calculations essentially using a form of triangulation called trilateration to determine its own and the satellites positions. Then, the device uses a fourth signal from a fourth satellite to verify this information.

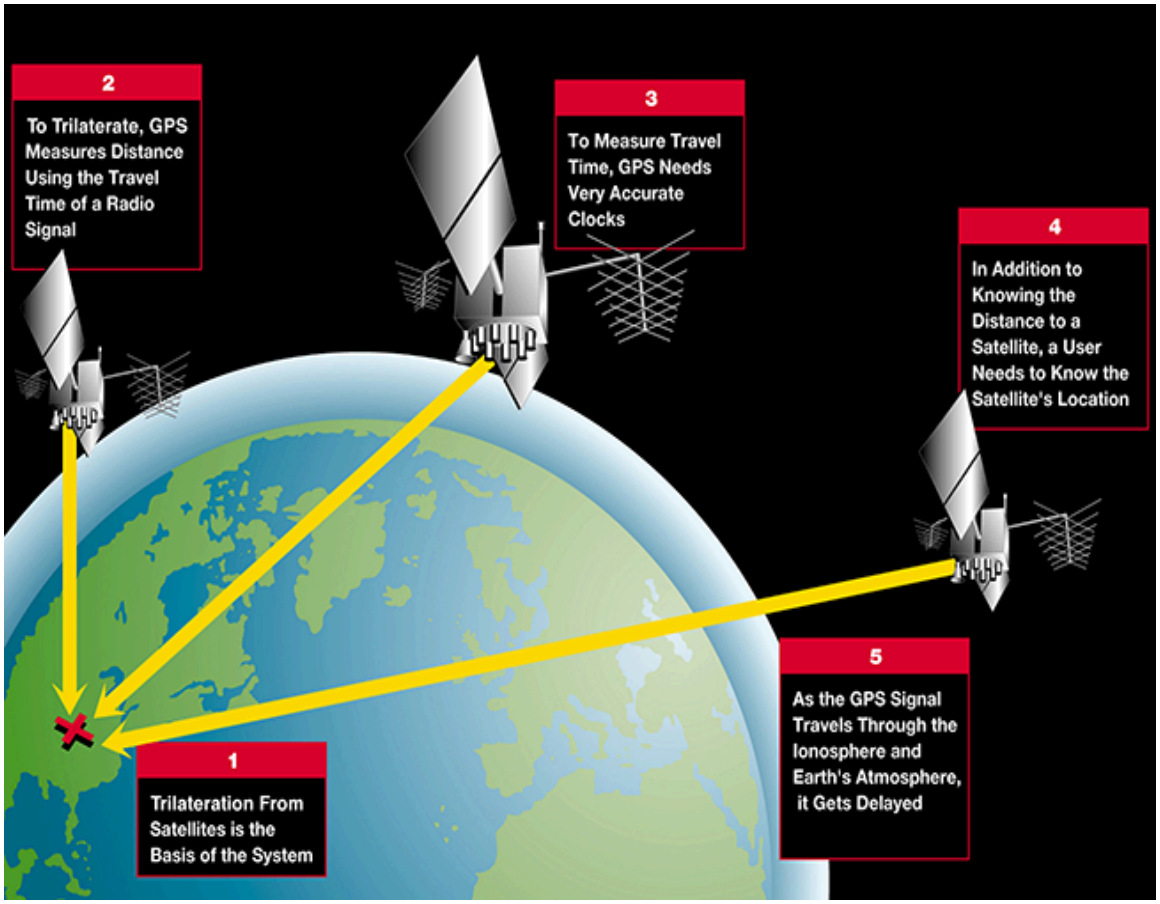


Figure 3.6: How a GPS utilizes satellites
Source: www.faa.gov

It is important to note that, for this project, we did not design our own GPS device as that is much beyond the scope of the project. We instead purchased a pre-built GPS device from among the wide selection available online.

3.4.2 Why Use a GPS?

The use of a GPS device for this project is very important to the success of the project. The GPS device is what is used to help the delivery drone navigate its way around the campus and allow it to greatly improve its pathfinding abilities. The GPS device will help the drone be able to find a valid path from a complex series of walkways and therefore allow the drone to move between locations successfully and properly with little to no issues. One of the main issues of the drone is relatively large-scale movement. This refers to the overall movement across larger distances rather than simply turning and moving forward or backward. The drone not only needs to know what its overall destination is, but also how to get there. The addition of a GPS device will allow the drone to do both of those things. The drone is able to make use of the information provided by the GPS device to know where it currently is located and then proceed to

compare that with where it needs to go and whether it is going the correct direction. The GPS device will also allow the drone to create an ideal path to follow before it actually starts moving which will greatly help its efficiency. The GPS device will also help with the overall safety of the drone since it will help the drone not run into things that could damage it or potentially harm others.

3.4.3 Aspects to Consider

All GPS devices involve wireless communication in regards to communication with the global satellite network so we do not need to consider the overhead for wired connections in regards to that.

1. Size

GPS devices are often relatively small and portable. Since they are usually attached to something or carried around on a vehicle, they cannot be too large or too cumbersome for the vehicle to handle. The device itself can vary in size from around the size of a standard CPU to the size of the average smartphone. To be more specific, GPS devices can range from around 1.5 x 1 x 0.5 in. to around 8 x 5 x 3 in. This range of sizes works very well for our delivery drone project. The drone itself is relatively small and must carry a package as well. Therefore, we cannot afford the GPS device to take up too much space if we want to stay within the size constraint of our drone.

2. Weight

GPS device weights can also vary respectively. Since the global satellite system, and by extension the GPS, is a relatively new concept compared to something like a pulley or a motor, it is often made with newer technology that consists mostly of silicon. This leads the weight of the GPS to match what is expected of something with a size like the one mentioned above. The device usually ranges from around 2 ounces to around 2 pounds and not too often more than that. This also works very well for our delivery drone project because this means that the GPS system should not add too much to the drones' total, no-package weight. Ideally, we want this no-package weight to be as little as possible so we leave a wider available range for possible packages. This is also very helpful in choosing what motor we want to use for the drone as more weight means we need a stronger motor.

3. Touchscreen vs. No-Screen

GPS devices often come in two types: touchscreen and no-screen. Touchscreen GPS devices are the more commercially common ones as those are the ones used the most when commuting between places. Delivery drivers often have them set up on the dashboard of the vehicle and use them to route the way to their delivery location. The touchscreen devices often land on the larger end of the range of the sizes and weights for GPS devices as they must include a screen for the user to interact with the settings of the device. No-screen GPS devices do not include this screen for user interaction and often solely consist of the technology for the device itself along with a way to read the information usually via some wireless connection. These GPS devices often range on the smaller end of the range of sizes and weights as the device itself only consists of what is required. These types of devices are often paired with some application, usually for a smartphone, which is where the user can interact with the device. Essentially the “screen” part of the touchscreen device is moved to be the same screen as your smartphone. Our delivery drone is unmanned once the package is placed on it and the delivery location is set so the no-screen type of GPS is the one we are using. We simply do not need the extra overhead that touchscreen GPS’ often come with if the user is not actually interacting with it. All a user needs to care about is where the package is going and not necessarily the route to get there. The user can also make use of the functionality of the external application to track where their package currently is. Simply put, the overhead of a touchscreen GPS is unnecessary so we will be using a no-screen GPS device for our delivery drone.

4. Cost

GPS devices can have a widely varying cost. They can range from around $20-30 to upwards of around $200. The range of prices varies due to not just the performance of the GPS, but also due to user accessibility. Touchscreen GPS’ often cost more than no-screen GPS’ because the device includes many more user functions that a no-screen GPS would not have included such as traffic information, weather information, voice-controlled settings, voice-assistance, etc. These functions would normally be useful for someone driving a vehicle, but for our unmanned delivery drone project there is no point to consider them as that information will not be useful at the scale we are aiming for. Our drone simply needs to know where to go and what path to take to go there which a lower-priced, no-screen GPS can easily provide.

5. Power Consumption

GPS devices are often relatively low-power because they are in the category of electronics. However, it is still important to consider the power consumption of the GPS unit because it will still affect our overall battery life and overall

operation time. The range of GPS power consumption is usually a standard voltage range of 3-5V and it is generally assumed that the current range is around 30-40mA. This is a relatively safe assumption, however when constructing our drone and considering overall power costs, we will be much more generous and assume a higher power consumption for the GPS device.

6. Implementation

The implementation process of the GPS device seems like it would be simple. However, there is more to it than it might seem at first. The main issue is how to transform the location information that the GPS device provides into information that can be used to determine a path. One of the touchscreen functionalities that is often included is Google Maps or Apple Maps or other map type support built in so that it is easier to visualize where you are and where you need to go. No-screen GPS' devices do not have that and yet that functionality would be very useful to our drone because it would help the drone's routing ability. Therefore, we must create a system that can do that data transformation.

The system should have three main parts: the GPS device, the data transfer, and the map connection. We will likely use Google Maps as our map connector because the API is free and open source and it is relatively simple to implement. We will go into depth on this process further into the paper but the main idea is: take the data readings from the GPS device and send them to the main controller which will make use of the Google Maps API along with another third party API to effectively simulate Google Maps which will allow the drone to create a valid route to get to its location. As mentioned before, this process will be explained in more detail in a later section of the paper

7. PDOP Value

Another important aspect of the GPS device is its PDOP value. PDOP stands for Position Dilution of Precision and this value essentially tells the user the accuracy of the measuring of the GPS device. Dilution of Precision is a term often used to describe satellites and their measuring capabilities. It describes how an error in the measuring of exact location can affect the actual determination of the location. If a location measurement had an error of a tenth of a degree in either latitude or longitude, then as long as the error of the other measurements is not too large, the actual location estimation should not be too far off. Remember the process that GPS uses to calculate a location uses at least four values; three for the proper trilateration and the fourth to check the accuracy of the other three. Therefore, any possible error in the measurements read by the GPS device is magnified if there is any error in the other respective measurements due to how the calculations are done. For example, if there is a lot of noise in the signal that the satellite sends that the GPS device is not able to filter out, then that noise will lead to possible error ranges in the calculations that must be considered and therefore lead to a less accurate overall calculation of location. Because all of

this is based on how well the GPS device is constructed and how good its capabilities to get around any possible error are, any error that does make its way to the calculations stacks up as more signals are read. All of this is culminated in the PDOP value which is actually divided up into two components: the HDOP and VDOP which stand for Horizontal Dilution of Precision and Vertical Dilution of Precision. More often than not is the HDOP value more important but since PDOP includes both values, both are usually considered. A low PDOP value is what is desired and that value usually caps out at around 7-8 for most people or applications. For our delivery drone's GPS device, we would like to aim for a PDOP value of around 2-3 so that we can be confident that the measurements that the GPS device is providing are correct. Small errors are acceptable for this application because we are more concerned that the drone makes it to the correct overall final location rather than being say a few feet farther from the doorway than expected.

8. Update Rate

Update rate is yet another aspect of GPS devices that must be considered when buying one. Update rate is simply how often the device is able to read a signal sent from a satellite. Initially it might seem like one would always desire a higher update rate because reading more signals more often can lead to more accurate data calculations. This is usually the case, however in the case of our drone we must also consider how much data our main control unit is able to parse and handle at once. If the GPS is reading and therefore sending too much data to the main control unit, it might get overwhelmed which will cause problems for the drone. We are aiming for a fairly average update rate of around 5-10 Hz which should not be too high for our main microcontroller to handle nor too low to not perform accurate enough calculations. In the case that 5-10 Hz is too much for our microcontroller we can always manually limit how much data is actually being sent to the microcontroller.

3.4.4 Broadcast Frequencies

GPS devices can also make use of different broadcast frequencies. As of 2021, there are three possible frequency categories that a GPS device can fall into. In order of date of introduction these are L1, L2, and L5. The L5 band of frequencies is a very new introduction and is not commercially available to us as students as of the time this project is being conducted so it will not be considered. The L2 band is also a newer band that is commercially available and consists of both the old C/A code, and older P code, and a newer L2C code. The L1 band is the original, legacy band that consists of the C/A code and the P code. The C/A, P, and L2C codes all have structural differences but their general uses are similar; they provide an effective way for the satellites to communicate data to the GPS devices on Earth.

P Code

The P (Precise) code was the first code developed for the Global Navigation Satellite System that was being developed. It was originally primarily for military use but that is no longer the case. The P code is a seemingly random string of ones and zeros that is repeated once every week. The string is actually not random and is instead particularly crafted at a rate of 10.23 million bits per second. Each satellite is assigned its own portion of the string and that portion helps a GPS device reading the data to know which satellite it is communicating with. For example, if the code that the device receives corresponds to the third section of the P code, then the device knows it is communicating with satellite three.

C/A Code

The C/A (Coarse/Acquisition) code is very similar to the P code except it is generated ten times slower at a rate of 1.023 million bits per second. The code is also repeated every millisecond instead of every seven days. This increased repetition allows for a more reliable and more consistent confirmation by the device. The C/A code is essentially a version of the P code developed for civilian usage and that is what it is primarily used for. Along with the P code, the C/A code is transmitted on the L1 range of frequencies and is the main L1 code that is used for civilian purposes. Despite being an older code, it is still used relatively often for an average GPS device.

L2C Code

The final relevant code that is used for civilian GPS devices is the L2C code. The L2 range of frequencies was always used in GPS since the start, however in 1998, the US government announced that the L2 range will be used for both civilian and military purposes instead of only military like it was previously. This was done by “separating” the current L2 code into two parts; one being the M code which is used in the military (as the M stands for military) and the other being the L2C code with the C standing for civilian use. The L2C code was introduced when the GPS Joint Program Office considered that it might be time for the C/A code to receive an upgrade as they found that it was very susceptible to distortion and interference that would cause much error in GPS devices. The L2C code improves upon the old C/A code by being divided up into two parts being the CM (Civil Moderate) and the CL (Civil Long) portions. Being divided into two parts allows each part to be longer than the C/A code and therefore provide better stability as it is easier for a longer code to be separated from the background. The CM and CL codes are transmitted at the same time using multiplexing which puts both of their effective transmit rates at 1.023 MHz which is the same as the old C/A code yet L2C is more stable and reliable than C/A because of what was just mentioned.

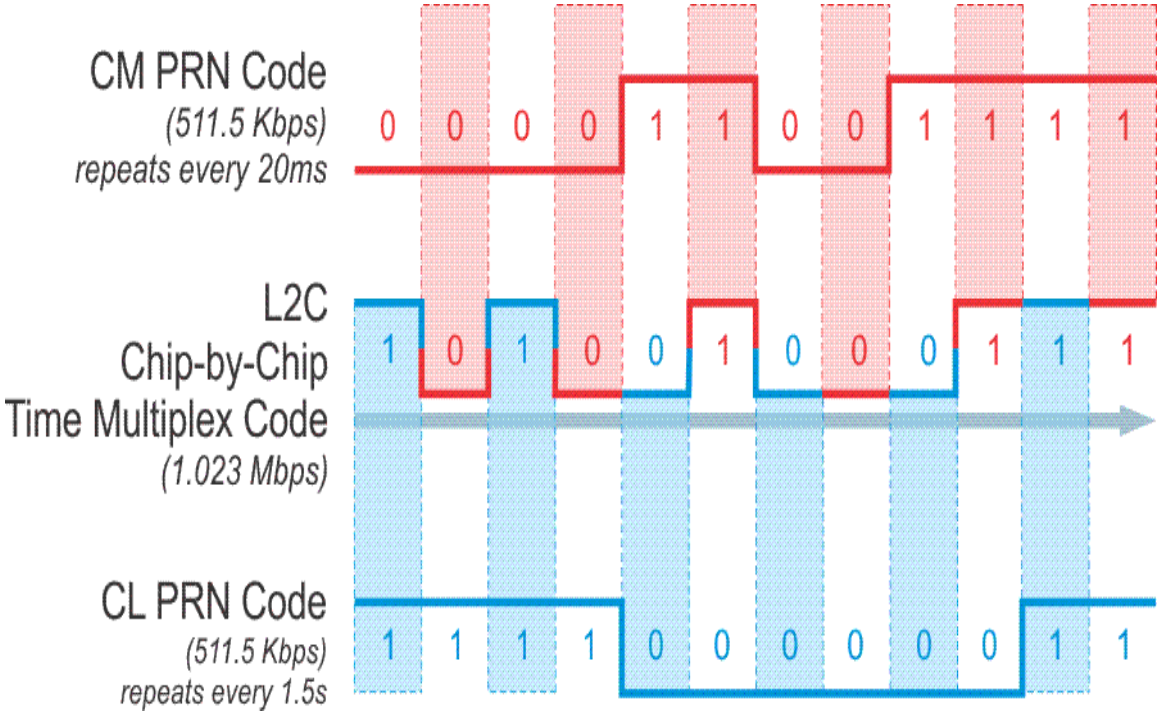


Figure 3.7: How L2C code is made
Source: www.e-education.psu.edu

This increased stability allows GPS devices that are centered around the L2C code to function more effectively in places where there is not a direct line of sight with the sky where previously the C/A code would cause GPS devices to struggle in this type of situation. The C/A code is still not completely useless however. Having two civilian level codes being broadcast at the same time, being the L2C and C/A codes, the impact of the biggest object of interference, the ionosphere, can be very greatly reduced by making use of phasing. Most current GPS devices will make use of this technique which allows them to provide a position measurement with a margin of error of about 5-10 meters along with the greatly increased effectiveness in areas where there is not a direct line of sight with the sky.

3.4.5 Standards and Requirements for GPS Devices

Since this project does not include building a GPS device from scratch, the engineering requirements and specifications for these devices do not necessarily apply to our project and, therefore, do not place too much of a limitation on what we are able to build and create for the project. That being said, there is one requirement specification that should be discussed.

The main specification is the operating frequency of the GPS device. GPS signals in the L1 band are designated by the International telecommunications

Union (ITU) to be in the range of 1559 MHz and 1610 MHz. To be more specific, the Navstar Global Positioning System, which is the Global Navigation Satellite System (GNSS) system that is operated by the United States is condensed to operate on the range of 1560 MHz to 1590 MHz with the signals being centered around 1575.42 MHz. This means that any operation outside of this range is not necessarily breaking any rules, but will lead to the device not performing as desired. In order for the device to receive what is essentially a “safety guarantee” from the ITU that promises protection against possible interference, a GPS device must have its receiver be limited to only receive signals from the above-mentioned designated range. This also means that any other devices that operate within this range are susceptible to possible interference from the GPS signals that are constantly sent out from the satellites.

We have come across three main potential options for our choice of GPS device to use for our drone. All three options are compatible with Arduino which is what the project will likely be centered around so there will likely not be any compatibility issues between the GPS device and the main controller. The three options are displayed in the table shown below along with some specification comparisons between the three.

Device Name	Arduino NEO-6M	Arduino NEO-7M	BS-280B
Manufacturer	MakerFocus	HiLetgo	Geekstory
Link	Amazon.com: GPS Module GPS NEO-6M(Ar duino GPS, Drone Microcontroller GPS Receiver) Compatible with 51 Microcontroller STM32 Arduino UNO R3 with IPEX Antenna High Sensitivity for Navigation Satellite Positioning	https://www.amazon.com/HiLetgo-Satellite-Positioning-Arduino-Replace/dp/B07X5GVW6Q/ref=pd_lpo_3?pd_rd_w=rUSiq&content-id=amzn1.sym.116f529c-aa4d-4763-b2b6-4d614ec7dc00&pf_rd_p=116f529c-aa4d-4763-b2b6-4d614ec7dc00&pf_rd_r=RGJJ0WSBSHC22ZHHT7B2&pd_rd_wg=nE4Jc&pd_rd_r=d610ec6d-5c68-4029-b394-6d59fbd728e5&pd_rd_i=B07X5GVW6Q&psc=1	Geekstory BS-280B GPS Module G7020-KT Chip with 4M Flash + IPEX Antenna, GPS NEO-6M Drone Microcontroller GPS Receiver for 51 Microcontroller Arduino Raspberry Pi High Sensitivity Navigation

Size	1.09 x 3.94 x 0.39 inches	3.82 x 2.32 x 0.35 inches	4.21 x 2.72 x 0.31 inches
Price	$10.99	$11.39	$19.99
Weight	0.32 ounces	0.634 ounces	0.317 ounces
Default Baud Rate	9600	9600	9600
Operating Voltage	2.7V - 3.6V	1.65V - 3.6V	3.6V - 5.5V
Average Operating Current	50mA	50mA	50mA

Table 3.6 GPS comparison

3.4.6 GPS Choice

The GPS we ended up choosing was the NEO-6M Module for Arduino. The reasoning for this lies primarily in the compatibility of the GPS module with the rest of the project. The three main options that were considered all had aspects and specifications that were similar to each other, therefore we had what was essentially free choice on which option we wanted as any differences between them were too minor to make a meaningful difference.

3.5 Printed Circuit Board

A Printed Circuit Board, or PCB, is one of the most used bases for making electrical circuits today. Before we started using PCBs, we used a method called point-to-point construction, which in simple terms, required big heavy machines that produced electronics that would be unfit for use based on size and reliability. Luckily, PCBs changed that world so we could work on creating smaller, more reliable electronics. That has allowed the creation of all the modern-day technologies we know today.

PCBs are used to assemble almost everything in the electrical circuitry domain. A Printed Circuit Board mechanically supports and links digital components using transmission lines, conductive tracks, and pads. These conductive lines are made of copper, which is etched or printed onto a substrate. They are generally sandwiched between a non-conductive substrate, allowing for multiple layering of different copper lines. The concept of being able to multi-layer provides for a

higher density/complexity of electrical components. The typical number of layers we see in the real world depends entirely on the board's application. For most smartphones, we can see around 12 layers; on average, we see anywhere between 4 to 8.

3.5.1 Current Issues

In light of the fact that there are currently shortages of essential materials that can be used to manufacture PCBs, the development and production of PCBs may suffer adversely as a result of the current global supply chain shortages. As a result, there is a shortage of PCBs being produced at the moment due to a lack of the essential materials necessary to manufacture PCBs. According to the current situation, copper foil and aluminum are the two resources that the country is lacking at the moment. They both play an essential role in the composition of printed circuit boards. We therefore anticipate that in light of this current shortage of PCBs, our order for a particular board will take longer to arrive after we place our order as a result of this shortage, which will also result in a higher production cost as a result of this shortage.

3.5.2 General PCB Makeup

As far as PCBs are concerned, there are four general substances that are used, no matter how many layers they may have. We have silkscreen, solder mask, copper, and substrate. When choosing between these substances, we are limited to silkscreen, solder mask, and copper when making our choice. There are a lot of options available on the substrate that can affect your board's performance.

Silkscreen

We will see that one of the common materials we will see for this is non-conductive epoxy ink. In the silkscreen process, we are able to print a layer of ink on top of the printed circuit board so that it can be seen from the outside clearly. As a result, the board can be easily printed and marked, since it does not conduct electricity. In the future, engineers will be able to identify test points, the polarity of parts, warning symbols, manufacturer's marks, and many other things with the help of this tool. When silkscreen marking is used, it will primarily be used to indicate components' locations and label pin functions. The ink can come in different colors and fonts, but the available colors are yellow, white, and black, considered standard among most PCB manufacturers when Silkscreen printing; there are three other methods that you could use to apply the ink. There are three methods available for printing a legend. The first method is manual screen printing, while the other is liquid photo imaging and direct legend printing. It is important to keep in mind that there are some guidelines that should be followed when using silkscreen, one of which is to keep the silkscreen on one side of the

board. Keeping an orderly board will also be easier if colors and shapes are held to the same standard.

Solder Mask

In a printed circuit board, a solder mask is a thin, generally opaque layer that is applied over the uppermost copper layer. This layer serves as a barrier against oxidation and prevents undesirable electrical paths from forming. Using a solder mask, PCBs help avoid unintentional electrical connections between traces on the board due to small solder blobs that can occur during the fabrication process. When hand-soldered assemblies are assembled, solder masks are not always necessary, however, when mass-produced boards are soldered automatically using reflow or wave soldering techniques, they are essential. When the solder mask has been applied, it is necessary to make holes in the mask where components will be placed, which is accomplished by using photolithography. There are various processes that can be used in the manufacture of soldier masks, but the most commonly used would be liquid photo imageable solder masks. The process in which the solder marks on the PCB are created utilizes photolithography techniques and masks in order to achieve the desired results.

Copper

It is true that copper is one of the most commonly used materials on a PCB to transmit signals inside the PCB with high performance and reliability. However, one thing to keep in mind is that copper's performance is affected by a variety of factors, including the quality of the resources, the layering on the substrates, and the roughness of the surfaces of the boards. There are many different types of foils that can be used in copper, which can be referred to as the quality of copper. There are two types of copper rolls that are commonly used: electrodeposited copper (ED) and rolled-annealed copper (RA). Electrodeposited copper is most commonly used in PCB applications where mechanical stress is expected. It is necessary to use ED copper in order to build a PCB that is strong, robust, and ready to withstand further fracture strains that may occur in the future.

However, rolled-annealed copper circuit boards have a reputation for being extremely flexible as well as having excellent thermal properties, particularly the ability to withstand thermal shocks of varying degrees. A choice between flexibility and rigidity can have a significant impact on the development of any application. As we design our PCB, we have to take into consideration the different requirements and needs our board has, and we should take that into account when designing our board.

Substrate

A substrate is a part of the PCB fabrication process, which refers to the bottom fabric where the electrical components and traces are placed. The substrate is an essential component of the PCB, as it serves as the underlying structure for the electrical circuit on the PCB. The most commonly used substrates include fiberglass-reinforced epoxy resin (FR-4), composite fabric consisting of woven

fiberglass cloth, and epoxy resin ligatures. Fiberglass-reinforced epoxy resin is frequently used as a substrate material. Aside from ceramic substrates, polyimide substrates and metal-core substrates are some of the other substrates that are utilized in PCB fabrication.

In order for a substrate fabric to be selected, there are a variety of factors to consider, including the type of application, the operating environment, and the performance requirements. For example, FR-4 substrates are commonly used for general-purpose applications. On the other hand, polyimide substrates are also preferred for high-temperature applications. Ceramic substrates are preferred for applications requiring enhanced thermal conductivity and chemical and mechanical resistance. Additionally, the choice of the substrate and other factors, such as the thickness of the substrate, the dielectric fixed, and the surface finish of the substrate, all play a crucial role in determining the electrical performance and the reliability of the PCB.

3.5.3 Design Software

Several software programs are available for PCB design, each with unique features and capabilities. Three commonly used software programs for PCB design are Autodesk Eagle, Altium Designer, and SolidWorks. This section is dedicated to understanding the key differences that come with each program, and determining the ideal one for this project.

