



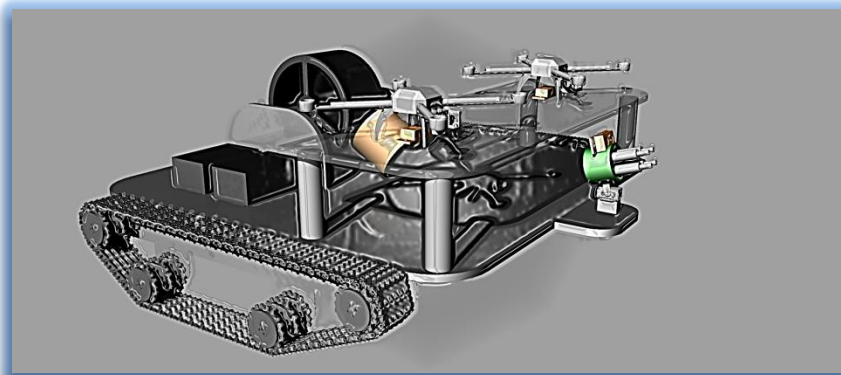
# DRONENET

## DRONENET

*Senior Design 1 Fall 2013*  
*Group 7*

*Brandon Frazer*  
*Ryan Borden*

*Joseph Howard*  
*Raymond Lueg*



Sponsored By:



**DUKE**  
**ENERGY®**



## Table of Contents

Table of Contents .....	ii
1.0 Executive Summary.....	1
2.0 Motivation and Goals.....	2
3.0 Specifications and Requirements .....	3
4.0 4WD Full Bridge Wireless Motor Controller .....	6
4.1 Goals.....	6
4.2 Platform Motor Control System .....	6
4.3 Platform Control Terminal .....	7
4.4 Software .....	8
5.0 Platform Touchscreen LCD Display .....	11
5.1 Design.....	11
6.0 Microcontroller Research.....	15
6.1 Platform Control Terminal .....	15
6.2 Platform.....	15
6.3 Quadcopter Controller.....	16
7.0 Bedini Motor .....	17
7.1 Motivation.....	17
7.2 Theory of Operation .....	17
7.3 Design Considerations .....	19
7.4 Monitoring System and Charging Batteries.....	20
7.5 Solar Panel(s) .....	22
7.6 Charging the Li-poly Quadcopter Batteries .....	23
8.0 Mobile Platform Power Supply .....	24
8.1 Goal .....	24
8.2 Hardware: Components Research .....	24
8.3 Hardware: Design .....	26
9.0 AES Security .....	28
10.0 Laser Guided Ultrasonic Autonomous Landing System .....	33
10.1 Concept .....	33

10.2	Hardware: .....	34
10.3	Software: .....	36
11.0	Wireless Camera System .....	37
11.1	Go-Pro HERO3:.....	38
11.2	Raspberry Pi with PiCam:.....	38
11.3	Foscam Pan and Tilt Network Camera .....	39
12.0	Passive Infrared Ultrasonic Target Acquisition System .....	41
12.1	Goals: .....	41
12.2	Components: Hardware.....	41
12.3	Components: Software .....	42
12.4	How it works: Hardware.....	43
12.5	How it works: Software .....	44
13.0	Quad Copter Locking System .....	47
13.1	Goal.....	47
13.2	Design: Concept Research.....	47
13.3	Components: Hardware.....	49
13.4	Design .....	51
14.0	Flight Controller Research .....	53
14.1	Goals .....	53
14.2	Hardware: Flight Controller Research .....	53
15.0	Hardware: Flight Controller Design .....	54
15.1	Gyroscope and Accelerometer .....	54
15.2	Digital Compass .....	55
15.3	Processor .....	56
15.4	On Board Programming.....	57
15.5	Data Logging .....	57
15.6	Altimeter .....	58
15.7	Turret.....	58
15.8	Landing System.....	58
15.9	Telemetry .....	59
15.10	Power Module.....	63

15.11	Speed controllers .....	65
15.12	Layout .....	65
16.0	Quadcopter Power Subsystem.....	67
16.1	Power Sources and Storage .....	67
16.2	Battery Technologies .....	68
16.3	Battery Charging .....	74
16.4	Propulsion System .....	75
16.5	Structural Subsystem .....	81
16.6	Quadcopter Remote Control System .....	86
17.0	Navigation Subsystem .....	90
18.0	Mission Planning Software .....	93
19.0	Systems Integration .....	96
19.1	Schematics .....	96
20.0	Systems Testing.....	98
20.1	Testing and Characterization Objectives .....	98
20.2	Testing Environment .....	98
20.3	Control Terminal and Testing.....	99
20.4	Mobile platform control terminal .....	99
20.5	Traditional Radio Controller .....	99
20.6	Ardupilot Software.....	99
20.7	Mobile Platform Testing .....	100
20.8	Quadcopter Testing .....	102
20.9	Low altitude flight test .....	103
20.10	System Testing .....	103
20.11	Waypoint testing .....	103
20.12	Auto home testing .....	104
20.13	Security and turret system testing.....	105
20.14	Auto landing test .....	106
20.15	Overall system test .....	106
20.16	Characterization.....	108
20.17	Range .....	108

20.18	Speed and battery life.....	109
20.19	Quadcopter load capacity .....	110
20.20	Bedini motor characterization - Duration of battery cycles.....	110
20.21	Test Summary and Recording .....	111
21.0	Milestone Discussion .....	122
22.0	Current Prototypes.....	125
23.0	Budget and Finances .....	127
23.1	Funding .....	127
24.0	Bibliography .....	129
25.0	Appendix A: Copyright Permissions and Authorizations .....	133
26.0	Appendix B: References for Figures .....	135
27.0	Appendix C: Table References .....	137
28.0	Appendix D: Equation References.....	137



## 1.0 Executive Summary

For our project, we wanted something that was both challenging and fun. There is a lot of experience with technology in this group, both design and developmental. Because of that, we had several ideas that we started tossing around before the semester started. Once we got into class, we looked in more detail at the ideas we had and down-selected to 3 potential projects. After some investigating into all of these projects, we decided on the one presented here.

Our project overview was to have some remote surveillance drones operating from a “base” drone that would be able to perform various tasks and home at the base drone. This eventually became two quadcopters and the mobile platform to support them. Because of limitations on quadcopters being able to last for more than 15 minutes in the air, we knew that a method of charging had to be provided and it couldn't be connected to a conventional charger. So we included a renewable energy design that we hoped would complement the portability of the mobile platform and provide sufficient energy for all of the vehicles.

During the brainstorming phase of this project, we had identified the challenges we knew would be part of this project. With that information, we divided our team to tackle the major challenges: charging system, quadcopter design, communications, and landing on the mobile platform. Most of our time and research has been spent in these areas. As our research was finalized on these aspects of the project, we moved on to filling in the gaps for the final implementation. During all of our research, we continued to keep in mind that we needed to have enough information to build our prototypes by the beginning of Senior Design 2. There is still a lot of work left to be done, this is an ambitious project; but, as of the writing of this summary, we are in a good position to make that happen.

Once the new semester begins, we will be spending a lot of time with testing and identifying what is working and what needs to be improved. We have anticipated that there will be a redesign involved during next semester as well. If we continue on the pace that we have established, testing should go smoothly. We are expecting only two major difficulties, the charging system and the landing system. With all testing, it is likely that there will be many more difficulties than those two, but we feel confident that we can handle anything this project gives us because of the efforts that we have put into researching the technologies this semester.

## 2.0 Motivation and Goals

The motivation for this project began before the group for senior design had formed. Two of the group members wanted to build a flying air vehicle (FAV) and were leaning towards possibly building a quadcopter or a quadcopter variant. Another group member wanted to build a ground vehicle because he had previously built a prototype remote control rover for generating interest in engineering with high school students. When the group was discussing the many different ideas for this project, several themes seemed to appear in most of the different ideas. Wireless communication, real-time navigation, a graphical user interface (GUI), obstacle detection, and sustainability were the themes that kept coming up in our discussions.

A quadcopter coupled with a mobile landing platform with navigation, a GUI, and renewable energy charging abilities appeared to be the perfect solution. The group also wanted to design something that would be fun, have potential real-world applications, and they wanted something that would be challenging. All four of the group members are electrical engineering majors and have held positions directly relating to engineering with companies such as General Electric, Lockheed Martin, Mitsubishi Power Systems, and OptiGrate. The group also has two 3D printers at their disposal personally owned by two of the group members.

The group believed that with their skills and knowledge, they could build a project that consisted of two quad-copters that could wirelessly communicate with a mobile landing platform with sustainable charging features.

The goals for the final product are to design and build two quadcopters capable of carrying a small payload and a guidance system that is accurate enough to land on a small platform. The quadcopters will be able to collect and relay telemetry and visual data to an all-terrain landing and charging ground vehicle. The quadcopters and ground vehicle will be able to navigate, negotiate landings, judge remaining flight time, and recharge using a renewable energy system. Range from the base station will be beyond visual contact. There will also be a graphical user interface (GUI) that will be a computer-based system, monitoring the quadcopters and the mobile platform. User(s) will be controlling the mobile platform and the quadcopters. The GUI will also be used to view all of the telemetry and visual data in real-time through wireless communication with the ground vehicle. This will also be where the guidance system lines up the quadcopters for landing by informing the user that they are positioned correctly for landing into the charging ports. We will also be implementing a target acquisition system and implementing security protocols in order to operate any of the systems.



### 3.0 Specifications and Requirements

This project has four main components, two quadcopters, wireless quadcopter control terminal, a mobile platform, and a wireless mobile platform controller. The specifications of our project include:

#### Quadcopter:

- Carry a payload of less than 5 lbs.
- Ability to collect data on range, altitude, weather, and temperature
- Wireless communication between the quadcopter and the mobile platform at a maximum distance of 1 mile.
- Real time location that is accurate up to 1 foot.
- Automated landing guidance on to the mobile platform that is accurate up to a few inches.
- Target Acquisition system

#### Quadcopter Control Terminal

- Control the quadcopters through wireless communications
- View real time data from the quadcopters
- Pre-plan a quadcopter flight pattern
- Communicate with the Mobile Platform Control Terminal

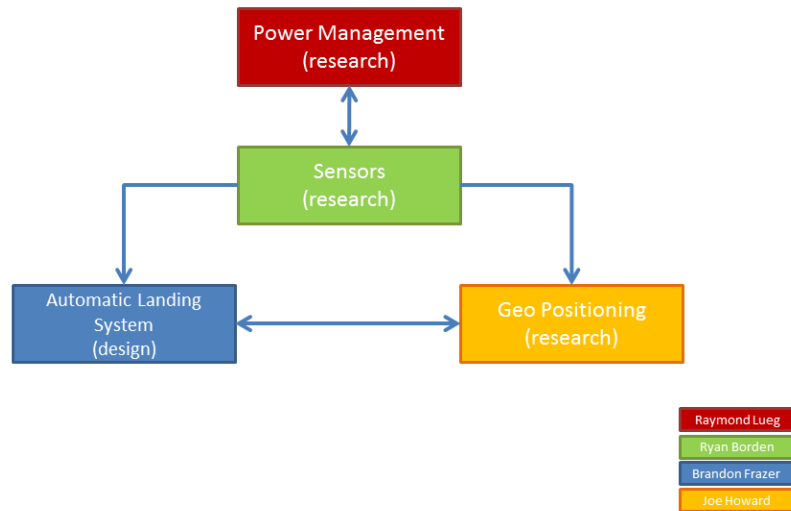
#### Mobile Platform:

- Ability to charge the quadcopter to extend mission range of quadcopter.
- Real time location between the quadcopter and the mobile base up to 1 foot of accuracy
- Have a user interface to display the information and location from the quadcopter
- Automated landing guidance system to allow the quadcopter to charge itself
- A turret defense system on the mobile base to protect the quadcopter during charging
- Ability to navigate through rough terrain

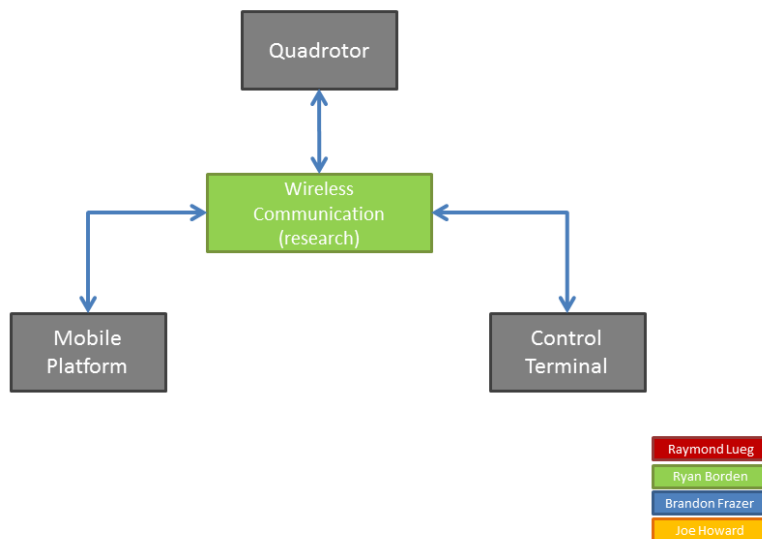
#### Mobile Platform Control Terminal

- Wirelessly Control the landing platform
- View all of the real time data streaming from the mobile platform
- Communicate with the Quadcopter control terminal
- Utilize both hardware and software based control systems
- Have a touchscreen LCD display for control and monitoring

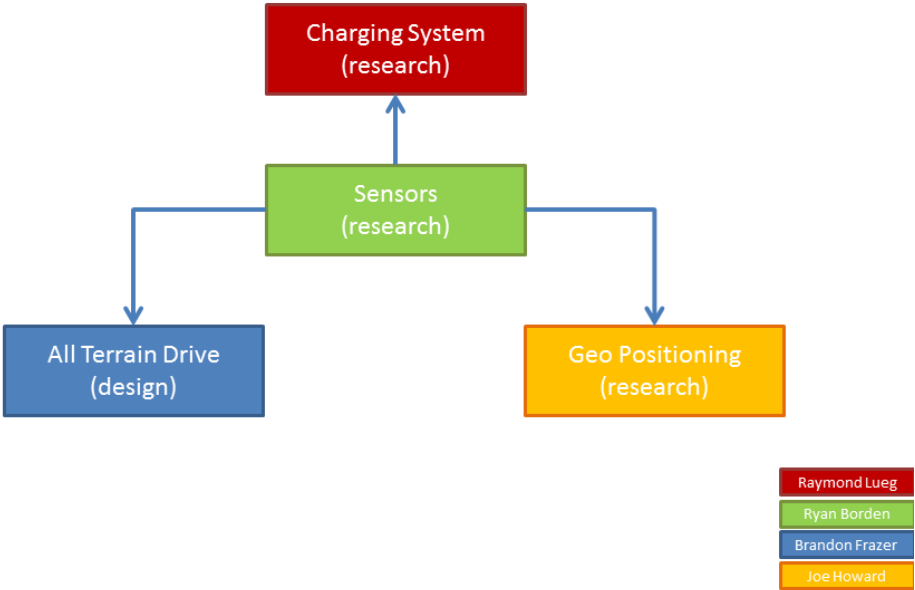
## Quadrotor Block Diagram



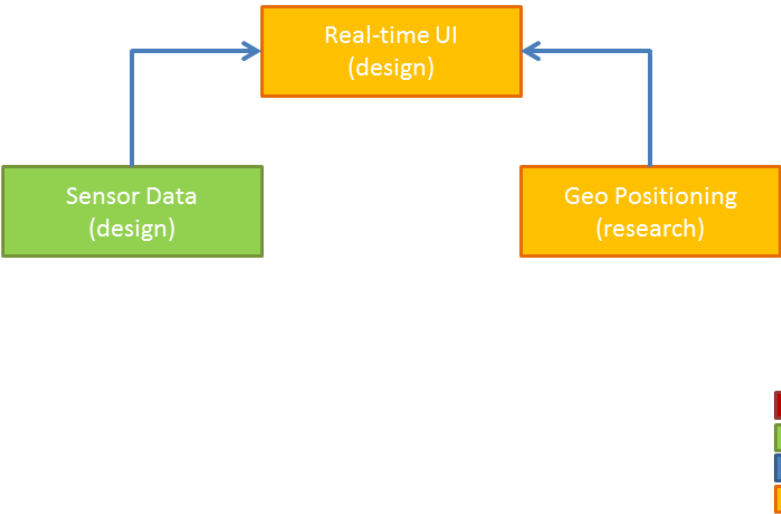
## Overall Project Block Diagram



Mobile Platform Block Diagram



Control Terminal Block Diagram



## 4.0 4WD Full Bridge Wireless Motor Controller

### 4.1 Goals

This part of the project will provide the motor controls for the platform. The platform will be running a 4-wheel drive all terrain system. We also wanted this to be controlled wirelessly through our GUI and with a hardware based controller combined with a software touch screen interface. There are two parts to this section of the project; the motor control driver system on the platform and the control terminal sending the commands wirelessly.

### 4.2 Platform Motor Control System

The platform consists of a Full-Bridge motor control driver. We were debating between a full bridge (ST) and the Half H-Bridge version (Instruments) . The reason we went with the Full-Bridge version is that it was capable of handling higher loads (4-6 amps total versus 2 amps total) and the built in current sensors. A single H-Bridge motor control driver will support only two motors. Since we will be running a four wheel drive platform, we will need two of the motor control drivers. The idea behind this system is based off the Sparkfun Motor Controller (SFE) and will taking the design and expanding it to be able to operate four motors versus two.

The platform also consists of a dedicated microcontroller being used to execute the commands sent from the control terminal. These commands are what control the motor speed using pulse-width modulation and also the direction. The direction is controlled by driving certain pins on the motor controller high or low depending on what direction is needed.

Additional parts being used are protection diodes across the outputs to prevent any unwanted EMF back feed. We will have LED's on each of the direction logic pins for visual debugging. Each of the four motors will require having a capacitor across the terminals for voltage stability and regulation during direction switching. The motor control drivers have a built in current sensor for each motor. The current sensors output an analog signal that is then combined with a resistor to create a voltage divider circuit. This will be used to monitor the motor current for protection and possible speed determination. Figure 1 below displays the platform schematic layout with the microcontroller left out.

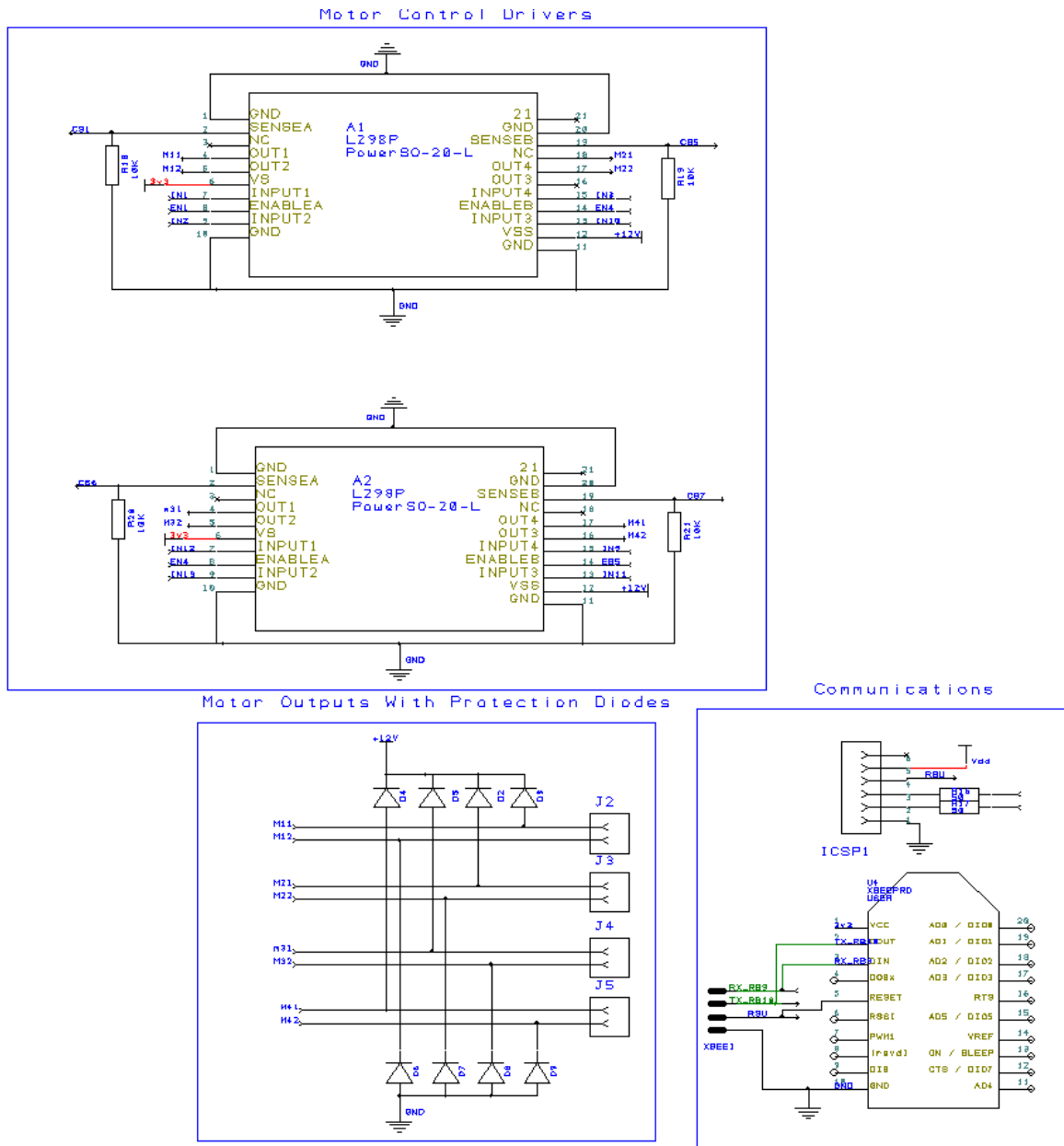


Figure 1: Platform Motor Control

### 4.3 Platform Control Terminal

The platform control terminal is the actual physical setup that we will be using to control the platform wirelessly. It is based on a fairly simple control setup using switches and potentiometers. We are going to be using a limit-switch (also known as “paddle switches”) based joystick for each direction. When you push the joystick in a direction, it provides a 0V (ground) signal to the microcontroller. In order to control the motor speed we will be using a slide-potentiometer. It is sent as an analog signal and then later mapped out to provide a speed control based on pulse width

modulation from 0-255. Figure 2 shows the schematic for the platform control terminal layout.

