Shape

Description automatically generated with low confidence

System Block Diagrams and Marketing Requirements

CPE 4800 – Senior Project Proposal  
Spring 2023  
Kennesaw State University  
Dr. Jeffrey Yiin

Team Members:

Grant Burke

Hanson Chaney

Cooper Newlin

Chris Turner

Contents

[List of Figures 2](#_Toc130758948)

[List of Tables 2](#_Toc130758949)

[1. Introduction – Product Description 3](#_Toc130758950)

[1.1 Description of Subsystems 4](#_Toc130758951)

[Phone-to-MightyMover Communications 5](#_Toc130758952)

[MightyMover Motor Control System 6](#_Toc130758953)

[MightyMover LED Control System 6](#_Toc130758954)

[Software Hub Diagram 7](#_Toc130758955)

[Battery Management System 8](#_Toc130758956)

[Marketing Requirements 9](#_Toc130758957)

# List of Figures

[Figure 1. Top-Level Block Diagram of MightyMover System 4](#_Toc130759523)

[Figure 2. Basic Flow of Phone-to-MightyMover Communication 5](#_Toc130759524)

[Figure 3. Motor Control System on MightyMover 6](#_Toc130759525)

[Figure 4. LED Control System on MightyMover 6](#_Toc130759526)

[Figure 5. LiDAR Sub System 7](#_Toc130759527)

[Figure 6. Software Hub Block Diagram 7](#_Toc130759528)

[Figure 7. Battery Management System 8](#_Toc130759529)

# List of Tables

[Table 1. List of Subsystems 4](#_Toc130759530)

[Table 2. Marketing Requirements 9](#_Toc130759531)

# Introduction – Product Description

The MightyMover, an autonomously following cart with GPS location control that eases the burden of carrying heavy objects. The user can access The MightyMover through a mobile app consisting of mode selection, battery information and location monitoring. Once the user selects a mode (location or following mode), MightyMover leaves standby and moves after checking initial conditions for emergency stop and object detection. The User can stop the mover during anytime using the app’s stop function, once the mover stops, it will then enter standby to await user input.

Under Location Mode, the mover will be following a set GPS course created by the user which will send updated coordinates via the app once the mover begins. If the mover leaves Bluetooth range of the user, a command file will run sending the mover to its home destination after a set amount of time.

Under Following Mode, the mover follows behind the user a set distance. During the Following Mode, the user can record the journey via waypoints under approval by the user through the app. This recording will be used to create paths to set locations for Location Mode. While the mover follows, multiple checks from our LiDAR and GPS inputs influence the mover’s behavior; as an example, if an object comes within a set distance to the mover, the mover will stop and wait for the object to move before setting off to catch up to the user. This LIDAR system will allow the mover to determine if it is able to fit into small spaces or doorways, if not, the user will be notified via the app.