Due to its pricing, Autodesk Eagle is a user-friendly software program popular amongst the more budget-saving crowd. It offers many components, such as a comprehensive library of parts and a simple interface that allows users to create high-quality designs efficiently. As part of Eagle's design rule checking (DRC) and 3D visualization features, designers can also check all necessary specifications and requirements based on the specifications.

This is a program that offers a comprehensive set of tools for PCB design. It is well known for its user-friendly interface system and general ease of use, which makes it a popular choice among professional designers who need a comprehensive set of tools for PCB design. With the help of Altium's advanced features, designers are also able to produce complex designs at a rapid and efficient speed, so it is possible for designers to achieve this feat. As well as ensuring circuit design compliance with all the necessary specifications that must be met to make the circuit work, the software also provides design rule checking (DRC).

SolidWorks PCB is a program that allows a creator to integrate a Printed Circuit Board design with a mechanical structure design. This will enable designers to create complex designs combining electronic and mechanical components. In addition, SolidWorks offers advanced features such as 3D PCB visualization, as well as integration for mechanical systems. These tools and features make it a

popular choice among designers who must create designs integrating electronic and mechanical components together.

Autodesk Eagle is a user-friendly and affordable software program popular among hobbyists and small businesses. Altium Designer is a powerful, comprehensive software program offering advanced features and a user-friendly interface. Finally, SolidWorks is a software program that integrates PCB design with mechanical design, allowing designers to create complex designs incorporating electronic and mechanical components. Ultimately, the choice of software program will depend on the specific needs and budget of the user. For the creation of our project, we will be going with Eagle.

Software	Design Specification	Price
Autodesk Eagle	999 schematic sheets, unlimited board area, and 16 layers of signals	$60 per month
Autodesk Eagle (free)	2 schematic sheets, 80cm2 board area, and 2 layers of signals	Free
SolidWorks	Unlimited board area, 23 signal layers	$1200 per year
Altium Designer	No limitations	$355 per month

Table 3.7 PCB Design Software

3.5.3.1 Design Using Eagle

There are a number of small steps that need to be taken when designing a PCB with Eagle, and each of them has significant importance to the overall creation of the PCB. The first step in creating our own PCB is creating a schematic or .sch file. A schematic is a diagram that shows how your circuit's components are connected together. You can use Eagle's schematic editor to design and create your schematic. Using the editor comes with various tools and libraries that allow you to search for almost any component available using a particular searching * placed at the beginning and end of whatever component you are looking for. Unless you have the exact name that will allow you to find that particular component, we will create our logical circuit diagram using the schematic editor after seeing the parts we wish to use. This allows us to make the schematic of the circuit we are trying to create.

Our next step is to create a PCB layout after we have completed the schematic diagram. The PCB layout represents a real-world board with actual dimensions

for the future board that is manufactured. Once the board has been loaded into the schematic, all our components are represented, and the dimensions, shape, and size of the board will be determined here. The first thing to do is to place all the components in a logical order and make sure the component placement is uniformly distributed across the board to ensure that the traces run smoothly. Having the board easily understood will allow for future corrections and troubleshooting, as well as allow for future board repairs. In Eagle, you have the option of using an auto-router that will place all of the components on your board automatically, saving you a lot of time and effort, especially if you have a lot of connections on your board. However, the auto-router tool may only sometimes produce the best results, and you may have to route some traces manually in some instances.

The next step in creating a PCB is to run the design rule check (DRC) as soon as you have completed the layout and ensured that all wires and components have been arranged correctly in order to achieve the PCB design that you have created. By using a DRC, errors can be prevented from occurring, such as short circuits, electrical errors, and other physical issues that may appear on the designed board. Additionally, the DRC can also be configured to accept any custom requirements the manufacturer has set in place, which are usually manually entered by the user, in order to make sure that it accepts them. In order to be able to generate the necessary files for manufacturing, it is necessary to supply a document that can be uploaded to Eagle's DRC and then move on to the creation of the necessary files after the DRC has been passed. When the manufacturing files have been created, it is essential to double-check them to make sure that everything was done properly. If the manufacturing files are created correctly, the manufacturer will be able to create the design according to the manufacturing files in a timely manner based on the design data.

3.5.4 Design Recommendations

There is an excellent need for PCB design recommendations because they provide guidelines and best practices that help ensure the circuit board's functionality, reliability, and manufacturing ability, in addition to ensuring that the board is functional, reliable, and manufacturable. With the use of design recommendations, designers are able to save a lot of time and money, as well as reduce costly errors in PCB design. These recommendations come from a variety of areas that affect different aspects of the board. For example, the trace width, the ground plane, and the test points are all examples of such areas. Furthermore, design recommendations can help ensure that the PCB will be able to be manufactured. It is crucial to consider the manufacturing process in designing a PCB, and design recommendations take this into account. Therefore, it is essential to follow these guidelines in order to produce a high-quality and cost-effective method of making a board.

1. Maintain consistent trace widths and spacing:
If you keep a consistent trace width and spacing in your board, you will be able to ensure that your board performs reliably and can be easily manufactured. To prevent electrical interference from affecting your board, you need to ensure that your traces are wide enough to handle the current they will be carrying and that you maintain the minimum spacing between your traces in order to avoid electrical interference.

2. Use proper via placement:
If you have a PCB with multiple layers and wish to connect them, vias can be an essential tool for joining them, but they should be used cautiously. Try to minimize the number of vias you use and place them in places where they will not interfere with other components or traces. If we were to use a via in the wrong place, it could adversely affect the full functionality of the PCB system. A few examples would be that the entire System could be grounded if misplaced

3. Minimize vias:
As mentioned earlier, vias can be an excellent tool for your PCB, but they can also be a hefty burden that adds complexity and costs. Consider routing a trace without using a via if it is possible to do so. Using too many vias can likely increase manufacturing costs, and the overall efficiency of board space will be affected.

4. Add a ground plane:
Your PCB should have a ground plane on both its top layer and its bottom layer in order to help reduce electromagnetic interference (EMI) that may occur on it. As far as your PCB is concerned, both the top layer and the bottom layer of your board should have a ground plane. Further, the ground planes will provide your circuit with a stable reference point, thus enabling you to eliminate EMI from the system.

5. Place decoupling capacitors near ICs:
To provide the maximum amount of power and stability to your ICs, you must use decoupling capacitors in order to reduce electrical noise and stabilize their power supply. Ideally, the decouplers should be placed in an ideal environment as close as possible to the integrated circuits, within a few millimeters of them. As a general rule, they should be within a few millimeters of the integrated circuits so that the distance between them and the ICs is as close as possible.

6. Avoid 90-degree corners:
The use of sharp 90° corners can be tempting and sometimes might happen, but it can lead to a number of problems. Firstly, these corners may create hot spots where the current density is highest, which may result in a trace overheating and eventually, a failure of the component. Additionally, sharp corners can increase

the risk of electrical noise, negatively impacting the circuit's performance. By using 45-degree corners or rounded corners, you can reduce these issues. It is possible to prevent hot spots by using 45-degree corners, since 45-degree corners spread current flow more evenly and reduce the concentration of current density. You can improve your circuit design reliability by using 45-degree or rounded corners in your traces.

7. Keep your PCB dimensions reasonable:
A smaller board is easier to handle and less expensive to manufacture, but it may also be more challenging to design and troubleshoot in the long run. You should find the right balance between size and complexity to suit your project needs whenever possible. We will have to follow our design constraints regarding dimensions, mainly enforcing weight and holding constrictions inside the vehicle.

8. Add test points:
Test points are essential for debugging and testing a PCB design to ensure it is in proper working order. These test points allow the designer to access specific nodes or components within the PCB circuit; ideally, these points are easily accessible for testing purposes. Another area to consider is that these test points should avoid interfering with the general use of the circuit. It will also be imperative to ensure the test points exist on all primary functions of the PCB systems, such as the ground plane, power plane, and significant vital components.

9. Place components in logical manner:
Logical component placement is a must when it comes to designing a PCB. It comes with many benefits that will allow overall functionality to improve. One of the main benefits of a board that has used logical component placement is the ability to perform easier debugging because the system is an easy-to-follow analytical method that makes following the traces and components simple. Another great benefit that comes from this is that when using logical component placement, it generally allows for shorter trace wires.

10. Use proper silkscreen and labeling:
It is imperative that your silkscreen is clear and legible and that your components and connectors are labeled to help you keep track of your design. You can identify components, track connections, and keep track of your design in seconds.

3.5.5 PCB Design Constraints

A high-quality PCB is a key to modern electronics. A PCB's longevity and performance are, however, affected by several factors. These factors include leakage resistances, IR voltage drops in traces, vias, and ground planes, stray capacitance, and dielectric absorption. In addition, due to their ability to absorb

atmospheric moisture, PCBs often contribute to parasitic effects differently from day to day as humidity changes. In this section, we will discuss the different ways in which a PCB can be made more reliable by incorporating proper grounding, partitioning, decoupling, thermal management, and tracing.

3.5.5.1 PCB Partitioning

Partitioning a PCB is a design process that allows designers to manage the complexity of their circuits. This process is achieved by breaking them down into smaller functional blocks. Based on the functionality of each block, functional partitioning divides the circuit into different sections, almost like figurative blocks. This mainly gives designers a more concrete concentration when dealing with smaller intravenous circuit designs. In addition, designing a board in this manner allows the designer to build and test each section separately.

It is very important to understand where to use partitioning in order to design a board successfully. PCB partitioning generally refers to identifying the function blocks in the PCB design and determining what type of functions need to be accomplished by these individual blocks through the partitioning process. Our goal is to separate five blocks in our design. Taking this as an example, let's divide the design into five blocks. Power management, signal processing, GPS, and wheel control are some of the uses of these blocks, as well as processing overall data and handling power problems with the help of the blocks. Following the identification of our critical blocks for functionality, the next step is to define the components that are involved in each block so that the functionality can be performed as planned.

Once we have identified the functional blocks, we can then begin to determine how the interface between each block will look once the available blocks have been placed. As a result of the interface between blocks, it will determine how a block communicates and contributes to the overall function of the board as a whole. A board's interface is an essential part of the whole board since it transmits communications signals between different sections of the board. Inputs and outputs of different blocks may be used as signal inputs, or signals may be used as signal outputs. As a result of this process, printed circuit boards can be tested and debugged for the purposes of testing and debugging.

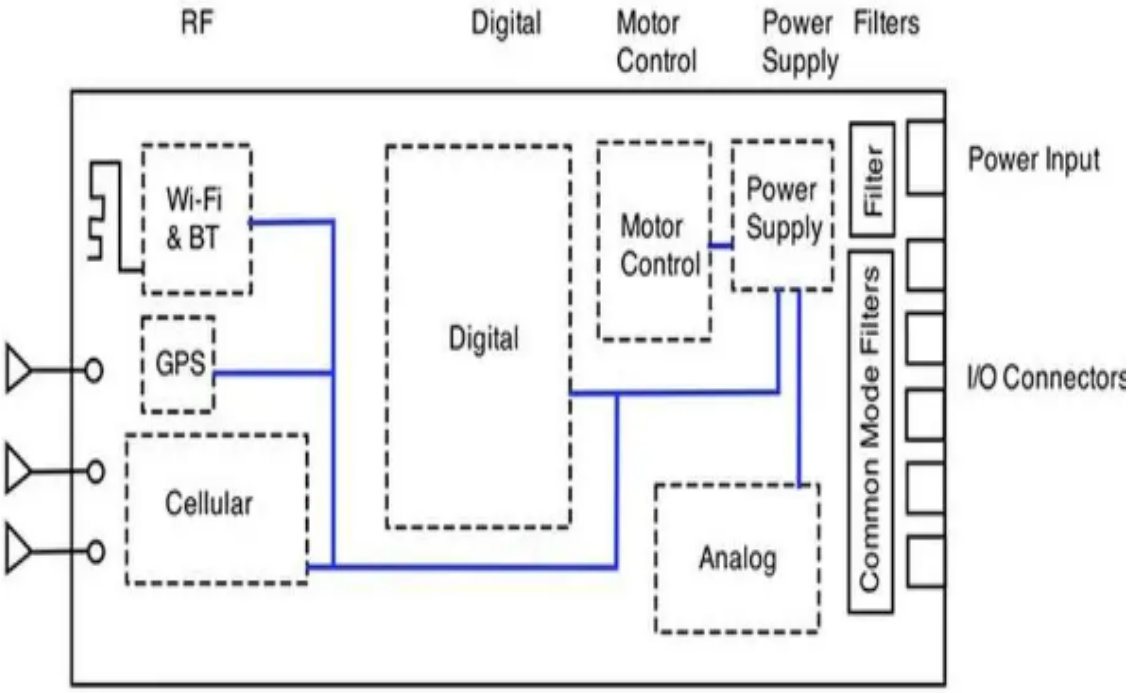


Figure 3.8: PCB Design Partitioning

A designer will be able to create complex boards with an assured method of solving problems by partitioning them. It will allow them to design complex boards with a wide variety of design possibilities. When a designer is creating a board that meets the desired criteria, they must make sure that every tool available to them is used to ensure that the board is going to function properly once it has been designed. It is one of those tools that will allow a designer to construct a PCB and test each partition simultaneously.

3.5.5.2 PCB Grounding

There are several elements to a functional PCB design, including properly grounding its components. The purpose of grounding is to provide a reference point for a potential voltage, as well as to allow all currents traveling through an electrical circuit to return to their starting point, which is the reason for grounding. In order to increase the longevity and lifespan of a PCB, proper grounding is essential, because it makes the PCB more reliable. Designers need to follow a few specific rules when designing a ground for their buildings. There are a few specific rules that designers need to follow when they design a ground for their buildings. These rules are listed below.

1. **Leave nothing unattached:**

If blanket spaces are left when designing and creating a PCB, it will have an adverse effect on the overall performance and functionality of the board. In order

to ensure that everything continues down to the ground, it is best to ensure that each section is either filled with copper or has a via that connects it together.

2. Minimize vias:
Vias are essential to the connection of any layer on a PCB to any other layer, which makes them an integral part of the design. The impedance of traces and vias is similar when it comes to how much impedance they carry. Due to this, we need to limit the number of vias used in the design to ground in order to avoid any significant potential differences in voltage in the design.

3. Keep ground layer whole:
The ground layer has to be kept whole when dealing with multiple-layer boards in order for the board to function correctly. In the event that the designer decides to run traces through the ground layer, this would result in a ground current loop in the circuit board, and this would ultimately lead to the board not working as it should. For this reason, the designer needs to avoid running traces through the ground layer of the plane so that the plane can be considered one whole ground layer.

4. Design ground before routing
As a rule of thumb, when designing a board, the ground is crucial since if it is not laid out correctly, then the entire board will be messed up, which is why it is essential to avoid any unnecessary mix-ups and design the ground first, and only after that run the traces on the board.

5. Understanding Current Flow
Current is the life of the circuit board; understanding how and where your currents are going is essential when it comes to grounding. Using Kirchhoff's Current Law, it becomes easy to understand how our current will travel through the board. Whether the current is leaving from the source or returning to the ground, it's fundamental to ensure all currents are accounted for in returning to the ground.

6. Mixed-signal floor planning
The most important thing that one should take into account when building a board in today's technology is that he or she will understand how analog-to-digital converters and digital-to-analog converters are used in almost every technology. Thus, when we plan our design, we must make sure that analog and digital components are separated. It is important for us to keep analog and digital signals separate when digital signals switch between states because a powerful signal can be emitted when they switch between states, which is why we do this. Further, if this emittance between states is strong enough, it can put a great deal of noise on an analog signal if it is brought near to it, since this emittance can be caused by state changes imposing noise on the analog signal.

The idea behind a mixed floor plan is to keep the analog and digital ground planes separate from one another. In other words, you can't just physically separate the ground planes from one another in order to keep the grounds separate. In other words, it is a ground where a potential difference of 0 volts exists between the analog and digital grounds. The result is likely that we will not be able to detect any fluctuations in the other components coming from the digital or signal planes. The separation of the two grounds also provides a common reference point for the comparison of values. It is possible to encounter differences in values when there is a physical separation between the two grounds. This is the reason why we try to avoid physically separating the two grounds from each other.

The solution to solving the mixed-signal floor planning is a simple task. In order to accomplish this task, we are going to use a method called star grounding. The star grounding method is relatively straightforward; it connects the analog and digital ground planes simultaneously. This system's goal is to ensure that all signals are routed back to a single point once the power supply returns. The idea behind this system is to prevent any signal from impinging on another by bringing all signals together at the same point. By using this method, it is possible to reduce the electromagnetic interference caused by a circuit and ground loops caused by it.

This philosophy of grounding is known as the "star". In a circuit, the entire voltage is referred to the same ground point, which is also called the star ground point. It does not necessarily need to appear as a star; it can simply be a point on the ground plane. The primary feature of the star grounding system is that all voltages are measured in relation to a specific point in the ground network, not just to an unknown ground. In my opinion, this is a rational approach to grounding. It is crucial to design a system that minimizes signal interaction, as well as the effects of high impedance signals or ground paths. However, implementation problems are often encountered in designing such a system. It is very difficult, for example, to implement a star ground system effectively if it is designed. The main reason behind this is the potential for ground loops that will cause everything connected to it to revert to a potential change of 0. Which would leave the overall circuit ineffective.

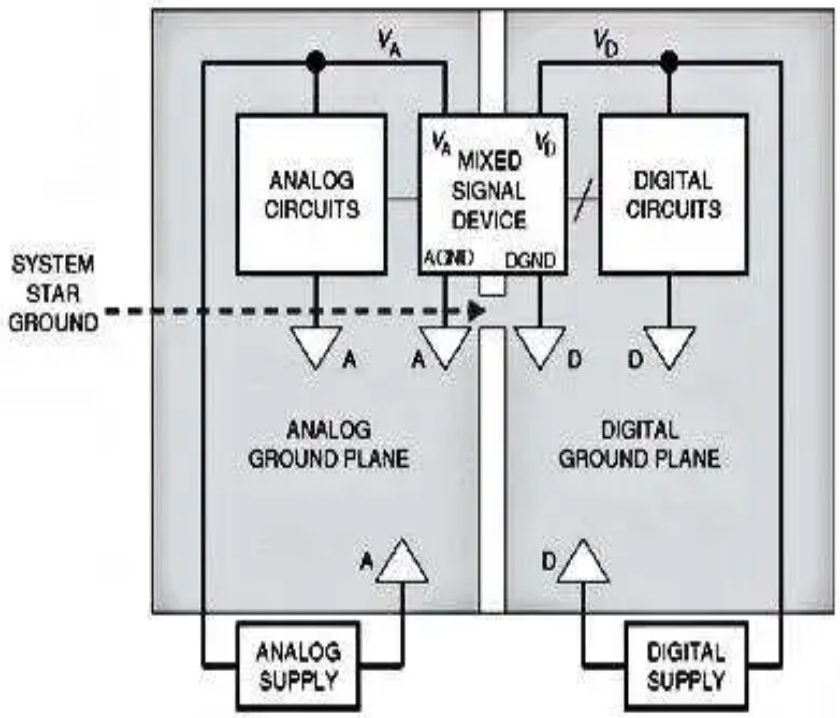


Figure 3.9: PCB Design Partitioning

3.5.5.3 PCB Thermal Management

Thermal management plays a crucial role between an operating system and a complete device failure. Currently, there is a significant increase in thermal hotspots among components as well as an ongoing trend of making said components smaller and smaller. As a result of thermal hotspots, a PCB can suffer from poor performance and a lack of longevity. Therefore, to increase the longevity of a newly designed PCB, the designer needs to take into consideration a variety of different thermal management techniques in order to improve the lifespan of the PCB.

In order to better understand PCB thermal management, it is essential to utilize thermal modeling and thermal simulation. The thermal simulation will show how much heat the simulated circuit generates and where it is being produced explicitly. In addition to its many benefits, thermal simulation can also show a current flow analysis because high current can be the primary cause of heat generation in a circuit and thus can serve as a primary cause for heat generation. However, because of circuit design requirements, it is sometimes impossible to reduce the high current. Therefore, the traces must then be re-routed away from any heat-sensitive components or parts that contribute to a high level of thermal generation when such a situation arises.

The reduction of thermal heat comes in many forms, but one of the most common is considering copper trace thickness and width when designing. It is

extremely important to consider the thickness of copper traces in order to determine the amount of impedance that the current will be able to pass through. Therefore, when designing, the designer should ideally be looking to optimize the trace thickness to provide a low impedance to avoid a high current density when avoiding thermal heat. In circuits with high current density, there is a tendency for heat to be generated and power to be lost, so as much as possible, it would be ideal for reducing these issues within one design as much as possible.

The most important way to prevent board damage is to make sure that heat dissipation is handled well. There are a number of thermal management techniques that can help prevent board damage. Still, one of the most important is to make sure that it is efficiently dissipated from the entire board, either through component placement or even through the board's structure as a whole. When considering how components will be arranged on a board, you should place them in the center of the board when considering the arrangement of components. A more even heat distribution will be achieved by locating a high-power component near the center. The components would suffer uneven heat dissipation if placed near the edge, which would adversely affect the system's performance. Changing the PCBs structure is another option that allows the designer to increase the thickness of the PCB, as increasing the thickness will increase the thermal conductivity, which in this case, will spread the heat generated by the component. As heat is applied over a wider area, the larger surface area is expected to increase heat dissipation.

A good thermal management method can also be achieved through the addition of components that are designed to increase cooling and heat dissipation. For instance, cooling fans are a commonly added component to PCBs that are capable of transferring heat convectively. By using this type of heat transfer, we can move heat away from components that are either heat sensitive or are heavy heat producers. The principle of conduction, which is primarily employed in heat sinks, can also be used effectively in a design since it is what we operate primarily in heat sinks. A heat sink's main objective is to provide a larger surface area for heat dissipation by drawing heat away from high-temperature regions and pulling it toward low-temperature measurements.

3.5.5.4 PCB Decoupling

We are accustomed to seeing both AC and DC signals in many of the electronics we use today. AC and DC signals are essential components for the operation of a PCB. However, when mixed, they can also have adverse effects. This is why it is necessary to separate these signals as much as possible. Otherwise, the designer will likely see an impact on both the signal integrity as well as the power integrity of the system. The best way to ensure that a signal is separated from each other is through decoupling, which is a standard procedure to accomplish this.

As a result of their inherent capacity to store energy, decoupling capacitors are widely used in both power supply as well as transient decoupling applications. For components such as processors, FPGAs, integrated circuits, and amplifiers to function properly, complex PCB assemblies require voltage regulation. As it turns out, capacitors can serve as decoupling devices on a board, providing current to the components so that the voltage levels on the board remain at a certain level based on their type and placement.

In order for a decoupling capacitor to be effective, its placement on a PCB needs to be carefully considered. The placement is entirely determined by the type of board that we are designing. As a general rule, it is necessary to take into account three types of boards when designing a PCB in general. The first type is the board that has no power planes, while the second type is the board that has a close-spaced power plane. Furthermore, when it comes to designing decouples, it is also essential to keep in mind that power planes are widely separated from one another.

1. Boards with no power planes:

There is no doubt that PCBs without power planes are a more straightforward solution when it comes to decoupling capacitors when it comes to decoupling. When designing a PCB, it is essential to place decoupling capacitors on each device individually so that the signal between all the active devices and the power input is as isolated as possible. This is what is known as the local decoupling capacitor. In order to filter out noise while at the same time allowing the device to operate at the desired frequency, local decoupling capacitors are best suited for all devices. In addition to the bulk decoupling capacitors, another critical component of the system would be the placement of a bulk capacitor at each voltage input to the system, which will ensure the system does not experience any voltage spikes in the input voltage.

2. Boards with close power planes:

Designing a board with closely spaced power planes can be challenging. When we design a board with these constraints, there are a few steps to take in order to maximize the performance of the board. First of all, we need to select the most significant nominal capacitance available, a capacitance that is greater than the parallel plate capacitance that exists between power and power-return planes. Secondly, local decoupling capacitor placement is less critical than boards without power planes; they need to be located close to the devices for which they will filter the effective operating frequency; otherwise, they will not function properly. The last rule for designing a board with close power planes is to understand that the number of decoupling capacitors is roughly inversely proportional to the effective connection inductance.

3. Boards with broad spaced power planes:

A broad distance may be a considerable distance when used in conjunction with the word. We must remember that we are dealing with a system that is measured

in millimeters, so when we refer to a broadly separated board, we mean a board in which the power and ground planes are separated by at least .5 millimeters. The reason why this distance is considered necessary is that, at this point, the inductance caused by the planes will be negated. For the bulk decoupling capacitor value to be evaluated as a function of the transient power requirements of the active devices on the board, it is necessary first to understand the rules governing a design such as this. Further, when placing local decoupling capacitors, the design requires that the capacitors be placed as close to the power and ground pins of the active devices as possible. Also, when working with decoupling capacitors here, it is essential that the designer avoids using any traces on the decoupling capacitors. As a result of this, you should ensure that there are no extra induced inductances between the mounting pads. Instead, it would be best if you located the via near the mounting pads for ease of use.

3.5.5.5 PCB Tracing

Tracing can be a versatile component when it comes to PCB design and how the overall system will function. In general, traces refer to the network of wires that are used to transmit signals from one device to another. For example, with traces, the entire system would be able to transmit signals to other devices. In general, the traces that we are referring to are the copper, insulation, and fuses inside the circuit board. In order to properly design a PCB, it is crucial to take into account the calculations that are used as well as the design of the trace that is used. Therefore, it is essential to pay attention to the following four factors when designing a PCB; thickness, trace resistance, trace inductance, and trace capacitance.

As part of the design process, trace resistance is a crucial design factor that must be taken into consideration and analyzed during the initial stages of the board's development. This is because all materials, including traces, have resistances and parasitic properties that can alter how the board is used. For example, we generally use copper for our traces, despite its many differences in composition and properties. As a result of these differences, the comprehensive implementation of the entire board, as well as its functions, can be altered. In other words, it is a crucial design step to understand the amount of resistance one expects during the design process because it can significantly impact the delivered current and interfere with power loss.