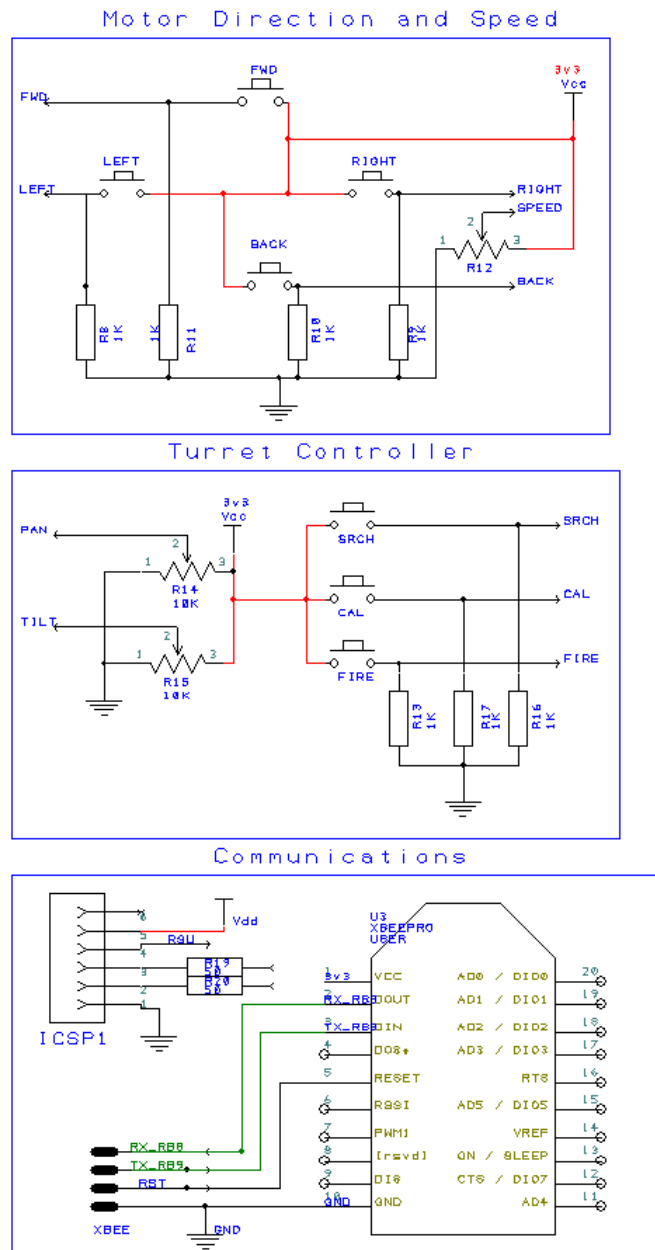


Figure 2: Platform Control Terminal Layout

#### 4.4 Software

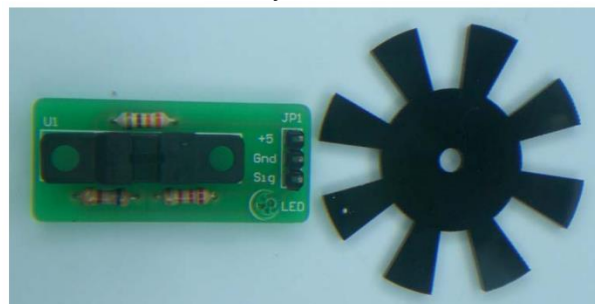
The microcontroller on the platform requires a total of six pins per motor control driver, requiring up to 12 pins total for the four wheel drive system. Eight of the pins are discrete values providing the logic to determine the direction of the motors. Table 1 below displays the logic behind direction. Four of the pins are pulse-width modulated (PWM) signals controlling the motor speed. The PWM signal controls the

speed by varying the output voltage going to each motor. It will output a PWM value ranging between 0-255 varying the voltage from 0-12V. The higher the PWM rate, the faster the motor will spin until it becomes maxed out at 12V.

<b>Direction</b>	<b>IN1</b>	<b>IN5</b>	<b>IN2</b>	<b>IN6</b>	<b>IN3</b>	<b>IN7</b>	<b>IN4</b>	<b>IN8</b>
<b>Forward</b>	0	0	1	1	0	0	1	1
<b>Right</b>	1	1	0	0	0	0	1	1
<b>Left</b>	0	0	1	1	1	1	0	0
<b>Reverse</b>	1	1	0	0	1	1	0	0
<b>Neutral</b>	0	0	0	0	0	0	0	0

**Table 1: Motor Control Logic**

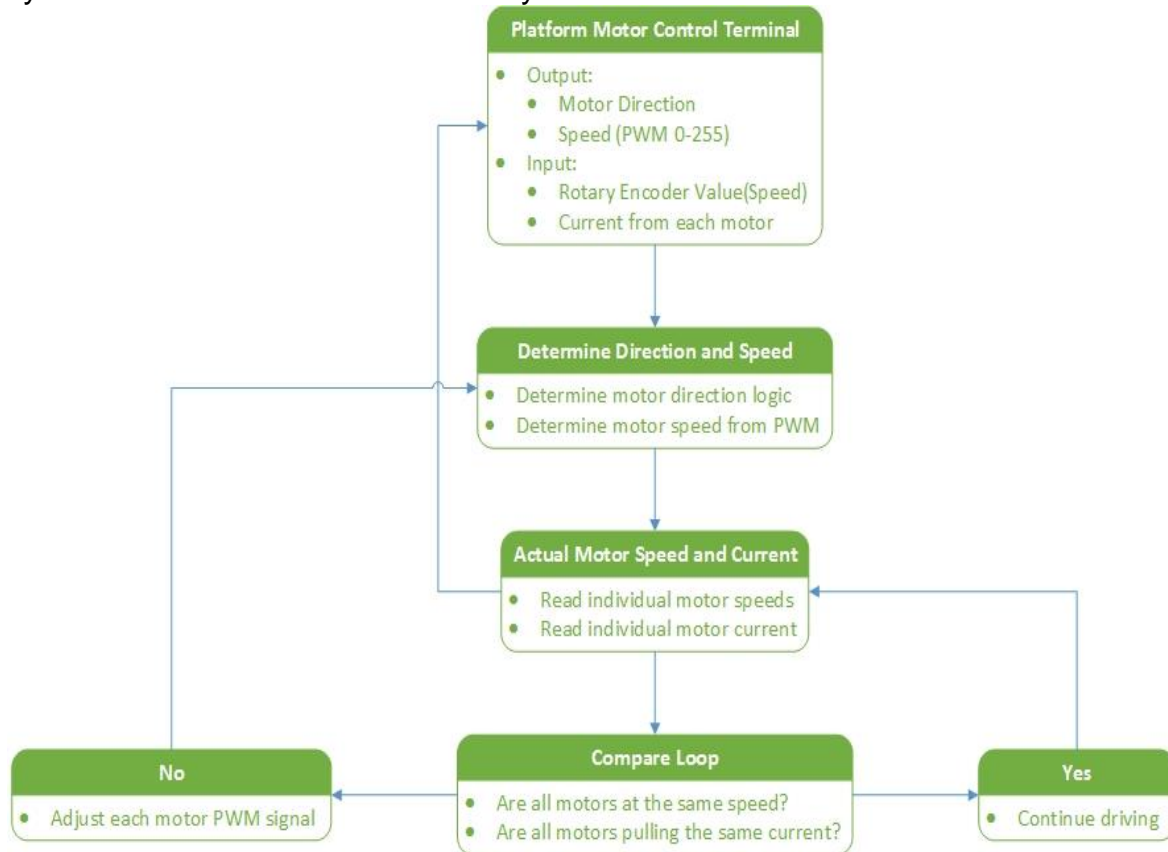
We will also require the use of the four analog pins to be used on the current sensors. Although we are controlling the motor speed using PWM, we wanted to be able to verify the motors are not trying to pull too much current and overheat or damage the motors. One of the mechanical properties of motors is that no two motors will spin at the same rate even if they are the same series. One of the ways we are trying to help avoid this issue is by monitoring the speed and adjusting the PWM signals for each one. Although the motors may be receiving the correct PWM value, we can offset the motor speeds to verify they are all spinning at the same speed and be able to correct the PWM values to achieve this. These speed values will be corrected through a closed loop PID control loop. It is possible to purchase a motor with an encoder built in and also possible to make a standalone optical rotary encoder. The encoder that is currently being researched is an optical encoder that can be added onto any motor. This is based off the *Cytron RE08A Rotary Encoder Kit* (Cytron) shown in Figure 3. This setup would consist of adding a slotted disk at the end of the motor shaft. While the motor is spinning, an optical sensor will count the amount of times it has seen the gaps in the slotted disk. It outputs a simple on-off based signal very similar to a pulse width modulated output. We are looking at combining the current sensing, PWM outputs for speed control, and using the optical encoder to have a complete motor control system.



**Figure 3: Cytron RE08A Rotary Encoder Kit**

We would use the PWM outputs from the microcontroller to adjust the speed of each motor as previously mentioned. The current sensing will be used to determine how hard the motor is trying to push itself and to also help monitor how much power the motor is drawing. If we are running low on battery power, speed and distance isn't an issue, we can slow down and conserve power. We would use the optical sensors

as a more accurate way to detect the speed at which each motor is spinning and then compare it to the PWM signal being produced to each motor. It would still utilize a closed loop feedback system where we can adjust the PWM values according to each motor to make sure they are all spinning in sync to be able to make sure the platform will drive straight and turn correctly. Figure 4 shows the basic feedback system of the overall motor control system.



**Figure 4: Motor Control Feedback System**

The next part of the software is the integration into the GUI for the motor controls. The control terminal has the hardware pieces mentioned above. Those will send the signal to the microcontroller to determine the function. It will then map out and flag the data to be sent to the platform microcontroller. The slide potentiometer will be read in as an analog value and the joystick will be read as discrete values as either high or low.

The GUI also provides full control to the motors but seemed a little difficult to control because of having to constantly move the mouse and click. Therefore the GUI provides us with a way to monitor all of the functions to see what is being sent and received between the platform microcontroller and the control terminal controller. The layout of the GUI is displayed in the Touchscreen LCD section.



## 5.0 Platform Touchscreen LCD Display

### 5.1 Design

We had originally planned on building a software GUI to run on the Raspberry PI. The aforementioned GUI is what we are currently using for development and debugging purposes. Fortunately we have received enough outside funding where it is now possible to be able to afford to implement a full color graphical touchscreen LCD display into our final release of the project. The LCD will simply run the previously mentioned systems. There are five main screens the LCD will have. These are the main screen, platform motor control, security, basic platform monitoring, and turret control. We will be implementing one of 4D Systems touchscreen LCD modules and programming them using their ViSi design suite (VISI).

The first screen is simply the overall screen which will allow you to select which screen the user wants to view. It has four buttons to select an individual screen as well as a picture and quoted text for visual purposes. Figure 5 shows the current progress of the main screen.

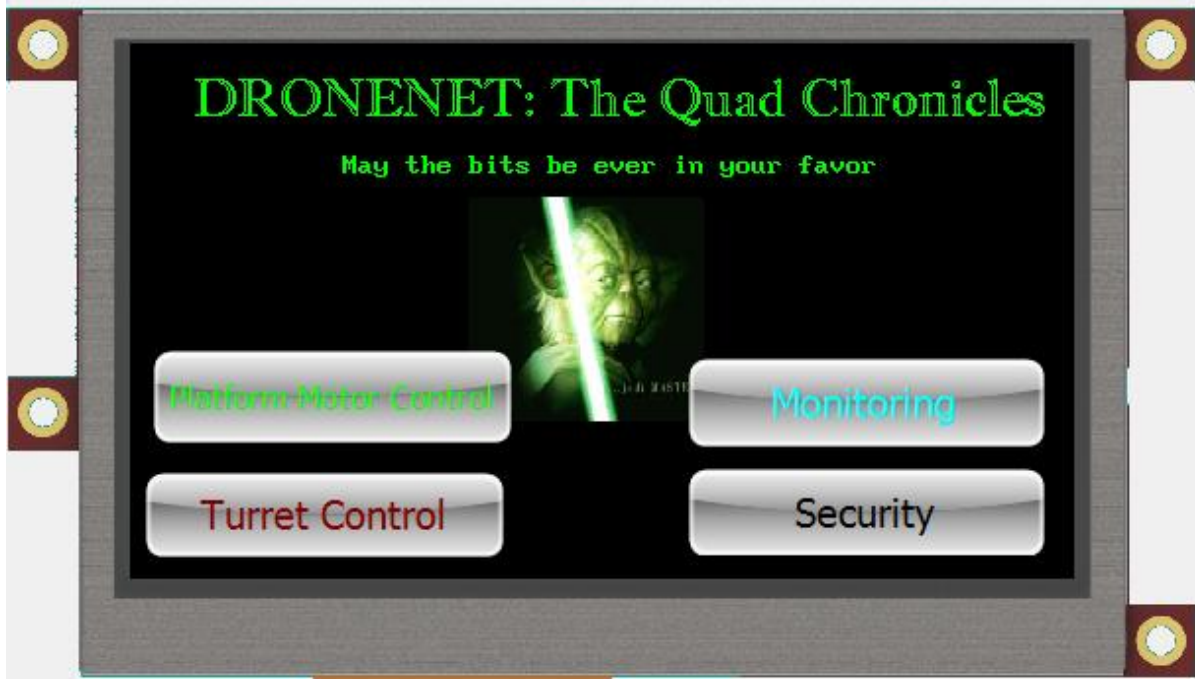
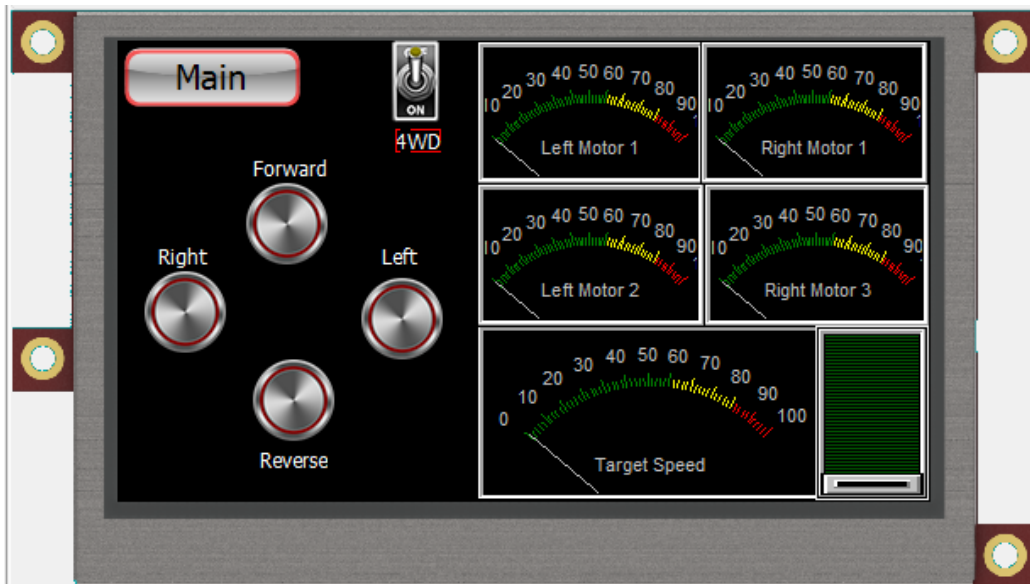


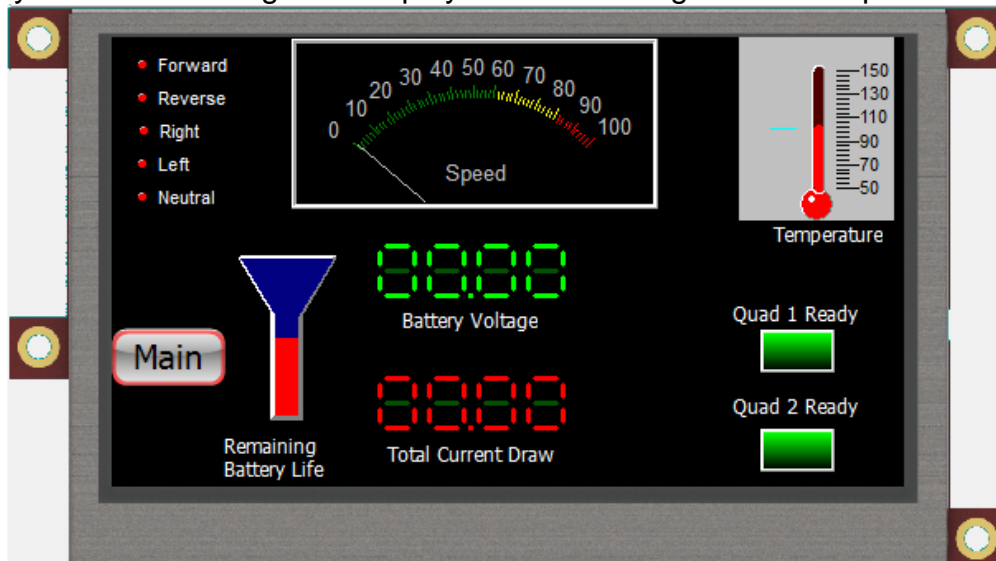
Figure 5: LCD Main Menu Screen

The platform motor control will be able to control and monitor the platform while still retaining the option to control it by hand with the physical controller. It will have the ability to control motor direction, speed and enabling the four wheel drive capabilities. It will display the desired speed as well as the actual speeds each individual motor is producing based on the rotary encoders attached to them. Figure 6 below displays the current setup of the motor control LCD display.



**Figure 6: Platform Motor Control Screen**

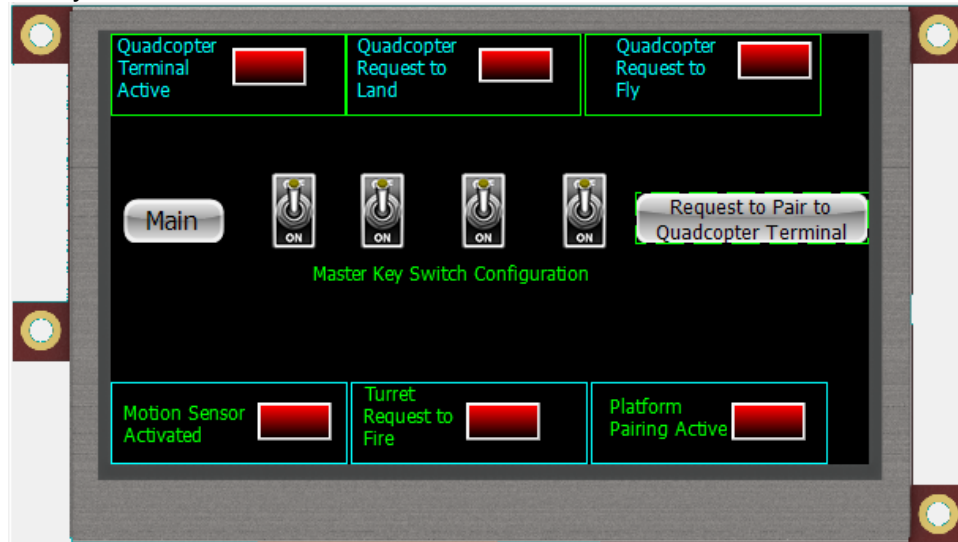
The next screen will be the overall platform systems monitoring. The screen will monitor the current draw on the main battery, the battery voltage, the remaining battery time, the ambient temperature, and whether or not the quadcopters are currently locked down. Figure 7 displays the monitoring screen setup.



**Figure 7: Platform Monitoring Screen**

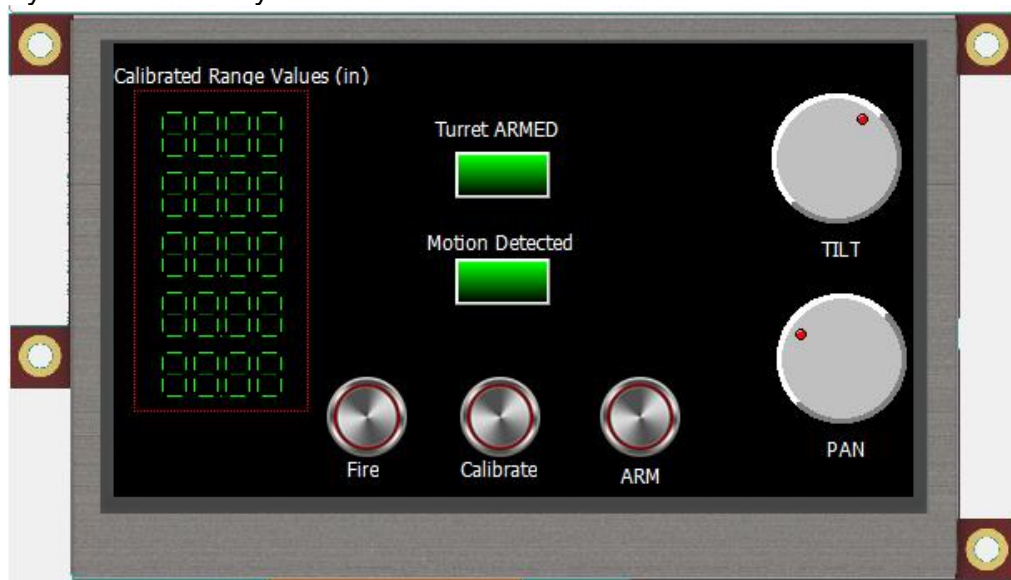
The fourth screen will be the security screen. It will monitor the current progress of the security functions of the system. It will display the current switch configuration used for controlling the master key, whether or not the platform control terminal and quadcopter control terminal are successfully paired together, if the pairing between the platform control terminal and the platform are currently active, and if any of the

requests to perform a function are active or not. Figure 8 displays the current setup of the security screen.