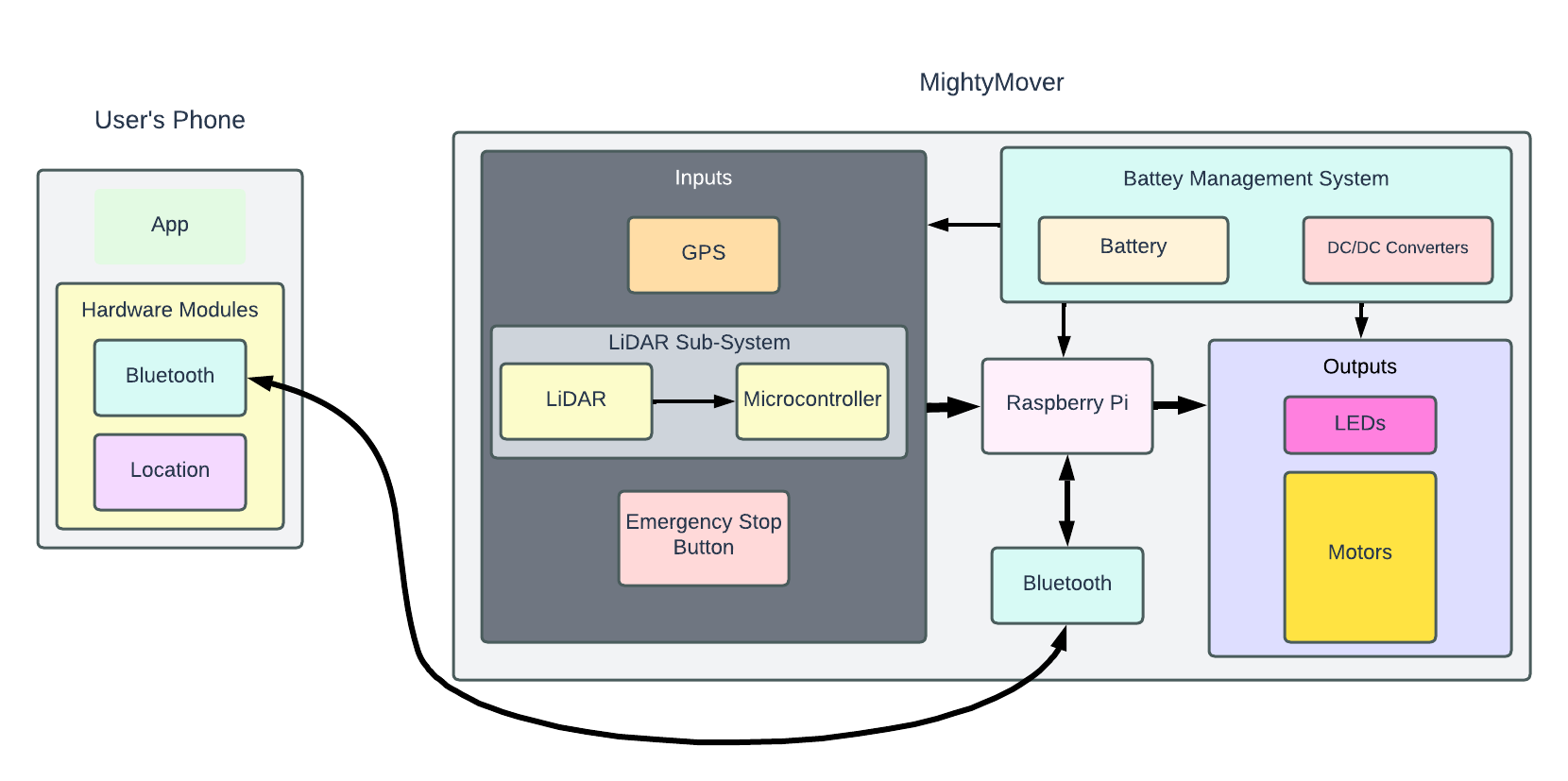


Figure 1. Top-Level Block Diagram of MightyMover System

The overall system should run as follows. The user’s phone will have information such as location and actions (from the App) being sent over Bluetooth. A Bluetooth module (separate from the RPi) will receive this data and will be forwarded to the RPi. The raspberry pi will also send location back using this Bluetooth connection. The inputs include a GPS module, an emergency stop button, and a LiDAR sub-system. These inputs will feed into our RPi and interact with codes which will control two outputs. The two outputs are LEDs and motors. The remaining system that is part of our diagram is the battery management system which will be used to power the entire system.

## Description of Subsystems

Table 1. List of Subsystems

|  |
| --- |
| **Subsystem** |
| Phone-to-MightyMover Communications |
| MightyMover Control System (Motors) |
| MightyMover Control System (LEDs) |
| LiDAR System |
| Software Hub Design |
| Battery Management System |