PCB trace resistance can be visualized as having two main components. The first component is represented in the equation as the resistivity of the trace material, which can be shown as P. In addition, the second part would be the trace area, which comprises several different elements. The temperature coefficient for the particular copper used in the trace can be expressed in terms of its thermal conductivity. Moreover, L, W, and T represent the physical dimensions of the trace. The resistance of a trace can be found by multiplying the resistivity by the area of the trace. This will give you the resistance of the PCB

trace.

$$R = \rho \cdot \frac{L}{T \cdot W} \cdot [1 + \alpha \cdot (temp - 25)]$$

Equation 3.2: Trace Resistance

As a general rule, it is essential not to miscalculate trace resistance, as this can result in a power loss, which eventually impacts the entire board, leading to a rise in temperature and a reduction in conductivity. This can all be avoided by calculating the trace resistance as best as the designer can and optimizing it as much as possible. There are many ways to maximize trace resistance, and one of those methods would be to increase the trace's total area. This would be helpful in the design of power circuit PCBs because they would allow for more consistent output of resistance, so little to no variation could be detected. Another helpful tool would be the addition of solder bridges, this would allow for an increase in current capacity and reduce power loss for power boards.

There is no doubt that the thickness of the trace has an extremely important impact on a PCB's design and operation, even though it seems unimportant. Many distinctions can have a direct correlation with the effect of the board. Therefore, it is essential that the designer takes into consideration the current requirements of the system before deciding on the trace width that will be required. This is the primary determinant of the overall maximum current amps that will be achievable. As part of the trace width calculation, another factor that must be considered is the expected temperature rise. Suppose this factor needs to be taken into account by the designer. In that case, the entire system could have problems with improper thermal management if the expected temperature rise is not taken into account. As a result, understanding the requirements of each device is essential so that the correct trace width can be selected that is suitable for the overall system requirements.

Temp Rise	10℃			20℃			30℃		
Copper	1/2oz.	1oz.	2oz.	1/2oz.	1oz.	2oz.	1/2oz.	1oz.	2oz.
Trace Width (:inch)	Maximum Current Amps								
0.01	0.5	1	1.4	0.6	1.2	1.6	0.7	1.5	2.2
0.015	0.7	1.2	1.6	0.8	1.3	2.4	1	1.6	3
0.02	0.7	1.3	2.1	1	1.7	3	1.2	2.4	3.6
0.025	0.9	1.7	2.5	1.2	2.2	3.3	1.5	2.8	4
0.03	1.1	1.9	3	1.4	2.5	4	1.7	3.2	5
0.05	1.5	2.6	4	2	3.6	6	2.6	4.4	7.3
0.075	2	3.5	5.7	2.8	4.5	7.8	3.5	6	10
0.1	2.6	4.2	6.9	3.5	6	9.9	4.3	7.5	12.5
0.2	4.2	7	11.5	6	10	11	7.5	13	20.5
0.25	5	8.3	12.3	7.2	12.3	20	9	15	24

Figure 3.10: Trace Width Chart

As a designer, it is important to understand what the width of copper traces will be based on the requirements of the circuit. A good example of common widths and sizes of copper traces can be seen in the graph above. This chart can make it easier for you to decide what functions and features are most important to you in designing a new PCB. As an example, will the decision be made by a significant spike in temperature, which would require a high amperage, or if the temperature rise stays low, but the amperage is constant? That is a decision that can only be made by the designer.

All PCB traces have some inductance, calculating the amount of inductance that is seen in the traces is essential for designing one's board. Depending on the amount of inductance found in the trace will determine the overall functionality of the designed board. A general rule of thumb when designing a PCB trace system is to ensure the inductance produced is within the desired tolerance of the system. In general, a designer is required to calculate two different inductances: microstrips and striplines. Microstrips are the traces on the surface, which can be seen on the top of the substrate of the PCB. While the striplines are the traces that are on an inner layer between two reference points.

Microstrips: $$L = 5.071 \ln\left(\frac{5.98H}{0.8W+T}\right) \text{(in nH/in.)}$$

Striplines: $$L = 5.076 \ln\left(\frac{1.9B}{0.8W+T}\right) \text{(in nH/in.)}$$

Equation 3.3: Microstrip Trace Inductance
Equation 3.4: Stripline Trace Inductance

A variety of units have an effect on the inductance, and we take into account the first unit, the copper weight, when it comes to the inductance. These units can change depending on the design of the system. The copper weight of the system can change based on the type used, and there are better units to change to meet the expectations of the system. Another unit is the layer thickness. This is a constant of the board and is a fixed constant. In general, manufacturers have a limited selection of widths for their products, whereas the designer can make these changes at the beginning. Finally, trace width, which is represented with W, can also be altered at the beginning. In order to meet our inductance requirements, we can change the trace width, but the designer needs to weigh many factors before he or she determines what to increase and what to decrease to achieve the ideal inductance value.

3.6 Chassis

The chassis of the delivery bot must fit the specifications defined in the introduction of our project. These specifications include a total weight not exceeding 20 pounds, which includes a payload weight not exceeding 5 pounds. This leaves the rest of the system not being able to exceed 15 pounds which included all other mechanisms such as the payload carrier, locking system, microcontrollers, and circuitry, etc. For this reason, chassis options exceeding 12 pounds were ruled out with a preference being given to lighter chassis to allow more room for error of the other components. In addition to the weight specification, there are also dimensions we have set that did not exceed 1ft x 1ft x 0.85ft. Lastly, other considerations that were used to determine the chassis that is used include the type of tires and durability as the delivery bot must be able to travel a distance of at least 2 miles. Cost was factored into the decision once all options that fit the previously discussed specifications were clearly laid out.

While researching options available for chassis, some considerations were made in regard to the areas of study that our team covers which only include Electrical and Computer Engineering. Acknowledging this aspect of our senior design team, we had a preference of purchasing a pre-built chassis or a set including most of our necessary components in terms of the mechanical portions of our

project. With this being said, some kits including components like motors, wheels, or batteries may clash with some of the requirements set forth for the success of our delivery robot such as the travel radius of 2 miles or maximum speed of 2 miles per hour. This will be further discussed in the motor research where the options will be compared and analyzed to see whether they fit the specifications of our project.

Another issue that may arise when purchasing a pre-built or set chassis is the steering mechanism available. While having dedicated axles and a steering mechanism may improve reliability and accuracy of the operation of our robot, steering by altering the speed of each wheel may be simpler to implement and requires less mechanical knowledge, which our team in general is missing. While options with dedicated steering will not be ruled out, the simplicity of building the robot was definitely considered and weighed against the improvements that may be made by using this steering system. And at this stage of research, these are all of the considerations that are laid out to focus the possible options we would like to work with.

3.6.1 Chassis Kit with No Dedicated Steering

This section will be dedicated only to chassis options that require a coding solution to steering by turning each motor on or off depending on the desired outcome. This category prioritizes the ease of building the robot at the cost of reliability or accuracy while it is operating. These options also may require more dedication towards the software aspect of our project. Furthermore, with these options being full kits, there may be some changes necessary to be able to fit our specifications for the project. Therefore, the modularity of the platform may be considered as we may need to exchange the given motors for more powerful ones, or the battery system for a higher voltage to allow higher payload weights. This explains why some options that were considered in this section even if the motors are not powerful enough as switching the components was factored into the final decision.

3.6.1.1 DFRobot 4WD Arduino Mobile Platform

This robot chassis kit seems to be a good general-purpose option, with many specifications that we are looking for, though the kit may not excel in each of these areas. For example, the shop page states that the size of the product is 200mm x 170mm x 105mm which fits well inside of our requirements. This means however, that the payload container must be smaller to fit onto the vehicle. In addition, the page states that the maximum speed is 90 cm/s which is about 2 miles per hour. This is perfect for our specifications, though this does not take into account any extra weight added, so depending on how the motors can handle load, the bot may not be able to support the payload weight at the same speed. The weight of the product is only 660g which means there is plenty of

available room for other components to take part of the weight specifications set forth. Also, the kit includes a plethora of mounting points and hardware which we will undoubtedly need to attach microcontrollers and the payload container. Another positive of this kit is that the electrical system is simply five AA batteries which make the electrical considerations for the motors and microcontroller that control the steering very straightforward, and more work can be dedicated to other areas.

Although the kit seems to be able to fit all of the specifications, there are some components that need to be considered before deciding on this option. As stated previously, the size is about half of the requirement which may make it more difficult to have room for the payload and other components. Also, the provided wheels, as stated by a customer review, may not be suitable for difficult terrain, which would be preferable so that the bot can handle some variations in its path like rocks, gravel, leaves, etc. In addition, the product name implies that the bot is only applicable with an Arduino board, which is not a downside of the product necessarily but must be considered when deciding on the final path.

Overall, at a price of about $40, this chassis kit fits many of our requirements and seems to be a decent option in totality. Some areas may need to be considered before fully committing to this option, but it is decent for our overall goal of the project.

3.6.1.2 Multi-Chassis 4WD Robot Kit (Basic)

This chassis is similar in terms of specifications to the last product, though it is a much different form factor. The platform sits much closer to the wheels and has a lower center of gravity which could reduce the possibility of the bot falling over. This needs to be considered as if the bot is too tall, with the extra height introduced by the container, the task of keeping the bot upright could prove to be difficult. Other positives of this choice are that it is also powered by 5 AA batteries which again makes the electrical considerations simple. With this kit however, the motors utilize a 48:1 gear ratio which increases the speed of the bot at the cost of reduced power. Depending on how weight can affect the motors, this could be a positive or negative as we need to have enough speed to reach the user in a reasonable time but also need enough power to traverse minor increases in the grade of the terrain. Again, this chassis also contains several mounting points.

While the form factor of this chassis may be beneficial, there are also some downsides. Since the bot has a lower center of gravity, it is much smaller than the previous chassis which leaves less room for other components and the payload. This could be fixed theoretically by getting a simple platform to add the extra components, which may be necessary. Overall, as a base this chassis is functional but seems to need many of the components to be switched out or added on to the chassis to fit our specifications. Depending on the direction of the other research though, this chassis could prove to be useful. With a price of

$34.95, this option could be useful but just looking at specifications the previous chassis looks to be an improvement over this one.

3.6.1.3 Hiwonder 4WD Chassis Car Kit with Aluminum Alloy Frame, TT Motor Smart Robot Car Kit

The previous chassis options featured only one level for all components to fit on while this chassis kit utilizes a two-tiered design that has all the components fit on the bottom level allowing the entire second level to be open for any additional attachments. This is a very useful design for our project as having a full platform to attach our payload mechanism would be beneficial while also having enough space to fit the microcontrollers and other components on the first level. This kit also utilizes 120:1 gear ratio on the motors which could be positive or negative as stated previously. In addition to the two-tiered design, this is the largest chassis kit at 7in x 5in x 3in which is a good option to have especially if we plan to expand some of the components or any of them take more space than is planned. As with the other kits, this also includes several mounting points, but this is most likely much more useful in this particular application as there is much more room to utilize the mounting points.

The specifications for this chassis kit look to be the best for our project though there does seem to be one major downside. This is that the product page states that the maximum load is 1500g. This a fairly small amount to work with especially considering the weight that other components outside of the payload could take up. This could be remedied with getting more powerful motors to maintain the speed of the bot. Also, the other kits do not mention a maximum load, so 1500g may be better than the other options but analyzing this product by itself, it leaves little room for the details of our project. However, at the cheapest cost of $28.99, this chassis kit still seems to be the best so far in most of the areas necessary for our project.

3.6.2 Chassis Kit with Dedicated Steering

This research section includes any chassis kit that has a dedicated steering system, meaning that the steering is not done through the speed of the wheels but instead the angle at which the wheels are turned. Theoretically, this section provides the best combination of reliability while driving and ease of building since the components are included in the kit to allow steering of the robot. The benefit of being able to directly turn the wheels is unclear other than having seemingly less logic coding for any situation the bot may encounter though this may change depending on the results of the other research areas.

This section does come with a caveat however, as though it seems easiest to both put together and code, the options are limited in the budget that we are

restricted to. Therefore, this section only contains one chassis kit essentially to keep any options possible open that could fit with our project specifications.

3.6.2.1 4 Wheel DIY Servo Robot Car 4WD Chassis Smart Car for Arduino Car Platform with Metal Servo Bearing Kit Steering Gear Control

The only chassis kit that was able to be found for this section was a small, high-speed bot that comes in rear-wheel drive with the two front wheels being the ones with the steering mechanism. The product page and accompanying test videos show that the chassis is very fast and accurate in its turning, and looks much more impressive than most of the prebuilt, non-steering chassis kits. The product is light at only 680g and small at 248mm x 146mm x 170mm. This does not leave much room for our payload but this can be fixed by attaching another platform to the bot. Furthermore, the high speed could counteract the added weight of the other components, effectively allowing a greater load than the other chassis kits could offer.

The major downside of this product is that the kit is only rear-wheel drive which could make traversing uncertain terrain a difficult situation. While the other kits have solutions to the downsides, this one does not seem to be able to be improved and we would have to work around it if it turns out to be unfit for our specifications. The potential benefits do seem to be great however, and depending on the other areas of research may be a better option than the others discussed. Overall, this option could be characterized as the high-risk high reward route where the steering, construction, and operation are all benefited at the cost of the question if it will work for our project. At $38.94 this option could work well depending on the other outcomes of research but at this time it is unsure whether it will be the best direction for our project.

3.6.3 Manual Construction

Although not preferable, constructing the delivery bot manually does have some positives that come with it. The main reason being that we can custom fit the parameters to match our specifications. For example, some of the chassis' kits have mostly good parts like the motors, wheels, and batteries, but have too small of a platform for our function. Problems like these could be solved by simply purchasing each part that comes with some of the kits discussed earlier to ensure that each part fits the necessary requirements that we have set forth. Furthermore, this could also reduce the cost of the overall chassis as we would avoid having to replace parts if the kits are not sufficient for our project. The worst-case scenario that detracts from buying a chassis kit would be that the entire kit is unusable meaning that we would need to find another kit to use. This would double our expenses on the chassis unnecessarily. In other words, the

chassis kits could lock our project into an undesirable position as we are relying on the quality of all the parts whereas if we build the bot ourselves by finding each part, we could test the reliability as we continue constructing the bot which could limit the extra expenses of having to swap out parts.

This option was at first not the best option due to our team consisting of only Electric and Computer Engineering majors but considering the chassis kits do not come prebuilt anyways, purchasing each component would essentially be the same amount of work in the actual construction of the bot, there just may be some extra work in finding the correct components for our application.

3.6.3.1 Wheels

While searching for wheels applicable to robotics project applications, the typical size is 65mm and are mostly constructed using a cheaper plastic. At a surface level, 65mm seems to be a bit too small for traversing the distance that is specified in the requirements defined in the project. Also, the plastic construction with little tread would likely not be able to traverse unlevel surfaces. Though uncommon, there are some options that are a bit larger with a tread designed more towards offroad travel. This would be a good option for the purposes of our goals but they do come with some considerations that will need to be addressed before finalizing a decision. Using these wheels could mean that modifications need to be added to the motors in order to fit as well as the platform, likely through the means of spacers to avoid friction of the components during operation. Also, larger wheels may require a stronger motor for movement and as a result, a larger power source. As stated previously, this option could be beneficial as it is tailored directly to fit our specifications but may require more effort in its construction.

3.6.3.2 Platforms

Unfortunately, online marketplaces do not seem to sell only the vehicle chassis frame and only include them in the chassis kits discussed previously. This realization likely means that manual construction of the bot would be an unreasonable option by ordering each necessary component separately. This idea could be supplemented however, by purchasing a chassis kit only based on the platform that is desirable and disregarding the other components, and replacing the other components like the motors, wheels, and power solution with higher quality parts. Other options regarding platforms like drafting a custom design are not available due to the group’s respective skills and knowledge being in different areas.

3.6.4 Part Selection

This section describes the final decision for the chassis kit that is used for the development of the project. From the parts researched, the Hiwonder 4WD Chassis Kit was chosen for the overall size which is within our parameters, the space given on each platform, the number of platforms for other attachments, the gear ratio of the motors, and the price. Not only is it the cheapest option, but it also offers several aspects that are useful for our application. Below is a table describing the comparisons between each chassis kit option.

Name	Steering	Motors	Size	Price
DFRobot 4WD	Software	TT Geared Motor (160 RPM)	7.87 x 6.69 x 4.13 in	$39.99
Multi-Chassis 4WD	Software	TT Geared Motor (1:48 gear ratio)	Unspecified (low-profile)	$34.95
Hiwonder 4WD Chassis	Software	TT Geared Motor (1:120 gear ratio)	7.09 × 5.51 × 3.5 in	$28.99
Wheel DIY Servo Robot Car 4WD Chassis	Hardware (Servos)	N/A	9.76 x 5.74 x 2.76 in	$38.94
Manual Construction	Hardware/ Software	TT/All metal construction geared motor	Custom	$25-60

Table 3.8: Chassis Comparisons

The highlighted row of the above table represents the team's decision to purchase the chosen chassis kit. As stated previously, this decision was made based on the price, size and quality of materials, and the type of motors. Some problems that developed throughout the project are that the steering must be done through software, which slightly complicatec the coding of our delivery bot as wheels may need to be turned off and on depending on the desired outcome. For example, to turn left, the left side wheels would be turned off while the right

side wheels would be on until a 90 degree rotation is reached. Since the kit is somewhat modular (in terms of switching motors or wheels) some combination of the kit and manual construction may be used depending on the effectiveness of each part in the kit. For example, if the motors lack enough power, they may be switched for more powerful ones. Overall, this was the plan moving forward with the part selection of the chassis.

3.7 Motors

Based on the project specifications, it is required that the robot is able to travel a distance of 2 miles to reach its destination at speeds of about 2 miles per hour. It is also preferred that the bot is able to traverse uneven terrain such as slight inclines and light gravel. To achieve these goals, not only do the wheels must be considered, but also the motors must be able to sustain these speeds in various environments with the most important situation being able to maintain speed and consistency under load or in other words, with the payload and carrier attached to it. For this reason, the motors that were considered must match a certain RPM which can then be calculated to find the speed at which it is able to travel. Most product descriptions on online marketplaces for robotics parts list the RPM of motors both without load and under load, so these numbers can be used to determine whether the motor is able to fit our project specifications. In this section, the motors for each pre-built chassis discussed in section 3.6 were analyzed to determine whether they can handle the speed and load in addition to some singular motors if the chassis kits did not meet the requirements. Once motors that met our requirements were found, the motor was then analyzed for attachment points and how it is powered to check if the motor can fit the chassis.

3.7.1 DC Motors

For robotics applications like our delivery bot project, DC motors are by far the most common solution especially for four-wheel drive vehicles. Most options, depending on the RPM range, are anywhere from 3V – 36V. The specifications for the project will likely not require anything above 12V.

3.7.1.1 TT Geared DC Motor

Included with most of the chassis' kits discussed previously, the TT geared DC motor is one of the smallest and cheapest solutions for small robotics projects. TT refers to the motor's construction which in this case is plastic. Also, the motor being geared means that it has a self-contained gearbox that increases torque at the cost of speed. Higher gear ratios result in greater torque and lower speed. These types of motors are desirable for our project as they are extremely cheap

and therefore easy to replace if there are breaks and malfunctions. Also, being a geared motor is likely necessary for our application as greater torque is required to carry the payload and carrier, as well as traverse inclines in its path. In addition, most of these motors have a variable speed system at different levels of voltage, allowing a higher RPM with increased voltage which again could be useful depending on the terrain and payload weight.

In general, the disadvantages of TT geared DC motors are that they are not durable since it is a plastic construction, and most options operate at a low voltage and RPM. While testing, they will most likely need to be replaced, whether due to mechanical breakage or insufficient torque and speed. The speed will be analyzed in a later section to determine if they are able to reach our speed goals.

3.7.1.2 TT All-Metal Geared DC Motor

As this motor is described, it is similar to the TT geared DC motor discussed previously, only in this case the gearbox is an all-metal construction. It is still housed in a plastic build, though the outside housing is likely not as important to switch to a metal material. This alleviates one of the problems with the plastic gearing as it will be much more durable and may help the overall system handle the payload weight and any other attachments added on to the bot. In addition, the metal construction is more likely to handle higher RPM and therefore higher speeds which will need to be checked to see whether the plastic motors can reach the speed requirement. If not, then the metal motors will be considered for reliability, speed, and durability. Also, the metal gears are able to deliver more torque, which in the case of our bot not being able to traverse certain terrain or not being able to handle load, would be beneficial for the project. Overall, this option is a mix between the previous motors and the next motors to be discussed as they offer a balance between price, durability, and power.

3.7.1.3 All-Metal Geared DC Motor

After considering the previous two DC motors, an all-metal construction of both the gearbox and housing can be an option if the others are insufficient to meet the specifications. Admittedly, this type of motor is the most expensive out of all of the options and is likely much more power and durability than is actually needed. For example, a quick search of these motors shows results of RPM exceeding 6000 which is definitely much more than we would need for our project. However, having the option of greater RPM speeds could be beneficial as we will not know the effects of attaching the carrier and payload until the bot is already constructed. With a greater maximum RPM, most motors like this have a voltage range that can slightly increase or decrease the speed of the motor. Having this adjustability could be useful when focusing the abilities of the bot to meet the requirements.

3.7.2 Analysis of RPM

In this section, the common RPM values found for the motors discussed previously will be analyzed to determine whether the specifications will match the requirements of the project. This will be done by taking the no-load RPM value found in the product descriptions and converting them to meters per second, and then to miles per hour to determine whether the load will match the maximum speed of our project. An online converter is used (found in references) which takes the RPM and radius of the wheel (a standard value of 65mm is used as it is a common size for robotics wheels) and outputs the speed expressed in meters per second.

3.7.2.1 TT Geared DC Motor (200 RPM)

For the most common motor that was discussed, 200 RPM seems to be the common value for the no-load speeds. For the calculation, 200 RPM was used with a wheel radius of 0.0325m (wheel diameter of 65mm). The output resulted in about 0.68 meters per second which is 1.52 miles per hour. This does fit into the maximum speed requirements set forth, though it must be remembered that this is the no-load RPM which does not include the weight of any parts of the other systems such as the chassis, payload, carrier, and microcontrollers. For this reason, the project will likely require a higher RPM both to maintain speed under load and handle various terrain. However, as discussed before, these motors are the most common, especially in the chassis kits that were found in section 3.6. This means that even if these will not be utilized in the end product, they can still be used during testing to see whether the real-world results match the theoretical numbers just found. Furthermore, these motors are the cheapest of all and therefore we may be able to supplement the speed by simply adding additional motors.

3.7.2.2 TT All-Metal Geared DC Motor (120 RPM)

For this section, the metal geared DC motor is analyzed. As the section title shows, this motor is only offered at 120 RPM at no load. This comes with some considerations but first, it will be analyzed for its speed in miles per hour. Again, the same value was used for the radius of the wheel and the converter output a value of 0.41 meters per second, which is only about 0.91 miles per hour. This is fairly low and likely not sufficient by itself. However, this motor is more useful as a supplementary motor to add extra torque to our system. This motor will essentially be used as an additional source if the other motors used are not able to handle load or inclines. With this motor also being a fairly cheap option, it is a decent solution to the chassis kits that may not be able to handle the specifications of our project.

3.7.2.3 All-Metal Geared DC Motor (330 RPM)

For the all-metal constructed motors, there are many options with even some reaching 6000 RPM. This is clearly outside of the necessary speeds and is not needed for the project. However, there are more reasonable values rating anywhere from 300 – 750 RPM which will fit our project at the higher end spectrum. At 330 RPM, the converter outputs 1.12 meters per second which is about 2.51 miles per hour. Although higher than the maximum speed specification, under load will likely reduce the speed and in addition, the voltage can be lowered also in order to reduce the speed of the bot.

As stated in the discussion about motors, this will be a good option if the plastic motors are not able to sustain the carrier and payload although it may be a bit overpowered for our application. In addition, these motors are a bit more expensive than the other two, which will need to be a consideration at the time of deciding the final components. Overall, these will likely be considered a sort of backup as all of the chassis' kits come with the TT motors and the all-metal construction will only be considered if the kits are not able to travel with the added load of our carrier and payload.

3.7.3 Motor Selection

This section determines the type of motors that are used for our design. This decision was based on the RPM and maximum speed output. For now, the team has chosen the type of motor that comes prepackaged with the aforementioned chassis kit which is a plastic construction TT Geared DC Motor with a maximum of 200 rotations per minute. This equates to around 1.5 miles per hour and is within the set parameters defined in section 2.3. The table below shows the comparisons between each motor.

Type	**Gearing**	**Max RPM**	**Speed (miles per hour)**	**Price**
TT Geared DC Motor	1:40 - 1:120	200	1.52	$2 - 10
All - Metal Geared DC Motor	1:298	120	1.12	$7 - 15
All - Metal Geared DC Motor		330	2.51	$10 - 25

Table 3.9: Motor Comparison Summary

Once the complete system was tested under load (with some weight to represent a package within the carrier), the effectiveness of the chosen motors was analyzed and reforme. In addition to these motors being included in the chassis kit, these are the cheapest motors to replace if a unit is broken or defective. The only aspect to keep in mind is that these are generally low power motors meant for small robotics projects and since our system will have a payload attached to it, they may not be enough to travel at a sufficient speed or up inclines. These motors are also compatible with the motor driver that we are intending on using that will allow the control of the motor's speed and rotation.

Section 4: Standards and Design Constraints

Standards and design constraints are the backbones of product development and engineering design. Standards are the general guidelines that developers of a product must follow. These standards can be different among different companies and nations. The only thing that remains the same when it comes to standards is that they are meant to ensure that any product developed will be safe, reliable, and tested to a set degree. On the other hand, design constraints are general constraints put into place by either the user or the designer. In general, design constraints are meant to be used to help design a product to meet the end user's needs. The parent company, engineering firm, or manufacturer generally sets these.

4.1 Standards

Standards are a crucial and necessary aspect of engineering: these standards and the establishment of guidelines and requirements for designing and manufacturing systems and products. Standards are an essential piece to ensure consistency and quality, that in the end, will increase performance and reliability. Without standards, engineering would lack the essential guide to create and ensure that any designed and developed product would lack in meeting basic safety and quality concerns. Lacking these would result in variations in quality, performance, and safety. Another great thing about standards is that they provide a common language for engineers to communicate and collaborate with each other. This allows for an easier way of sharing knowledge and technology with engineers, even if the physical speaking language differs from each other. Overall, standards are the basis of engineering that help promote safety, quality, and consistency through innovation.

4.1.1 ISO 10218-1

The ISO 10218-1:2011 standard specifies the requirements and guidelines for the inherent safety of industrial robots, protective measures, and information regarding their use. In the document, the primary hazards associated with robots are described. They also provide requirements for eliminating or adequately reducing the risks associated with these hazards. However, the document does not address the robot as a whole. For example, noise emission is generally not considered a significant hazard of the robot alone, so ISO 10218-1:2011 excludes noise from its scope, as it is considered a minor hazard. The ISO 10218 safety principles can be applied to nonindustrial robots, but these other robots do not fall under this classification.