**Figure 8: Security Systems Screen**

The fifth screen is the turret control screen. It will allow the user to control the pan and tilt motion of the turret, selecting between the different turret controls modes: calibrate, arm, and fire; the current range of any object above it, confirmation to fire the turret and if the PIR motion sensor has been active as well as the calibrated perimeter range values. If the PIR motion sensor becomes activated and the user is on the main screen or any other screen a warning will flash on that screen indicating the turret has locked onto a target and request the next step to fire or ignore. Figure 9 displays the current layout of the turret control screen.



**Figure 9: Turret Control System Screen**

We are still discussing the possibility of needing more screens. We want to keep the LCD as minimal as possible but also still make sure it includes all of the necessary features. The LCD will communicate all of the data by using the onboard UART data pins connected to the microcontroller UART pins.

## 6.0 Microcontroller Research

Our project will require having to use multiple microcontrollers throughout the different subsystems. Each one will be performing specific tasks and we need to pick out the most appropriate ones to meet our specifications. Table 2 displays the features we were looking at when determining a microcontroller.

MCU	ISA Size	SRAM	FLASH	ADC	DIO	UART	SPEED	PRICE
PIC32MX320F128B	32 Bit	16K	128K	12	43	2	80Mhz	\$3.22
ATMEGA328P	8 Bit	2k	32k	6	14	1	16Mhz	\$2.50
MSP430G2553	16 Bit	128b	16k	4	10	1	20Mhz	\$2.00
AT91SAM3X8E	32 Bit	96k	512k	12	54	4	84Mhz	\$10.54

**Table 2: Microcontroller Research Comparison Specifications**

### 6.1 Platform Control Terminal

For the platform control terminal we needed a controller that can perform serial communications at very high speeds with very little error. We also need it to be able to handle external interrupts with ease and be able to compute all of this as quickly as possible to provide accurate control of the platform. We will only need up to four analog inputs, 12 digital inputs, and two UART communication ports, one for the LCD touchscreen and one for sending/receiving the wireless data. We originally tried using the simple ATMEGA328 but found that it lacked the processing power for the UART and continued to send false data. We then tried the PIC32MX320F128B and found that it handled all of the processing and interrupts with ease. It also provides us with multiple UART ports and provided a Microchip exclusive called Peripheral Pin Set (PPS) (Microchip) allowing us to change the configuration of our pins without have to do it in software.

### 6.2 Platform

The platform will consists of multiple microcontrollers to keep certain processes isolated and help lower power consumption. For example, the motor controller will have its own dedicated microcontroller. All of the motor monitoring and control for four motors will end up using up a lot of pins. We also won't need the motor controller to be on if we are not moving. We will either be putting the controller in sleep mode or simply holding it in reset until we need to run it. This also added protection for the rest of the platform since motor controllers are notorious for getting fried for various reasons and we did not want to fry the entire platform system due to a motor malfunction. We also wanted a DIP style microcontroller that we could just use a socket for quick swapping if we needed to change out. Since we do not need heavy processing, memory, or expanded UART ports; we will be choosing the ATMEGA328P.

The next microcontroller on the platform will be the controls for the laser guided landing system and turret controllers. It will perform all of the functions necessary to

operate the servos, range finders, send and receive UART data, motion sensor, lasers and perform the math on the autonomous landing system algorithm. It will require two pulse width modulated outputs, one UART, four analog inputs, and up to six DIO. Due to the possible fast and heavy math operations and the amount of data being sent over the UART port we felt it was best to go with the PIC32MX320F128B again.

The last microcontroller on the platform will be the “main brain”. We are still deciding if we want to use an FPGA for this or keep with the microcontroller for uniformity. This processor will perform the voltage monitoring for the Bedini motor, switching between which batteries are charging and the security algorithm as well. We will be using this processor to control the other two processors to hold them in reset until we need them. We are still deciding how we want to do the UART communications but we wanted to be able to have at least two UART ports just in case we want to communicate with the other processors on board. We will be using the PIC32MX320F128B due to the need of accurate results over the UART for the security system and the voltage monitoring application. For the voltage monitoring, we need something to be able to know when to switch the batteries after they are fully charged and will be using this controller for those tasks.

### 6.3 Quadcopter Controller

The quadcopter controllers were a pretty simple choice. We looked into writing our own firmware for the quadcopter controls but soon discovered how daunting of a task it was going to be. We realized that we did not need to reinvent the wheel and therefore will be chose a compatible microcontroller with the Arducopter firmware. We also needed the firmware to be open source so we could add our own functions and easily be able to change out parts or add things. We will go more in depth in section 0 , with all of this in mind we decided to go with the ATMEGA2560.

## 7.0 Bedini Motor

### 7.1 Motivation

During random internet searches over the past year, the Bedini motor was discovered and we were intrigued. The Bedini motor claims to be a highly efficient design that transfers energy from electrical into mechanical and then converts that mechanical energy back into electrical energy to charge other batteries. A lot of people on the web try to claim that this is an over-unity device (i.e. perpetual motion machine); that is, this device creates more energy than it uses. However; the creator, John Bedini, denies this claim. He calls it a “battery desulphator” and a “highly efficient motor.” (Bedini Motors)

### 7.2 Theory of Operation

The Bedini motor works on a principle of a pulsed DC voltage. The idea is that an electromagnet circuit is switched on and off based on sensing the proximity of a permanent magnet as it rotates on a rotor. Initially when the rotor is not spinning, the system is “off” and no current energizes the electromagnet. As can be seen in Figure 10, once the rotor starts spinning; the permanent magnet moves towards the center of the coil, an induced voltage into the sense winding (red) biases a NPN transistor and increases the base voltage such that the transistor turns on. This supplies current to the electromagnet and produces a magnetic field that is the same polarity as the permanent magnet. The repelling magnetic field accelerates the rotor in the same direction that it was moving. As the permanent magnet moves away from the center of the coil, the induced voltage starts to reduce, eventually below the turn on voltage of the transistor. When the transistor turns off, no current flows through the coil, but the rotor continues to spin due to momentum. This completes the drive portion of the circuit.

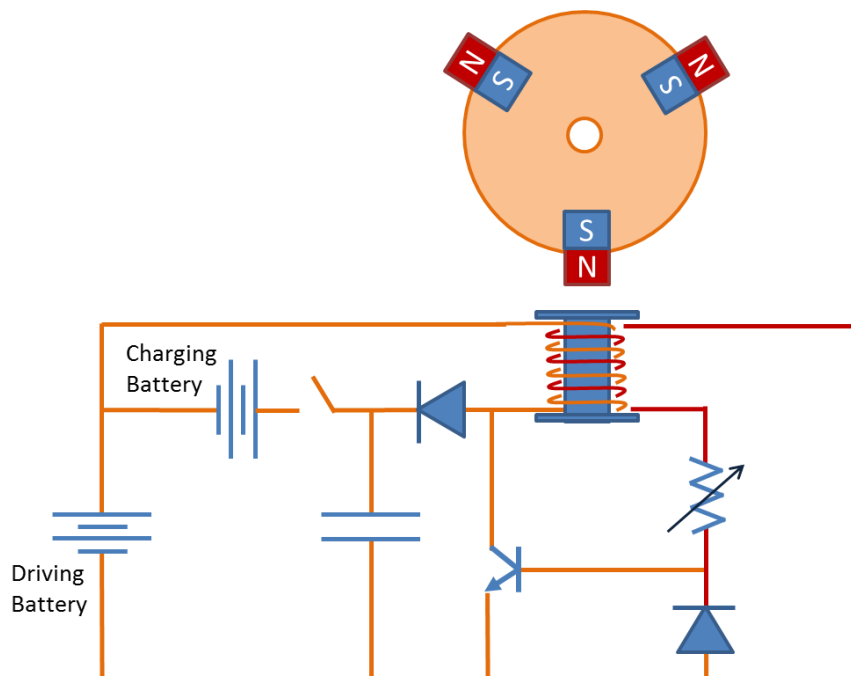


Figure 10: Illustrative schematic of a simple Bedini motor circuit.

The battery charging comes from two mechanisms. The first mechanism occurs due to the magnetic field developed in the coil. When the power is removed, there is an inductive kick from the coil due to the collapsing magnetic field. However, since the transistor is off, the only path allowed for current flow is directed into the capacitor bank.

The second mechanism happens due to the interaction of the rotor and the coil. Generator action happens when a magnetic field is moving relative to a conductor. This induces a voltage into the windings of the coil, aiding the current flow into the capacitor bank. The voltage produced by these mechanisms is rectified to ensure the proper polarity for charging the batteries. A controlled switching mechanism allows the energy to build up in the capacitor bank and periodically dumps this stored energy into the battery to be charged.

What differs about this device from other charging devices is that this is designed to charge batteries using a pulsed DC voltage. Traditional battery chargers utilize continuous DC current as their charging mechanism. The problem with using continuous current as the charging mechanism is that the  $I^2R$  losses generate heat in the battery and cause a variety of issues. These include sulfate buildup and evaporation of water (Pistoia). With the Bedini motor, the pulsed DC voltage can be much higher than continuous since the duty cycle will be much less than 100%. Since:

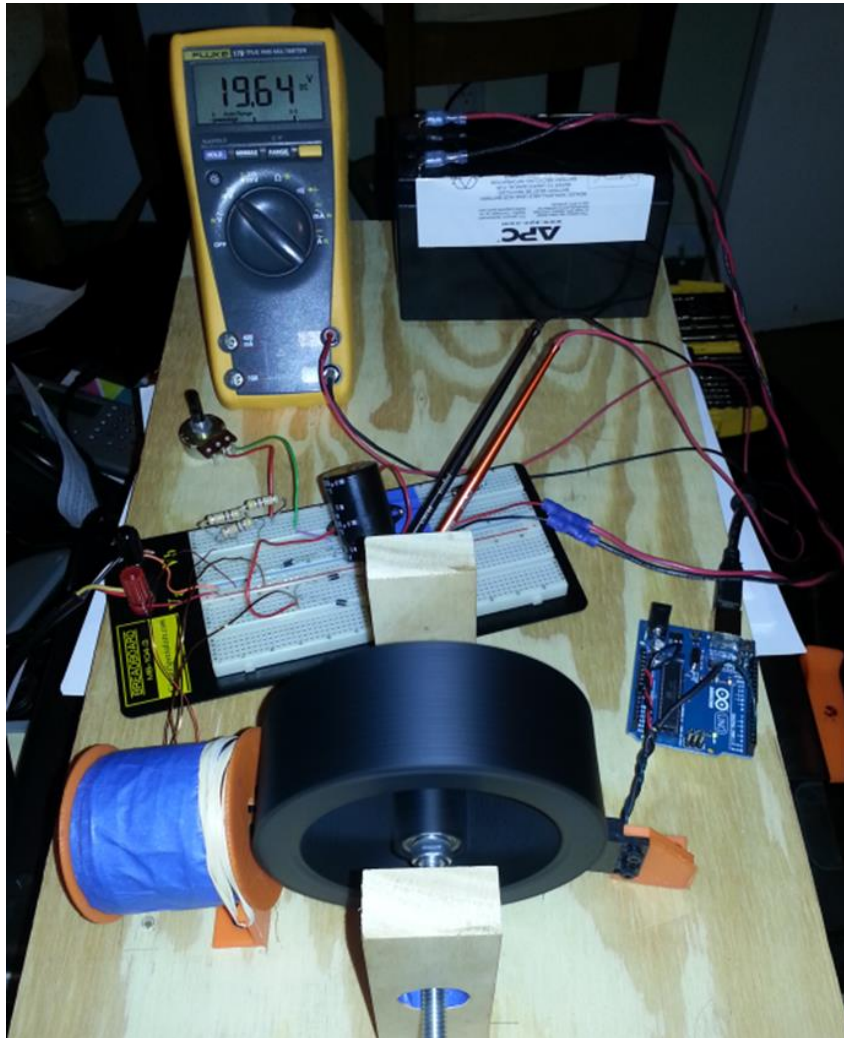
$$E = P \times t$$

The 't' will be a fraction of continuous, which implies more power can be applied for a short period of time to get the same energy into the battery. Since the current will not be continuous as well, the current over time will be reduced, which means the heat generation will be significantly less from the  $I^2R$  losses.



### 7.3 Design Considerations

Initially our hope was to be able to put this on one of the quadcopters to directly extend its flight duration. We quickly realized that was not feasible due to weight considerations for the quadcopter. The most critical reason for this is the motor has only been tested to run with sealed lead acid (SLA) batteries. Additionally, the components that make up the circuit are for power circuits, and tend to be heavier and bulkier than what we would want to have on the quadcopter. Instead we decided to move the charging system to the mobile platform and will connect to it after the quadcopter has landed.



**Figure 11: Development of the Bedini motor.**

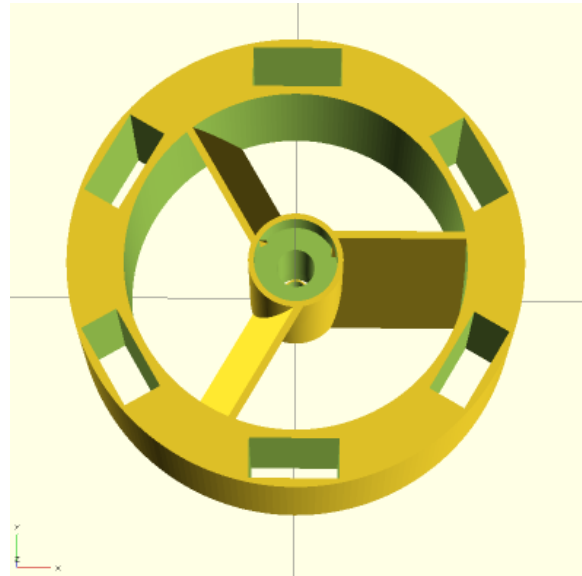
Figure 11 shows the development platform for the Bedini motor. The Arduino is utilized as a meter for different parameters to determine the effectiveness of the motor. This allows us to simultaneously monitor several different parameters and that data is output to a computer screen for viewing. With this information, we can

monitor the status of the various batteries and can swap them around as needed. Additionally, a Hall Effect sensor is used to monitor the rotational speed of the rotor.

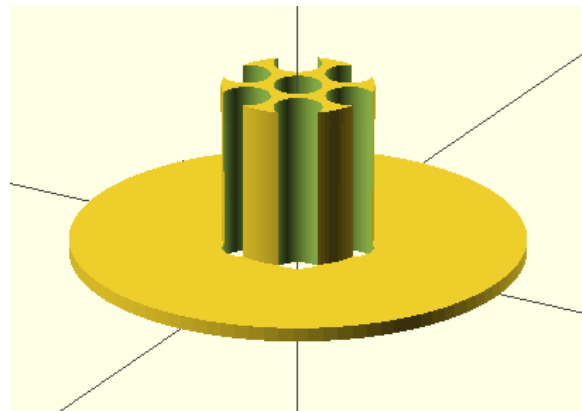
The permanent magnets are  $\sim 1'' \times 2'' \times 3/8''$  and are mounted in a 3D printed rotor assembly Figure 12. The spool holder for the dual coil was also printed using a 3D printer and has holes for inserting ferrous rods for the coil Figure 13. Additionally, the spool holder was printed in two pieces and zip tied together to hold the coil on the spool. In lieu of using batteries to drive the motor, a 12V power supply is being used to prove out and optimize the rest of the circuit. Once the circuit has been optimized, there will be tests performed utilizing the battery driven circuit.

Part of the optimization process will be to find capacitors that will be able to hold the charge from the rotor and quickly transfer that energy into the batteries. During preliminary testing, the currently used capacitors charged quickly over their rated voltage of 100V.

A switching circuit will also be integrated into the development circuit to control the charging of the capacitor bank and subsequent discharge into the batteries. In his patents, Mr. Bedini initially used a mechanical switch that was coupled to the permanent magnet rotor. For this design we will use the Arduino for the timing of the switch, to minimize excess mechanical devices on our mobile platform.



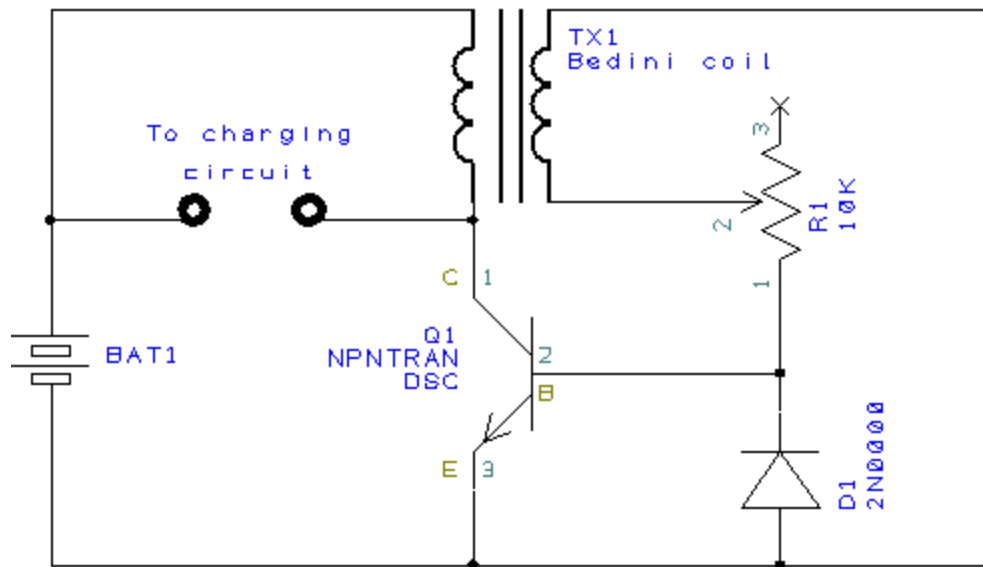
**Figure 12: Model of the rotor holding the permanent magnets.**



**Figure 13: One half of the spool holder for the dual coil.**

## 7.4 Monitoring System and Charging Batteries

In order to automatically switch between charging batteries and the driving battery, a monitoring system must be implemented. This monitoring system will measure the input voltage of the driving battery as well as the voltage of each of the batteries being charged. In this system, we will have four batteries to swap between. It will also measure currents so that we can measure the amount of energy being moved through the system.

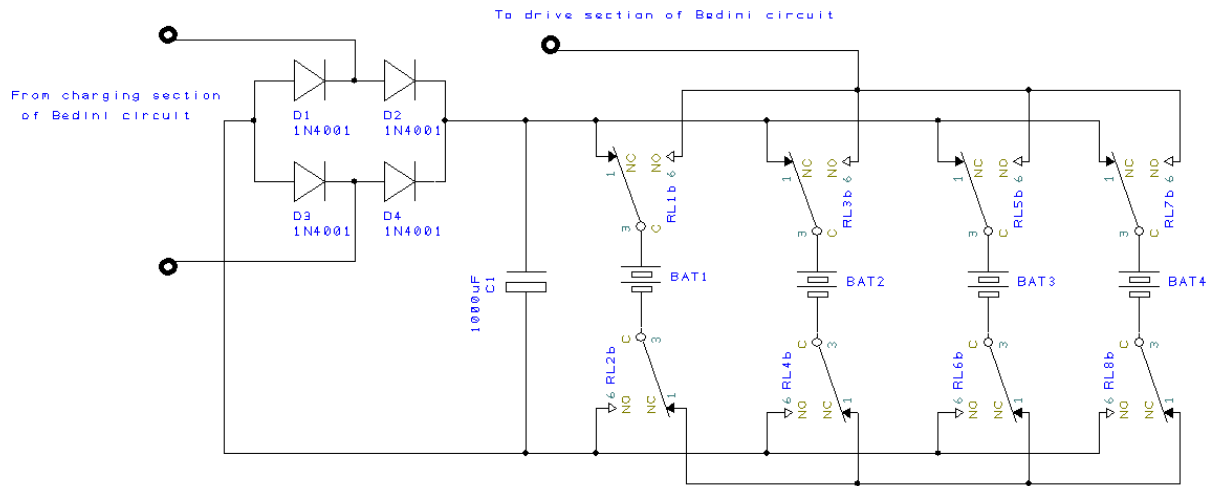


**Figure 14: Bedini motor circuit including monitoring system.**

We will have two voltmeters to monitor the voltages of the driving battery and the charging batteries. The first voltmeter will be monitoring the driving battery and will be used to determine when that battery will need to be swapped out. The second voltmeter will monitor the capacitor bank. By monitoring the capacitor bank, we will be able to see the voltage that is being sent to the batteries when they are being charge and also allow us to ensure the charging capacitors do not exceed their rated voltages from the charging of the Bedini motor. Measuring the voltage going to the batteries will mostly be for determining efficiency of the system and will help us to optimize the system, as needed.

It would be desirable to have four independent voltmeters, one for each battery; however, this adds a level of complexity to the system that is not sensible at this time. The reason for this is that the capacitor charging bank will see voltages up to, and possibly greater than, 100V; which makes the monitoring circuitry more complicated. Having one of these circuits will be enough for our purposes.

There will be two ammeters implemented in this circuit as well. The first ammeter will be connected to the driving side of the Bedini motor circuit. In combination with the voltmeter, these measurements will allow us to monitor the input energy being supplied from the driving battery. While the voltage monitor will be used to determine when the driving battery needs to be swapped out, the current monitor will only be used for the input energy measurement. The second ammeter will be used to monitor the current coming from the capacitor bank and going into the charging batteries. As with the voltmeter connected to these batteries, the ammeter will be used in determining the energy being sent to the batteries for determining efficiency of the system.