### Phone-to-MightyMover Communications

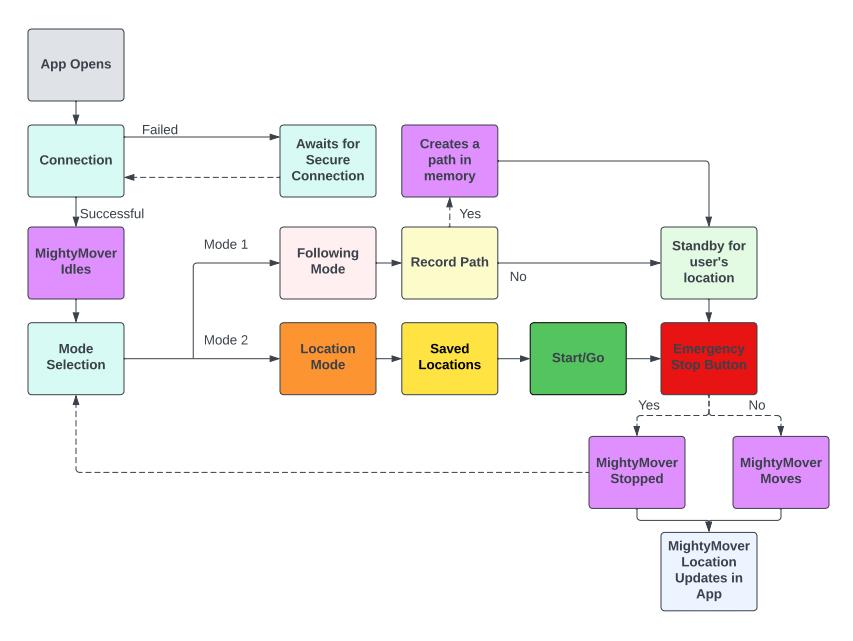


Figure 2. Basic Flow of Phone-to-MightyMover Communication

The first sub-system demonstrates the process in which the mobile app communicates to MightyMover based on user input. First, the app must make a successful Bluetooth connection, then the MightyMover idles while getting inputs from user. Depending on the user’s inputs (different colors), the MightyMover will respond (in purple). In this case, the red emergency stop button is both in the app and physical.