For the system we are creating ISO 10218-1:2011 has a multitude of safety systems that must be considered and utilized in the creation and design of the Deliver Me Something robot. Our first concern with ISO 10218-1:2011 is that it is going to have to include a limiting device, which will help restrict the robot in terms of speed and space that it can operate. As a result of this restriction, we are able to achieve a higher level of safety when the operation and control of our project is taken into account. As it is in a constrained space, it will need a limiting device to ensure that the robot does not go outside of its bounds. A protective stop will also be a necessity for the robot's safety, which is implemented as a push button emergency stop button that is easily accessible. There are several factors that will go into the idea of a protective stop for Deliver Me Something bot. One of those factors is a safety measure for if the robot exceeds the maximum rated speed which is defined into the system, an automated protective stop will be activated.

4.1.2 ISO 10218-2

The ISO 10218-2 specification specifies safety requirements for integrating industrial robots and industrial robot cysts, based on ISO 10218-1, along with industrial robot cells. In addition to the design, manufacture, installation, operation, maintenance, and decommissioning of the industrial robot system or cell, these integrations also include the design, manufacturing, installation, operation, maintenance, and decommissioning of the system or cell. Also, the information necessary for the design, manufacture, installation, operation, maintenance, and decommissioning of an industrial robot system or cell is included. As a final point, it is included as part of the industrial robot system's component devices.

It is essential to take many precautions from the ISO 10218-2 for the Deliver Me Something, mainly from the point of view that its original design was intended for industry, and the overall concept of the bot is to deliver items, regardless of the industry it finds itself in. Our bot is a collaborative tool, so we must consider the potential issues that could arise, the safeguards that are in place, and the

collaborative nature of our bot. Our Deliver Me Something robot is based on the ISO 10218-2 definition, which describes it as a collaborative robot that will interact with humans in a closed environment. As part of our standard, we are required to include safety measures to ensure the robot operates in a safe manner in order to limit and prevent possible harm or endangerment to those around it. Understanding the interaction between the bot and the human is imperative.

According to ISO 10219-2, a safe state refers to a situation where a machine or piece of equipment is not going to present any imminent hazards. In order to understand what is going on in the problem, it is imperative to note the operation of a safe state in order to ensure that the robot, as well as any other particular items or people within the robot's range, are protected. The ideal implementation of this is adding restrictions to areas where the robot is limited so the objects or individuals there will not be harmed. The overall benefit of implementing safeguards is that it will put restrictions on the robot, but the overall advantage is that it will protect the robot and bystanders as well.

4.1.3 ISO 9283

The ISO 9283 standard relates to robot pose and path planning, in particular how many factors go into the planning of a robot's root path. When it comes to a robot's root path, ISO 9283 is a standard that relates to that. To be able to find a safe and direct path to the desired location, the Deliver Me Something robot will need to have an essential planning process. By applying the ISO 9283 standards to our robot, we can make sure that it is able to reach the desired location in a timely manner.

This standard involves a lot of fine details but one of the ones that particularly play a role in our robot's planning system would be, pose. A robot's pose represents its general orientation, so without our system being able to understand the robot's current pose, we will not be able to plan the robot's next moves. The next step would be to identify clusters, so the robot can remember where it has been since it knows its current pose, in order to be able to remember where it has been before. Having this knowledge will be useful in improving the delivery time in the future and can also help with path planning so that the robot does not operate down paths it had decided in the past would not be a viable option.

4.1.4 ISO 8373

There are a number of different types of robots and robotic devices that are used in industrial and non-industrial settings. According to this standard, robots are defined as programmed actuators that perform tasks independently of human intervention. It is a reprogrammable multipurpose manipulator that is capable of moving on three or more axes and is used in industrial environments for a wide

variety of purposes. A robot in an industrial setting consists of manipulators, controllers, as well as software and hardware that are used to program them or instruct them.

The concept of modularity describes the ability to separate and recombine a system into modules. A module is a component that can be removed or added to a system in a defined manner. It can be hardware or software, or a combination of both. Per the design our robot is in the essence a modular style robot. The main aspect that Deliver Me Something robot modularity is in relation to the delivery system in this robot. Plans to improve the current design would allow for a removable module that would allow for delivery of an item to be easier and more proactive to the user.

4.1.5 OSHA 1910.95

It is important to understand that OSHA 1910.95 refers to noise and the safety risks associated with prolonged listening to a piece of equipment. Particularly if the participant does not wear ear protection. This standard has been put in place in order to ensure that the recommended protection around equipment is effectively regulated, to ensure that no loss of hearing or other dangers result from the use and operation of our robot.

This standard is an essential driving element towards the success of a robot that we use in our design process. The Plan is to test the Deliver Me Something robot for longer durations and ensure that the sound produced from each system is in compliance with OSHA 1910.95, and therefore anyone exposed to the Deliver Me Something robot will not be required to wear any form of earwear protection at all. It is important to follow this standard in order to ensure that the robot is able to operate for long periods of time around people without causing any harm to them.

4.2 Design Constraints

Every project, no matter the size or scale, has constraints that it is limited by and our delivery drone is no different. There are multiple different constraints and multiple different types of constraints that are placed on our group when designing the drone. These constraints limit multiple aspects and steps of the project. They limit the range of parts that we can consider using for the drone. They limit how exactly we are able to build the drone and how and where we are able to use it. They limit the extent that our drone can perform. They also force some important ideas onto the project such as safety and reliability. These design constraints limit what the group is able to do overall in terms of the creation and use of the delivery drone. Despite all of that, they are still necessary and important to the project as a whole as they help to provide a structure to the

process. They also help ensure that the delivery drone is a respectable and reliable system that no individual would have to be concerned about. While it may seem like an annoyance to place such restrictions on the project, they are necessary and would be applied to anyone in the same position.

4.2.1 USB

The project at hand will likely make use of the Universal Serial Bus (USB) data communication method. USB was first introduced in 1996 by Intel and has since been one of, if not the most popular data transfer/communication methods in the world. Since its release it has constantly been implemented in many electronic devices in many different fields and our delivery drone is no different. It is often the main form of data communication between a PC and any external hardware. This is also often true for microcontrollers and integrated circuit boards.

4.2.1.1 Impact of Using USB

In order to effectively utilize USB communication, the Arduino board that is chosen must come with USB support. We must also be able to know how the data is being transferred and be able to properly manipulate any incoming or outgoing data that is transferred via USB. All of our group members have had general, high-level experience with a USB before, however many of us have a lack of experience with low-level USB data transfer which may or may not cause some difficulties in the project.

Another main aspect of USB that must be considered is the version of USB that is being used. Since its creation in 1996, USB has had multiple upgrades. As of the time of this project in the year 2023, USB has gone through four different version iterations with even a wireless form of USB being currently researched. Once we know that both the main Arduino board and the external hardware that are connected to the board can effectively and properly make use of USB, we must consider the versions of each item. Luckily for us, the older USB versions have been mostly discontinued and most devices are equipped with one or two of the relatively newer and currently popular versions. If the USB versions for the main Arduino board and the external device happen to be different, there have been many cables created that can help bridge that verison gap such as one of the more notable examples which is the cable that is used for charging many Apple products which connects USB-3 to USB-C.

Another aspect of USB that might be considered is the transfer rate of the USB cable. At first this may seem like a problem, however USB cable transfer rates are explicitly a maximum and if we want to be sure that too much data is not being sent to the main controller, we can simply manually limit the data transfer rate on the devices themselves.

4.2.3 I2C

Another popular method of data transfer is the Inter-Integrated Circuit (I2C) method. I2C is older than USB, being first developed in 1982 by Phillips Semiconductor. Since its initial creation I2C has undergone few major developments.Despite this, I2C is still a relatively popular method of data transfer as it is fast, reliable, and very useful for many embedded systems.

I2C's relative lack of major changes over the years is both a cause and effect to it being significantly stiffer than USB. To put it in other terms, I2C's formatting and standards are very strict and non-malleable with the only major adaptations being modes that are faster in order to keep up with the increase in computing power over the years. The base idea along with all of the inner processes of the communication protocol have not changed at all since its initial release. This lack of change is due to the advantages of I2C that were mentioned above, more specifically the one about it being useful for embedded systems. I2C's rigidness allows for solid reliability and with how relatively low-level it is when it comes to data transfer, it is no surprise it is one of the most popular methods of data transfer among embedded devices.

I2C is one of the methods that can only be properly utilized if one knows exactly how the process works. With USB, you could simply just plug in the correct USB cable into the respective port and let the data transfer handle itself. However, in order to effectively utilize I2C, you must set up the system with respect to I2C's protocols and processes. A specific example of this would be whether you are working with a MISO system or a MOSI system.

The main features of I2C that have helped it be as robust as it is are:

- The use of a master/slave system where there is one master device that controls the flow of data
- The use of a designated Start/Stop bit
- The requirement for a slave device to acknowledge when the data has been transferred/received
- Specific designation of whether the master is reading or writing data and is communicating with which slave device

The master/slave system is one of the more defining features of I2C communication. It is a simple system where one device is considered the "master" and controls the flow of data in and out of the rest of the devices. The rest of the devices are considered "slaves" as they must always obey the commands of the master device and have little to no say in the overall control of the system. This type of rigid structure helps to clearly define the role of each device in the system which can help prevent confusion when working with the data transfer. In the case of our delivery drone, the GPS module and sensors

would be the slaves as their only role is to provide data to the master or receive data from the master which would be the main Arduino board, or more specifically the main microcontroller. This is also where the previously mentioned MISO/MOSI systems come into play. MISO stands for Master In Slave Out and MOSI stands for Master Out Slave In. They are both descriptions of the direction of the data flow. They also are indirect descriptions of the hardware connections as I2C-enabled devices usually have designated ports for each specific system.

The designated Start/Stop bit is exactly what it sounds like. In most cases, the data being sent over I2C is restricted to a single byte at a time meaning for a given cycle, only 8-bits of data are sent at a time. This is one of the previously mentioned rigid structures of I2C that have allowed it to be as reliable as it is. Having the data being sent be evenly divided into a constant size helps the system recognize when data is being sent and when it should listen for the data. The Start/Stop bit can be thought of as similar to a regex. Once the Start bit is sent, the bus is in use by the two devices communicating data and cannot be used by any other devices until a Stop bit is sent. In the picture below, the Start and Stop bits can be seen at the beginning and end of the data transfer. In between when those bits are sent and received, all of the data being written and read is exclusively between two devices and only those two devices.

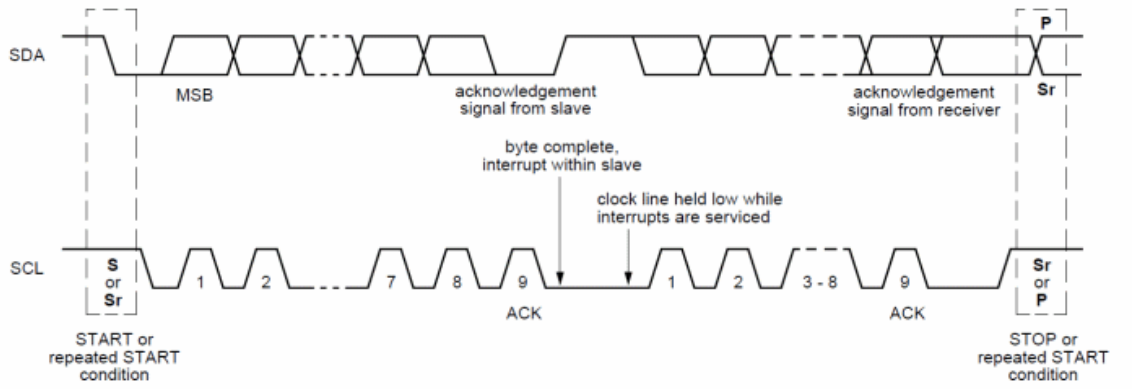


figure 4.1: Shows the complete structure of the I2C process

Image # also shows the use of the Ack bit or Acknowledgement bit. This is another aspect of I2C that helps with its reliability. This bit is exclusively used for acknowledging if the data is sent or received. Once the start bit, address, and read/write signal are sent, that respective slave device is expecting to either send or receive data. The first Ack bit is always sent by the slave device so that the master is confident that the slave knows that it is that device's turn to send or receive data. This is one of the more defining aspects of I2C as it helps guarantee that not only is the data not lost in the transfer process, but that the correct device is being communicated with. There is essentially no possibility of transfer error in those regards because of this Acknowledgement bit.

Along with this Ack bit is the use of the slave address and read/write bit. As mentioned before, the first Ack bit is sent by the respective slave device only

after the previously mentioned Start bit, a slave address, and the read/write bit. The slave address and read/write bit are the final pieces to the structure of I2C. It might seem redundant to send the address of the device you are transferring data between to that device, but it can be considered a checksum. The device can compare this address with its own address to confirm that the data is going to the intended location. The read/write bit is the slightly more important part as it designates whether the slave device is going to be sending data to the master or receiving data from the master. This bit is sent exclusively after the slave address and is the last thing that the device receives before sending the initial Ack bit which will begin data transfer. Image #+1 below visually demonstrates this structure for the transfer of data between the master and a given slave device.

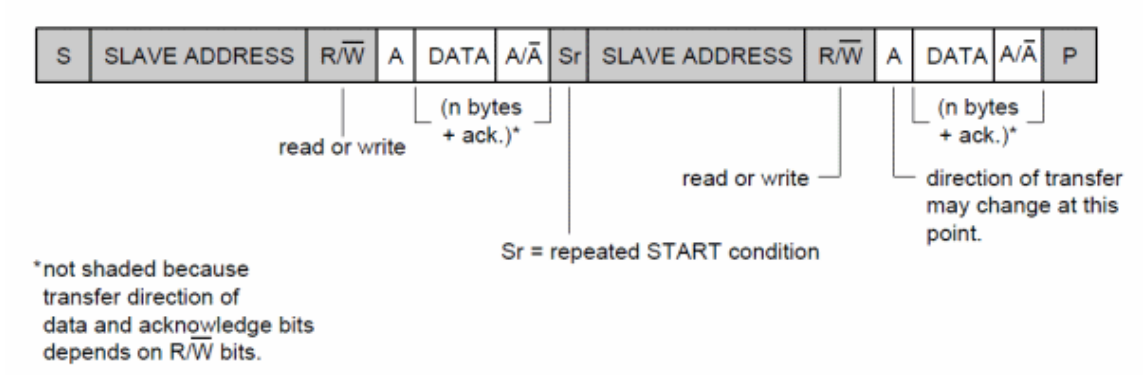


figure 4.2: The structure of the data packets that are sent using I2C

It can be seen from the image above that this slave address and read/write bit are sent multiple times along with multiple Start bits. This is because this process occurs whenever the conditions of the data transfer differ from the previous conditions. For example, if the data transfer changes direction meaning we go from reading to writing or vice versa, then a new set of condition bits will be sent to confirm the change with the device. If the conditions of the data transfer do not change between packets, then all that is required is an Ack bit from the slave device which designates to either keep sending data in the case of a read or that it has received the data and is ready to receive more in the case of a write.

I2C is a relatively old method of data transfer and it can be reasonably said that it is generally outclassed by newer methods. However, the strict structure of I2C is very enticing and is generally very good for data transfer. In order to keep up with the evolution of computers, enhanced versions of I2C have been developed. These versions consist of fast mode, high-speed mode, and 10-bit addressing mode. Fast mode was the first of the three to be introduced and simply increased the speed of data transfer from 100 kbits/second to 400 kbits/second.

After which came high-speed mode and 10-bit addressing mode. High-speed mode significantly increases the data transfer rate up to 3.4Mbits/second. While that may not seem significant by today's standards, it is a significant increase to the transfer rate of what is considered one of the more robust ways to transfer

data, thus keeping it alive and useful for some situations. High-speed mode makes use of several improvements such as spike suppression and a lack of clock synchronization, however these are not very relevant to our project. High-speed mode itself however is certainly relevant to our project as the master device will have a lot of data constantly coming in from all of the sensors.

10-bit addressing mode simply increases the number of devices that can be connected to the I2C system from 112 to 1024. This is not relevant to our project as we would not be hitting the initial 112 device limit let alone a 1024 device limit.

4.2.3.1 Impact of Using I2C

I2C seems like a very reliable and robust method for data transfer and paired with the relatively new high-speed mode, it seems like it should be a perfect pick for our delivery drone project. However, there are two major issues with I2C as a choice. The first is ease of use. Not only is USB much simpler to implement, it is also not too much more costly in terms of money. Essentially, I2C requires a high level of low-level control over the devices and will likely involve much more troubleshooting and fine tuning than USB where you can simply plug in a cable and it does most of the heavy lifting for you. The other major downside of I2C is the added overhead. I2C's rigid structure leads to a very robust communication system, but in doing so it also adds a fair amount of overhead. Along with the actual informational data being transferred, there are four other data aspects that must be transferred along the same line, those being the Start/Stop bit, the slave address, the read/write bit, and the Ack bit. These additional aspects add up to at least 10 extra bits of data every time information needs to be sent across the wire. If the transfer of information is not perfectly optimized, then many instances of having to swap between reading and writing may occur leading to that 10-bit overhead being applied more often. There is also the issue of only one slave device being able to communicate with the master device at a time. Even if we were to implement daisy chaining, which is a setup that involves chaining the slave devices together to create what can be visualized as a circle of information flow, that still only lets one device directly or indirectly communicate with the master at a time. If we want to implement I2C as our main method of data transfer and communication and take advantage of the reliability of it, we would need to put more effort into optimizing the data transfer between the GPS module and all of the sensors.

4.2.4 UART

Universal Asynchronous Receiver-Transmitter (UART) is the final data transfer and communication option that we can consider using. Just like I2C, UART is a method that has not changed too much since it was first introduced in the 1960s by the Digital Equipment Corporation. UART is a relatively simple method only making use of three main features:

- A Start/Stop bit
- The actual informational data being transferred
- A parity bit

The Start/Stop bit for UART functions the same way as it does in I2C. It designates when the informational data is actually being transferred. Its location can be seen in the image below.

The Data Frame consists of the actual informational data that needs to be transferred and can be between five and nine bits long. It comes after the start bit and before the stop and parity bits.

The parity bit is what differentiates UART and is essentially its signature feature. The parity bit is sent by the transmitter and it represents how many logic-high bits, or ones, exist in the Data Frame. It is sent after the Data Frame and before the final Stop bit as seen in the picture below. If the parity bit is a zero, then there is an even number of ones in the data packet being sent. If, after receiving and reading the data, the UART receiver finds that the number of ones does not match the evenness or oddness of the parity bit, then it knows that there was an error during the data transfer. The parity bit is the only semi-complex feature of UART which has led to it being one of the most popular methods of data transfer in embedded systems.

Figure 4.3: The structure of the data packets sent by UART are shown here.

4.2.4.1 Impact of Using UART

UART is one of the more simple communication and data transfer methods as it only utilizes a self-explanatory start and stop bit and a parity bit along with the actual informational data. The reliability that the parity bit brings along with this simplicity leads UART to be a fair contender to be the main method of data transfer for our delivery drone project. The overhead only involves three extra bits so while that will have to be considered, it would not make too much of an impact on the overall system. Along with this, the rate of data transfer is only limited by the baud rate which is usually manually set to 9600. The only major issue that we would have to consider is, yet again, the sheer amount of data being sent to the main controller at once. With multiple sensors and the GPS module constantly sending data, a 9600 baud rate may be too much for the main controller to handle which would require us to lower the overall data transfer rate leading to a slower overall processing and reaction time.

4.3 Economic Constraints

Arguably the most important constraint of the project is the economic or monetary constraint. Since the project is self-funded, all of the costs are coming directly from the pockets of the group members. While we may be able to contribute a few thousand dollars per person, we have all agreed that we do not want that to be the case. We want this project to be economically efficient so we need to be considerate of the parts we are choosing to make the prototype and the final delivery drone. Although the COVID pandemic has mostly come and gone, there are still leftover remnants of its effects in terms of overall part and device production. Much of the project will consist of pre-build and purchased parts so this is something that we must be aware of when prototyping.

Our delivery drone project can be considered a commodity or a luxury. It is not a crucial part of any system nor does anything that it can be applied to completely rely on it functioning properly. Therefore, we have a slightly greater amount of leniency for how often the drone completely functions. It does not *need* to work 100% of the time, however that is obviously preferred by everyone. Along with this, the drone itself is not very large in size and, therefore will require less materials overall to produce. There is also a leniency in how well each of the parts of the delivery drone work. Because the drone is considered a commodity and it is not very large in size, it is not too detrimental if errors occur during the delivery process. This is specifically referring to minor errors such as the most optimal path not being taken or the drone taking slightly longer than expected to reach the desired location. Errors such as crashing into something or someone, delivering to the wrong location, or causing damage to the package are obviously not acceptable. This type of leniency with regards to the performance of the drone in the specifically mentioned fields allow for the group to consider parts that may not be one hundred percent accurate and/or reliable. It lets us consider parts that may cost less and deliver less overall performance so that we can lean into the drone being more cost-effective. Despite all of this, the group desires to deliver a completely reliable and working product and will therefore, consider parts that may cost more but give better performance. The overall desired cost limit that was decided for the project is not necessarily a hard constraint but rather a preferred option. The cost limit that was chosen allows for some room in terms of the quality of the parts we choose and the cost of said parts.

4.4 Time Constraints

The amount of time that we have to finish the project is more than what is generally expected. While this report had to be done within the first five months of the year, the prototyping and building phase has significantly more time. Since everyone in the group plans to graduate in the fall of the year 2023, we technically have the entirety of the summer to work on the project even though it is not specifically designated or required. Normally time would be a major

constraint upon a project of this type as a working prototype needs to be finished by the end of the graduating semester. In the case of this delivery drone project, we technically have two semesters to work on the prototyping and building of the drone which we the group plans to take full advantage of. It is a circumstance that is not often available to many of our peers and therefore, we will utilize it as much as possible to help improve the overall quality of the project.

4.5 Safety Constraints

Safety is always a top priority when designing any system that interacts directly or indirectly with people. It is always absolutely necessary that a project not intend to cause direct harm to any individual and it is very heavily prioritized that a project not unintentionally cause harm to anyone. Safety being a reliable aspect of our project is part of what makes the drone able to properly do its job in the first place. Without this heavy importance placed on safety, we would not be able to advance as much as we could and be where we are now.

With all of that being said, our delivery drone is not limited by the safety constraints as much as other projects may be. Our drone intends to be relatively small in size and move fairly slowly. These features are essentially self-restricting safety constraints as the small size and low speed help to assure that the drone is almost completely incapable of causing harm to an individual. Despite this, there are still obviously unintended ways of the drone causing harm such as someone tripping over it. The group intends to minimize any and all possibilities of the drone being the cause of harm to an individual.

4.6 Manufacturing Constraints

Constructing the drone itself is a major part of the project as a whole and comes with its own set of constraints. The first main one being where and how we are going to construct it. We are fortunate enough to have support from the University of Central Florida in terms of providing environments that contain the tools necessary to construct a project of this caliber. UCF openly provides access to a Senior Design Lab that is allowed to be utilized by any student who is currently working on a senior design project. Along with that is the TI Innovation Lab which is also open to students for free use. The scope of the project also does not contain any particularly complex manufacturing as most of the process involves putting together pre-built parts. It would be safe to say that we do not have to worry about a lack of opportunities in regards to the ability to physically build the delivery drone.

4.7 Ethical Constraints

Like safety, ethics is also a top priority for the group. Every member wants to assure that the use and/or production of this drone does not cause any ethical concerns among anyone who may encounter the drone. Due to the scope and size of the project as well as the nature of the delivery drone itself. There is not too much to consider in terms of possible ethical issues with the drone functioning in a live environment.

4.8 Sustainability Constraints

Our delivery drone intends to be able to be active in both indoor and outdoor environments. This brings up the concern of the reliability of the parts used to build the drone in these situations. Any parts that are used to build the delivery drone must be able to withstand a high-heat and high-humidity environment due to the location in which we intend to make use of the drone. The parts, and by extension the entire drone, must be reliable and not malfunction or fail to work when exposed to the previously described environments. This issue places a restriction on the list of parts that are available to us which leads to the issue of sustainability. If a part were to malfunction and/or fail to work properly, we would have to order a new one. Ideally we would reorder the same part so as to not cause any complications in regards to compatibility of parts as well as our familiarity with the parts we are using. However, if that is not possible for whatever reason, this restriction might cause some issues with figuring out a replacement option.

Section 5: Hardware and Software Design Detail

The system design is done in different sections that together will incorporate the entirety of the design and prototyping including the PCB design, the overall schematics and case diagrams of the product.

5.1 PCB Design

For this project, we were required to create a PCB design that would be implemented into our project. This PCB design was created using Eagle Software. Many components of this PCB will be discussed thoroughly below, including the Voltage Regulator, I2C Transmission, General PCB layout, Sensor PCB Design, Motor Control Design, and GPS System Design.

5.1.1 Voltage Regulator Design

It is widely available on the market today several voltage regulators, including the LM317, that are widely used because they provide a variable output voltage within a specific range of the regulator. As well as regulating the voltage across a variable resistance, the LM317 also uses a stable voltage of 1.25 volts as a reference voltage. In the feedback circuit of the regulator, this voltage is compared with the output voltage and is fed back into the system. In order to achieve a steady and precise output voltage, the voltage of the input is adjusted until it equals the voltage of the reference voltage.

There are three pins on the LM317: the input pin, the output pin, and the adjustment pin. The input pin is connected to an unregulated voltage source, and the output pin is connected to a load, while the adjustment pin is used to adjust the output voltage. In order to determine the output voltage, a voltage divider network is connected to the adjustment pin and ground. The voltage across the resistors in the voltage divider determines the output voltage. With the output voltage and the reference voltage being used as inputs into the regulator's feedback loop, the resistance of the adjustable resistor is adjusted in order to accommodate the output voltage, which changes the reference voltage at the same time as the output voltage changes.

The LM317 has the ability to produce output voltages that range from 1.25 volts to 37 volts, which is why a voltage divider is needed to set the desired output voltage. The resistance divider should be sized appropriately in order to determine the desired output voltage. It is important to remember that the adjustment pin must be connected to a smaller resistor than the output pin should be connected to. Due to the internal current flowing through the adjustment pin of the LM317, there is a voltage drop across the resistor due to the LM317, which is the reason for the LM317. As a result, there is a voltage across the adjustment pin of around 1.25 volts. In order to calculate the value of the resistor, which is connected to the output pin, it is necessary to subtract this voltage from the desired output voltage.