**Figure 15: Schematic of the charging and switching circuit connected to the Bedini motor circuit.**

The last aspect of the monitoring system is a RPM sensor for the rotor. Because of the presence of the permanent magnets in the rotor, a Hall Effect sensor is used to calculate the RPM. By counting each magnet as it passes the sensor and timing them, we can determine the approximate speed of the rotor and convert it to RPM.

All of these measurements will be sent to an Atmega chip and programmed in the Arduino environment. Using the information from these devices, energy can be measured and efficiency can be calculated to monitor the performance of the motor. Additionally, the Atmega chip will be used to control the eight SPDT switches that will be used to determine which batteries will be used to drive the Bedini circuit at any given time. Because we will not be monitoring individual voltages of each battery, each battery will be used as the driving battery sequentially. As the driving battery falls below a specified voltage, the next battery in the sequence will replace it as the driving battery. Additionally, a supercapacitor will be placed in parallel with the driving battery in order to maintain power to the system while the batteries are being switched.

## 7.5 Solar Panel(s)

In addition to cycling through batteries and recharging them, we plan to place a solar panel or two on the mobile platform to supplement the power needed. With the goal of extending the operational lifetime of this system in the field, the solar panels provide us a large reservoir of energy to use. The solar panel will be able to replace the driving battery in the Bedini motor, allowing us to charge all of the batteries, if needed, while still providing enough power for the rest of the electronics (except possibly the motors for moving the platform). It will also be possible to connect these up to charge the batteries for the quadcopter directly, depending on the needs of the system. Combined with the regenerative ability of the Bedini motor, we should be able to extend the duration of this system greatly.

Another opportunity to use solar panels is to operate the control terminals. With the exception for the laptop that will be running the Ardupilot software, all of our electronics are designed to run off of a 12V supply. For the power side of the control terminal, a solar panel would be a renewable source of energy would supplement our dependence on batteries for these devices.

## 7.6 Charging the Li-poly Quadcopter Batteries

Due to the restrictions for charging lithium polymer (Li-poly) batteries, the Bedini motor will not be able to charge these directly. Unlike traditional Ni-Cad or NiMH batteries, Li-poly batteries have an issue with overheating. Because of this, there has been a lot of research on how to charge them efficiently and safely; and there have been many ICs that have been engineered to control the charging for these batteries. There are two options that we are investigating on how we are going to charge the quadcopters from the mobile platform.

The first option is placing one of these ICs on the quadcopter's PCB and supply power from the charging circuit of the Bedini motor. The advantage of this method would be in simplifying the connection to the quadcopter. We would incorporate the IC onto the quadcopter and would add the extra circuitry we would need to not disconnect the Li-poly batteries when they needed to be charged. From the mobile platform, we would extend a wire that would supply the high voltage to this circuit and disconnect it when it's done; a minimal transition to charge the quadcopter. Unfortunately, this would add complexity and mass to the quadcopter that may affect the quadcopter's flight time.

The second option is to place the charging circuit on the mobile platform and have a cable available to plug into the Li-poly batteries from that charging circuit.

The disadvantage is that it would add complexity to the transition to charge the quadcopter because we would have to unplug the batteries from the PCB and then plug it into the charging circuit on the mobile platform. The advantage would be less mass and complexity for the quadcopter PCB.

An idea that we have been considering is attempting to make the transition to charging automated, as part of the landing and locking sequence. This would be an ideal feature because it would make the entire system automated. To do this we have been discussing a magnetic connector that would be strong enough to make contact when the quadcopter landed, but would let go when the quadcopter was launched. For this reason, we are leaning towards the first option. We feel that we can accommodate this feature, but we won't know for certain until we have the initial prototypes built and are able to test the feasibility of this idea.

## 8.0 Mobile Platform Power Supply

### 8.1 Goal

For the mobile platform we will be using a 14.4 volt battery to power the whole system. However we will also need five volts, 3.3 volts, and 12 volts for the components in the system. Therefore we need to come up with a way to power these devices. The devices that need to run off of a 14.4 volt system will be the motors and the motor controls. The 3.3 volt, five volt, and 12 volt regulators for the rest of the system can run straight off of the main power supply.

The devices that to run off of a 12 volt power supply are a voltage meter and red, green, and blue LED strips. The devices that need to run off of a five volt supply are an LCD screen, a rangefinder, four MOSFETS, and some of the servo controllers. The devices that need to run off of a 3.3 volt supply are two XBEE wireless modules, two PIC microcontrollers, two motor control driver IC's, 12 optocouplers, eight LED's, an FTDI cable, and two push buttons.

### 8.2 Hardware: Components Research

In order to provide the right voltage and current to the components that need 3.3 volts, five volts, and 12 volts we can either have multiple batteries or step down our power supply to three different levels, once to 12 volts and the second time to five volts and a third time for the 3.3 volts.

Having multiple batteries would definitely increase the overall capacity and consequently prolong our mission time. The main downfalls of this solution is that it will increase the payload of the mobile platform, and the batteries that would be available for the 3.3 and five volts would not have a huge capacitance, and they would also take up valuable real-estate inside the mobile platform. Clearly from the amount of components that need to run off of the 12 volt, five volt and 3.3 volt systems, having separate batteries will drastically increase the payload of the quadcopter and they will also take up a large amount of space.

From our research we have determined two methods that will be able to step down the voltage are to use either a linear voltage regulator or a switching regulator. Each component has its own benefits to stepping down the voltage for our system. The linear regulators require less board space which will be essential for our PCB design. While the switching regulators require big bulky components, they are much more efficient then the linear regulators. From our research the typical efficiency of a linear regulator is between 40% and 60%, while the typical efficiency of a switching regulator is between 70% and 90%. Since our project requires that we get the most out of our battery power supply, being able to step down the voltage efficiently is the most important benefit. The linear regulators also generate a large amount of heat, and the performance of the regulators decreases as a function of increasing temperature.



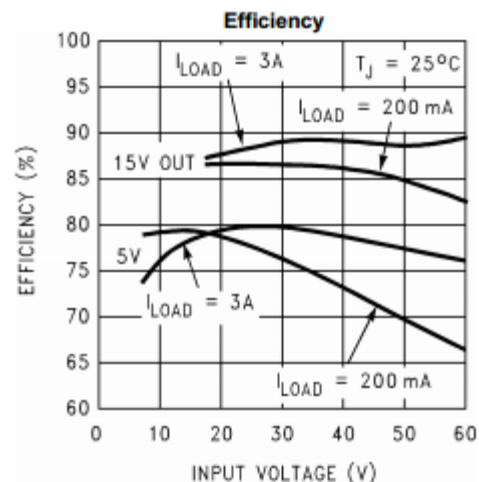
Some of the linear regulators that we looked into are the LM 2931 CT and the 78M12CT. The LM 2931 is an adjustable linear regulator that will allow us to achieve our 12 volts, five volts, and 3.3 volt specifications. The 2931 requires output capacitors for device stability as well as a resistive network to act as the adjustable output. The downfall of this regulator is that it can only support up to 100 milliamps of output current, which is significantly less than for our application (LM 2931 Data Sheet). The 78M12CT is also a linear regulator; it comes in separate packages for 12 volts, five volts, and 3.3 volts. This regulator only requires two capacitors for stability, yet it only allows for 500 milliamps of current draw. For the inability for these regulators to supply the amount of current that we will need for the power supply we will be using switching regulators (78M12 Data Sheet).

The switching regulators that we are looking at are the Texas Instruments LM 2576 as well the LM25576, the LM 25005, and the LM 2650. One of the downfalls of the LM 25576 is that we can only get an output voltage as high as nine volts. While the LM25005 package is larger than the 2576 and is less efficient. The LM 2650 has many more components to adjust the output voltage than LM 2576. The rest of the comparisons and the characteristics that we deemed important to this project are in Table 3.

Regulator	Average Efficiency	Output Voltage(s)	Max Output Current	# of Outside Components	# of Pins
LM 2576	88%	12,5,3.3	3A	4	5
LM 25576	85%	Up to 9	3A	13	20
LM 25005	90%	Adjustable	2.5A	13	20
LM 2650	85%	Adjustable	3A	15	24

**Table 3: Switching Regulator Comparisons**

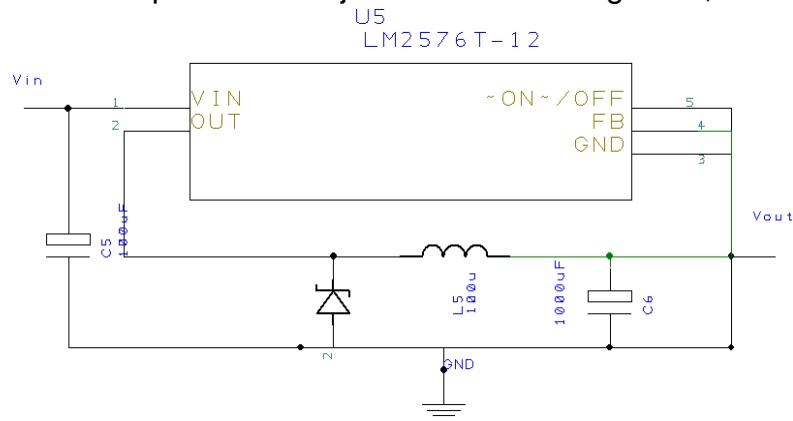
The switching regulators that we will be using to create the power supply are the LM 2576-12 LM2576-5 and the LM2576-3.3 from Texas Instruments. As the name of each indicates the first will step the incoming voltage to 12 volts, the second to five volts and the last will step the voltage down to 3.3 volts. We chose these regulators because they only require four external components which will help with limiting the size of our PCB, they have a efficiency of approximately 85-90% between a load of 200 milliamps and three amps as shown in Figure 16 from the LM2576 data sheet, and due to the efficiency they have a relatively small power dissipation through heat and they are simple to use and build. (LM 2576 Data Sheet).



**Figure 16: LM2576 Efficiency (reprinted with permission from Texas Instruments)**

### 8.3 Hardware: Design

The design of the power supply circuit is shown in Figure 17. The configuration is given from the LM2576 data sheet provided by Texas Instruments. One of the great things about using the LM2576 series of switching regulators is that all three circuit designs for the three different output voltages are exactly the same and require the exact same outside components with just the different regulator, see Figure 17.



**Figure 17: LM2576 Configuration (reprinted with permission from Texas Instruments)**

The difference between the 3 separate regulators is a voltage divider within the IC with different values for each output, making the power supply system very simple. We have each one of the regulators running off of the 14.4 volt supply rather than cascading them and feeding the stepped down output to the input of the subsequent regulator. I wanted to do this for a few reasons; one of the main ones is having the isolation from each of the subsystems that are connected to the power supply.

For safety purposes I have put in a diode along with a fuse at the input of each regulator, this will protect all of the components from an accidental voltage reversal. If the 14.4 volt battery is accidentally hooked on wrong all three of the fuses will break, this method will require more parts and will cost slightly more. The reason for doing this is to protect each of the individual systems from affecting the others; if one of the regulators malfunctions, its own fuse will blow and will allow the others to stay powered.



## Mobile Platform Power Supply

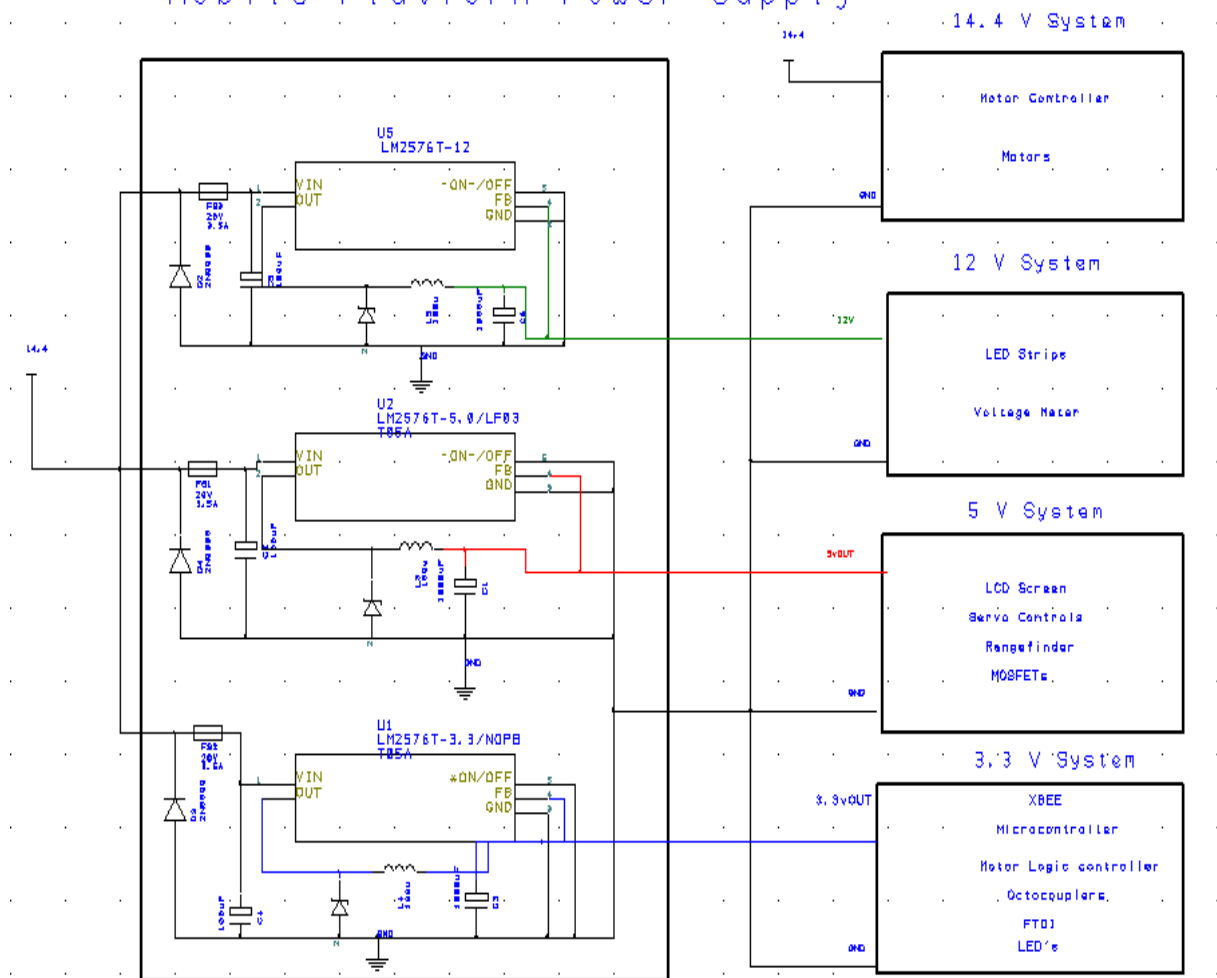


Figure 18: Power Supply Schematic

## 9.0 AES Security

Since this is being modeled after a defense project and has the ability to fire at objects, it became apparent that it is important to secure the systems. If an outside hacker were able to get into any part of the system in real life, the outcome could become fatal. For these reason we are implementing a secure algorithm that requires several layers of verification using dynamic keys. The authentication techniques and encryption methods will be utilizing AES algorithms. This system was derived from projects used while taking the *Hardware Security and Trusted Circuits Design* at UCF taught by Dr. Yier Jin (Jin), *Cyber Security Awareness Week* competitions, and an *AES Security algorithm for embedded devices* derived from Brian Gladman (Gladman).

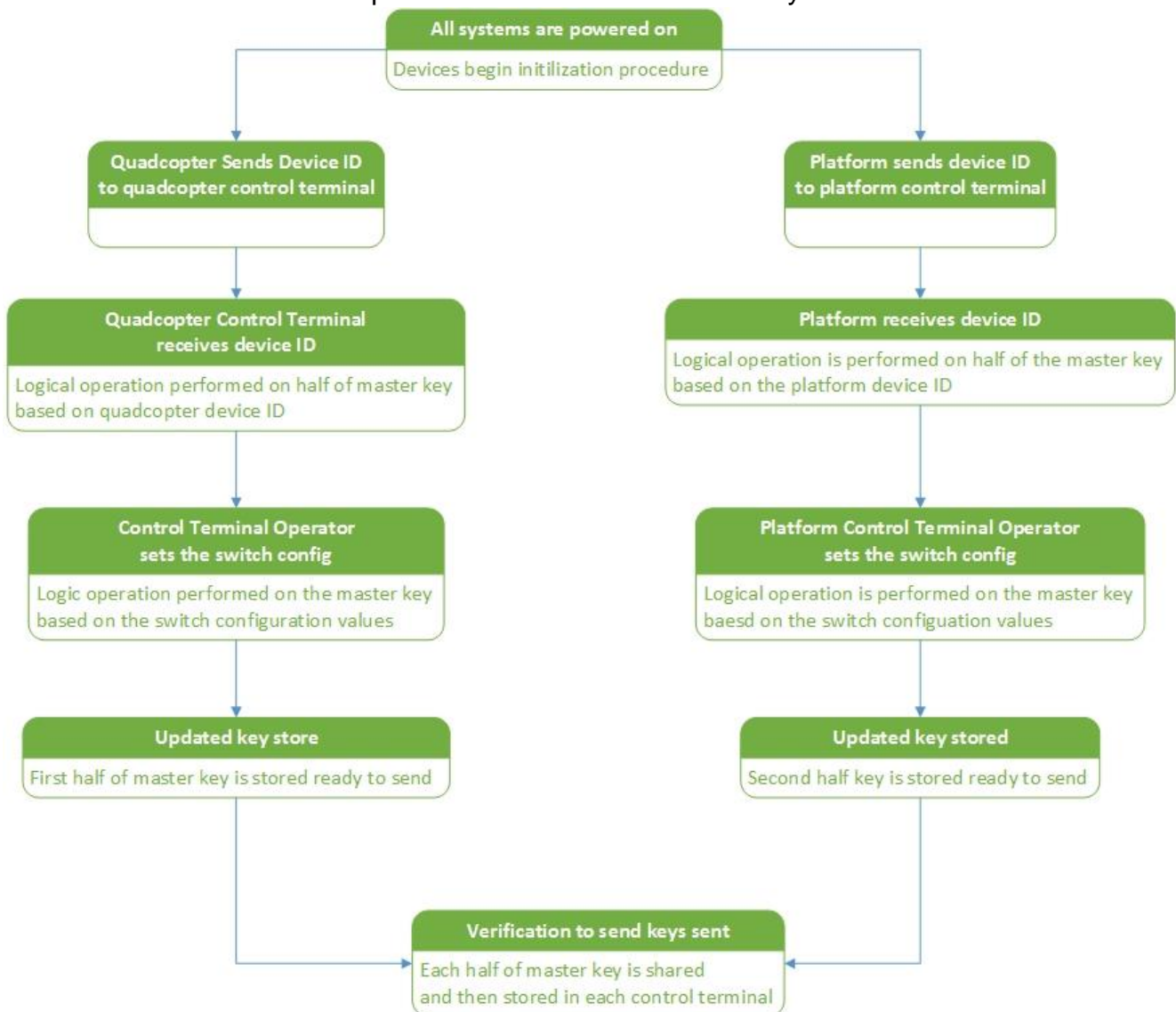
The system will have a certain number of verification requests. These happen during the initial startup, when a quadcopter wants to launch on a mission, when a quadcopter wants to land on the platform, and when the platform wants to fire the turret.

We had to have a way for the master key to be common among all of the components but still become dynamic to help eliminate vulnerabilities. There are ways we have gone about achieving this using several different techniques. The first technique is each of the devices that are being controlled is assigned a device ID. For now this device ID is going to be a static variable for the purpose of simplification. Each control terminal, the quadcopter control terminal and the platform control terminal, contains half of the master key. On each terminal there is a special set of switches used for being able to change the key. The reason for this is to prevent software hackers from getting in the system and figuring out the keys. The switch configuration is updated every one minute between each terminal once the initial verification is complete. In the case of an incident where an outside hacker got into the system, this would allow the control terminal operators to change their key instantly and would make it harder to re-obtain because it is done on a hardware level.

The initial startup is where all of the keys are sent and configured. The quadcopter will send its device identification number to the quadcopter control terminal. Once the device identification is obtained on the control terminal, it will then take the device ID and perform a logical operation to the half of the master key that it has inside of it. This helps prevent the key from always being the same in the case where if we configure a way to make the device ID's dynamic and still stable. After that the quadcopter control terminal operator will set his switch configuration. There will be four switches that just use simple on-off logic (1 and 0). Once the switch configuration is set the half of the master key is then performed one more logical operation based on the switch configuration. The quadcopter control terminal key is now set and ready to be sent to the platform control terminal.

The platform control terminal works in a very similar fashion as the quadcopter control terminal. The platform is assigned a device identification which is sent to the platform control terminal upon startup. The logical operation is then performed and the switch configuration is then set. The half of the master key that the platform control terminal has is now complete.