### MightyMover Motor Control System

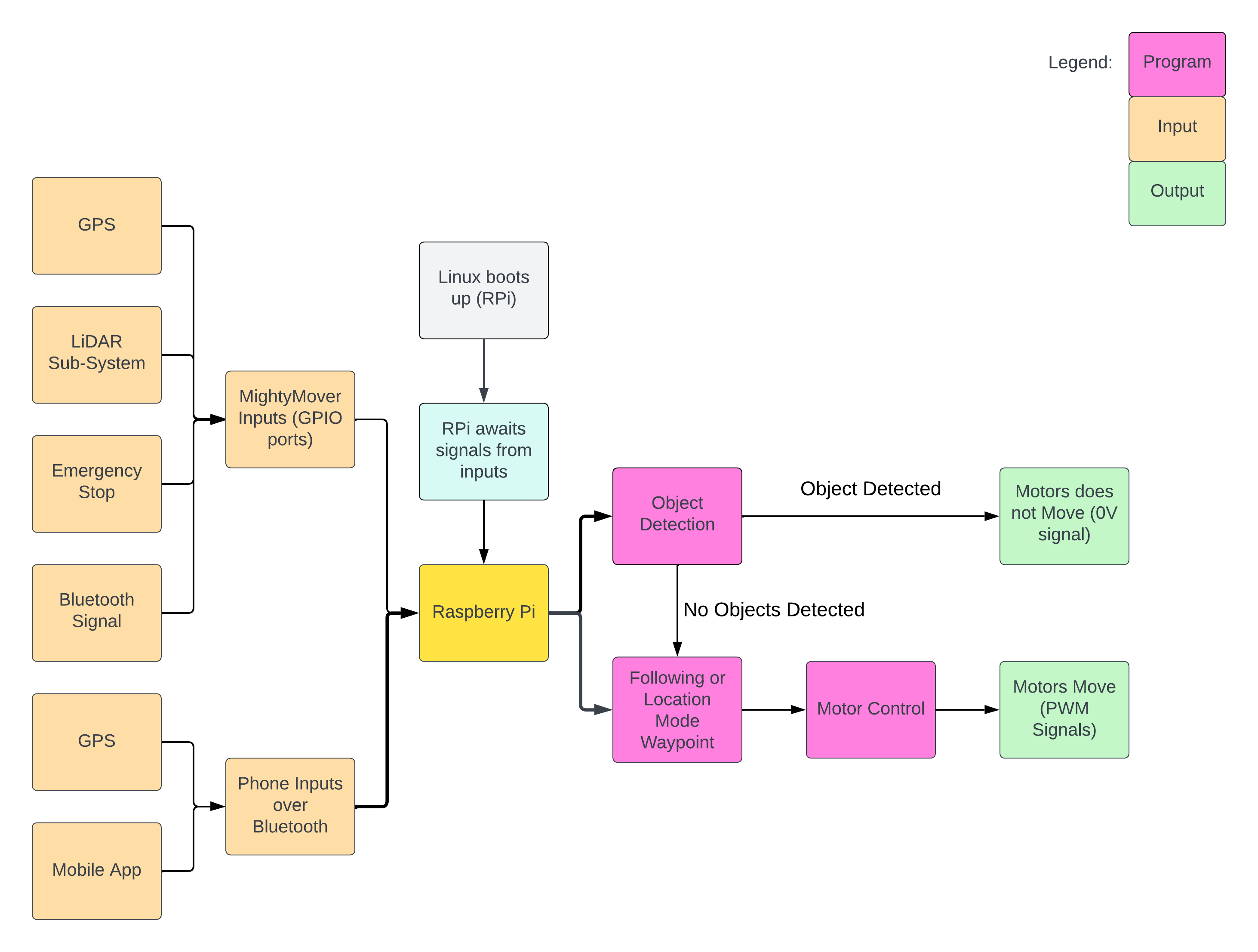


Figure 3. Motor Control System on MightyMover

The motors determine where the MightyMover moves, and this is partially the main function of the cart. The motor control system is specifically designed by the code on the raspberry pi. Before this code is executed a lot of parameters are being weighed. This includes previous codes and inputs both from MightyMover and the user’s phone. Overall, the diagram above shows what the system controlling the motors is.

### MightyMover LED Control System

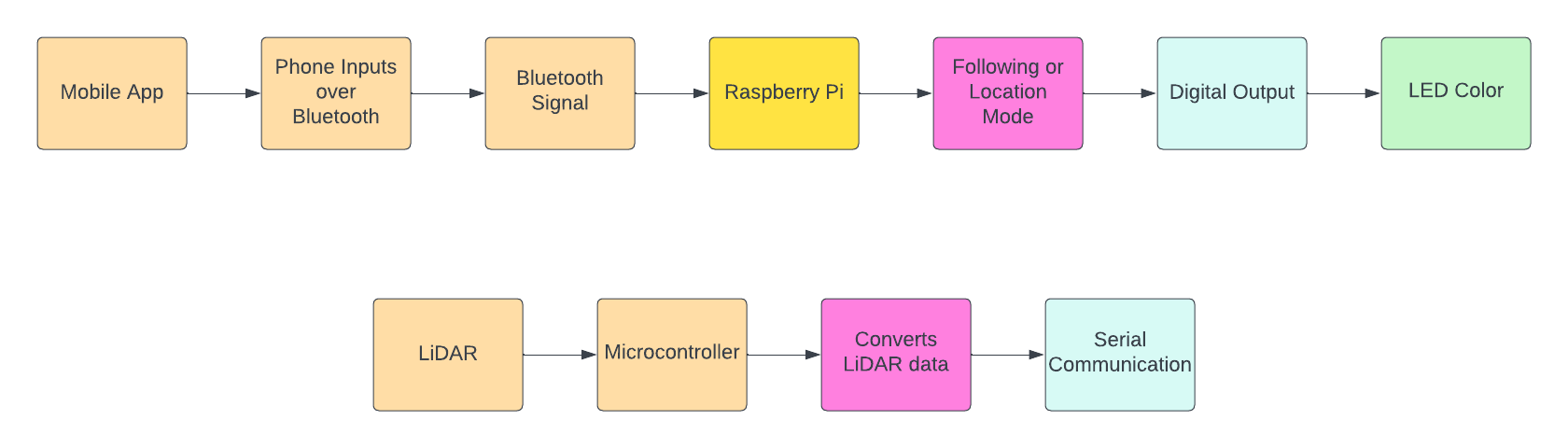


Figure 4. LED Control System on MightyMover

The LEDs show which mode the cart is currently in and if your system is on. This will help the user understand what the cart is doing and will help the team in debugging.

LiDAR System:

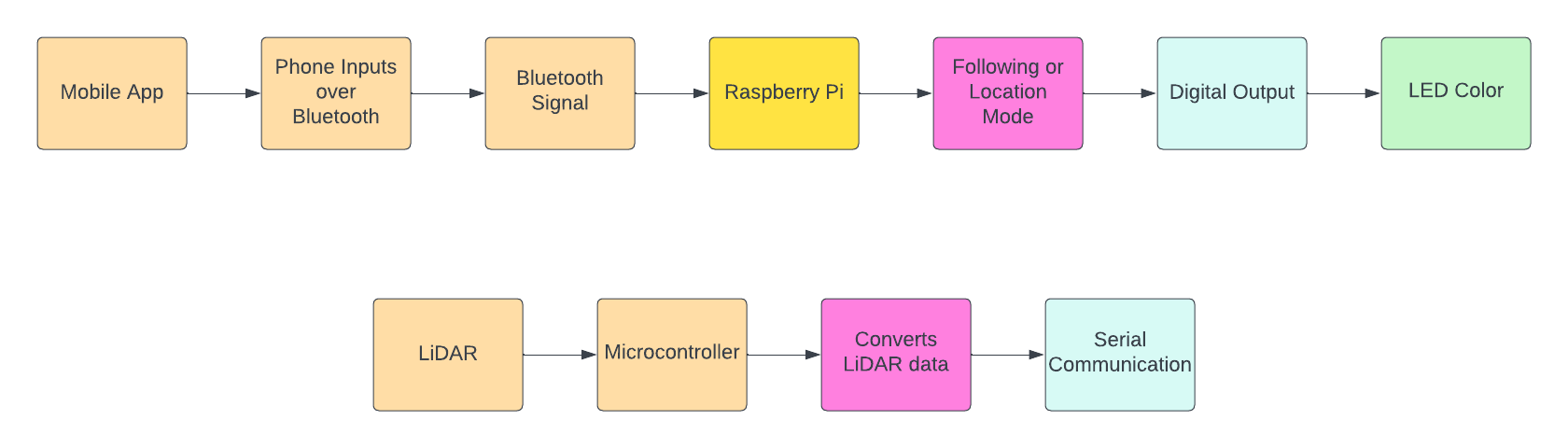


Figure 5. LiDAR Sub System

The LiDAR system will include a microcontroller to process the LiDAR data to convert into serial communication. This will help determine objects around the path of MightyMover. We are thinking that the microcontroller may also be able to run the object detection code that we initially have on the Raspberry Pi. For now, we plan to have the raspberry pi running the code for simplicity.

### Software Hub Diagram

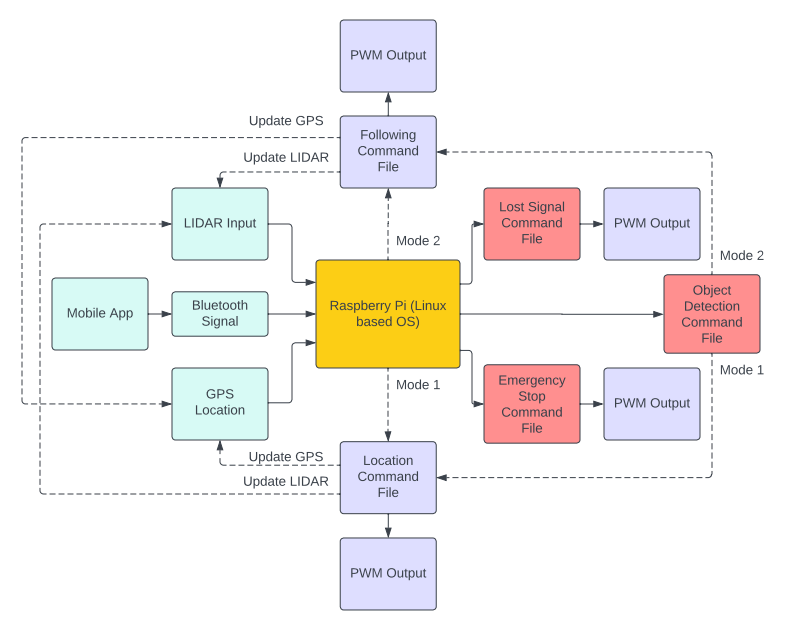


Figure 6. Software Hub Block Diagram

The Raspberry Pi will take in information from the mobile app via Bluetooth signal, GPS location, and LIDAR input from its microcontroller. These inputs will feed our multiple command files when they are executed. As the command files run, they will then continuously pull data from each sensor and produce a PWM output to the motors. The emergency stop, lost signal, and object detection command files will hold higher precedence over the location and following command file. If any condition within the red colored blocks (command files) are met, they will run thus stopping the following or location command files from running.