Overall, the LM317 voltage regulator is based on the principle of regulating voltage across variable resistances, as described above. Based on the reference voltage of 1.25 volts, the output voltage is compared with the reference voltage, and the output voltage is adjusted to match the reference voltage. A voltage divider network is connected to the adjustment pin to set the output voltage. The feedback loop adjusts the resistance of the adjustable resistor to maintain a steady output voltage using the voltage divider network. This LM317 can regulate output voltages between 1.25 volts and 37 volts. To adjust the output voltage, it is necessary to select the resistor values appropriately to the desired formula of $Vout = 1.25 * (1 + \frac{R1}{R2})$. Using this formula we present the two PCB designs below for a 3.3, and 5 volt regulator.

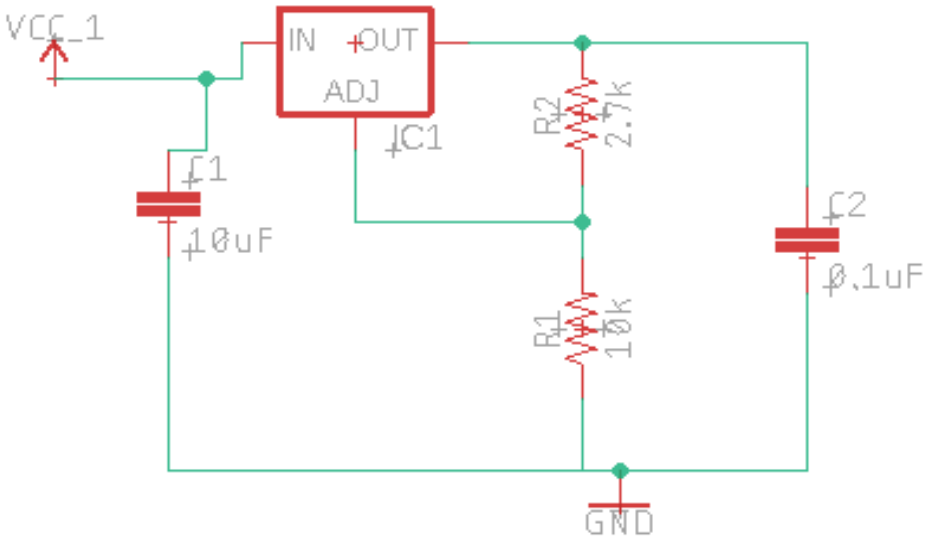


Figure 5.1: 5V Voltage Regulator

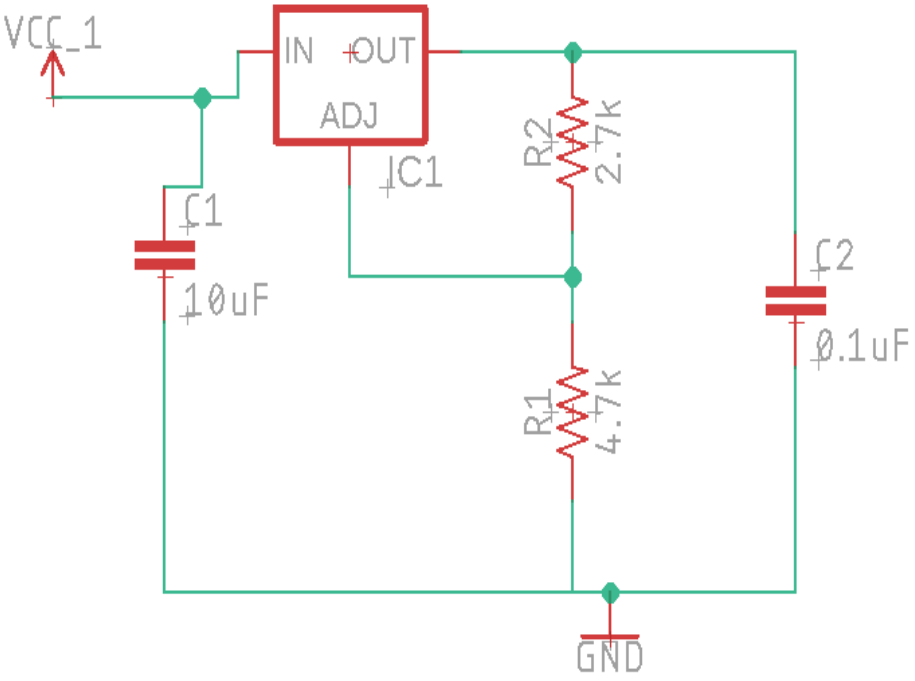


Figure 5.2: 3.3V Voltage Regulator

5.1.2 I2C PCB Design

The design of our PCB requires a shared information pathway that allows data and warnings to be presented to each microcontroller. I2C is often a good choice for connecting short distanced, low speed devices. With the primary usage of Input and Output to peripheral devices like sensors the system we are designing calls for. For instance I2C is used to establish communication between two or more integrated circuits, these integrated circuits being the different microcontrollers.

I2C has the advantage of multi-master and multi-slave communication over other communication protocols, such as serial port communication and SPI, in many ways. Thus, more than one master IC can communicate and control a slave IC, speeding up the embedded system's process and adding functionality. I2C is also capable of addressing slaves. A component can be added to the bus without using CS lines. As a result, CS lines are unnecessary, and members can be added to the bus more quickly.

Among other advantages, I2C can establish communication between several devices using the I2C protocol. Only two bidirectional signal lines are required. Additionally, due to its simplicity, the I2C protocol has a more robust error-handling mechanism than any other communication protocol worldwide. There are two types of acknowledgments in I2C: ACK and NACK. This feature enhances error detection and correction using ACK/NACK error correction.

Lastly, I2C is an adaptable protocol that allows it to work with both slow and fast ICs, thus making it suitable for various applications. In addition, it has several advantages that make it an efficient method of short-distance intra-board communication. These particular features allow the I2C to be considered a reliable source of communication.

For our design, since we were incorporating the same microcontroller, to avoid confusion and keep the programs about the same. With this runs the issue of shared I2C address which would inflict significant matters on the designed communication network. To overcome this issue, we implemented an I2C Multiplexer that will change the I2C address of each microcontroller to allow clean and clear communication between master and slave MCUs.

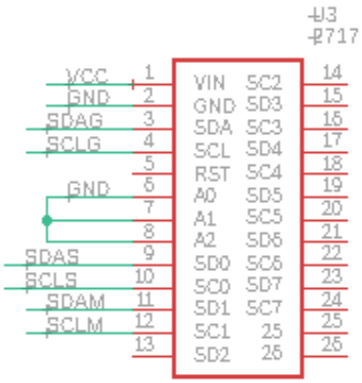


Figure 5.3 I2C Multiplexer

5.1.3 MCU PCB Design

As a result of the design of this project, we have concluded that more than one Microcontroller is required to handle the amount of data and calculation required in order to build a fully functional vehicle. As a result, we separated the partitioning into three separate sections: the sensors, the GPS, and the motor control systems. As we proceed through each of these sections, we will discuss the uniqueness and the requirements met in each of these designs for the system.

It is very important to understand that before we dig into the PCB, each unique function of the Microcontroller will require a basic setup to be able to function. For this design, we are going to follow the standard rule that the direct power source connected to the MCU must also have a decoupling capacitor attached to it. One of the most important factors in ensuring a successful system is power, and the first step is power. In each design, you can see a 47 nF capacitor attached directly to pins 7,20, and 21. Attached to this input is a 47 nF capacitor that is directly connected to the ground. As a result of this setup, the MCU is protected from potential electrical damage, which may occur at any time.

There is one of the key components in every great microcontroller design, and that is adding an oscillator to the board. The purpose of adding an oscillator is for timing and control of the system as a whole. The microcontroller executes all instructions in sync with the oscillator's clock signal. In the absence of the oscillator, our entire program can start to fail, and parts could become missing during the operation of the vehicle. Besides the overall usefulness of Crystal Oscillators, they offer a different advantage that can be calculated into the comprehensive choice that they offer. This is a good choice for any design because of the stability and reliability offered to the project. It is a good choice for any design because of the accuracy, low cost, low power consumption,

compactness, and high frequency generation. The crystal we use for this project is a 16 MHZ crystal connected in parallel with two 22pF capacitors that lead to ground; this will ensure the MCU runs at 16MHZ.

A reset switch is a crucial component of any engineer's design when creating a masterpiece of a PCB design. The designer needs to ensure that the reset switch is always included in his or her design. For this project, the PCB designs were equipped with a reset switch so that a restart could be performed if necessary. Initially, it would be used to restart the system if a part had disconnected or, for any other reason, the system was experiencing difficulty. With this feature, the team would be able to reset the MCU and prototype while allowing other parts of the board to continue working. Our PCB is a relatively simple one designed so that the 5 volt power supply is fed continuously into pin one until the reset switch is pressed. Whenever the reset switch is pushed down and connected to pin 1 to the ground, a momentary interruption and connection occurs. This interruption occurs when the switch is pushed down and connects pin 1 to the ground, which initiates the board's reset.

I2C as mentioned earlier in our design process is the main form of communication between Microncontrollers for this project. Each board is required to be connected to the I2C Multiplexer to allow communication between the different systems and functions. These connections are connected to pins 27, and 28 using the names of SDA and SCL respectively. Because of the design of I2C there is a potential for open-drain interference that could cause a loss in signal. To avoid this interface and the potential loss of a signal we add a pull up resistor to each input of SDA and SCL as seen in each design.

5.1.3.1 PCB Sensor Design

The design we have planned requires five separate sensors to allow for obstacle avoidance. The sensor chosen for this, as mentioned previously, is the HC-SR04 sensor. For the specific sensor in our design, it requires two digital pins to be able to operate and function properly. Along the lines from the digital I/O, we add a pull-down resistor so that it helps to reduce noisy and erratic readings in the sensor signal, which results from the way resistive sensors work.

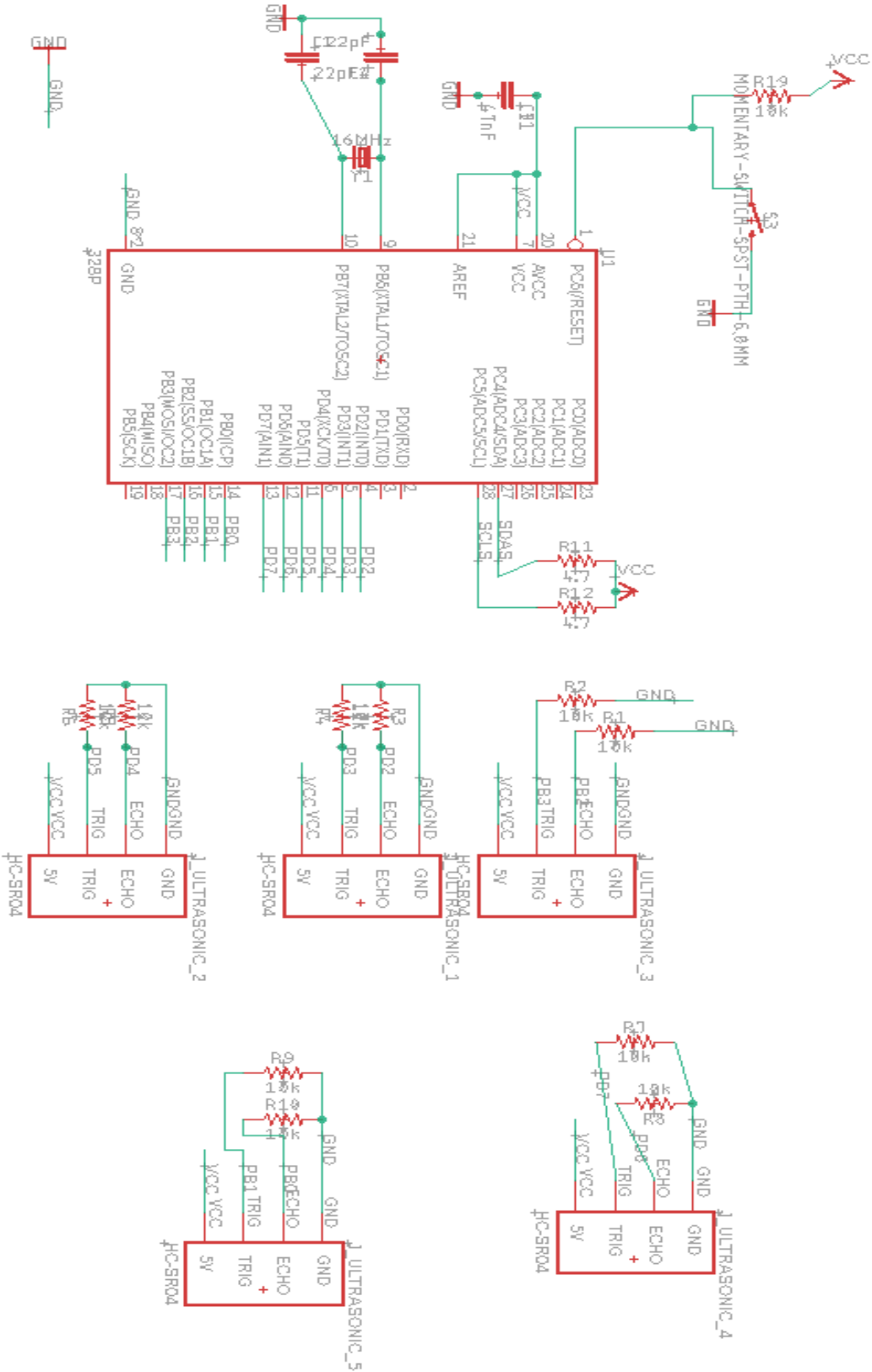


Figure 5.4 PCB Sensor

5.1.3.2 PCB Motor Control Design

The design for this system is a relatively simple one. Using basic DC motors, we only need a directional input and a speed input. Knowing this, we used a dual motor driver, in which we ran four input signals and two EN signals to the driver. From this, the Motor driver outputs four different outputs, which outputs 1 & 2, will run to the right side of the motors, which are connected in parallel. While in production, 3 & 4 will be run to the left side of motors, which will also be run in parallel to avoid extra drivers required.

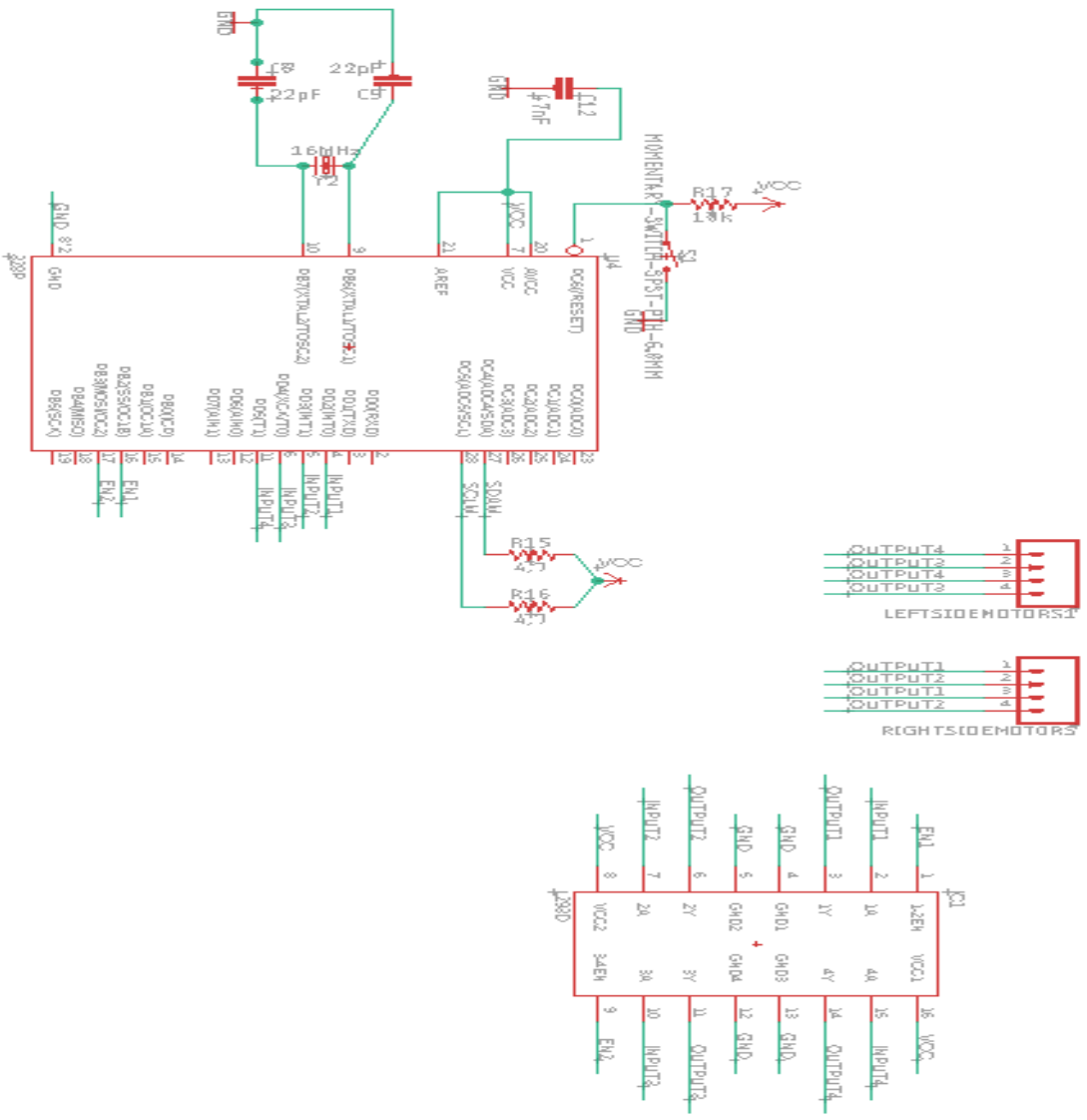


Figure 5.5 PCB Motor Control

5.1.3.2 PCB GPS Design

The design for the Microcontroller in charge of the GPS is a relatively simple one. Using the RXD and TXD pins, we connect those pins to pin 2 & 3 on the GPS module. Besides that the power supplied to the GPS module is 3.3 V from the 3.3 voltage regulator.

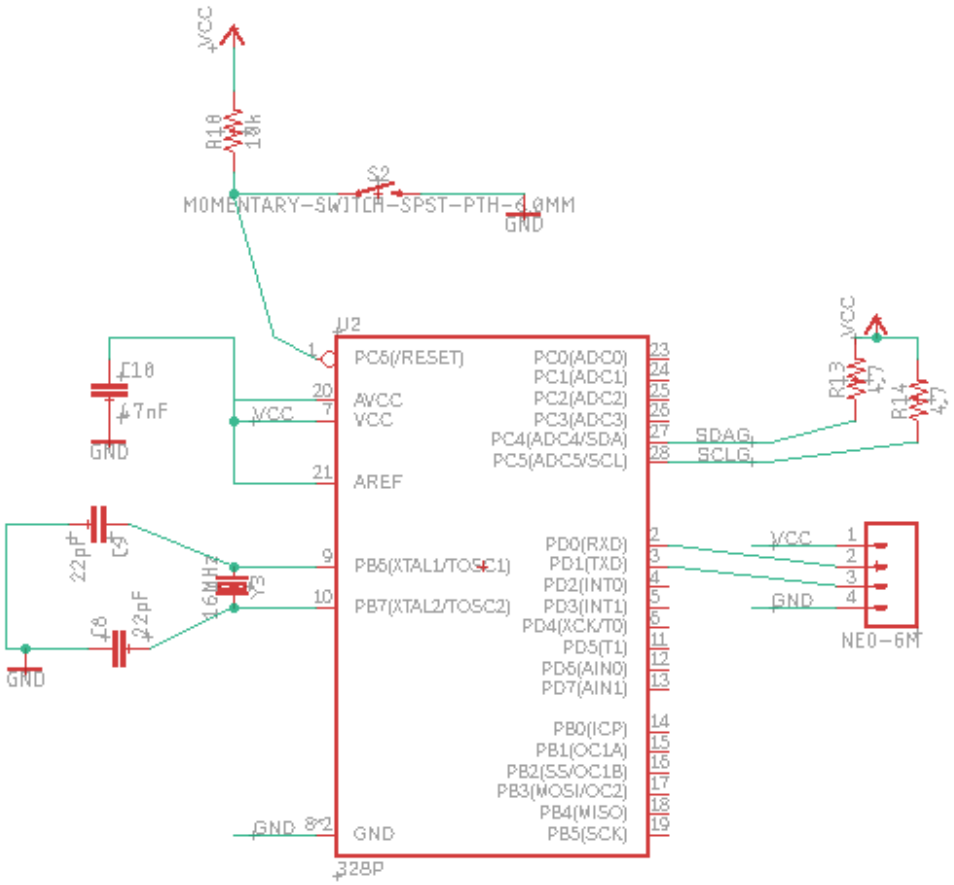
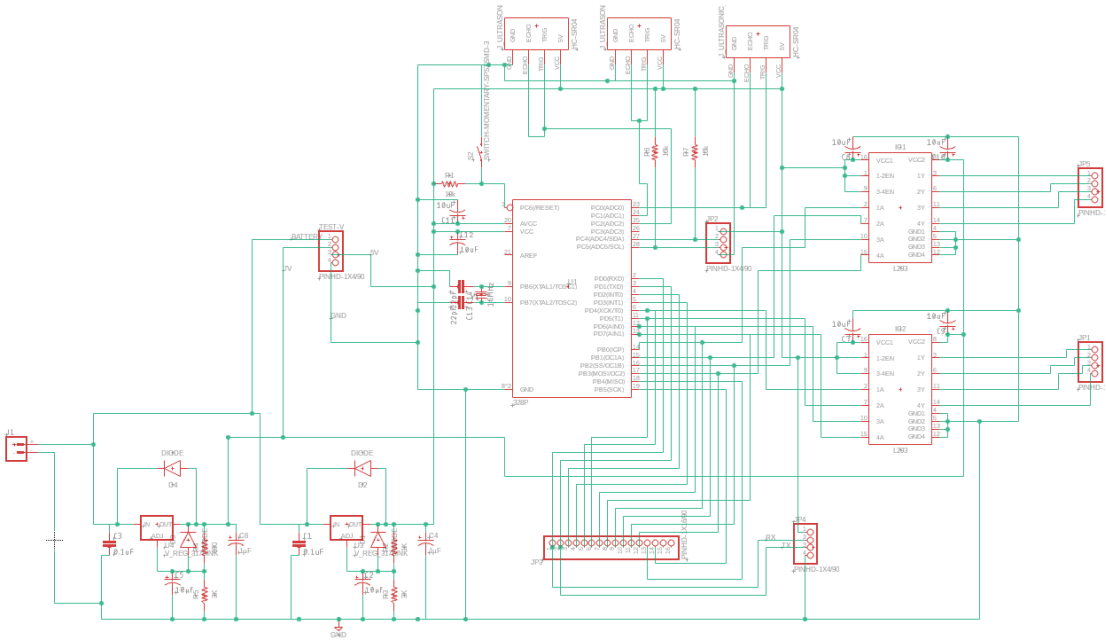


Figure 5.6 PCB GPS

5.1.3.3 Total PCB Design



5.2 Hardware Design

This section details the physical construction aspect of the robot design which includes the chassis, battery system, and motors among others. The following information will go into depth about how the chosen components of the system are able to fulfill the constraints set forth for the completion of our project and how each component adds to the functionality of the delivery bot. Each component is presented and analyzed in relation to its effectiveness and how the component fits into the design of the overall project.

5.2.1 Chassis Design

Initially, the idea for the chassis was to purchase a robot kit from a seller that matched a four-wheeled chassis with several mounting points to allow attachment of our payload carrier as well as the PCB and microcontrollers necessary for the operation of the delivery bot. This meant that the chosen chassis kit must fit within the size constraints of being smaller than 1ft x 1ft x 0.85ft while also having the aforementioned aspects. Section 3.6 of this document details the process of research in which several chassis kits were selected and analyzed to determine whether the components would be sufficient for the purposes of the project. Four different chassis kits were picked and discussed in addition to the idea of constructing the robot by purchasing each component separately, however, the final decision ended in the Hiwonder 4WD

Chassis Kit. The below figure displays the chosen kit, both fully constructed and with the components laid out.

This particular chassis kit features the chassis platform, an extension bracket, wheels, motors, and all of the necessary screws and nuts to construct the robot.

Fig 5.7: Project Chassis and Components

The dimensions of the kit are well within the requirements at 7.09 × 5.51 × 3.5in with the maximum load of the kit being 1500g which is also within the load requirement specification. The material that the kit is made of is aluminum which was durable enough for any situation that was used for testing the robot. The planned use of the space for the chassis is that the top level platform is used mostly if not entirely for the payload carrier box and locking mechanism while the bottom platform will house the microcontrollers, battery system, motor attachments, wiring, etc. At the end of the project, the top level consisted of only the carrier box and the GPS module and antenna while everything else including the microcontroller, sensors, and PCB were located on the bottom level.

For other attachments to the chassis, the carrier payload is constructed using a simple plastic or wood material taped on the top level of the chassis. Furthermore, the locking mechanism is attached to the carrier mostly self contained in order to reduce issues of tampering with the wiring and connections. The motors are attached below the circuitry level to avoid any collisions with the components. As it can be seen in the fully constructed figure, the motors are below the entire system and leave the bottom level platform to be completely free space for the microcontroller, PCB, battery, and all of the necessary wiring to have the system functioning.

The wheels of the chassis are given as shown by the figure. They are small enough to allow space for the wiring to pass through between the bottom and top level platforms and should be large enough to traverse the expected terrain including light gravel and small inclines. While testing the drivability of the system, the effectiveness of the tires will be analyzed and may be switched for more suitable tires that have a more off road tread to avoid slipping and improve

grip on other surfaces. Other than these considerations, the chassis design is relatively simple as a kit is being used and will work with the other components that we have chosen for the project. The only other consideration that may need to be reviewed is to add extensions to the platforms in the case that the components take too much space with the given materials.

5.2.1.1 Chassis Design Summary

Below is a simple table summarizing the chosen chassis design and its aspects (for ease of viewing in the final document).