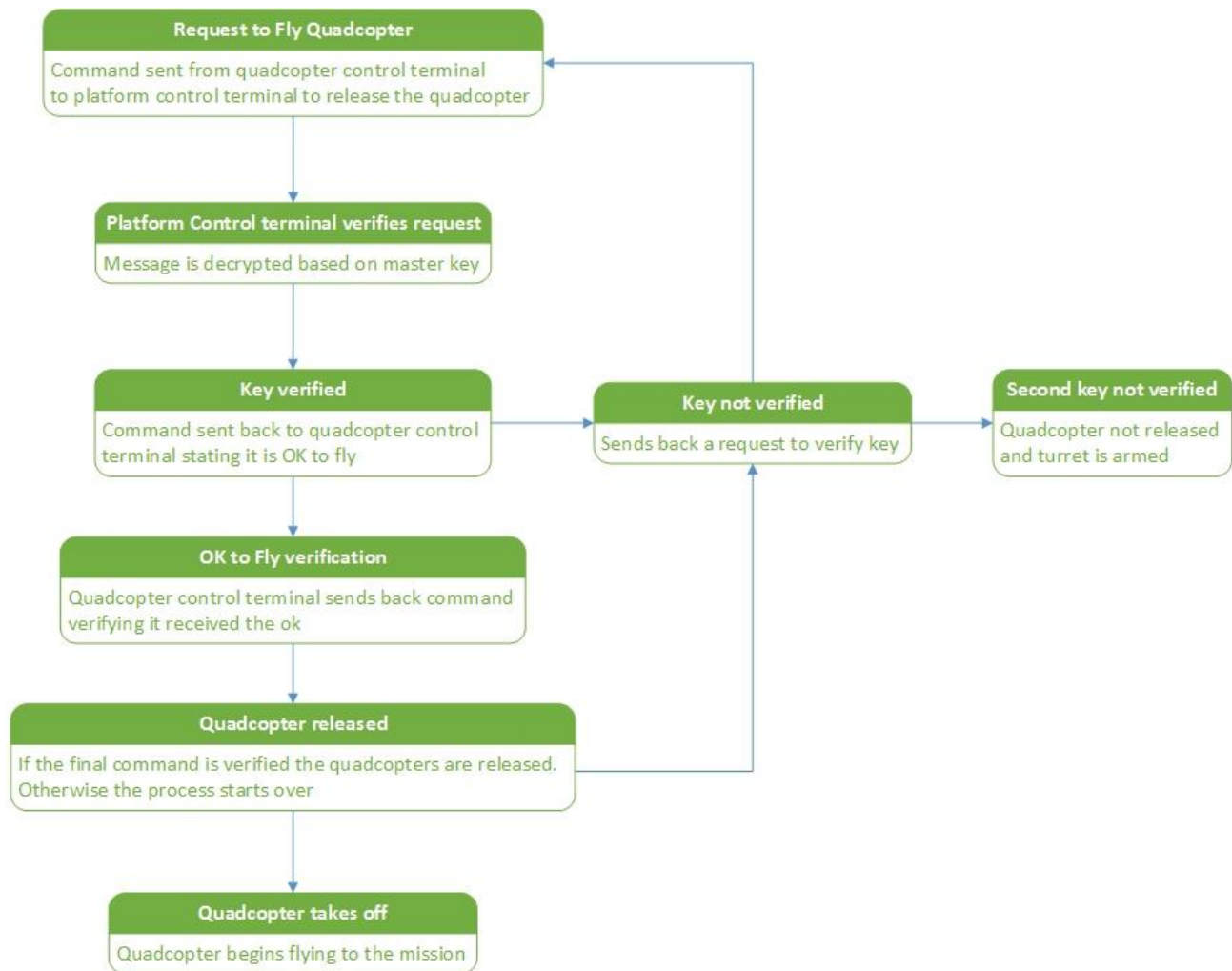
Now that both of the control terminals have their halves of the master keys set they need to send them to one another to be stored for later use. There is a special switch configuration that has to be on each terminal for this next step to happen. They will send each a request to share master keys. This will simply begin with each one sending each other's switch configuration and verifying it is correct. Once they are verified each half of the master key is then sent to one another and concatenated to the respective halves and the master key is set.



**Figure 19: Security Initialization Procedure**

Once the initial master key is set it can be changed anytime and will automatically update every minute. The switch configuration is sent to each terminal during every update and the logical operation is performed based on the switch configuration and the master key is updated on both control terminals. The initial security procedure is outlined in Figure 19.

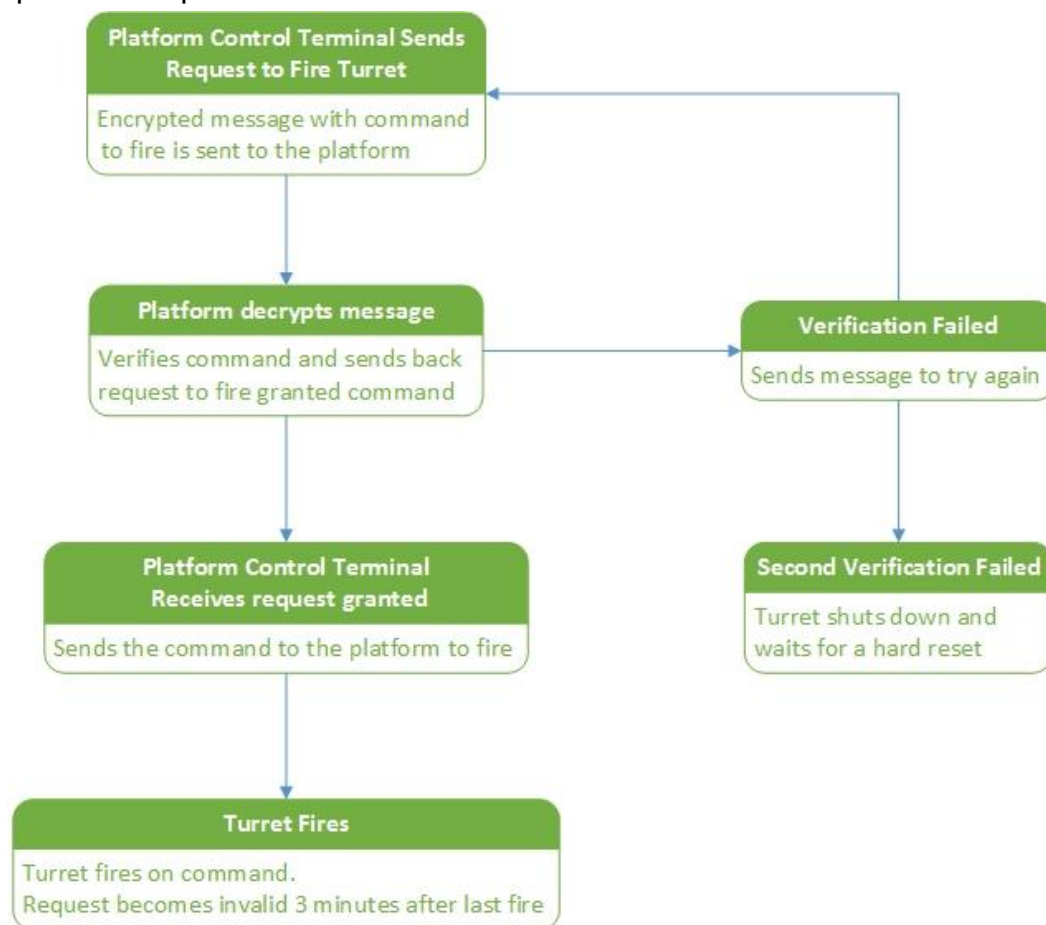
Since the quadcopters are locked down on the platform, the platform control terminal operator will have to release the quadcopters in order for it to take off. When the quadcopter wants to fly, it will need to send a “Request to Fly” command to the platform control terminal. This procedure begins a “handshake” between the two. The quadcopter control terminal will send the request to fly in an encrypted message. The platform control terminal will then decrypt the message and send back the response if the message matches the request to fly command. It will send an encrypted message back to the quadcopter control terminal indicating it is clear to take off. The quadcopter control terminal will then send back one last message verifying that it received the message. Once that command is decrypted and verified



**Figure 20: Request to Fly Security Protocol**

in the platform control terminal it will send the command over to the platform to release the locking mechanisms. This is also a decrypted message that the platform will have to decode and verify before releasing. The quadcopter is now ready to fly and can take off and perform its original plan. Figure 20 demonstrates the layout of the request to fly security procedure.

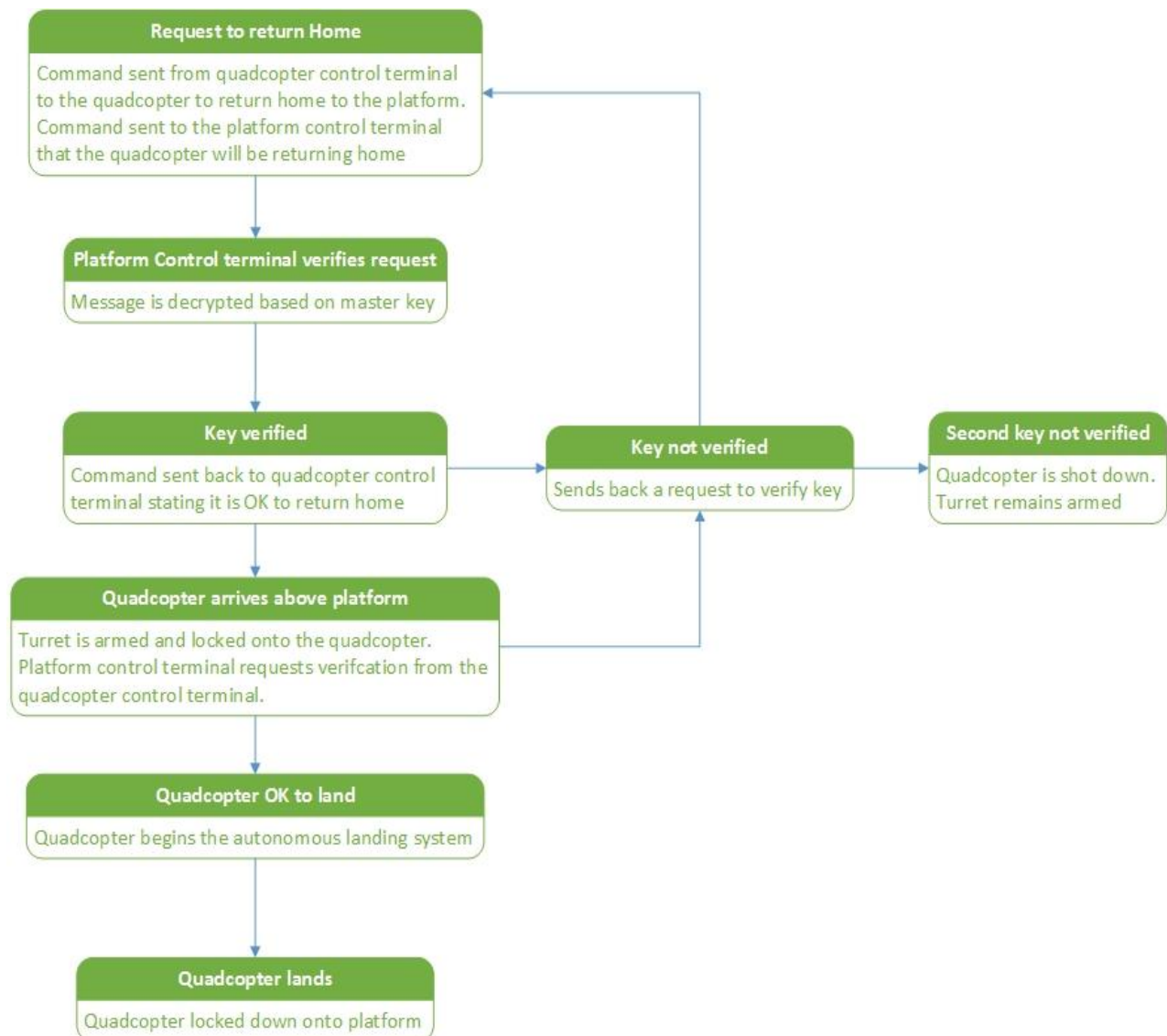
The next layer of security is on the platform turret. It performs a very similar handshake when it wants to fire. If the control terminal operator wants to fire the turret, they will have to send an encrypted command to the turret controller to fire. The turret controller will then decrypt this message based on its device identification to verify the command. If the command is verified it will send back a verification code verifying that it is clear to fire. The operator can now fire at will. Figure 21 outlines the request to fire protocol.



**Figure 21: Request to Fire Protocol**

In order for the quadcopter to return to the platform it will need to verify itself. Once the quadcopter control terminal sends the command to return home it will also send the command to the platform control terminal. The same encryption and decryption methods are performed from the ready to fly commands only a different command set for landing. Once the platform verifies that the quadcopter will be returning home

the turret will then perform a perimeter calibration and arm itself waiting for the quadcopter to get close. Once the quadcopter is overhead the turret will perform the perimeter check and lock onto the quadcopter waiting for it to send the second verification. When the quadcopter is overhead the platform it will have to send the command to the platform controller a request to land command. It will perform the same encryption and decryption methods previously discussed. If the verification fails the turret will lock onto the target and send a warning that it will fire if it fails on more verification. The quadcopter will then send out the verification again. This time if the quadcopter fails verification it will be instantly shot down. Otherwise the quadcopter will perform its autonomous landing technique. Figure 22 demonstrates the protocol required to land.



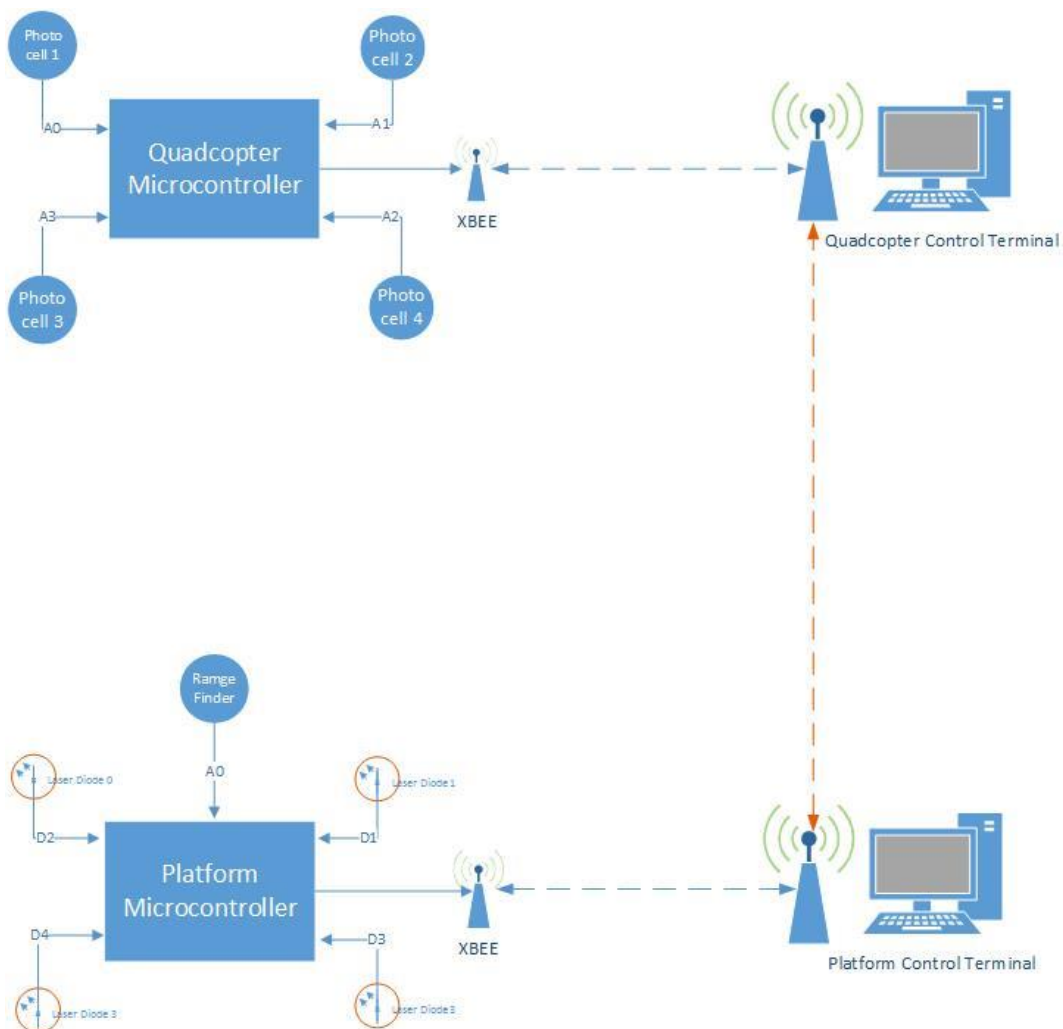
**Figure 22: Request to Land Protocol**



## 10.0 Laser Guided Ultrasonic Autonomous Landing System

### 10.1 Concept

This part of the project utilizes a method for an autonomous landing system. The components being used are four laser diodes, four photocells, and one ultrasonic range finder. The overall concept is to get the quadcopter hovering over the platform at a certain distance using the GPS to find its way home. Once it is at that distance, it will then continue to rotate until the lasers line up with the photocells. Once the lasers line up, the quadcopter will then continue to land on the mobile platform. The overall layout of the autonomous laser guided landing system can be seen in Figure 23 below.



**Figure 23: Laser Landing System Layout**

The setup for the lasers will be integrated into the mobile platform. We decided to put the lasers on the platform instead of the quadcopter to maximize the available

power for the quadcopter. The lasers will represent the landing pattern we need the quadcopter to line up with before landing. The photocells are a varying resistance based on the amount of light emitted onto them. Putting the photocells on the quadcopter helped reduce any ambient light that could possibly interfere with the readings of the photocells.

When we bring the quadcopter home, it will get as close to the platform as possible. We may have to do some of our own manual steering to get it where it needs to be positioned. Once the range finder finds the quadcopter, the quadcopter will hold static hovering over the platform. While it is being held stable, it will continue to rotate in circles until the landing position is determined. The way the landing position is determined is when all of the lasers have hit the photocells, the photocells will all have the same outputs readings. There is one photocell on each arm of the quadcopter aiming towards the platform. Then the platform will turn the lasers on and wait for all of the photocells to be lined up. When the photocells are lined up, the quadcopter will stop rotating in circles and begin lowering itself to land on the platform. We will be using a PID control loop to control how fast the quadcopter lowers itself based on the distance from the platform read by the range finder. If at any point the quadcopter photocells are not line up with the lasers on the platform, it will stop lowering itself and then continue to rotate in circles until the position is lined up again. It will repeat this process until it lands onto the platform.

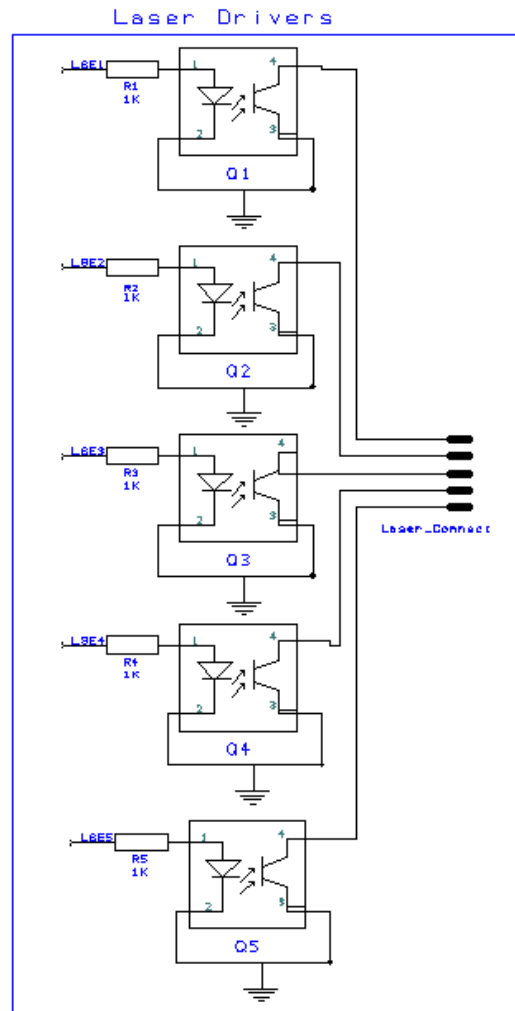
In order to prevent errors or the quadcopter getting stuck in an infinite loop trying to correct itself perfectly, we will have a threshold that will be based off a timer. As long as the photocells are lined up for a certain amount of time it will flag the quadcopter to be OK to land. While lowering itself to the platform it will only check the photocells at a timed interval. Once the quadcopter is at a predefined distance above the platform, it will no longer need to check for the photocells being lined up and will continue to land.

## 10.2 Hardware:

The platform will consist of four to five lasers per quadcopter, one ultrasonic range finder, optocouplers, and a microcontroller running off an Xbee for the wireless communications. We decided not to hardwire the lasers to always be on since the lasers will only need to be on during landing. The optocouplers are being used to drive the lasers to isolate them from the microcontroller and to help alleviate pulling too much current from the microcontroller IO pins. This will help with power management since we will also be using the optocouplers as a way to control turning them on and off. The ultrasonic range finder is to detect the range the quadcopter is above the platform. We will determine whether or not a wide beam or narrow beam ultrasonic range finder will be used during testing. The concerns with using a wide beam ultrasonic range finder is that it could pick up the quadcopter before it is actually close enough to the platform to land. The concern with using a narrow beam ultrasonic range finder is that it may require the quadcopter to need to be too close, causing the quadcopter to have a hard time finding the platform. The microcontroller



is being utilized to control the lasers turning on and off and to read the data coming in from the rangefinder and sending the distance back to our control terminal. The Xbee is just providing wireless communications. Figure 24 shows the schematic of the controller being used on the platform to drive the lasers and range finder.



**Figure 24: Platform Controller Laser Driver**

The quadcopter only requires the photocells, microcontroller, and an Xbee. The photocells use a voltage-divider system to detect the light being emitted on them. The outputs are then fed from the voltage divider output into the analog pins of the microcontroller. Once the lasers line up with the photocells the analog value will change indicating it is lined up. The value of the analog signal will depend on ambient light and also the distance from the platform the quadcopter is hovering. The calibration of the ambient light is discussed during the software section. The Xbee is simply providing a means of wireless communication between the quadcopter and the platform.