### Battery Management System

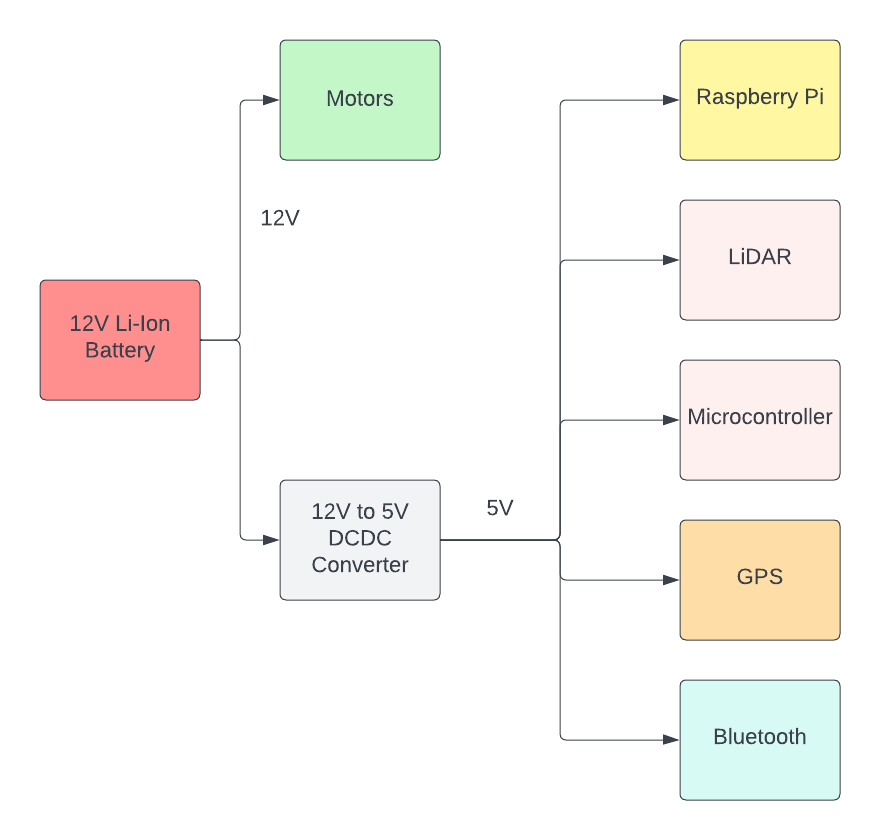


Figure 7. Battery Management System

The battery management system is simple to understand. We are still in the process of picking parts out, so most of these materials are still being determined, but this is the idea so far. If things change to a low voltage system, we may need a different DC to DC converter.

# Marketing Requirements

Table 2. Marketing Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement ID:** | **Requirement** | **Type** | **Criticality** |
| MR-FNC-01 | The cart will follow the user which will allow the user to get to their destination without having to push or pull on the cart. | Function | Must Have |
| MR-FNC-02 | Cart will be able to navigate to set waypoints instead of following the user. | Function | Must Have |
| MR-FNC-03 | The cart shall include an emergency stop system to prevent movement during loading or operation. | Function | Must Have |
| MR-FNC-04 | The cart shall possess the ability to detect objects in its way. | Function | Must Have |
| MR-FNC-05 | The cart shall have an indicator of the status of the battery. | Function | Nice to Have |
| MR-PER-01 | The cart shall have a suspension system suitable for driving on uneven terrain. | Performance | Must Have |
| MR-PER-02 | The cart shall not have problems moving through off-road terrain. | Performance | Must Have |
| MR-PER-03 | The cart shall possess a sturdy frame for longevity uses. | Performance | Must Have |
| MR-PER-04 | The steering of the cart should complement off-road uses. | Performance | Nice to Have |
| MR-PER-05 | The cart shall provide at least 4 consecutive hours of use-time. | Performance | Must Have |
| MR-FIT-01 | The system shall provide a mobile application to control the operation of the cart. | Fit and Form | Must Have |
| MR-FIT-02 | The electronics of the cart shall be housed in a 3D-printed water-resistant enclosure. | Fit and Form | Must Have |
| MR-IFC-01 | All cart controls shall be accessed remotely via Bluetooth. | Interface | Must Have |
| MR-IFC-02 | Emergency stop shall be accessed in app and physically on cart. | Interface | Nice to Have |
| MR-CON-01 | Bluetooth range shall be at least 10 meters. | Constraint | Must Have |
| MR-CON-02 | The market cost of the cart must not exceed $2000. | Constraint | Nice to Have |
| MR-CON-03 | Total weight must not exceed 100lbs. | Constraint | Must Have |