Chassis Design				
Construction	**Operation**	**Motors**	**Size**	**Max Load**
Aluminum Alloy	4WD	TT Geared DC	7.09 × 5.51 × 3.5in	1500g

Table 5.1: Chassis Design Summary

5.2.2 Motor Design

This aspect of the hardware details the motor functionality and the connections that will be necessary to operate the delivery bot. The motors that come packaged with the chosen chassis kit are labeled as 12V TT DC Motors. This means that the motors are plastic construction with a maximum output voltage of 12 volts each. Additionally, each motor is geared internally with a ratio of 1:120. In this case, in order to allow control of the motors including the speed and direction of the rotation, another component is necessary between the interface of the motor and microcontroller. This will be discussed in the next section of the motor design. Each motor is wired only with two wires designed to be connected to a negative and positive terminal of a battery. As stated previously, this only operates the motors with no ability to change the speed or rotation (unless the positive and negative terminals are switched which would not be possible during operation). Additionally, the voltage of the motors are typically not safe to be interfaced directly to a microcontroller. This is where the motor driver is implemented in order to give control to these simple DC motors as well as act as somewhat of a voltage regulator before being connected to the microcontroller.

Other than addressing the problem of giving control to the DC motors, the actual design of the motor is a very simple plastic geared motor typically used in robotics projects similar to the one we are developing. They are typically given gearing in order to increase torque and allow the system to traverse while under load. In this case under load could mean going up inclines or with additional weight introduced onto the vehicle. As stated in section 3.7, the analysis of the RPM that is the typical output for these types of motors is around 200 RPM which translates to a speed of about 1.52 miles per hour. This fits well within our speed

limit and should be able to maintain with additional weight as this calculation is with only one motor while our system will have four.

5.2.2.1 Motor Driver (L298N Dual H-Bridge Motor Control)

As mentioned previously, this motor driver is necessary for the control of the motors from a microcontroller which in this case is an Arduino board using the ATmega328p microchip. The motor driver has two ports for motors meaning that each motor driver can handle two separate motors. This means that two L928N motor drivers are necessary to control the rear and front wheels of the delivery bot. Below is a simple diagram showing how the motors interface with the motor driver and the motor driver then interfaces with the microcontroller. No specified pins are shown as more detailed representations will be discussed in an upcoming section.

Fig 5.8: Connections between Arduino and Motor Driver

Again, this particular setup allows the motors to be controlled through the board rather than varying the levels of voltage to give more or less power to the wheels. This means the variables like rotation and speed can be more accurately dialed into any situations that the delivery bot may encounter like obstacle avoidance. Furthermore, the power delivered to each motor can be adjusted if for example there is an abnormally heavy payload, the motors can be tuned up in order to account for the extra weight.

The L298N Motor Driver utilizes both a Power Width Modulation technique in order to allow for speed control of the DC motors as well as an H-bridge circuit to switch the polarity of the voltage to the motor allowing for the ability to change the direction of the rotation. The PWM technique for controlling speed involves controlling the output time of the voltage of on/off components like our DC

motors. The PWM sends a series of on and off signals known as duty cycles. The width, or time that the signal is on represents the average voltage sent to the motor. This means that a lower duty cycle will result in a lower average voltage sent to the motors and as a consequence, reduces the speed of the motors. With a higher duty cycle, the average voltage and speed increases. Below is a diagram explaining this idea used in the motor driver.

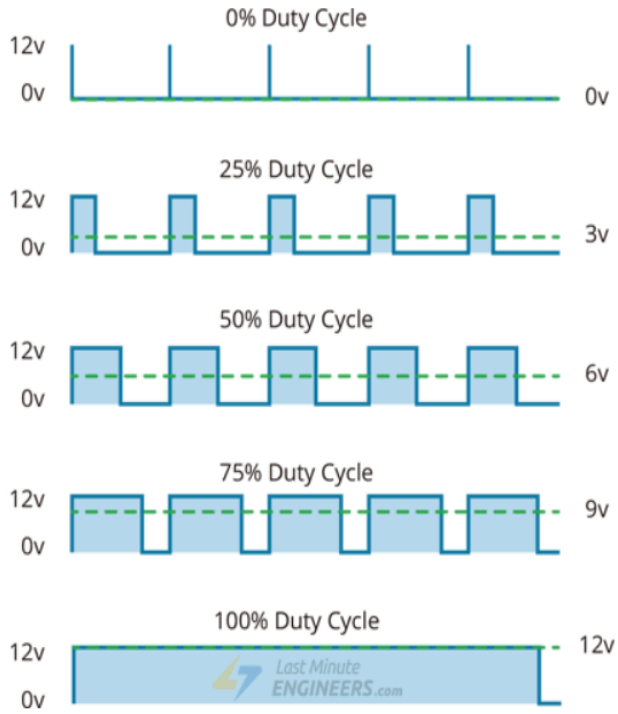


Fig 5.9: Representation of Duty Cycle using PWM

As stated before, the width of the duty cycles represents the average voltage output which is shown by the green dotted line. This allows the DC motors that are typically only on or off at all times, to be controlled more precisely.

The H-bridge aspect of this motor driver is the component that allows the rotation of the motor to be switched. Without this component, in order to change the rotational direction of the motor, the positive and negative connections to the battery would need to be switched, which while operating is not possible. So, this circuit involves a series of switches that close and open depending on the desired outcome, automatically switching the polarity and thereby changing the rotation of the motors. Below is a figure showing this circuit.

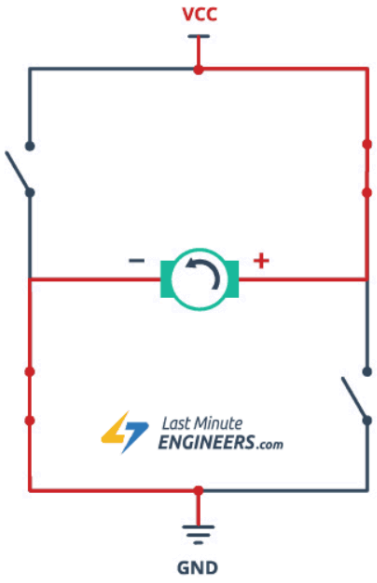


Fig 5.10: Representation of H-bridge Circuit

In this example, the motor is rotating counterclockwise with this switch configuration. By opening the highlighted switches and closing the others, this swaps the polarity resulting in a clockwise rotation instead. This H-bridge circuit on the motor driver is essential for the operation of the delivery bot as this will allow a reverse function which is helpful in avoiding obstacles. In addition, this component could improve the drivability as sharper turns could be made by reversing one side and leaving the other in the forward rotation. This allows the delivery bot to rotate in place which would help in tight spaces.

5.2.2.2 Motor Driver Option 2 (L293D IC Motor Driver)

This motor driver type is a bit different from the previous option in terms of form but it has the same type of technology built into it in order to allow the simultaneous control of two motors in any direction. Each unit is able to control four motors in a single direction but only two in any direction. Therefore we will need two units for the operation of our delivery bot. The design of this particular motor driver is an integrated circuit that also has the same H-bridge and PWM control as the L298N motor driver discussed previously. This design is a bit more simple and compact which is most desirable for our project. Some recommendations on this device state that the unit may increase in temperature while under load. Since our delivery bot will likely have an average running time of 1 hour at one time with several changes in speed and direction to the motors, this presented a problem and caused some issues with itself and other components. However, this is the first motor driver that we used due to its simplicity, price and size. If the motor driver has issues with overheating, we will

then use the previously mentioned L298N motor driver as it has a built in heat sink to allow for heat dissipation.

Although a different form factor, the L293D driver has a simple connection to the Arduino board and motors that we used. Below is the pin diagram for the driver.

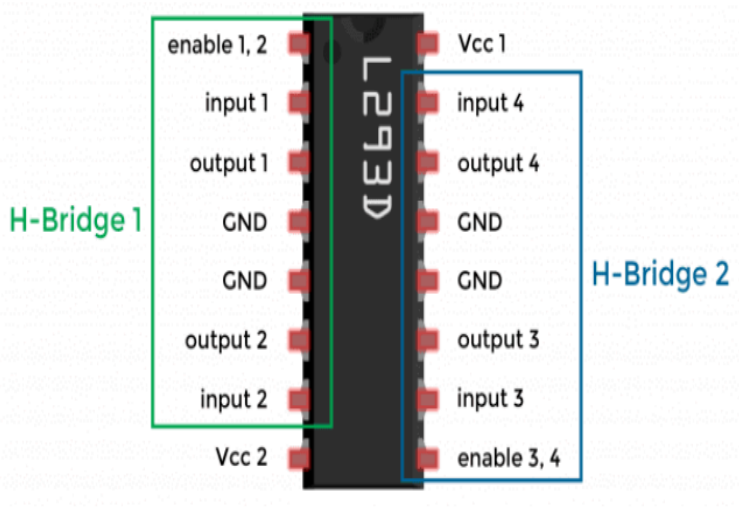


Fig 5.11: Pin Functions of Motor Driver

Vcc 1 supplies the unit's internal logic with voltage (5V) while Vcc 2 supplies the motors voltage of up to 36V. As the diagram displays, the left side of the unit controls one H-bridge and the right controls the other allowing for the control of two motors at once. To control the speed of the motors, a high signal can be sent to the enable pins. The motors are connected to the output pins and are controlled by the input pins. To change the direction of the motors, a high signal is sent to one input pin and a low signal is sent to the other. To reverse the direction, simply switch the high and low signal to each input pin. As stated previously, the control system is the same for the other H-bridge to manipulate an additional motor and as a result our delivery bot will require two of these motor drivers.

5.2.2.3 Motor Connections with Microcontroller

Due to our implementation of a motor driver to control the speed and direction of the motors, there were not any direct connections from the Arduino microcontroller to the motors. Instead, the MCU will connect to the motor driver, supplying voltage, connection to ground, and to the enable pins, using the ones that allow for PWM control as this enables control of the speed of the motors as stated in the previous sections. Since the motors we are using are rated up to 12V, we supplied the motor driver with 5V from the Arduino to allow for the maximum speed of the motors if necessary as 5V is the maximum output voltage for our microcontroller selection. The average output voltage can be modulated

as explained before using the PWM pins and utilizing a smaller duty cycle, meaning the voltage received by the motors from the motor driver unit will not always be at a maximum.

To view the pin diagrams of the Arduino Uno board and the L293D Motor Driver, refer to their respective sections. The connections from the microcontroller to the motor driver are the 5V output pin on the Arduino to the Vcc 1 (pin 16) of the motor driver. This supplies the motor driver with voltage for the internal logic of the H-bridges. Next, ground is connected by connecting the GND pins on the Arduino to pin 4 and 5 on the motor driver. To allow speed control, the PWM pins on the Arduino (pins 3, 5, 6, 9, 10, or 11) are connected to the enable pins on the motor driver with pin 1 being speed for one motor and pin 9 being speed control for the other. Since the direction of the motors do not need a variable voltage supplied and only need a high or low signal depending on the desired rotation, any unused pins from pin 2 to pin 13 on the Arduino may be used (whether PWM enabled or not). These are connected to the input pins on the motor driver (pin 2, 7 and 10,15 respectively) and are controlled by sending voltage to one of them on each side at a time in order to change rotation. Lastly, the final connections to be made to enable the motors are the actual motors. This is done simply by connecting the motors to the output pins on the motor driver (pin 3, 6, and 11, 14 respectively). Once this connection is made, the motor system is ready to be used once the Arduino board has power supplied to it, which will be discussed in the next section on Battery Design.

5.2.2.3 Motor Design Summary

Below is a table describing the final decision on the motor design we will be utilizing in our project. The attributes were found based on the chassis kit we are using. If it is found that the motors are insufficient for our use case, if for example, they are not powerful enough or the current draw is too much for the chosen batteries, then they can be switched for the other options found in section 3.

Motor Design Summary						
Motor Type	**Construction**	**Gear Ratio**	**Voltage**	**Current Draw**	**Maximum RPM**	**Motor Driver**
Geared DC	Plastic	1:120	3-12V	150-200 mA	200	L293D

Table 5.2: Motor Design Summary

The motors that come in the chassis kit have the above attributes and should have all of the relevant hardware necessary to attach to the chassis. Furthermore, the motor driver we are using is an integrated circuit type that allows control of speed and rotation through the H-bridge switch and the PWM

pins on the Arduino microcontroller. The voltage we are using the motors at is 5V as that is the maximum voltage output available on the microcontroller.

5.2.3 Battery Design

The following section will explain the type, capacity, and connections to the system of the chosen battery for our application. This will include descriptions of the battery, simple equations to determine the current draw of the system (and the resulting capacity to support the draw), and how the battery will interface with our system. Since the battery will be purchased, the design and technology of the chosen battery will be analyzed and explained as to why the particular battery was chosen.

5.2.3.1 Battery Type

The battery chosen for the operation of the delivery bot as explained in section 3 is a singular Lithium Polymer battery. A singular battery pack was chosen as opposed to having multiple batteries for the different systems (i.e. one for motor control, one for sensors, etc.) because this will ensure that only one thing needs to be charged in order for the system to operate. This also will prevent some components from losing power before others. For example, since the motors will require a higher draw from the battery, this would mean the battery controlling the motors would lose power before the other components resulting in loss of function prior to the rest of the system. This does mean however, that the battery pack must have a high enough capacity to allow all components to be able to draw from it for a sufficient amount of time, which will be discussed in the next section.

The material decided for the battery is a Lithium Polymer. The reasons for this decision are most importantly that they are rechargeable. It is more cost effective for this project and for testing to have a battery that can be used several times so this automatically removes alkaline batteries from this decision. Another consideration that would be necessary is size and weight. Since lead-acid batteries are typically very large and heavy, this would reduce the available weight for the carrier and payload and would also increase the load on the motors. For this reason, lead-acid batteries were also decided against. The last few battery options were Lithium Ion and Nickel-Metal Hydride batteries. These were both good options as they were lightweight, rechargeable, with decent capacity. However, Lithium Polymer was the final decision as it also has these desirable aspects in addition to it being less susceptible to heat issues than Li-ion and better discharging characteristics for our project.

The technology for the Li-Po battery is similar to a Li-ion battery except the liquid electrolyte solution between the anode and the cathode is swapped for a porous gel polymer material. This is then put between two laminate layers that contain one cell. The cells are then stacked and bound and results in the final battery.

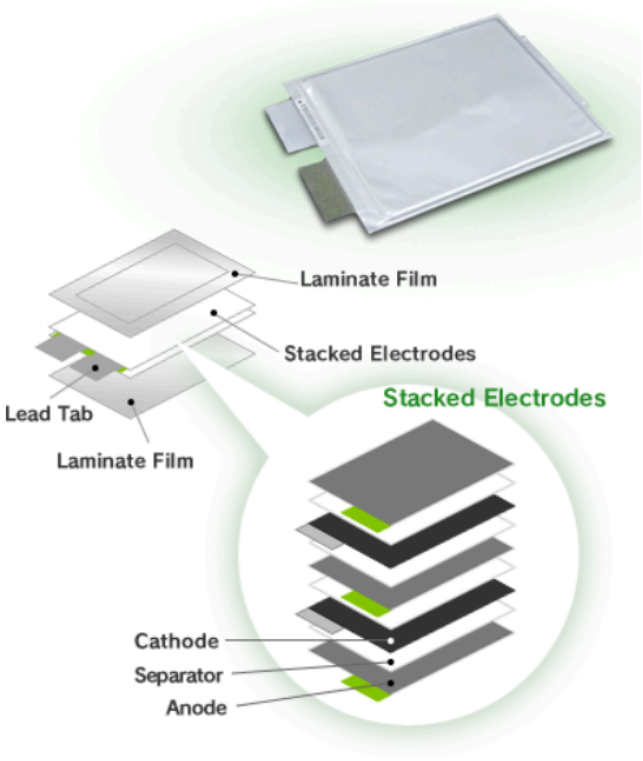


Fig 5.12: Li-Po Construction Diagram

The figure above represents the construction of a Li-Po battery as each layer can be seen stacked and bound by the laminate film. Our particular battery utilizes a 3 cell stack of 3.7V each for a total of 11.1V. The details of the specific battery will be discussed in the following section.

5.2.3.2 Battery Capacity

The maximum voltage output from the Arduino Uno is 5V which can be supplied to the motors. This also supplies voltage to the L293D motor driver. For other components, like the GPS system, sensors, locking mechanism, and processors, they are all within about 3-5V each. As stated in section 3, we utilize a Lithium-Polymer battery due to its characteristics of heat dissipation, size, and relatively high and continuous discharge rate. An 11.1V rechargeable Li-Po battery was the best for this use case as Li-Po batteries come in cells of 3.7V each meaning this pack would have 3 cells. To determine how long the battery

would last, an analysis of current draw would be needed to find how long a certain capacity would last under a certain load. For example, each motor would likely have a max current draw of about 100 - 200mA and the microcontroller and other components likely have a smaller current draw of about 20-40mA each. The motors have a current draw of about 1 - 2A, but only when stalled so this case will not be very common. The motors will not be running at the maximum load at all times so assuming an average current draw of about 200mA per motor, and another 100 - 200mA for all the other components, the system will have a total current draw of about 1 - 2A. With a voltage of 11.1V, and a desired runtime of about 1 hour, a battery capacity of about 2200mAh should be sufficient and is helpful to have an additional 500mAh to work with in the case that the motors are under higher than normal load or the additional components of the system are have a higher current draw than expected.

5.2.3.3 Battery Connections

Since the Arduino Uno board that contains the processor we are utilizing accepts a 12V source, the 11.1V Li-Po battery can connect directly to the microcontroller and provide power to all of the relevant components. The Arduino also comes with a built-in voltage regulator so there should not be any worries about overloading the components or board itself. Since the Atmega328p takes 5V as input the voltage regulator implemented in the PCB lowered the input voltage from 11.1V to 5V, to avoid burning down the 5V components while having a different voltage regulator for the motors.

5.2.3.4 Battery Summary

The overall construction of the battery is an 11.1V, 2200mAh Lithium Polymer battery that is about 200g based on similarly constructed batteries available for purchase. This particular battery composition is not the best at handling heat issues but it is better than some of the other considerations. While risk of failing due to heat is a real concern, this type of battery shouldn’t need anything more than a simple cover to protect from heat sources of other components or the environment. The capacity of 2200mAh was decided on based on the expected current draw from the rest of the components, with the motors being the highest draw of about 150mA each and a maximum draw (if stalled) of about 1.5A. This situation of stalling should not be a problem unless the delivery bot encounters terrain that it is not planned to traverse. Below is a table summarizing the design attributes of the battery.

Battery Design Summary				
Voltage	**Capacity**	**Weight**	**Composition**	**Heat Attribute**
11.1V	2200mAh	200g	Lithium Polymer	Positive/Negative (Worse than some, better than Li-ion)

Table 5.3: Battery Design Summary

Overall the battery design is sufficient for this type of robotics project.

5.3 Microcontroller Design

The microcontroller is an important part of the integrated circuit that is included so it can control the specific devices on the system like the sensor system, the power management and the motor. The chosen microcontroller selected for controlling these subsystems and the specific functionalities of this project is the atmega328p created from Atmel corporation, it is a high performance single chip microcontroller based on a 8 bits RISC processor core. It is designed to have read and write capabilities with 32KB internal flash memory, this specifications makes it the perfect fit as it is capable of performing complex tasks while maintaining the power consumption low.

This microcontroller is very versatile and easy to use making it essential to use when reading input data from sensors, processing the data and also being responsible for performing the output to the actuator components that are attached to them, like the motors to maneuver the bot to reach its destination. This microcontroller supports ADC, UART, SPI and I2C interfaces, the project uses these built in peripherals to communicate with the other components and subsystems.

5.3.1 Arduino Uno's Hardware

The Atmega328p comes included and is the main component in the Arduino uno development kit, the kit brings a software library that makes it faster and simpler to develop programs for the microcontroller, this platform facilitates the development of the project in the regard of the area of using and developing for the microcontroller as it provides many advantages.

The arduino unos is incredibly convenient specially for the prototyping phase of the design as the board itself includes a bunch of built in features thanks to the hardware included in the development board, making it simpler to connect to other components and basically ready for fast prototyping of complex systems.

The board has 14 digital I/O pins plus 6 analog input pins for easy connection to other components that can collect input data and to output data with a simple connection, it also possesses the capability of taking in “shields” which are basically board add ons that can be implemented if needed to it very easily like wireless connection shield can be tested on the arduino board.

The arduino board is compatible with standard breadboards, making the prototyping less complicated because this makes the process of connecting the components through jumper cables less definitive and eliminates the need of soldering components into the prototype phase also it lets the components be re used on other systems to test if a system is not working because a component might be damaged but if is working in another system then the system is the one having the issue. If changes need to be made to the prototype the accessible nature of the breadboard allows the change without any major issues. That is mainly why the breadboard compatibility of the arduino board is so useful.

The biggest major arduino uno advantage is the development environment that it offers for programming the microcontroller with the necessary code for controlling the subsystems, it provides a familiar IDE comparable to texas instruments’ code composer studio which every member of the team has used previously on courses taken. This permits for simple and easy load of the program into the microcontroller for easy testing and debugging. If the microcontroller itself presents too many issues for using, as a last instance the team could use the entirety of the arduino board in the project it will just be slightly more expensive. The following schematic is of the Arduino Uno development board from the official Arduino website.

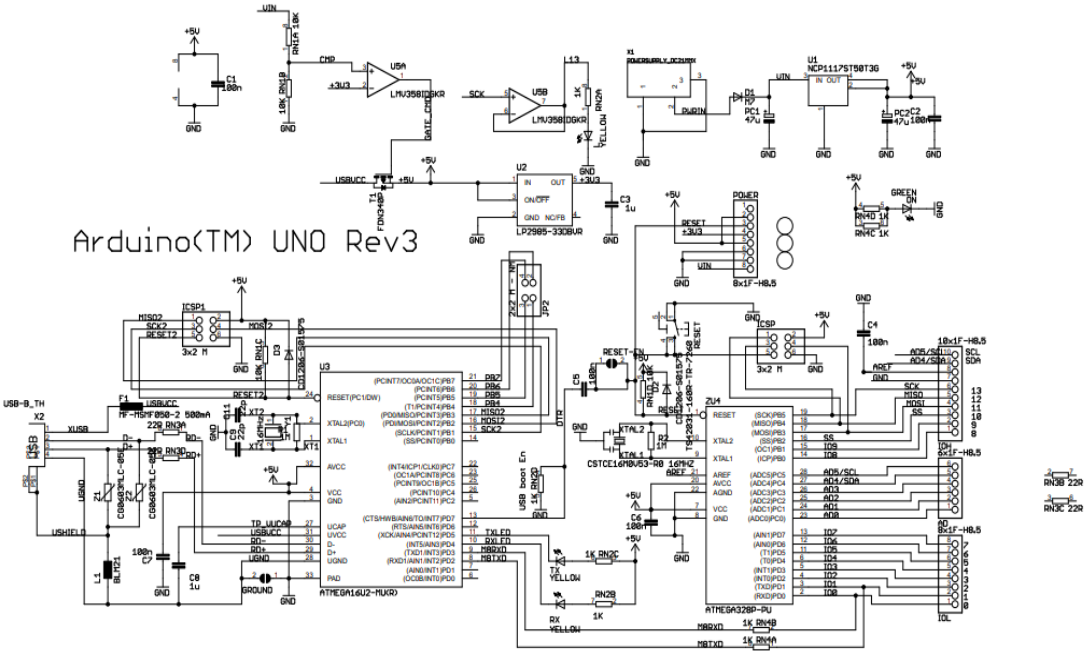


Figure 5.13 arduino uno schematic

5.3.2 Microcontroller Description

The Atmega328p has 28 pins that are divided into different functional groups like the digital I/O pins, the analog input pins, the power supply pins, etc. the Atmega238p includes a very useful clock system configuration can be used for some different frequencies and is based on a crystal oscillator. The clock can be configured in different modes: Normal, Fast pulse width modulation, phase correct pulse width modulation or clear timer compare match. For this bot the clock configuration that was chosen to use is the Normal mode which entails that it works at a frequency of 1-20 MHz to provide the adequate amount of computation power while also maintaining a stable power consumption, so the battery does not get drained needlessly making the clock configuration an important part of the microcontroller as a piece of hardware because the appropriate selection speeds up performance efficiently for every subsystem that the Atmega328p is used for.

The communication interface in specific the I2C uses two communication pins on the Atmega328p in specific the serial data (SDA) and the serial clock (SCL) pin so it can communicate back and forth to the other components, these pins require a pull-up resistor on each pin to make sure the signal is high when a device is not driving the line since the pins are open drain type. This interface has a dedicated hardware module that facilitates the implementation of the I2C

communication called “Two wire interface (TWI) “ which allows the microcontroller to communicate with multiple components using just the two pins mentioned above by assigning every device a different 7 to 10 bits address. This will permit the project to use less pins for communication between different components and therefore simplify the design by a little if possible.

5.3.3 Microcontroller Functions

The microcontroller unit is responsible for multiple functions throughout the entirety of the design, so multiple of them are present on the entirety of the project and positioned strategically on the PCB board to save footprint and money on components and PCB printing.

The microcontroller will be responsible mainly for the control of the sensor system in the aspect of collecting the data registered by the sensors and then sending signals when the data reaches certain threshold, there was a plan to have five to nine sensors using one to two microcntrollers per sensor depending on the location of the sensor, so if the sensor distribution is one or two sensor placed around the bot’s sides accompanied by one microcontroller. In the final design only one ultrasonic sensor was used with one pin of the microcontroller, reducing the need of using multiple microcontrollers as well.

Another microcontroller function will be the collection of data sent by the sensor and the CPU and then sending the signal to the motors to indicate if they should go forward or backwards and the amount of power driven to the motors this task will only require one microcontroller. Another function that will be assigned to a microcontroller is the control of the direction by turning the wills according to the data received from the sensors, for example depending which sensor sends the obstacle alert signal the bot will turn to the opposite way where the signal is coming from and depending of the data collected from the GPS module to send the signal to turn the wheels to follow the route.

The last microcontroller function was to ensure the safety of the bot ensuring the bot will not get tampered or left stranded by collecting data like the battery monitoring sending an emergency signal if the battery is really low, collecting data of the status of the bot if its stuck in the same place too long send emergency signal, if someone tries to forcefully open the parcel before it arrives its destiny also send emergency alert. In the final design this feature was

scratched and substituted with the feature of using a compass for self orientation and pointing to the GPS desired location.

Only one microcontroller, the Atmega328p is used for the complete system.