### 10.3 Software:

The platform software system will mostly be for triggering, photocell calibration, and determining the range the quadcopter is above the platform. Once control terminal triggers the “Home” command, the range finder will continue to wait until the range has changed from its previous value. Once the value has changed, the quadcopter will send a “request to land” message to the control terminal. If the request is relevant (relevance determination is discussed in the “Security” section) the microcontroller will then activate the laser system. The laser system will then continue to run along with the range finder until the quadcopter is lined up and landed. As soon as the quadcopter is hovering above the platform it will calibrate the photocells. This will be used to kick out any ambient light. The calibration will just read all the current values before the lasers and use that as the comparison value.

The quadcopter will consist of a closed loop feedback system triggered by the range finder value. We will have a pre-determined value for the distance the quadcopter needs to be at before the feedback system runs that controls the quadcopter becomes active. Once all of the photocells values line up from the quadcopter, it will begin the correction loop the quadcopter will need to go through. The correction loop consists of rotating the quadcopter until the photocells are all lined up. It will rotate a specified amount of time until the photocells are lined up. The rotation interval time will be determined during testing and will also be configurable on the control terminal GUI. Figure 25 below demonstrates the different states of the total algorithm.

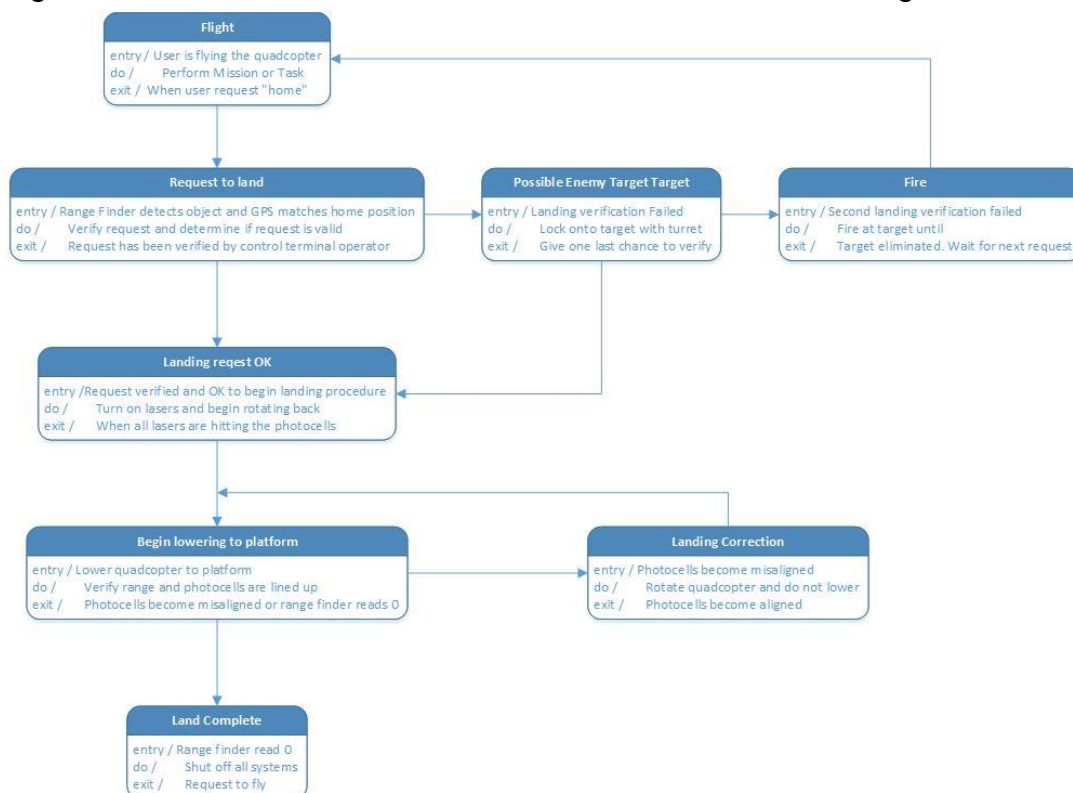
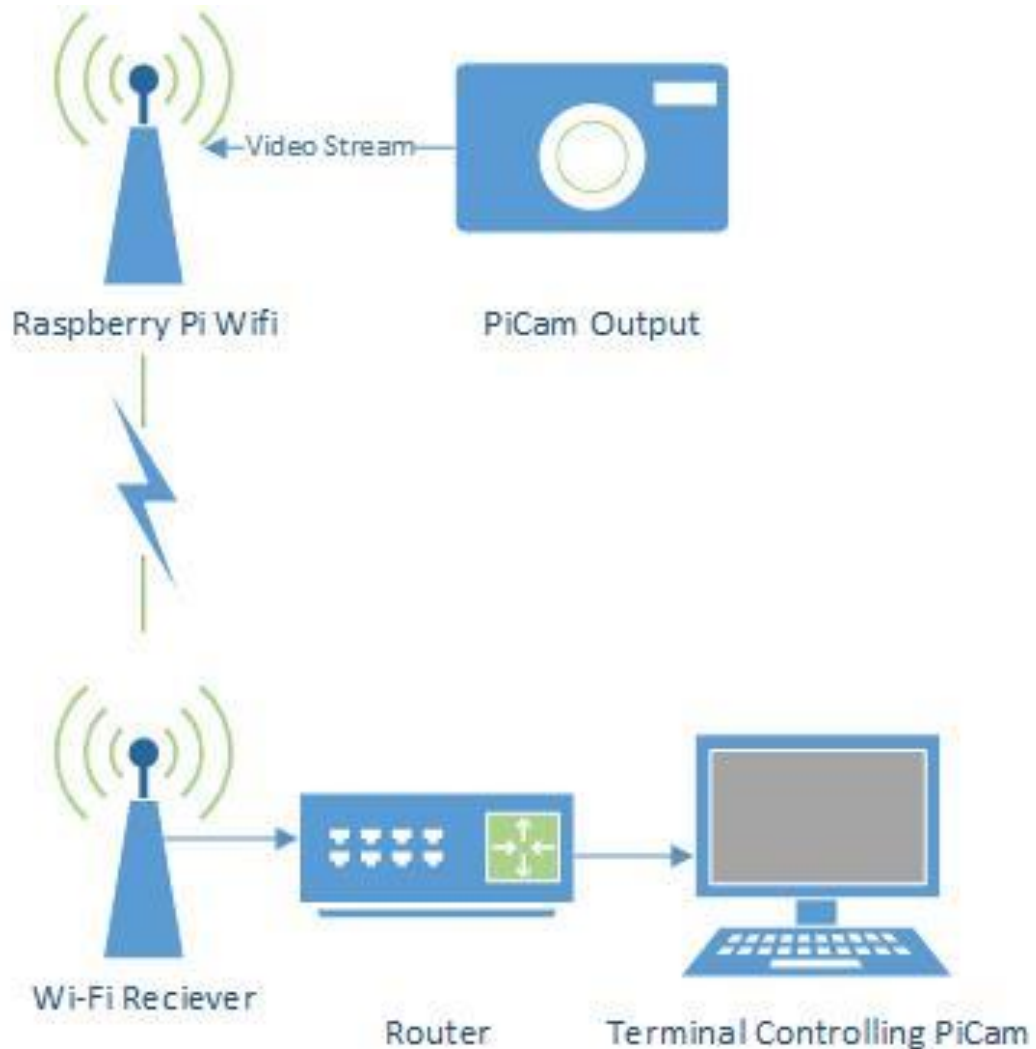


Figure 25: Laser Landing System Software FSM

## 11.0 Wireless Camera System

We will be utilizing a basic camera setup for the quadcopter. We originally wanted to be able to view the entire flight as a live video stream. We then looked further into it and realized how high the power consumption ramps up due to how much data would be transferred through the wireless system. The following Figure 26 displays the basic layout of the camera system.



**Figure 26: Wireless Camera System**

We chose three options; Go-Pro HERO3 (GoPro), Raspberry Pi with PiCam (Pi), and a Foscam Network Camera (Foscam). Before we discuss our decision we will review each camera setup and compare the pros and cons of each. Afterwards we will narrow it down and determine which of the systems and why.

## 11.1 Go-Pro HERO3:

### *Pros:*

- The most durable camera system
- Highest image quality
- Many quadcopter frames support the go-pro and already have mounts built in for them
- the 3D printing community offers huge support for different mounting methods
- Go-Pro App allows being able to view and control camera over Wi-Fi

### *Cons:*

- Very expensive (\$249 for the bare minimum)
- High power consumption (1.2 amps at 3.7V)
- Weight (4.8 oz.)
- Size (1.6" x 2.4" x 1.2")
- Less flexibility for being able to mount it anywhere due to its shape and size
- Proprietary software for controlling and viewing

## 11.2 Raspberry Pi with PiCam:

### *Pros:*

- Flexible mounting options with the option of mounting to a servo for changing viewing angles
- Having a full Linux System on Board the quadcopter leaves us room for future additions
- Price (\$35 for the Raspberry pi and \$29 for the PiCam)
- Weight (1.6 oz. without the camera, 3.0 oz. with the camera)
- Power Consumption (750mA at 5v with the camera running, >400mA without it running)
- Familiarity with being able to learn more about Linux
- Easily able to embed to a GUI
- Able to switch between Video Stream and JPEG

### *Cons:*

- More of a learning curve for to make it run
- There are not currently available mounts so we will need to make our own
- It comes as a bare circuit board which means it is not as durable out of the box

### 11.3 Foscam Pan and Tilt Network Camera

#### *Pros:*

- Built in pan and tilt motors
- Price (\$79.95)
- Wi-Fi built in
- Durable
- Audio playback
- Night Vision

#### *Cons:*

- Weight (25 ounces)
- Size (3.9" x 2.3" x 1.5")
- Mounting options don't exist
- Proprietary software for controlling it
- 5V at 2.0 Amps

We are still currently trying to fully decide between the Raspberry PiCam and the Go-Pro HERO3. The original decision was on the Raspberry PiCam largely due to cost, mounting options, future addition options, and power consumption. Now that we have found out we will be receiving outside funding, we are debating on going back to the Go-Pro HERO3 setup. The reason why we might be able to get around the power consumption issue is because the Go-Pro has its own built in battery separate from the power supply that the quadcopter will be using. The new version also includes an app to view and control the video which would eliminate us from having to worry about the software. The last issue is the weight. It comes down to only a few ounces, but in a quadcopter a few ounces can make the difference of a stable flight or not. Although the Go-Pro HERO3 is the best out-of-the-box solution, the Raspberry PiCam still offers everything we are looking for and is still currently the top winner on the list. The extra workload seems so far like it would be worth it for the added benefits. The Go-Pro will make our project easier but we would be cutting out the learning experience of being able to run a Linux network camera system.

The communications for the camera system are going to be separate from our data stream that will be controlling the quadcopter. The reason for this is to keep the data stream as clear as possible to prevent any information problems and the quadcopter crashing or having erratic behavior. The camera stream will be sent via the onboard Wi-Fi of the Raspberry Pi. We will be using the built in commands embedded into the Raspberry Pi's Linux terminal (GNU). The Linux terminal makes it possible to take control of Raspberry Pi and gives us several different options to view our live stream. We can control the time the stream is viewed, if we want to save the video or just view it, and also to be able to switch just taking a JPEG image instead of video. There will be times where a JPEG will be all we need or actually ends up

better such as viewing a target after it was marked. Being able to view a JPEG will drastically reduce the power consumption. Being able to control these options is powerful for us because we can be able to switch between modes during flight without any modifications to the actual quadcopter providing us with an all-in-one solution for our quadcopter.

We are also discussion the options of being able to change the viewing angle of the camera. Most quadcopters just have the camera pointed underneath it. In most scenarios this is all that is needed but since we are going to be using this as a targeting system, we need to be able to view the surrounds and possibly what is above the quadcopter. Because of this we will probably be implementing a servo controlled mounting system on the camera if the weight will allow us. This is also another reason why the weight and size is important. The less weight and size, the less powerful the servos need to be.

## 12.0 Passive Infrared Ultrasonic Target Acquisition System

### 12.1 Goals:

The first step of this system is to be able to defend the mobile platform. Due to the types of systems and data that the mobile platform has on-board, it could be prone to attack. In order to prevent this we have designed a missile defense system.

### 12.2 Components: Hardware

The system's main physical components consist of: passive infrared motion detection, ultrasonic sonar transducer, servos, DC driven pump, MOSFET, optocouplers, and a 1mW laser. For physical control the system utilizes an analog joystick, a set of pushbuttons, wireless communication device, and microcontroller; and bridges off of the on-board power system. Figure 27 displays the component layout.

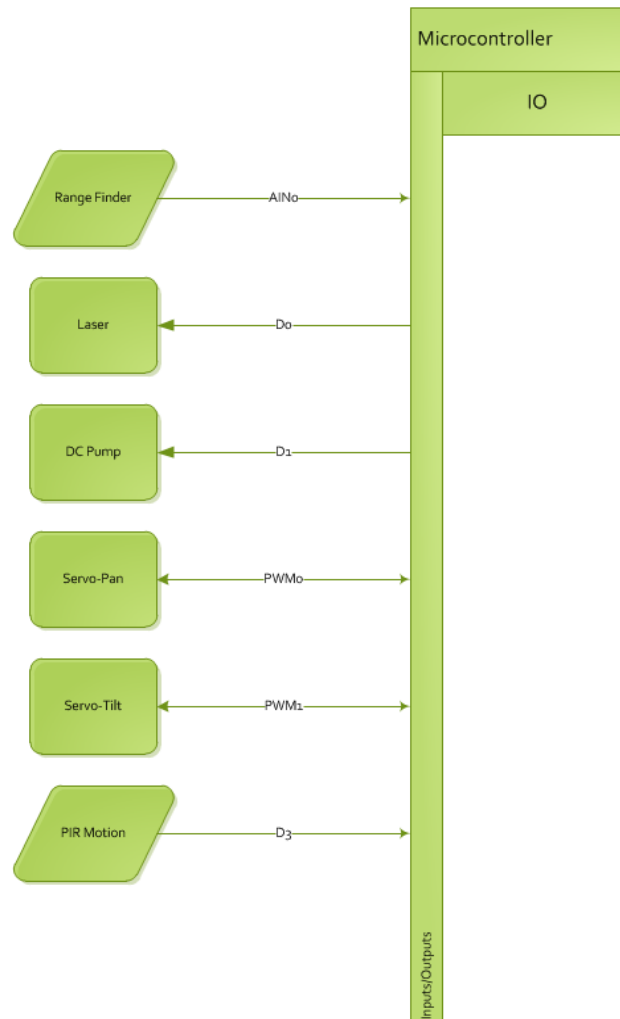


Figure 27: Targeting System Component Layout

### 12.3 Components: Software

The software requires a user interface where the operator can arm, calibrate, control, and view all of the data the turret is utilizing. The computations are done inside of the onboard microcontroller and communicating via RF wireless. Figure 28 shows the functionality of the system.

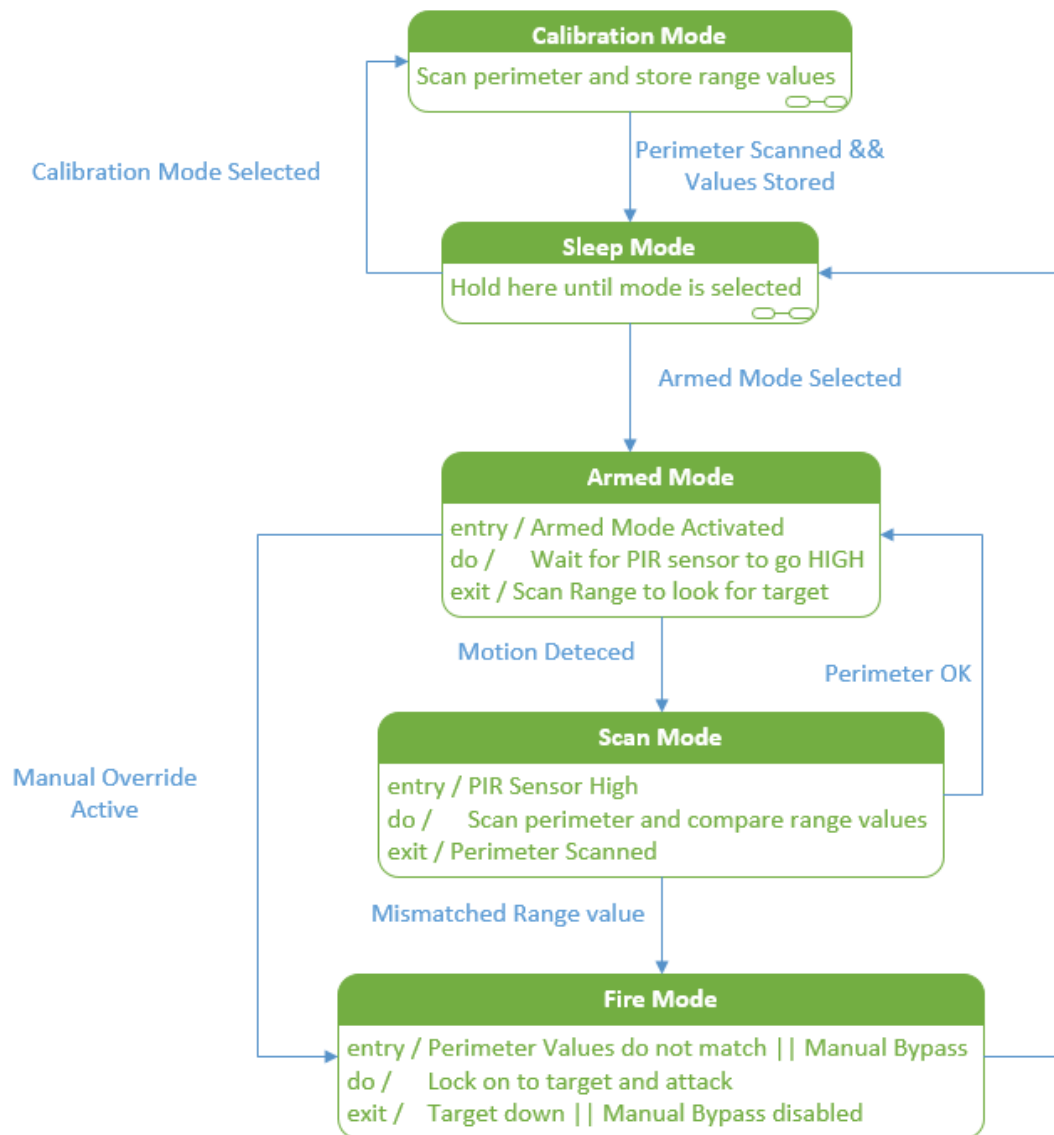


Figure 28: Target Acquisition FSM



## 12.4 How it works: Hardware

We will first discuss the overall functionality of the system. The turret has two ranges of motion (pan and tilt) and knows its position through the servo feedback. The passive infrared sensor and the ultrasonic sonar transducer are being used to determine the distance of the object as if something has moved overhead. The laser is just there for visual aiming purposes. The servos will move in a pattern and scan 10 zones. It will then record the range at those 10 zones and store it in an array. Once a motion is detected it will then perform another scan to determine if there are any differences in the previous ranges resulting in some type of object overhead the platform. If it detects any differences, it will lock in at that position and fire. The 10 zones will provide the turret with accuracy of up to two inches.

There are a total of three motors on the turret, two servos and one D.C. motor. Servo one is the “pan” servo and controls the turret moving 180° east to west. Servo two is the “tilt” servo and controls the turret moving 100° moving north to south. The servos are both driven through the PWM signals coming from the microcontroller and 5 volts from the power supply system. The DC motor is driven by a MOSFET and drives the gear to pump the springs for loading the missiles. As the motor spins; a gear system will pressurize the chamber until it reaches a point where it releases the missiles one by one. There is a reed switch inside the chamber on a thin piece of metal that reacts to the pressure inside the chamber. Once the pressure required to release the first missile is reached, the pressure causes the metal on the reed switch to close, closing the circuit indicating the first missile has been launched. This can be repeated up to three times. There is a laser attached to the top of the turret but it is currently only being utilized as a way to know where the turret is aiming.

The motion activation system is utilizing two main components, an ultrasonic sonar transducer and a passive infrared motion sensor (PIR). The main features of these will be discussed on the software side in the next section. The PIR sensor is being utilized as a motion sensing system. Once an object flies overhead the platform, it will then send a discrete “1” signal the microcontroller for the turret to perform a perimeter scan. Since this is a battery powered system, we did not want the turret to continually have to run perimeter scans and setting it to run at timed intervals also seemed unnecessary. This is why we decided to use a motion sensing system. The ultrasonic sonar transducer is what is calculating the range of the object from the platform. It sends out a sonar wave and based on its response time, it will determine up to 654 inches and outputs an analog signal. Figure 29 below displays the schematic being used for the platform turret control system.

There will also be a set of 12V RGB LED strips on the platform. This will be used as a visual indicator for when the system is arming. It will change colors depending on the function the microcontroller is performing at the time. The MOSFETS driving the RGB strips are triggered from optocouplers for any back feed protection and to prevent drawing too much current from the microcontroller pins.

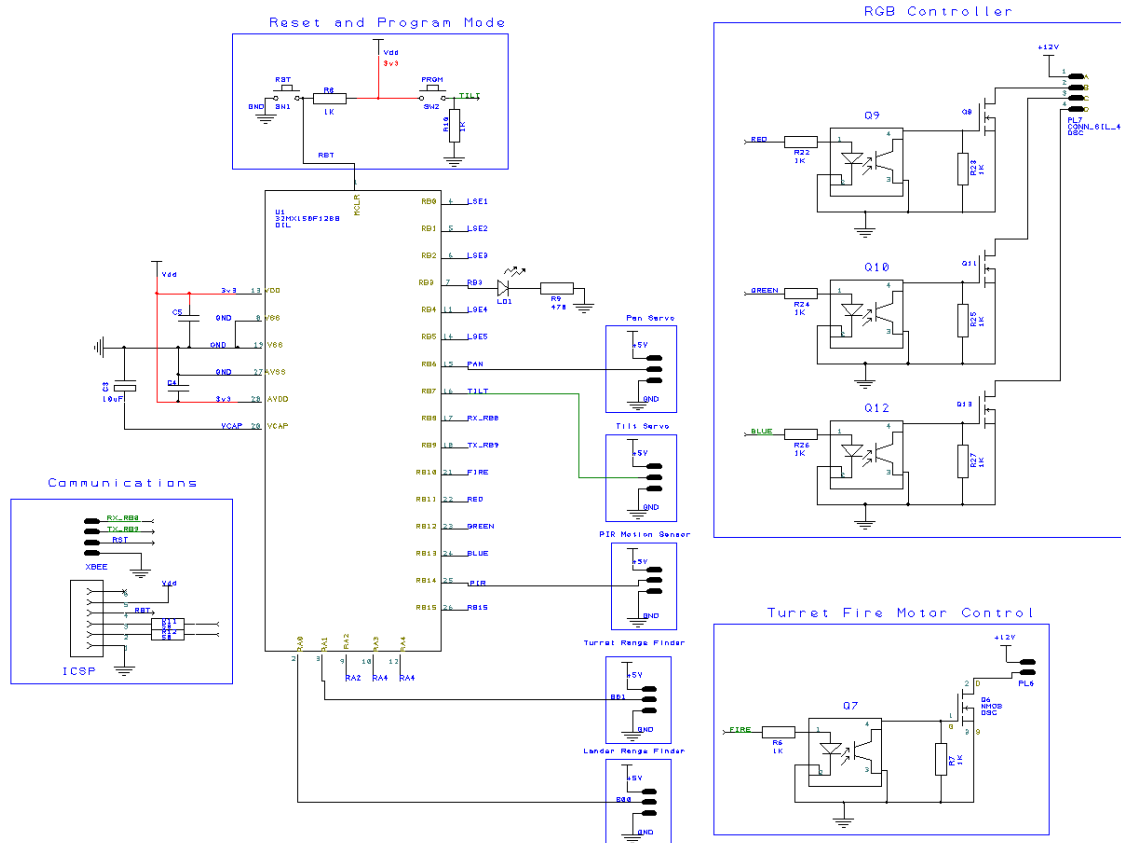


Figure 29: Platform Controller Turret Control System

## 12.5 How it works: Software

The software is monitoring the range, motion, motor positions, and calculating the algorithms. There is a user interface that will display all of the data and also allow full control of the system. When the program starts up, it will do a sweep of the perimeter and save all of the 10 range values into an array and keep them for later reference. At the moment, the perimeter is only dynamic in the East $\leftrightarrow$ West direction and the North $\leftrightarrow$ South direction is fixed aiming above the landing zone. Now that the turret has been calibrated, it will go into “sleep mode” waiting for further instructions. There are three ways to pull the turret out of sleep mode.

Method 1: Calibration Mode. This can be used if you want to change the position of the range and calibrate it to look at a different zone. This is simply activated by a button on the GUI and is using the same subroutine the startup routine uses.

Method 2: Autonomous Mode. You will need to arm this system on the GUI by pressing a button. Once the turret is armed, it will remain in “sleep mode” while waiting for a motion. If a motion is detected above the platform by the PIR sensor, it will then perform a perimeter scan of the 10 ranges. If at any point there is a difference from the calibrated value at that range within the threshold, it will then lock-on to the target in that zone and fire the missiles. It will then re-perform the

perimeter scan until all of the targets are eliminated and the values match up to the original range values.

Method 3: Manual Override. It will bypass all of the sensors and calibration modes and allow the user to move the turret to any position and fire at will. It does not override the calibrated ranges so the system is able to be put back into sleep mode and retain all of the original values. The GUI has sliders and buttons for maneuvering the turret and firing the missiles.

The GUI has basic features for the turret system. It will allow the user to adjust the position of the turret (the servos) using sliders that are mapped to output a PWM signal between the minimum servo angle to the maximum servo angle. We had to omit certain angles due to the potential issue of crashing into objects on board or the servos colliding with each other. Other features worth noting are being able to put the turret system into different modes. There is an individual button for each mode; Calibration, Armed, and Manual. Each of these modes has already been previously discussed. You will also be able to view what values are stored at the 10 zones of the perimeter and be able to compare the calibrated value versus the new value. If the turret has to fire during any of the zones, it will keep track of these zones and view them so the user can know where the attacker was coming in from. Due to inconsistencies with the range finder and real-world error, we had to implement a threshold. This just helps eliminate false triggers. Sonar can sometimes be affected by outside objects that aren't attackers. To help alleviate this, we can adjust the value that the turret triggers from to be within a range. This threshold sometimes needs to vary, so we implemented it onto the GUI. There is also a way to adjust how many zones are in the perimeter up to a user defined value. The last thing implemented was the scan speed of the perimeter. The faster the scan speed, the higher we had to raise the threshold due to movements and the range finder not being able to obtain accurate readings. We found that the most stable and accurate readings were about 400 milliseconds. Figure 30 displays the current GUI we are using for the development process (Hobye).

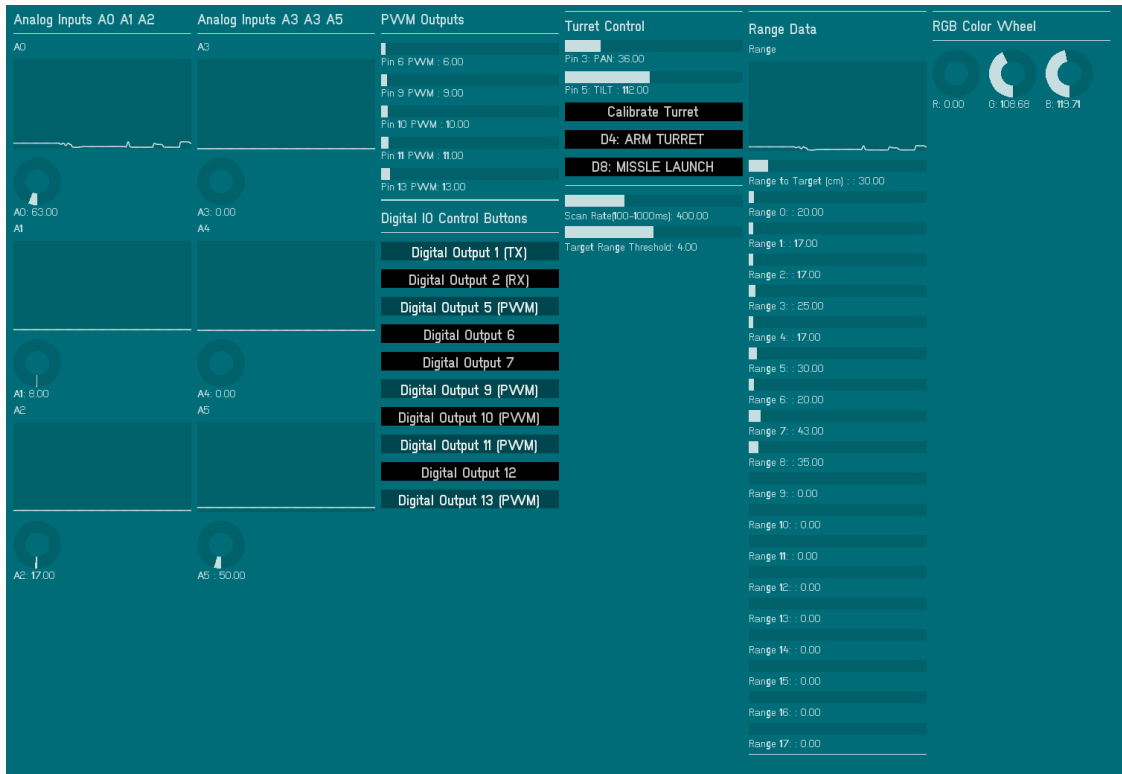


Figure 30: Development GUI for Platform Controls

## 13.0 Quad Copter Locking System

### 13.1 Goal

The mobile platform will be used to store the quadcopters during transport and while they are charging. The rough terrain that the mobile platform might encounter during this process can jar the quadcopters out of place and they can possibly fall off or become dislodged during the charging process. Both of these scenarios will jeopardize the critical components of this project, the quadcopters themselves, and the ability to increase mission time of the quadcopters.

In order to accomplish the task of locking the quadcopters in place we must abide by some restrictions. Some restrictions we must be wary of are; battery consumption, disengaging the locking mechanism, allowing the quad to take off from the platform, and size allocation. Of all the ideas that we came up with there are only two differences, using magnetic force or mechanical force to secure the quadcopters.

### 13.2 Design: Concept Research

One of our ideas includes using an electromagnet to lock the quad down to the mobile platform. We originally thought of this idea so that we can easily turn the magnet on and off depending on whether or not we want to lock the quad or let it loose to fly. One of the downfalls of this method would be the large power consumption that the electromagnet would consume. In order for the electromagnet to be used as a locking mechanism it would have to be on constantly and would subsequently consume an abundant amount of power. Another disadvantage of this method is that an electromagnet with a sufficient amount of windings on it to give us the force we need would get quite large and we are trying to minimize the amount of space this feature takes up.

With this in mind we came up with multiple methods to raise and lower the platform. One of the main ideas is creating two platforms that are hinged in the middle of the platform and connected to a servo with an eyehook shaft attachment. The attachment will allow the servo to raise and lower both platforms using one servo. This idea was one of the front runners because it minimizes the amount of extra components that we would need to use. It would also be simple to control; one of the down sides is that we would not be able to control the plates separately. This is a problem because if we wanted to only launch one quadcopter, the other quadcopter would be disengaged from the locking mechanism too which could cause the quad to be less stable on the platform.

One idea that we are considering to use is creating a separate platform underneath the quadcopter landing platform that will raise and lower permanent magnets. The platform will make contact with either a permanent magnet on the feet of the quadcopter or some ferrous metal on the feet of the quadcopter. We will be using this idea because it allows us to use fewer parts and take up fewer pins on our

microcontroller and save power. We contemplated a few ways to do this. One of the ones we were looking at pursuing was using four linear actuators at the posts that connected the landing platform to the rest of the platform. Then we would attach the linear actuators to a secondary platform with the magnets in place. In order to raise and lower the actuators we would use a 4 relay system for each actuator, giving us a total of 4 relay modules.

This will be our backup plan in case something goes wrong with our primary idea. The actuator that we will implement would be the one displayed in Figure 31. It has a five lb. thrust and a two inch stroke (Linear Actuator). This will allow us to lift the platform up and down two inches which will allow plenty of separation between the two magnets. Some of the reasons why this implementation would be successful include; the five lbs. of thrust will be more than enough to raise and lower the platform, each one pulls around 200 milliamps which is acceptable. One of the reasons this is our backup plan is because each one of these servos cost \$80.



**Figure 31: Linear Actuator (permission pending)**

One idea that we will be pursuing for the locking system will be using electromechanical solenoids attached to permanent magnets to lower the magnets when the quadcopters need to take off. The electromechanical solenoids have a plunger that stays in one position, either extended or contracted, and switches to the opposite state when an electric field is applied. We would want to get solenoids who's off state is when the plunger extended, therefore we can keep the quadcopters locked and only need to consume power for them to take off (Linear Solenoid Design Guide and Technical Information).

In order to accomplish this task we will be getting permanent magnets that have a hole cut out in the middle, and creating a harness that will allow us to attach the electromechanical solenoid to the magnet. One of the downfalls of this method is that it would require an electromechanical solenoid for each leg of the quadcopter which would give us eight total solenoids. This method also allows us to separate the locking of each quad, unlike some of the ideas using a single platform.

### 13.3 Components: Hardware

While researching magnets to use I came across two types of magnets, ceramic and Neodymium. Neodymium magnets are the strongest available, they can be 10 times stronger than the strongest ceramic magnet (KJMagnetics). Since the goal of this feature is to have the quadcopters be as stable as possible, we will need the stronger Neodymium magnets rather than the ceramic magnets.

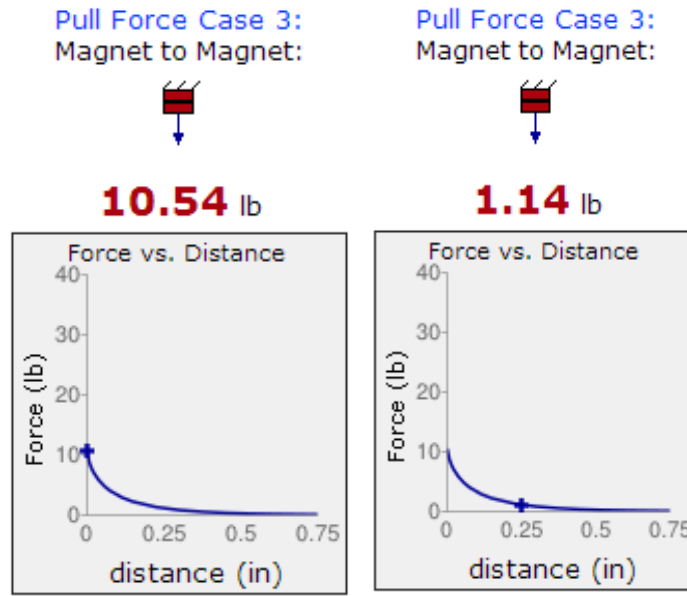
Neodymium magnets are given grade based on the material that it is made of, for instance N48. The higher the number after the letter indicates a stronger magnetic force. Some grades have a second letter after the number which will indicate a temperature rating; no letter indicates a standard temperature (KJMagnetics).

The magnets that we will use in this project will be Neodymium magnets of grade N48. The dimensions of these magnets are  $\frac{3}{4}$  in x  $\frac{1}{4}$  in x  $\frac{1}{8}$  in for the outer diameter, inner diameter, and thickness respectively. See Figure 32.



**Figure 32: Neodymium Magnet**

According to the N48 grading they can have a pull force of 10.54 lbs. and this force will be closer to 1.2 lbs. with the separation of  $\frac{1}{4}$  in from the Plexiglas platform that will be between the two magnets, see Figure 33. The approximate weight of the quadcopters will be around 3 lbs.; having four sets of magnets will give us a pull force of 6 lbs. which will keep our quad steady (KJMagnetics Magnetic Field Calculator).



**Figure 33: Magnetic Pull With and Without a Gap (reprinted with permission from KJMagnetics)**

For this feature we will use a pull electromechanical solenoid so that its rest position will have its plunger fully extended. This will allow us to attach a magnet to the plunger and be able to lock the quadcopters without using any power.

We tried to prototype this design using a smaller solenoid to test the pulling strength required to separate the two magnets which were put on opposite sides of  $\frac{1}{4}$  inch Plexiglas. We used the ROB-11015, see Figure 34, from Sparkfun. This part was supposed to have a pulling power of around two lbs. at 12 volts, however when we tried to separate the magnets they were not strong enough to overcome the magnetic force to move the plunger.



**Figure 34: Small Solenoid**



We will need a solenoid that will be able to separate the magnets, therefore being able to pull more than 1.5 lbs. The pulling power of a solenoid is determined by the stroke size and its power consumption. The stroke is the distance the plunger will travel when energized; the smaller size stroke, the more pull power. We will be using a push-pull solenoid from spark fun that has a 10mm stroke and can pull approximately two lbs. We will be applying 14 volts to this solenoid and it draws approximately two amps. The part that we will be using is a solenoid, product number ROB-10391, which is shown in Figure 35



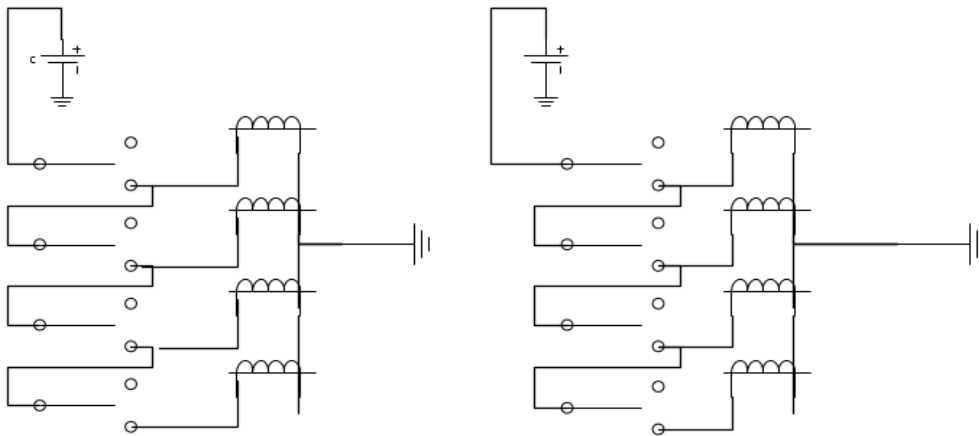
**Figure 35: Push-Pull Solenoid (reprinted with permission from Sparkfun)**

### 13.4 Design

The quadcopter locking system will be controlled using the Pic32 microcontroller that was selected to run the mobile platform. The solenoids cannot be powered from the microcontroller, therefore we will activate the solenoids using a 30 volt 10 amp single pole double throw relay module as shown is Figure 36.

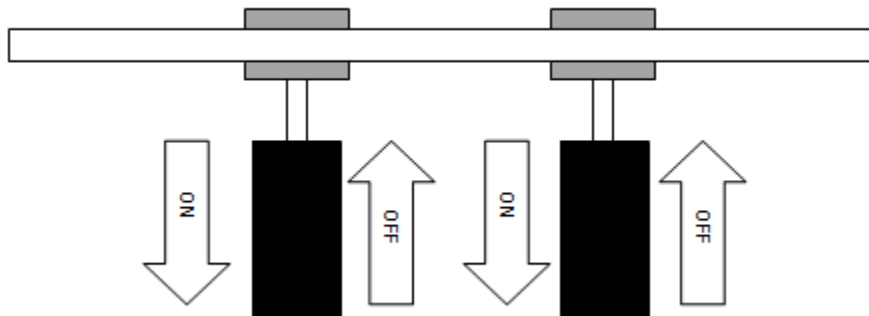
The relays are wired so that each set will be activated at the same time. This is important because if any of the solenoids were not energized while the quadcopter was trying to take off, the quadcopter would have to generate more lift, which means the motors would have to draw more current, shortening the flight time on the charge of the battery. If more than one solenoid was not activated the quadcopter would not even be able to overcome the magnetic force. They relay modules wired this way would not have this issue. We also isolated the solenoids systems for each quadcopter as to have independent processes so that we can launch one quadcopter while locking the other.

In order to power the solenoids we will use the main battery. We will have to do this because each solenoid can pull up to two amps. This is a problem because the power supply design for distributing power to the different subsystem uses switching regulators that can only handle three amps.



**Figure 36: Relay and Solenoid Configuration**

The locking system relay configuration is displayed in Figure 37. The top magnet will be permanently attached to the bottom of the quadcopter. The bottom magnets will have to be attached to the solenoids using a separate mount so that the magnet is secured. When the solenoids are energized they will retract, creating a larger gap. As demonstrated in Figure 33, the larger the gap between the two magnets the weaker force they exert on each other. This decrease in the pull force between the magnets will allow our quadcopter to take off without having to increase the output from the motors, saving the charge of the battery.



**Figure 37 : Side View Model of the Quad Locking System**

Once the quadcopter has cleared the landing platform and the locking system, the microcontroller will turn off the relays and return the solenoids to their non-energized state. Since the solenoids and magnets will be locked in place, any adjustments must be made by the quadcopter.

## 14.0 Flight Controller Research

### 14.1 Goals

For the purposes of this project we need an open source flight controller with preloaded firmware, it needs to be able to have gyro stabilization and also self-leveling. One of the main factors that we considered for the flight controller of the quadcopter was cost. Therefore we came up with a few different flight controllers.

### 14.2 Hardware: Flight Controller Research

The first flight controller board we looked at was the UDB5- PIC UAV development board, this flight controller was under consideration because it fits the criteria of low cost and had the gyro stabilization and self-leveling. We also looked at this board because it was based on PIC processors. This flight controller is based on the PIC33F from microchip. This processor has 85 programmable digital IO pins. This processor is based on the modified Harvard architecture, runs at 80 MHz, and has 8 PWM channels (UBD5- Pic UAV).

Another flight controller was the Hobbyking KK2.0 flight controller board, this particular flight controller was under consideration because it is very low cost compared to most of the flight controllers. The main issue with this controller is that the schematic is not open source and we could not find the firmware to program it on a PCB (KK2 Flight Controller Board).

The series of flight controllers that we have decided are the APM flight controller, either the 2.5 or 2.6 version. This flight controller is completely open source and runs using the APM firmware that we can use to program the PCB. The main difference between the 2.5 and 2.6 versions is that the 2.5 has the compass built onto the board and with the 2.6, the compass is connected externally along with the GPS (APM 2.6).

Having the compass built in onto the board was very hazardous to the navigation and flight controlling systems of the quadcopter. The electronic speed controllers were causing a high amount of magnetic interference with the compass being so close. Also having the compass connected to the ground plane of the rest of the PCB also throws off the magnetic orientation of the compass. Having these magnetic interferences affecting the compass it will alter the reading that the flight controller is taking in from the compass. These inaccurate readings can cause the quadcopter to give false representations of its position and will affect the flight controls.

## 15.0 Hardware: Flight Controller Design

### 15.1 Gyroscope and Accelerometer

The controller design incorporates a six-axis MEMS device with a gyroscope and accelerometer built in; the part is called MPU-6000. Having a gyroscope and accelerometer are essential for any multi or single rotor copter. The three axis gyroscope allows the controller to know how it is rotating in an X, Y, and Z orientation. One of the downfalls of the gyroscope is that each calculation has a minimal amount of error in it, and the subsequent readings are calculated by integrating over the time period, therefore over time the measurements become more inaccurate. The accelerometer will give the quadcopter the orientation of the quad with relation to the Earth's surface. As does the gyroscope, the accelerometer only has one downfall. The negative effects of the accelerometer are that it cannot differentiate between the acceleration due to Earth's gravity and that which is being applied to the quadcopter and accelerometers are very sensitive to vibrations. The three axis accelerometer in combination with the three axis gyroscope allows us to know exactly where the quadcopter is heading and how its motion is changing in each of the three directions. Having both of these components calibrate each other to get the correct readings.