Table 5.4: function subsystems

Subsystem using microcontroller	Number of microcontroller used	Main function
Sensors	1 (previously 4)	Collect data from the sensors placed around the bot and send alert signal back
Motor	1	Collect the data from CPU and sensor subsystem and send power to motor
Navigation	1	Collect data from GPS and sensor modules and send data to navigate route
Self orientation (previously security)	1	Collect data from compass and react accordingly

5.3.4 Microcontroller Schematics

The Atmega328p provides a 28 pin package which is in a dual inline modality and they are arranged in two rows having 14 pins per row. The pins of the microcontroller function vary from eight digital I/O pins, another eight analog input pins, three pins for serial communications, three timer and interrupt pins, a reset pin, a couple of voltage supply pins and a couple of pins that can be connected to a external crystal or resonator for improved timing.

The plan for implementation using the schematic was connecting components that perform the same task to the one microcontroller and if its a component from a different subsystem a different unit of the same type of microcontroller will be used the UART peripherals allow for multiple connections to one microcontroller and making use of the at least eight general purpose I/O pins that are provided

by the Atmega328p. In the final design only one microcontroller was connected to all the subsystems.

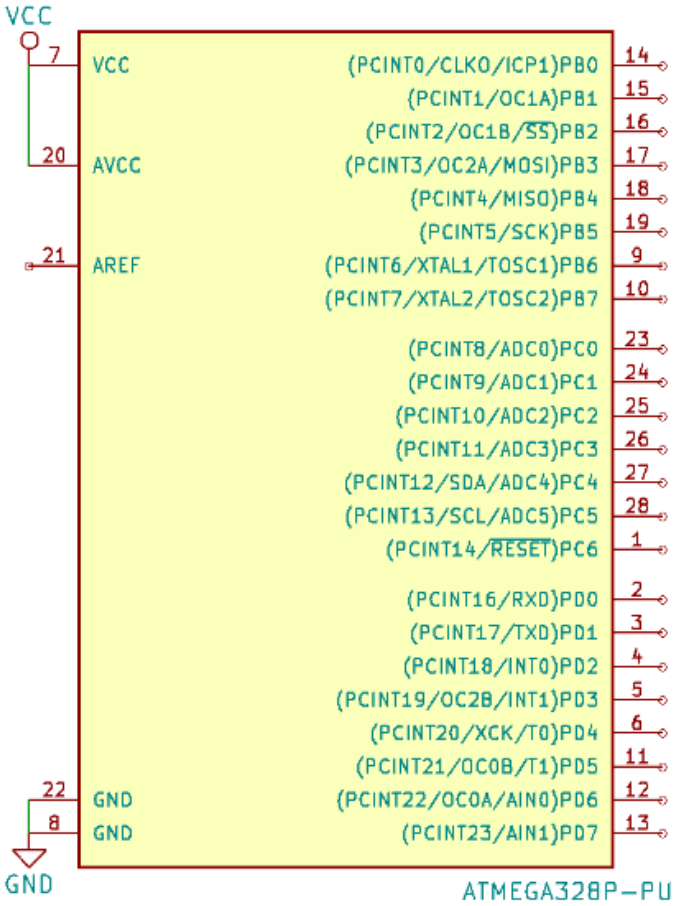


Figure 5.14 Atmega328p

5.4 Ultrasonic Sensor Design

The ultrasonic sensor prototype design was done to integrate the ultrasonic sensor with the MCU to make it able to detect distance of objects from the sensor and then record that information. The seamless integration of the sensor components including the integration of the microcontroller component into the design. The schematic for the sensor system represents the input and output layouts of the components. After this information is collected it is used to design a small breadboard prototype with the parts and an arduino uno development kit to load the microcontroller.

The design of the basic system sensor prototype is composed of 4 major components which are the ultrasonic sensor in this case the chosen one is HC-SR04, the development board containing the microcontroller in this case the

arduino uno development board containing the atmega328p microcontroller chip, and also implemented in this design was a simple red LED with a 220 Ohm resistor so the LED does not get burned out with the incoming electricity flow, as well as a total of four jumper cables to connect the components through the breadboard.

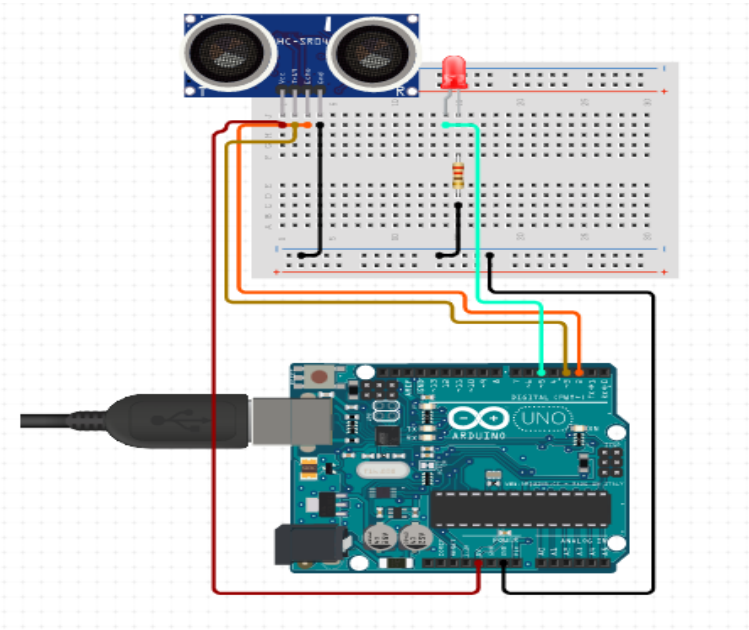


Figure 5.15 Sensor system schematic

5.4.1 Sensor Breadboard Design

The components that were used to set up the very basic breadboard design were purchased to compete the idea of testing the ultrasonic sensor functionality of detecting objects that come in close proximity to it and sending a signal after a foreign object approaches the sensor under certain threshold and check the functionality of the microcontroller being able to handle such request. The following figure shows the mentioned components.

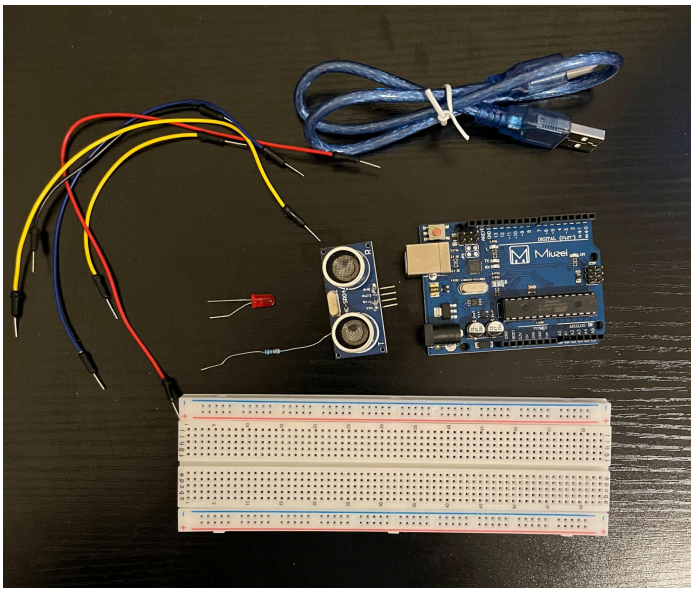


Figure 5.16 components used to set up the basic breadboard prototype

The next step was then assembling the components into a system attached together using the breadboard to represent what is going to be the sensor system in the land delivery bot and will collect the ultrasonic sensor's data, and then this data will be sent across the bot system to make decisions kind of like humans when collects data from their eyes or ears and send a signal to the brain to decide what kind of decision will be taken.

In this case as it is only a basic prototype testing of the sensor data collection, the arduino uno board is loaded with a program that collects the distance from the ultrasonic sensor to a foreign object, if the object approaches the sensor and gets into the proximity of 30 cm or less to it, a signal is sent to the LED connected to the system and turn on to show that an alert signal will be sent if the bot approached an obstacle in a certain distance determined that can be adjusted accordingly. The following figure shows the assembled components in the breadboard ready for the testing environment.

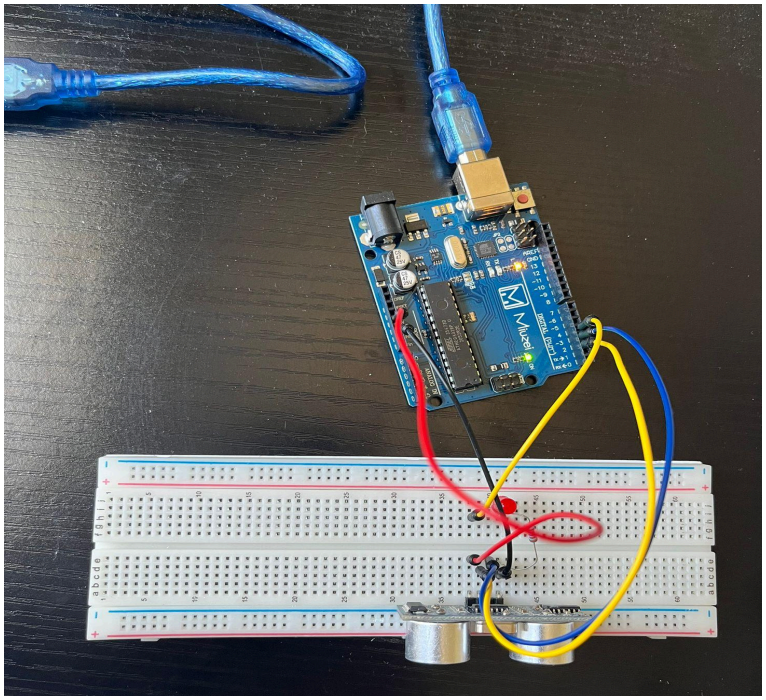


Figure 5.17 Sensor system with LED after merging the components

The code was edited and compiled into the arduino IDE and then loaded into the arduino uno board to load the functionality so the microcontroller is able to understand the task given of collecting the distance data from the sensor and displaying it into the terminal of the serial monitor, only one Arduino uno should be needed to load the sensor program into the five microcontroller units for the five sensors if this proves to be too difficult the arduino uno development kit or similar will be purchased for every sensor, but this would make the project a little more expensive as the the development board is more expensive than just the microcontroller but it will still stay in the budget. The Atmega328p did its job perfectly without the need of the development board besides bootloading and loading code.

The loaded program made use of the microcontroller attached to the ultrasonic sensor to send a signal that alerts the system of an obstacle in this case a LED simulates and symbolizes the arrival of the signal that an object is closer than the established threshold which for prototyping purposes it was set to 30 cm, and when the LED turns on it means that the signal of an incoming object was deployed. When the whole system is integrated with the rest of the bot components this signal was changed to instead of turning on LED when it is sent to stop the bot from going any further and hitting the obstacle by stopping the motors.

The initial tests performed were a success considering it was very early on the development stage the breadboard assembly was pretty straight forward as only four components were really needed to test the sensor system hardware and for even though the bot was going to use five sensor systems they all had the same

principle as this one, with the difference that they will be pointing at different directions to have a broader range of detection, they will function with the proximity signal principle. All the other components were subjected to the same procedure of testing before submitting it to the PCB creation. Instead of 5 sensors it was decided to just keep one sensor as with the integration of the compass five sensors was an overkill.

The breadboard design met the requirements established on the design constraints for the hardware design of the sensors system, it has been established that the alert signal is connected to the LED as a placeholder but the goal is to integrate the sensors system with all five sensors and instead of LEDs use a connection to the motor system and the MCU. Then depending on which sensor is sending the alert proximity signal the bot will react accordingly in which direction the bot will move if there is an obstacle present if there is an object in front of the bot it stops, rotates right and then left and attempts to proceed again.

The location of the sensors was to be one at every cardinal point, so one at the front, one at the back and one at each side plus one on the front bumper pointing downward at a 330 degree angle approximately so it can detect if there is a hole or a gap on the road. This downward facing ultrasonic sensor will work a little different as instead of detecting if an object comes into a proximity threshold send a signal this one will work in the opposite way, so it will send the alert signal to the system if the distance to the object(in this case the floor) exceeds a threshold of certain distance, so if there is no road in front of the vehicle it will send the stop signal. This proved to be a stretch goal for the future but for purposes of our design only one sensor in front of the bot was applied. The following table contains the components that were used in the basic breadboard developing prototype.

Part Number	Type	Specification
HC-SR04	Sensor	Ultrasonic
CFR-25JB-52-220R	Resistor	220 ohms
Atmega 328p	Microcontroller	Arduino Uno development kit
C5SMF-RJF-CU14QBB1	LED	Red 1.7-2.0V

Table 5.5 Components used for sensor system breadboard

5.5 GPS

The GPS we ended up choosing was the NEO-6M Module for Arduino. The reasoning for this lies primarily in the compatibility of the GPS module with the rest of the project. The three main options that were considered all had aspects and specifications that were similar to each other, therefore we had what was essentially free choice on which option we wanted as any differences between them were too minor to make a meaningful difference.

5.5.1 Hardware

The GPS we ended up choosing was the NEO-6M GPS Module for Arduino. This device consists of four pins that are used for connections. The first pin is the VCC pin which is used to supply power to the module. The second and third pins are the TX and RX pins which are the receiver and transmitter pins respectively. They use serial communication to send and receive data to and from the main Arduino module. The fourth and final pin is the GND pin which is used for a ground connection. This low amount of pins allows for a simple and easy setup and connection to the main Arduino module.

The NEO-6M also comes with a built-in 3.3V regulator so we do not need to worry about configuring an external regulator.

The NEO-6M communicates with the GPS satellites by using a small patch connected to an antenna which is configured to around 161 dB. This antenna connects easily to a section on the module which further emphasizes the ease of use of the module. It is possible to get a more accurate reading with a higher quality, more sensitive antenna, however that will not be necessary for the scope of this project.

The GPS module also comes with a built-in battery which is used to power the battery-backed RAM section of the module. This battery-backed RAM allows the GPS module to achieve a much faster first-fix time than usual as it contains the clock data and location data of the previous reading. Storing this data allows the NEO-6M to lower its first-fix time to just around one second compared to around ten seconds without the battery-backed RAM. The RAM is obviously not permanent, but it can store data for up to two weeks without power which is plenty of time to overcome and reasons it may even power off in the first place.

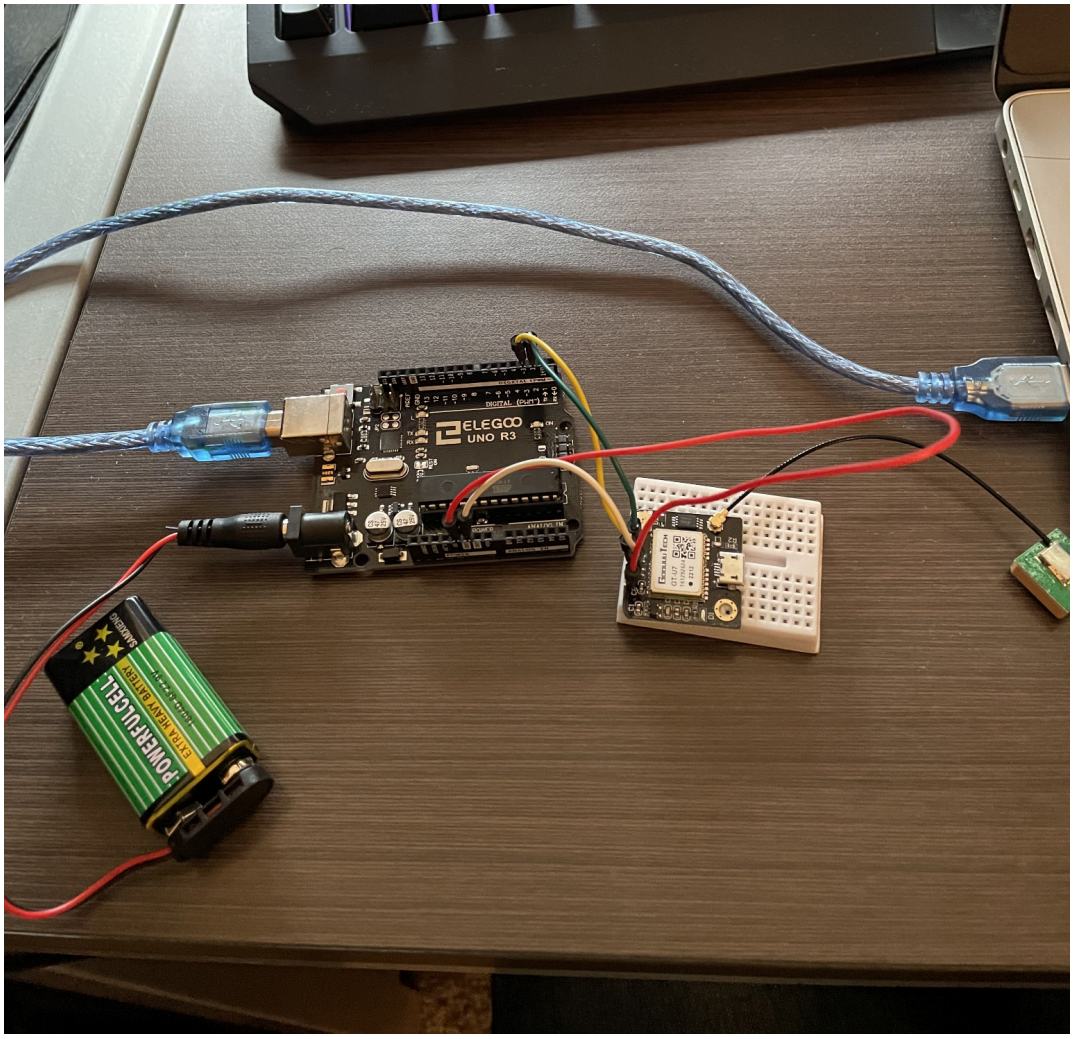


Figure 5.18 GPS system breadboard prototype

5.5.2 Communication

The NEO-6M comes with four communication protocols which are: Universal Asynchronous Receiver/Transmitter (UART), Universal Serial Bus (USB), Serial Peripheral Interface (SPI), and Inter-Integrated Circuit (I2C). This allows for a variety of potential communication methods as well as, if necessary, the combination of multiple communication methods working together. The pin configuration for the GPS device can be seen in the table below. While this is not relevant in terms of hardware connections as the NEO-6M comes pre-soldered onto an integrated circuit board where the connections are simplified to the four main ones mentioned above, these pins are relevant for debugging purposes as well as creating specific configurations for the chosen communication method. For example, if we want to make use of the USB communication method, we will need to consider pin five and pin six when attempting to debug.

The SPI configuration settings consist of the standard settings of 9600 baud rate, 8 bits with no parity bit, and one stop bit. The USB settings consist of the regular, standardized settings. Since both of these communication methods follow the regular standard, there will be no trouble configuring them.

Pin Number	Name	I/O	Description
1	Reserved	I	Reserved
2	SS_N	I	SPI Slave Select
3	TIMEPULSE	O	Timepulse
4	EXTINT0	I	External Interrupt Pin
5	USB_DM	I/O	USB Data
6	USB_DP	I/O	USB Data
7	VDDUSB	I	USB Supply
8	Reserved		Reserved
9	VCC_RF	O	Output Voltage RF
10	GND	I	Ground
11	RF_IN	I	GPS Signal
12	GND	I	Ground
13	GND	I	Ground
14	MOSI/CFG_ COM0	O/I	SPI MOSI Configuration Pin
15	MISO/CFG_ COM1	I	SPI MISO Configuration Pin
16	CFG_GPS0/ SCK	I	Power Mode Configuration Pin/SPI Clock
17	Reserved	I	Reserved
18	SDA2	I/O	DDC Data
19	SCL2	I/O	DDC Clock

20	TxD1	O	Serial Port 1
21	RxD1	I	Serial Port 1
22	V_BCKP	I	Backup Voltage Supply
23	VCC	I	Supply Voltage
24	GND	I	Ground

Table 5.6 MCU Pin Configuration

5.5.3 Special Features

The NEO-6M also comes with a built-in 3.3V regulator so we do not need to worry about configuring an external regulator.

The NEO-6M communicates with the GPS satellites by using a small patch connected to an antenna which is configured to around 161 dB. This antenna connects easily to a section on the module which further emphasizes the ease of use of the module. It is possible to get a more accurate reading with a higher quality, more sensitive antenna, however that will not be necessary for the scope of this project.

The GPS module also comes with a built-in battery which is used to power the battery-backed RAM section of the module. This battery-backed RAM allows the GPS module to achieve a much faster first-fix time than usual as it contains the clock data and location data of the previous reading. Storing this data allows the NEO-6M to lower its first-fix time to just around one second compared to around ten seconds without the battery-backed RAM. The RAM is obviously not permanent, but it can store data for up to two weeks without power which is plenty of time to overcome and reasons it may even power off in the first place.

5.5.4 Software

The software side of configuring the NEO-6M module is just as simple as the module configuration and makes use of the C++ programming language. One would simply install an Arduino Integrated Development Environment (IDE) from the main Arduino website which is already configured to work with any Arduino brand device. We will also make use of the TinyGPSPlus library to help with configuring the GPS and parsing the data.

5.5.4.1 Software processes

The processes for configuring the module, reading the data being sent in, and evaluating the respective data are all relatively simple as well. The general overview of the process are as such:

1. Assign RX and TX pins
2. Declare and initiate a serial connection
3. Parse the data that is received
4. Send the data to the main controller
5. Make a decision on what to do based on the data
6. Send the action to the respective parts

Assigning the RX and TX pins is a simple variable assignment so there is not too much to consider there.

The main library being used to help configure the GPS is the TinyGPSPlus library which contains many useful methods and objects that can help make working with the GPS much easier. One of those objects is named SoftwareSerial and it is an object that represents the serial connection that we need. We simply create the object that represents the serial communication and then use the Serial.begin() method that is a part of the TinyGPSPlus library to start the communication with the satellites. We initiate the serial communication at a baud rate of 9600 and set up a loop that will constantly read in and parse the data while the serial communication is active. We can also set up a condition statement that will catch a possible lost transmission by checking if five seconds have passed with the transmission pin not receiving any data.

Once the data has been received, it must then be parsed. When the data is read by a receiving pin sent via the transmission pin, it is sent in a structure known as a NMEA sentence. The National Marine Electronics Associate created a standardized format for GPS data to be transmitted in that contains most of, if not all, of the useful and relevant data. There are two main formats that the data will most likely be sent in. The first is the $GPRMC format and the second is the $GPGGA format. The $GPRMC format (which will hereby be referred to as Format 1) returns many different aspects of data. The ones that are relevant to us are time, latitude, longitude, and estimated velocity. These can simply be parsed from certain sections of the message that is returned and then sent to the main controller. Format 1 also returns aspects such as altitude, date, and track angle, however those are not relevant to our situation and will therefore be ignored. The $GPGGA format (which will hereby be referred to as Format 2) returns many of the same data aspects with a few differences. These differences consist of aspects like GPS fix count and the number of satellites being tracked. This format also returns the basic information such as latitude and longitude and the differences are not relevant to our situation and so it does not have to be considered. Format 1 is the one we used for the drone. The TinyGPSPlus library has many useful methods that we can utilize that take care of much of the parsing for us. This helps to make sure that we do not have to worry about

formatting errors when parsing and evaluating the data. The first majorly useful method is the location.lat() method which gets the current latitude of the GPS. Similarly, there is also a location.lng() method which gets the current longitude of the GPS. The second useful method is the speed.value() method which returns the current ground speed up to hundredths of a knot. We will still have to transform that into meters per second to be able to properly evaluate it, but it is still useful to get the initial reading. There are many other methods in the library that can perform similar tasks such as hdop.value() which shows the current HDOP value and age() which shows how many milliseconds have passed since the last update. We can use this in the previously mentioned condition statement that checks for a lost fix.

Once all the relevant data is received, it will then be transmitted to the main controller via the TX pin. The main controller will then use that data along with the other sensor data that it is receiving to properly evaluate its current state and make a decision about what to do next. For example, if the current location is to the left of the target location and the sensors' data indicate that there are no obstacles that would prevent it from turning left, then the main controller will have the drone do so. This type of condition statement is the main way that the main controller makes the necessary decisions as they come up. We will utilize the GPS information along with the sensor information to have the main controller follow the shortest effective path to the target location. Once a decision is made, all that is left to do is to send signals to each of the respective parts in order to perform the action.

The Integrated Development Environment (IDE) that we used to write the code for the drone is the Arduino IDE 2.1.0. The initial setup for the Arduino board is relatively simple as all we would need to do is simply search for and select the board that we are using. We then write the code for the communication with each of the sensors and with all of the external parts. After setting up the initial code for the main Arduino board, the code for the NEO-6M GPS module can be implemented. The TinyGPSPlus library will have to be installed into the IDE as it does not come with the base IDE installation, but that is as simple as searching for the library using the library search function within the IDE.

Section 6: Prototype Construction and Coding

This section is an overview of how the coding process was completed, tested, and finalized in the development of our project. This will be a generalized explanation and any detailed code blocks can be viewed in the appendices. For the purposes of this area of the report, the process of the coding will be explained initially, then any specific area of the system will be explained further. These sections will contain first, the hardware components in each section, the desired actions or behaviors, then the first testing phase, and lastly the interaction between other systems and finalizing the complete project. Some

simple pin connections may be introduced, however, detailed explanations of connections can be viewed in section 5 that explains the hardware designs of the system.

6.1 Prototype Coding

This section will detail the logical processes behind the coding of each of the components of our project system. This includes the logic for sensors, motors, locking mechanism and GPS system. These will simply be explanations and descriptions of how the code operates rather than any full code snippets (which may be viewed in the corresponding appendix section). The overall operation of our system intends to compose each aspect of the system separately, then once all code works as intended, communication will then be transferred across an I2C communication bus to allow commands to be sent from the given component to any other relevant components. This will be explained in further detail in the following section.

6.1.1 Coding Process

As stated previously, the overall process for coding of the whole project involved first coding each component separately, then once it was found that the code operated how it was expected, the code was then implemented together and communicated to the system as an entirety. The plan for code development was to proceed from the most simple component (in terms of coding difficulty) to the most complex. For this reason, the sensors were developed first, then motors, then compass and finally GPS. Once all sections were complete, communication between each component was developed to ensure that the expected behavior is still achieved.

In section 3, the team determined that the microcontroller that we are using is the Arduino Uno with the ATmega328p processor. For this reason, we used the supplied Arduino IDE. All coding logic will happen utilizing this microcontroller and tested with simple indications that the program is working as intended.

6.1.2 Sensors

The development of the sensor logic came first as it was the easiest to determine the effectiveness as well as change the logic if any other behavior was desired. The initial output from the sensors is to indicate that an object or surface is within 30 cm of the sensor by activating an LED on the Arduino microcontroller. The effect of this is two-fold: it ensures that the sensors are working properly and the only change in the code to implement it into the whole system is to change the communication process from activating an LED to communicating to the motors that an object is close. Only one sensor was used in testing at first, then all of the sensors were attached at once and the code was altered if the code was not producing the expected output.

To create this behavior from the sensor, code from author MIUZEI was altered. The code involved first establishing the pins that are utilized for the sensor which are pins 2, 3, and 4, and are labeled as the trigger, echo, and LED pin respectively. The pins are then set to the initial state before entering a loop that is running continuously until the robot has finished delivery. This loop involves sending a high signal to each pin in the following order: trigger pin first to send an ultrasonic wave, echo pin next to read the resulting distance, and LED if it is determined the distance is within a certain threshold. In addition to the LED, the distance is printed to the IDE console at all times in order to debug if any anomalies occur (for example, not triggering the LED if a distance within the threshold is read). This is the initial code design for the sensors of the system.

Once all the sensors were tested and the communication between components is necessary, the code will change in the following areas. First, the pins were changed in order to incorporate the motor driver which facilitates the communication between the sensors and the motors. Secondly, the sensor will determine a different behavior for the motors. For example, if the sensor detects an object, the delivery bot should demonstrate an action of turning left. Therefore, instead of activating an LED, the sensor will send high and low signals to the desired pins of the motor driver to activate the corresponding motors (Note: there are two motor drivers for each side as one motor driver can only control two motors at once). The details of each motor direction will be explained in the following section where the code logic for the motors are located. After the desired output was reached with the sensors, the motor logic was reviewed to alter its original configuration. Even though the sensor system was successfully implemented for the use of multiple sensors, it was dialed back to using just one sensor as the logic for the extra ones was not really necessary for the purposes of this project.

6.1.3 Motors

This section details the coding logic behind the motor aspect of our project. To begin, the motors were tested initially without any communication to the other components. This is to ensure that each motor is working properly and is connected to a power source. First in one direction, then the polarity was reversed to ensure that both directions are functional. This initial testing provides a baseline so that when an unintended action happens, we know it is through the software side and does not involve malfunction from the hardware.

When the operation of the motors was confirmed, the coding process began. First, functions were defined for each behavior that is desired for the motors. As explained in the previous section, when sensors detect an object is within a certain distance, it will send a signal to each side of the motor drivers that will determine the action. At this point in the project, at least four functions were necessary: *moveForward(), stopMotors(), turnLeft(),* and *turnRight().*

This was done through sending a reverse signal to the motor driver of the front left motor, low signal to the back left and front right, and high signal to the back right motor. By reversing the front left and activating the back right, the output behavior resulted in a counterclockwise rotation. When this behavior works properly, the signals are changed for each encounter (object in front, right, left, back, etc.). The detection of objects by the sensors were sent to the microcontroller responsible for the motor driver and the functions activated the corresponding pin to attain the desired action.

After these actions were confirmed as operating as expected, the actual testing on the delivery bot began. This involved traversing a random environment and ensuring that the code is executing as it should. Modifications to the code were implemented after this step and retested as this was the most accurate method in determining the drivability of the bot. With the motors and sensor detection being the largest aspects of the operation of the bot, these sections are crucial in coding.

6.1.4 Locking Mechanism

The locking mechanism as a whole was scratched and replaced with different features that ensured a better functionality for the bot and the complexity of the implementation and cost of materials for this feature.

Another aspect of coding that may be necessary, but is more of a stretch goal for our project, is to implement a locking mechanism to ensure that a user's package or items are protected during transport and will not fall out of the carrier or be stolen. To do this, a simple servo will be sufficient with a small keypad to allow users to enter a code and open the carrier box. The servo and keypad will be connected to the Arduino microcontroller and a simple keymap will be developed for the numbers on the keypad in the form of a 2D array. Then, as the keypad is used, the key presses are checked with the password. Once the end of the password is reached without any errors, the servo will be moved and the lock will be released. In addition to the password check, an implementation can be made where a separate symbol, such as a pound sign or asterisk can be utilized in order to lock the system. This would allow users or companies transporting the package to the customer to set the package and lock it, then by giving the password to the customer, will only be unlocked once the destination is reached.

Another possible implementation of the locking mechanism can be instead based on geolocation rather than a keypad. Since our system will already be using a GPS system, the information can be relayed to the microcontroller and once a destination is reached, the servo motor activates and automatically unlocks without any other interaction. This may simplify the process in that only the location of the delivery bot needs to be checked rather than a keypad mechanism. The only downside to this implementation is that there is no confirmation that the correct person received the package. For example, if the location given to the delivery bot is a large apartment building and the

mechanism unlocks when it arrives, the potential for anyone opening the carrier is fairly high and may reduce the effectiveness of the security measures in the first place.

Regardless of the implementation used, the coding process will be the same for both. Once the components are connected, the locking mechanism will be tested to ensure the hardware has no issues. Then once the behavior operates how it is expected, a location will be determined. As the bot travels to the location, the lock will be tested continuously to ensure that the mechanism is working properly. Then, in the case of the keypad, the password will be entered and the mechanism will be tested both by entering the correct and incorrect password. For the GPS method, the mechanism will only be tested to make sure it has been activated at the correct destination.

6.1.5 Coding Overview

This section summarizes the previously explained areas of coding including the processes for the sensors, motors and locking system. Below is a generalized diagram of how the delivery bot is planned to operate.

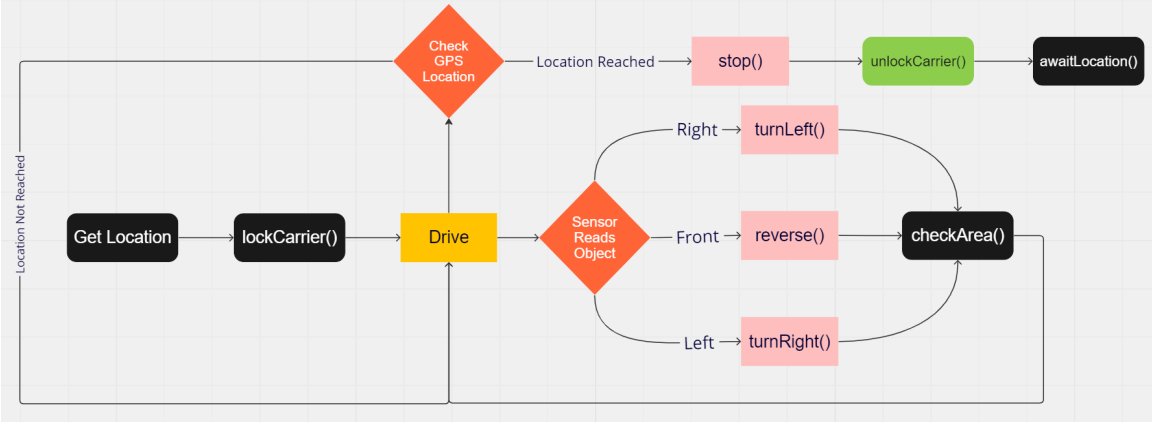


Fig 6.1: Code Block Diagram

As can be seen by the diagram, the system is relatively simple. Most of the focus is directed towards the sensors and motors at first, with no location. Instead, the delivery bot is free roaming to ensure that the sensors are able to process any objects and send the correct commands to the motor driver and that the motor driver transfers them to the corresponding motors. Finally, once the delivery bot reaches its location stops.

The coding process for each component is essentially the same process where the component is coded and tested alone to ensure that the behaviors are achievable with the hardware. When the component is tested on its own, the system is slowly connected one by one to be able to test the communication between each aspect of the system. Once all of the components are working

together properly the entire system was tested together and evaluated to fix any other issues that arise.

Some other areas of the coding involved stretch goals such as speed monitoring, additional security measures, and other aspects like these. These areas were to be only be worked on if the entire system works properly as stated above and no bugs were contained in the code.

Section 7: Prototype Testing Plan

Testing is an essential aspect on every type of project to ensure that the functionality of the project is successful and help push the development forward and find any type of issues in a timely manner to make the process of fixing it faster and easier. Finding an issue with the project early was very helpful as it let the team solve it with more time and before it piled up with other issues and became a bigger problem. Once the prototypes were done testing started taking place and the test results were evaluated, to make any type of adjustments needed to get the required results.

The first step for the testing was testing the components and the circuits on breadboards before printing anything to make sure that they worked according to the plan and adjusting any changes that needed to be done.

7.1 Hardware Test Environment

The hardware test environment for the project consisted of these elements: testing the hardware of the bot itself, controlled testing environment, control center, sensors detection, endurance. These 5 elements together constituted the hardware test environment in which the project was evaluated and further details of each testing element in this phase are specified below.

Bot hardware was tested along the other 4 elements of this testing phase, the main components tested where the wheels with the motor, the cpu, the sensors, the power source and the MCU, basically this testing element is to make sure that all the hardware components are communicating with each other correctly and the bot is responsive and move autonomously.

The controlled testing environment was a very important testing element in this phase, it was designed to simulate real world conditions and it was divided into two phases: indoor and outdoor testing environments. The indoor testing environment consisted of a flat surface with some obstacles like walls, doors and ramps, this was to simulate different types of situations the bot may encounter, the indoor environment was well lit. the outdoor testing environment consisted of less obstacles to detect like walls and such but the terrain was rougher and less

even to test durability and control on harsher conditions, the lighting will vary as it relied on natural lighting and the outdoor elements, small rocks and other types of obstacles were encountered in this testing environment to test the bot against them.

The control center testing consisted of the control of the bot to at least get it started and stop it, the team also implemented a way to put the destination on the bot, the control center was tested by turning the bot on and off multiple times to test, and testing. Another factor that was tested for the control center is while the bot is moving the movements and sensor functionality can be monitored.

The sensor detection was tested on the indoor and outdoor area with collision detection to make sure the bot can detect and avoid possible obstacles that may be encountered, the bot reaction was evaluated and analyzed to make sure the bot is taking the correct path after detecting the obstacle with the sensors and see if there is any flaw and rate of success/accuracy detecting.

The endurance is the last testing element on this phase that was evaluated specially on the outdoor environment, endurance testing consisted on use on rough terrain making sure the bot can handle the vibrations of the terrain, if the bot were to collide with something or an object might hit it by accident will it whistand the hit, will the cargo be affected and will the bot be able to reach its destination or return to homebase.

7.2 Hardware Specific Testing

Hardware specific testing describes the testing that took place specifically for the individual hardware components of the project and make sure that they work individually, to maximize the chances of the whole project working as a unit correctly and each component meets the requirements established on the design phase. The hardware specific testing consisted of 6 elements: the mobility and GPS test, obstacle detection and sensor accuracy, package handling test, battery life test, durability of cargo test and motor limitations test.

The mobility and GPS test consisted of testing the bot’s ability to go where it is intended and reach the destination of the delivery. To perform this testing the bot had to complete different routes, the routes were composed of different types, short(10 meters), medium(100 meters), long(200+ meters). The bot was tested progressively from short routes but then there was no need to escalate up to the long routes. The testing of the accuracy of the navigation system in different circumstances and distances were also included, the route’s levels and distances helped determine the project’s distance limitations after the design specification has been achieved.

The obstacle detection and sensor accuracy were tested in this element to verify the detection of distances by the sensors and then preceding to intentionally put the bot on routes crowded with objects of all sizes to test the object detection. In the first phase of testing, distances were measured using measuring tape and compared to the sensor's distances and detection threshold to verify its accuracy. In the second phase obstacles were placed on the way of the bot starting with big obstacles, like closed doors, then the obstacles did progressively become smaller like a bench, to a big rock, to a shoe, etc. to find out the smallest object that will be evaded and if it was not evaded make sure is not a problem for the traversing of the terrain.

Package handling test consisted of testing the limitations and capabilities of the bot being able to accept the package that is to be delivered, with ease, carry it with no issues around the established route, different sizes and weights of packages will be placed inside the cargo to make sure the desired design specification is met as the maximum size and weight and less than it should be accepted and carried with no issues as well. Furthermore the extraction of the package by the receiving party will be tested as well to make sure it is an easy and uneventful experience with packages of different sizes and weights.

Another element to test was the battery life which evaluated if the power source system and battery can provide enough power for the bot to continue to operate for the expected time. The way this testing proceeded is very simple as it was done just by running the bot continuously for the maximum amount of time until it runs out of power, this period of time was measured with a stopwatch to make sure it met or surpassed the design specifications established earlier.

The cargo durability was tested to verify the bot's hardware's component will withstand wear of use or if accidents happen how safe is the cargo from being damaged, this was very important as the cargo's safety is top priority. This test was completed in 3 phases of physical stress to simulate possible conditions. The first phase was shock-like wear and tear like checking the bot's durability from being dropped from different heights, being tilted over, crashing with obstacles. The second phase was centered more around weather related wear and tear, like water that could happen with rain, strong air that could happen with windy conditions, heat from the sunny outdoors. The third phase was centered more around security and human interference, so it simulated if someone maybe tries to mess with it by kicking it or trying to steal the package.

And the last hardware specific test element to be examined is the motor limitations, this is a very straightforward test that explores the bot's ability to speed up and slow down using the motor. The motor capabilities were tested by making the bot move at different speeds and comparing which one was the more appropriate for power consumption, cargo safety, reliability and wear and stress placed on the motor.

7.3 Software Test Environment

The software test environment was completed in testing like any other software it did include unit testing, integration testing and system testing. This was done to make sure that the every unit module is working successfully after being designed, the the integration of all the module units its necessary to check if the units work together, the module units were worked on by different people on the group there were issues integrating so integration test was key, then system testing was done to make sure that the software works well as a system, how its supposed to, and the results were documented to solve any issues.

The first phase of the software test environment was to test the software's modules on the programming environment to make sure it compiles and runs smoothly on that environment, testing the modules using unit testing, then testing the interaction using integration testing. The bot was loaded with the software and tested to see it performed the assigned tasks.

The bot was placed and assigned to deliver a package, in this test the GPS software was tested to see if the bot will go in the correct direction, then when the bot started moving, the MCUs software was tested if the bot's motors activate at the appropriate speed and if the sensors were working, the bot does not collide with the obstacles.

Another test that was completed was the failure alerts software. These messages are supposed to alert if there is a hardware component like the sensors that was not working properly or if there is an interruption (the software crashes) a failure alert should be received. This was tested by trying to run the both with a component disconnected by making the bot stop moving by force.

7.4 Software Specific Testing

Software specific testing did focus on testing every single software with hardware components individually; it was done on a breadboard and/or the compiler environment to make sure that the software was running smoothly and free of bugs or crashes. The MCU components were tested by compiling and loading the program with the help of the arduino uno board development kit, different programs were compiled and if successful it was loaded into the development board and tested.

The sensors software was updated and connected to the MCU through a breadboard and the sensor integration was tested and made sure the software was collecting the data from the ultrasonic sensors and checked if the software was able to process the data; the accuracy of the processed data was tested to confirm the reliability of the software.

There was a software performance test where the software was tested with edge cases to see if it could handle multiple request at a time and make sure it will not crash if unexpected events were to occur and respond quickly if things change at a fast speed for example if something falls in front of it or something hits it, the software was able to change accordingly.

The GPS software was tested by evaluating if the software was able to map the environment and plan the route to the destination of the delivery. The battery software was tested just by making sure that the battery level shows up on the display and is accurate, this will be done by checking the battery level and comparing it to the one shown in the display.

Section 8: Administrative Content

There are two key topics we need to discuss before we begin this section. First of all, we need to talk about the budget. As this project is self-funded, each member will contribute the same amount of money. We discussed this project collectively to determine if it could be accomplished within the budget of everyone. As a second step, the team needed to set milestones in order to make sure we did not fall behind and that we could keep track of any dead ends that could be followed up with Webcourses or even testing. Our team made sure that we kept in mind all of these crucial deadlines and typical costs, so that every team member felt at ease and prepared to cooperate in accordance with the procedures that had been defined for this portion of the project.

As a team, we had to discuss the budget, one of the most important aspects of this project. We needed to determine whether the project would be financially feasible as well as whether each member of the team could contribute the necessary amount. It is imperative to take into consideration all the costs involved in a complicated and delicate matter like finances, including materials, tools, and any other potential expenses that may occur.

As part of the budgeting process, it is crucial to set milestones for the team so they can stay on track and not fall behind schedule once the budget has been established. In order to accomplish this project gradually over the course of the spring and summer semesters, it is necessary to create a plan. As a result of these milestones, the team can stay on top of any potential issues that may arise, follow up with advisors on any mistakes that have been made, and correct any problems well in advance of any other issues. To ensure our final project is completed on time and within budget, we created these milestones to help us stay organized and focused.

8.1 Milestone Discussion

Our Senior Design Project is no exception to the rule that having a well-planned and detailed timeline is essential to the success of any project, and this is no different in our project. In order to stay on track and ensure that we are making progress towards our ultimate goal, we recognize the importance of clearly defined milestones for our project.

A key part of our timeline is ensuring that all documentation is completed by the end of Senior Design I, including detailed reports, research findings, and any other relevant information that will be needed to ensure the project's success. It is our intention to continue to learn and research throughout the entire process so that we can refine our design and ensure it meets the project requirements as the project progresses.

One of the most crucial milestones is completing at least one PCB design by a specific date. This allows us to test the various components we implement over time and make any necessary adjustments. Moreover, it will provide us with the opportunity to test all the major components at the same time in order to develop a more precise and robust product.

As a result of our timeline, we are able to stay on track and meet our project goals by staying on track. We will need to develop a clear plan in order to develop a functioning and effective autonomous delivery robot in order to ensure that we continue making progress towards achieving our goal. In order for us to be able to deal with unforeseen challenges, we recognize the importance of flexibility and adaptability.

8.1.1 Semester 1 Milestones (Spring 2023)

The milestones that are to be completed in the first semester of senior design are broken up to successfully complete the project planning document that was used in senior design 2 for the assembly of the project.

Description	Duration	Due Date
Brainstorming	2 weeks	January 20
Project Selection	1 week	January 27
Divide & Conquer	1 week	February 3
Research & Documentation	7 weeks	March 17
General Table of Contents		March 17

60 Page Rough Draft	1 week	March 24
Design Research	2 weeks	April 7
Rough Draft Finalization	2 weeks	April 24
Final Paper		April 25

Table 8.1 spring milestones

8.1.2 Summer 2023 Milestones

Given our team's particular situation of taking Senior Design 1 in Spring and Senior Design 2 in Fall, we have an opportunity to plan, order, and construct aspects of our project to get a good start and save time in the fall. The majority of this time will include the following: Ordering components, testing each individually, and small coding areas. The current goal for this stretch of months is to have all of the components in before the start of the fall semester and to have at least tested some key components of the system such as the sensors and motors. If possible, simple coding will be completed as well (for example, code for sensors to read objects or motors to turn in certain directions).

As our team plans to work over the summer additionally, we will keep a planned schedule of meeting at least once a month. This will allow for about four meetings between the two phases of classes and will hopefully assist the progress of the project and set a good precedent for testing and completion in fall. Contact was kept all throughout the summer in order to ensure the parts are being ordered and any simple tests have been completed. Below is a table describing the milestones for the time between classes.

Description	Duration	Date
Part Ordering	8 weeks	June 27
Simple Testing	4 weeks	July 25
Initial Coding	4 weeks	August 22

Table 8.2: summer milestones

8.1.3 Semester 2 Milestones (Fall 2023)

The milestones for senior design 2 are designed for the second phase of the project which include the assembly, prototyping and testing of the project to make sure it is working appropriately and the final presentation of the completed product.

Description	Duration	Date
Prototyping	3 weeks	September 11
Testing	1 week	September 18
Redesign	2 weeks	October 2
Finalize Prototype	2 weeks	October 16
Final Presentation		November 30

Table 8.3 Fall milestones

8.2 Budget and Finance Discussion

The budget and financing aspects of this project are important to discuss for many reasons. One of those reasons is to ensure that the team is not overspending on the given budget. Additionally, once the project is complete, or at least the components are ordered, it is important to track the prices of each component as the prices could fluctuate and be more or less expensive than what we had originally planned.

8.2.1 Expected Spending

Our current budget for the project is set at $750. We anticipate that our actual expenditure will be much higher, as we are facing supply chain issues and increasing product prices. In addition to this, prototyping and redesigning will likely incur an additional cost. Taken together, these factors mean that our original estimate was not accurate in terms of how much money we actually need to spend on the project.

Component	Price
Cables/Wires	$0
Sensors	$60
PCB	$80-100
Motor	$20

Wheels	$15
GPS System	$125
PSU	$10
Chassis	$20
LCD Display	$25

Table 8.4: Budget

8.2.2 Total Spending

The purpose of this section is to the amount of budget we have left after purchasing components. The previous section explained the planned spending budget, however, as the team continued to research, there were components that were either more expensive or cheaper than what was expected. Therefore this section is necessary as it helps to track spending and analyze how accurate our expectations are.

Part	Price	Price Difference
GPS Module GPS NEO-6M	$70.99	-$54
Hiwonder 4WD Chassis Kit	$28.99	+$8.99
Motors	Included with chassis	-$15
Wheels	Included with chassis	-$20
Atmega328	$60	$0
Ultrasonic Sensors (hc-sr04)	$93.75	+33.75
PCB	$243.97	+143.97
PCB Components	$340.29	+340.29

Table 8.5: current spending

The above table describes the current components purchased along with the price difference between the expected and actual expenses. The parts that were

cheaper are highlighted in green and the parts that were more expensive than initially thought are highlighted in red. As the table shows, our actual expenses are turning out to be much cheaper than we initially thought, likely due to some components already being purchased before this class, or simply being much cheaper after researching more deeply into the hardware details.

Section 9: References

[1]8 Robots Racing to Win the Delivery Wars Of 2019

[2] PCB Art- The Ultimate Guide to How PCBs Are Better

[3] Your PCB substrate: a guide to materials | Essentra Components US

[4] The History of Printed Circuit Boards

[5] Things You Must Know About Silkscreen Printing of PCBs

[6] Ten golden rules of PCB design

[7] Functional Partitioning in PCB Design

[8] Staying Well Grounded | Analog Devices

[9] Tips about PCB design: Part 2 - Single-card grounding for multicard systems - Embedded.com

[10] 12 PCB Thermal Management Techniques | Sierra Circuits

[11] PCB Trace-The Importance of PCB Traces In the PCBs

[12] Global Positioning System (GPS) Standard Positioning Service (SPS) Performance Standard - 5th Edition, April 2020

[13] GPS

[14] Satellite Navigation - GPS - How It Works | Federal Aviation Administration

[15] GPS, LNA, Sensitivity, Jamming, Cohabitation, TTFF

[16] What Is GPS (Global Positioning System)? Meaning, Types, Working, Examples, and Applications

[17] GPS Trackers and Power Consumption - Myth Busted! – Pro-tekt

[18] GPS System : Working, Types,Trackers & Its Applications

[19] GPS/GNSS Buying Guide - SparkFun Electronics

[20] Dilution of Precision

[21] What is PDOP? And Why it's Obsolete

[22] GPS Receiver Specifications - Inside GNSS - Global Navigation Satellite Systems Engineering, Policy, and Design

[23] The P and C/A Codes | GEOG 862: GPS and GNSS for Geospatial Professionals

[24] L5 | GEOG 862: GPS and GNSS for Geospatial Professionals

[25] L2C and CNAV | GEOG 862: GPS and GNSS for Geospatial Professionals

[26] GPS.gov: New Civil Signals

[27] NEO-6 u-blox 6 GPS Module

[28] NEO-7 - u-blox 7 GNSS modules

[29] BN-280

[30] Types of Distance Sensors and How to Select One? - Latest Open Tech From Seeed

[31] What is Ultrasonic Sensor: Working Principle & Applications – Robocraze

[32] Work principle of the ultrasonic sensor.

[33] How an ultrasonic distance sensor works | Practical Python Programming for IoT

[34] Top Sensors Used in Autonomous Delivery Robots | Arrow.com

[35] 3 Types of Autonomous Vehicle Sensors in Self-driving Cars

[36] Lithium-Ion vs Lithium Polymer Battery - Robocraze

[37] Lithium Ion vs. Lithium Polymer Batteries – Which Is Better? - RAVPower

[38] Advantages and Limitations of I2C Communication - Total Phase

[39] ISO 10218-1:2011 - Robots and robotic devices — Safety requirements for industrial robots — Part 1

[40] ISO 10218-2:2011 - Robots and robotic devices — Safety requirements for industrial robots — Part 2: Robot systems and integration

[41] ISO 9283:1998 - Manipulating industrial robots — Performance criteria and related test methods

[42] NEO-6 u-blox 6 GPS Module

[43] In-Depth: Interface ublox NEO-6M GPS Module with Arduino

[44] Guide to NEO-6M GPS Module Arduino | Random Nerd Tutorials

[45] TinyGPS++ | Arduiniana

[46] USB Standards: USB 1, USB 2, USB 3, USB 4 » Electronics Notes

[47] What is I2C? I2C Protocol and Interface Explained | Arrow.com.

[48] I2C Bus Specification

[49] UART: A Hardware Communication Protocol Understanding Universal Asynchronous Receiver/Transmitter | Analog Devices

[50] Huy, Vo & So, Seongjoon & Hur, Jaehyun. (2021). Inorganic Fillers in Composite Gel Polymer Electrolytes for High-Performance Lithium and Non-Lithium Polymer Batteries. Nanomaterials. 11. 614. 10.3390/nano11030614.

[51] Hiwonder 4WD Chassis Car Kit

[52] DFRobot 4WD Arduino Mobile Platform

[53] 4WD Chassis Smart Robot Car

[54] Multi-Chassis 4WD Robot Kit

[55] RPM Converter

[56] The Future Of Autonomous Delivery Bots

[57] Why Amazon, UPS and even Domino's is investing in drone delivery services

[58] Convoy Trucks Will Be the First Autonomous Semi-Trucks to Reach Market

[59] TT All-Metal Geared Motor

[60] TT Geared Motor

[61] Interfacing DC Motor with 8051 Microcontroller
[62] PWM: Pulse Width Modulation: What is it and how does it work?

[63] Interface L298N DC Motor Driver Module with Arduino

[64] Control DC Motor with Arduino and L293D Motor Driver IC

[65] HOW TO CONTROL DC MOTORS WITH AN ARDUINO AND AN L293D MOTOR DRIVER

[66] How Do I Choose a Battery?

[67] How to Calculate Battery Run Time

[68] Choosing Batteries for Robots

[69] What is Lithium Polymer Battery?

[70] DC Motor Control using Arduino UNO and IR Sensor

[71] Arduino Basics: Making a Locking Mechanism

[72] Flowchart Designer