The 6 axis MEMS device that we will use for the gyroscope and accelerometer is the MPU-6000. This device will run on 3.4 volts and has a logic level of 3 volts. During normal operation, using the gyroscope and the accelerometer at the same time, the MEMS device will draw around 3.9 mA's of current giving us a power consumption of 13.2 mW.

The MPU-6000 can communicate with our Atmega 2560 via SPI or I2C, for our board we will have the MPU communicating using SPI because with SPI it is much easier to control the other devices with the master input slave output (MISO) and master output slave input (MOSI). Using SPI allows us to access any auxiliary sensors that will be connected to the MPU-6000 such as a magnetometer or compass directly as displayed in Figure 38 from the MPU-6000 data sheet. The MPU will act as a slave for the processor and a master for the auxiliary components. We will also be configuring the SPI connection between the MPU, the Atmega 2560, the altimeter and barometer, and other auxiliary sensors in a manner described in Figure 38 (MPU 6000 Data Sheet).

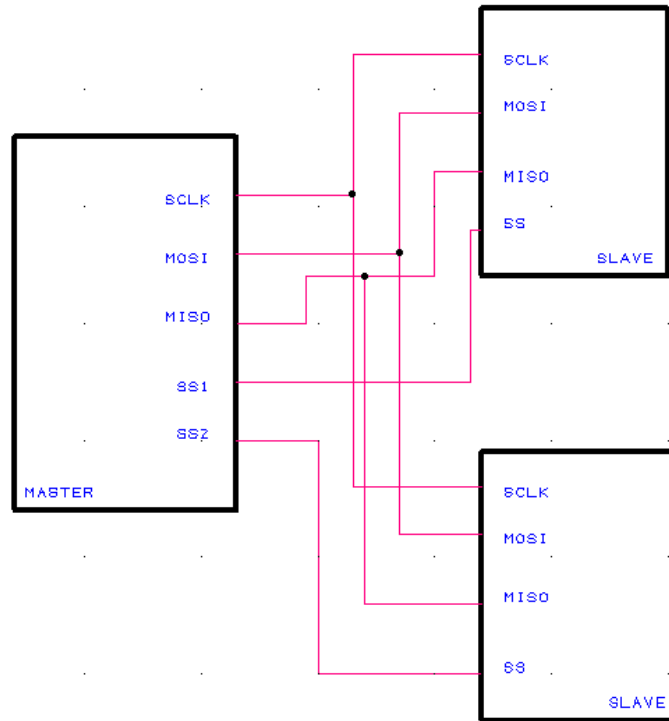


Figure 38: SPI Configuration

## 15.2 Digital Compass

The next component that we will be adding on to the flight controller is a three axis digital compass. The accelerometer and the gyroscope give us the measurements needed to adjust the pitch and roll of the quadcopter; therefore we are missing a component to give us the yaw. The yaw is perpendicular to the direction of Earth's gravity, see Figure 39, and thus cannot use the accelerometer alone.

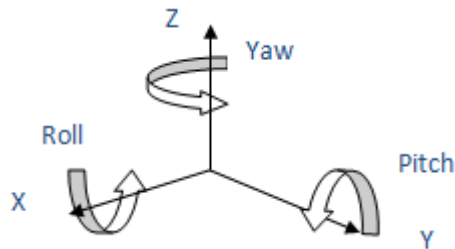
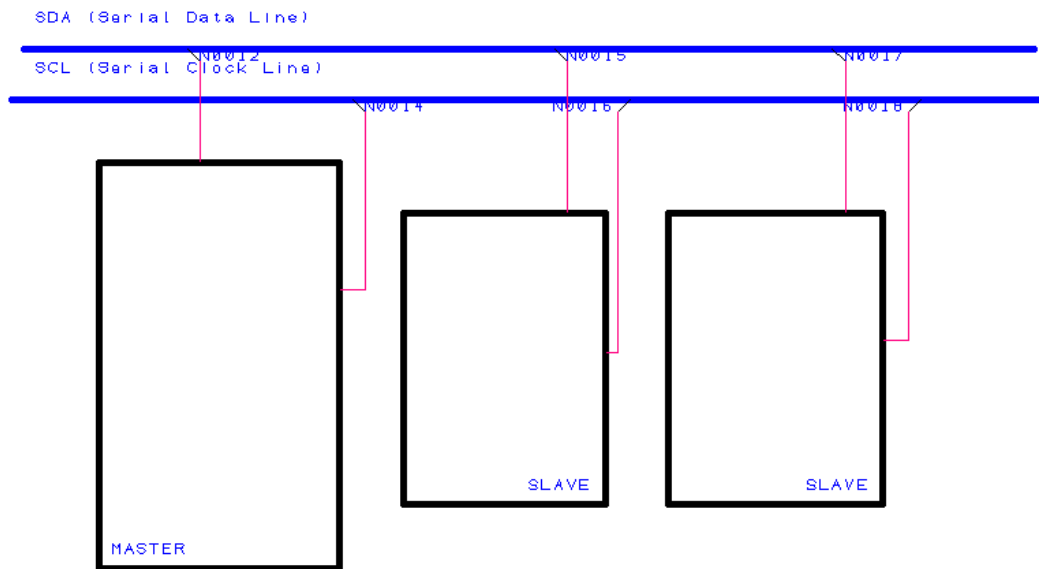


Figure 39: Definitions of roll, pitch, and yaw

The digital compasses use magneto-resistive sensors to calculate the magnetic orientation of the compass. The magneto-resistive sensors are made of a nickel-iron film that is patterned to act as a resistive element. The magneto-resistive sensors work by passing a current through a ferromagnetic material and when an external magnetic field is applied it changes the current flow through the material. It then takes the data and uses an application-specific integrated circuit (ASIC) chip that amplifies the signal and then the chip turns the analog data into digital data.

The digital compass that we will be using in our flight controller design is the HMC5883L. The electrical characteristics of this part include having a supply voltage of three volts and a current draw of approximately 100  $\mu$ amps. This component has a maximum output rate of 160 Hz. Since the compass is very sensitive to a changing magnetic field it is important to have this part away from any ferrous metal including most of the components on our board. Therefore we must have this part be off of the main board that will be our flight controller. This part can only communicate using I2C, which setup is displayed in Figure 40, therefore we will also need to incorporate an I2C bus voltage level translator so that we do not fry the compass while interfacing with the Atmega 2560. The part that we will be using for this is the PCA9306DC. The HMC588L has a range from plus or minus eight gauss, and a resolution of two milli-gauss (HMC588L Data Sheet).



**Figure 40: I2C Setup**

### 15.3 Processor

The APM 2.6 controller is run using an Atmega 2560 which has an open source schematic that we can use, the controller design will allow us to put a 3-axis gyro, an accelerometer and magnetometer, and a barometer included on the board. It also will allow us to use an off board GPS and compass system, we must put these off the board for the purposes stated earlier. The Atmega chip has multiple RX and TX pins which will allow us to use XBEE's for our telemetry.

The Atmega 2560 will be the brain of our flight controller. The Atmega 2560 is an eight bit AVR architecture which is Atmel's own architecture that is based on the Harvard architecture. The Atmega 2560 runs on five volt logic while most of the sensors that we will be using use I2C or SPI connections which all run on 3.3 volt logic, therefore we will have to use a bi-directional level translator to convert between the 3.3 volt and five volt logic. The Atmega 2560 specs are located in Table

4: Atmega Specs. One of the main reasons that we will be going with the Atmega 2560 is because of the amount of GPIO and PWM channels that it has. The amount of GPIO is important because of all of the sensors that we will be implementing onto the flight controller. The PWM channels are essential for the quadcopter application because it allows us to control the motors, the additional channels also will allow us to put in the two servos that we will need in order to put a turret system on the quadcopter (Atmega 2560 Data Sheet).

MCU	Speed	RAM	GPIO	PWM channels	UART	Flash	ADC channels
Atmega 2560	16MHz	8KB	86	12	4	256KB	16

Table 4: Atmega Specs

## 15.4 On Board Programming

The AMP 2.6 comes with another Atmega chip on it, specifically the ATMEGA32U2-MU. This chip is used for on board programming of the Atmega 2560 via USB. We are not going to add this chip onto our design to save space on our PCB. Instead of the separate Atmega chip we will program the Atmega 2560 by breaking out the UART, reset, five volt, and ground and program the chip from an Arduino Uno.

## 15.5 Data Logging

The flight controller design will also include an AT45DB161D-MU DataFlash chip that will be used as separate flash memory. Since we plan on collecting sensor data, we will need the DataFlash chip for data logging purposes. The data that we would like to log on the DataFlash includes altitude, GPS coordinates and flight path details.

DataFlash is different from the traditional Flash ROM. Traditional ROM uses multiple address lines and a parallel interface. The DataFlash uses a serial interface and accesses the data sequentially. The benefits of accessing the data sequentially include; reducing the active pin count, reducing the package size, and minimizing the switching noise.

The DataFlash chip will be interfacing with the Atmega 2560 using SPI and is capable of operating up to 66 MHz. Since it communicates through SPI, it is activated using a chip select pin that will be sent from the microcontroller. This chip runs off of a three volt power supply and consumes seven milliamps when on and 25 micro amps while in standby mode (AT45DB161D-MU Data Sheet).

## 15.6 Altimeter

The flight controller design will need an altimeter. The altimeter that we are using is technically called a barometric pressure sensor, but it serves the same purpose. The sensor takes in the changing barometric pressure and from there can determine the change in altitude.

There are two barometric pressure sensors that are under consideration for this project. The first one is the MPL1151a; it has operating characteristics of a supply voltage of 3.3 volts and 5  $\mu$ amps. This sensor has a range of 50 to 115 kPa which translates to an altitude range of 16,000 ft. above sea level down to sea level. This part has an accuracy of plus or minus 1 kPa, with a resolution of two significant figures.

The second part that we will be looking at is the MS5611-01BA03 barometric pressure sensor. This sensor, as do most of the sensors that we use, runs on 3.3 volt logic. The operating characteristics of the sensor are perfect for our low power requirements; it runs on 3.6V at 1 $\mu$ A. This chip has an operating range of 10 mbar to 1200 mbar which translates to about sea level up to 85,000 ft. above sea level, and its accuracy is plus or minus 1.5 mbar with a resolution of 4 significant figures.

One of the downfalls of the MPL1151a is that it requires additional parts to operate, such as some external capacitors. It also has a lower resolution and a lower accuracy than our second sensor. The first sensor also consumes 5 times as much power as the MS5611. Because of these specifications we will be designing our flight controller using the MS5611.

The MS5611 typically uses either SPI or I2C for communication between the sensor and the Atmega 2560, for our design we will be using SPI. For the communication we will be running the MOSI and MISO lines through the MPU-6000. The SPI configuration will allow the MS5611 to interface its output up to 20MHz. The chip select line for the SPI configuration of the MS5611 is active low (MS5611).

## 15.7 Turret

The turret system that we will have to defend the quadcopter will be controlled using the Atmega 2560. The only thing that we will need to use to control the turret is to use two PWM outputs from the microcontroller to control the two servos, one for panning the turret and one to tilt the turret.

## 15.8 Landing System

In order for our landing system to work we will need to put some photocells onto the quadcopter that will need to communicate with the flight controller. In order to do this we will just need to hook up the four photocells to four of the analog IO of the microcontroller.



## 15.9 Telemetry

For our quadcopter design one of the most important features is being able to communicate with our mobile platform and to be able to transmit the data and the video feed that we will be collecting. In order to accomplish this goal, we will set up a telemetry system that will be able to send the data long distances to our mobile platform.

After determining the goals we needed for the telemetry system we decided the major factors to take into account are as follows: low power, the ability to transmit/receive data packets over at least one mile, and be able to transmit large packets of data accurately.

From our research there are mainly two operating frequencies for most telemetry devices, 900 MHz and 2.4 GHz. One of the benefits of using the 900 MHz transmitting frequency is that using the same transmission power it can transmit much farther than the 2.4 GHz version. This is due to the path loss from the attenuating signal, where the path loss is directly proportional to frequency, as modeled in Equation 1, therefore the higher frequency of 2.4GHz will have higher path loss and will limit its effective transmit distance (Friis Equation). The lower frequency of 900 MHz also gives an added advantage of greater penetration distance, meaning the lower frequency and the longer wavelength the 900 MHz signal can penetrate walls much easier.

$$P_r = \frac{P_t \lambda^2}{(4\pi R)^2} \text{ Watts}$$

### Equation 1: Received Power

However for the purposes of our project neither of these factors will require us to use 900 MHz; we plan on having a relatively clear transmission path, no walls, and a modest transmitting range of one mile. Therefore the added benefits of the 900 MHz will not actually be very beneficial to us. A 2.4GHz frequency will give us the one mile line of sight range to meet our specifications; therefore the tradeoffs between both options are not very important.

From our research we have narrowed down the wide market of telemetry sets to two different kinds, XBee-Pro modules and a 900 MHz telemetry set that is available from the company that we are using for our flight controller, Figure 41 shows the telemetry set from 3D Robotics.



**Figure 41: 3D Robotics Telemetry System (reprinted with permission from 3D Robotics)**

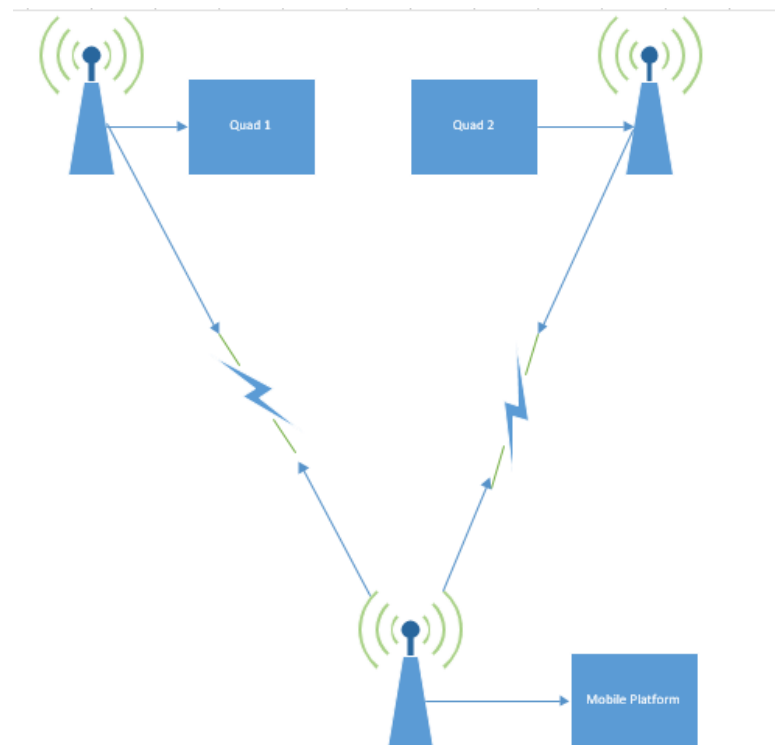
What we are looking at in the differences of each set are transmission power, price, transmission rate, the communications protocol, and range. The XBee-Pro modules have a transmitting power of approximately 18 dBm which translates to roughly 63mW and the 900MHz model has a transmitting power of approximately 20 dBm which is around 100mW. There is clearly a very large gap in the power required to transmit our data with each module, one of our biggest limitations is our battery power on the quadcopter and maximizing efficiency (Xbee Pro Data Sheet).

As for our other specifications, both modules have an effective line of sight range of 1 mile and both have a transmission data rate of 205 kbps. Another limiting factor for these two modules is the price, a set of the transmitter and receiver of the 900 MHz set is \$100, while purchasing two XBee-Pros would cost around \$60. One of the great things about the XBee-Pro is that it uses a standard IEEE 802.15.4 communication protocol, while the 900 MHz set uses its own protocol called MAVlink. Therefore it is a logical solution to choose the XBee-Pro modules. The full list of options we considered and the criteria that we used are summarized in Table 5. (3DR Radio Telemetry).

Protocol	Product	Frequency	Transmit Power	Range	Data Rate	Price
<b>IEEE 802.11</b>	Xbee Wi-Fi	2.4 Hz	16 dBm (40 mW)	N/A	1-72 Mbps	\$35
<b>IEEE 802.15.4</b>	Xbee Pro	2.4 GHz	18 dBm (64 mW)	1 Mile	250 Kbps	\$32
<b>Zig bee Pro</b>	Xbee ZB SMT	2.4 GHz	8 dBm (6 mW)	4000 ft.	250 Kbps	\$28
<b>MAVlink</b>	3D Robotics Module	915 MHz	20 dBm (100 mW)	1 Mile	250 Kbps	\$30
<b>ZigBee Pro</b>	Xbee Pro ZB	2.4 GHz	18 dBm (63 mW)	2 Miles	250 Kbps	\$29
<b>DigiMesh</b>	Xbee-Pro 900HP	900 MHz	24 dBm (251 mW)	9 Miles	200 Kbps	\$39

**Table 5 : Telemetry Comparison Chart**

Now that we have determined which telemetry system we are going to use, we now have to figure out how to implement them onto both the flight controller of our quadcopter as well as on to the mobile platform. The way we will set up the Xbee modules will be as multipoint communication. Simply meaning that we will be able to have multiple XBee's being able to communicate with each other, see the diagram in Figure 42.



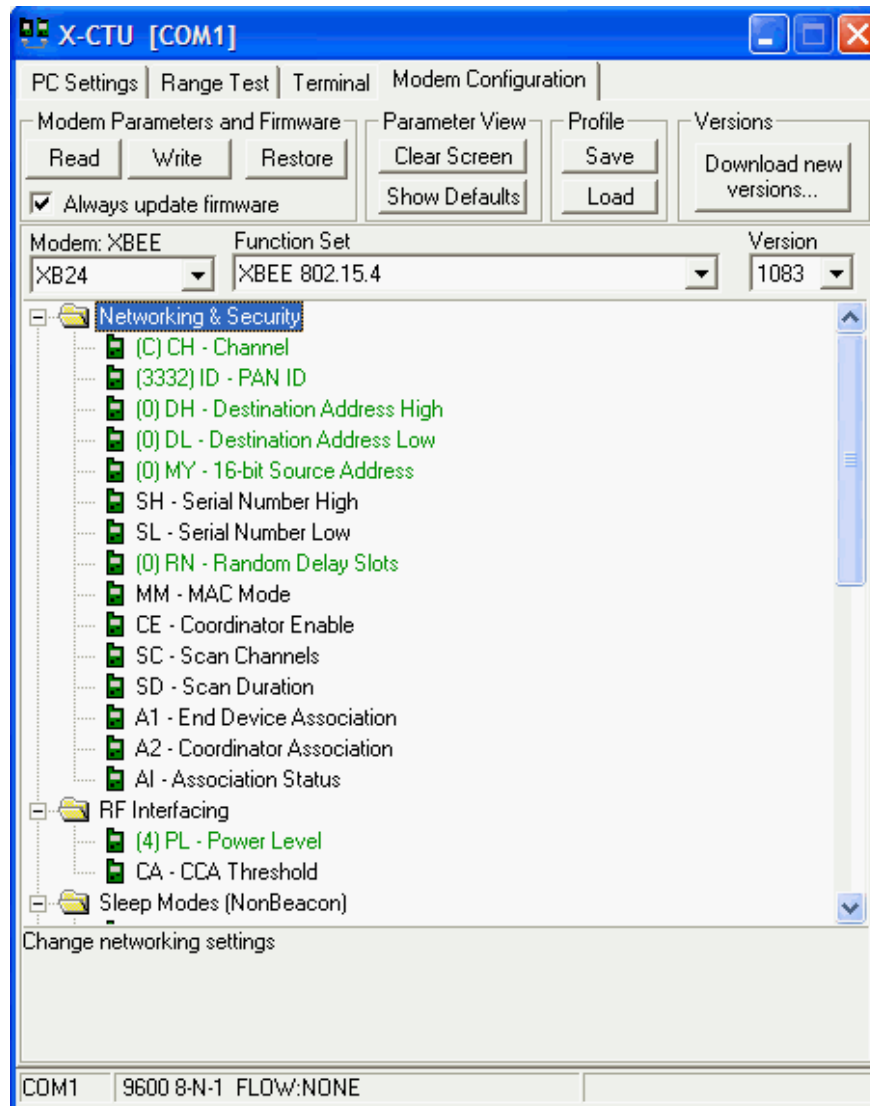
**Figure 42: Telemetry Diagram**

The Xbee modules really only need to be hooked up to the power and ground of the controller they are designated for and to be hooked into the respective transmit and receive pins. The Xbee chips that we will be using are pictured in Figure 43.



**Figure 43: Xbee Pro Modules (permission pending)**

In addition to the ease of installation of the Xbee hardware, they are also extremely easy to program and configure. In order to configure the Xbees we must use a program called X-CTU which is free and easy to use. In Figure 44 is the modem configuration tab of the X-CTU software. In this terminal there are a few numbers to change around to get the Xbees to communicate with each other. The first number that must be changed is the PAN ID number. This number is used to set up the XBee network and all of the XBees that need to communicate with each other need to have the same PAN ID number. The next numbers of concern are the DH and DL numbers which are the destination high and destination low numbers. The destination high number and low numbers are on the back of each XBee. There are two eight bit numbers; one on top of the other. The one on top is the upper ID and the one on the bottom is the lower ID. When programming one Xbee you put the upper and lower ID of the other XBee you want to communicate with. So if you have Xbee 1 and 2 and you are programming Xbee 1, then you put the upper and lower ID of Xbee 2.



**Figure 44: X-CTU Modem configuration (permission pending)**

There are also options that are not in this screenshot that list the pin outs of the XBee's input pins. To send the data that we want we will have to send the data through the flight controller processor or directly from the sensors that we will be working with to the various pins.

## 15.10 Power Module

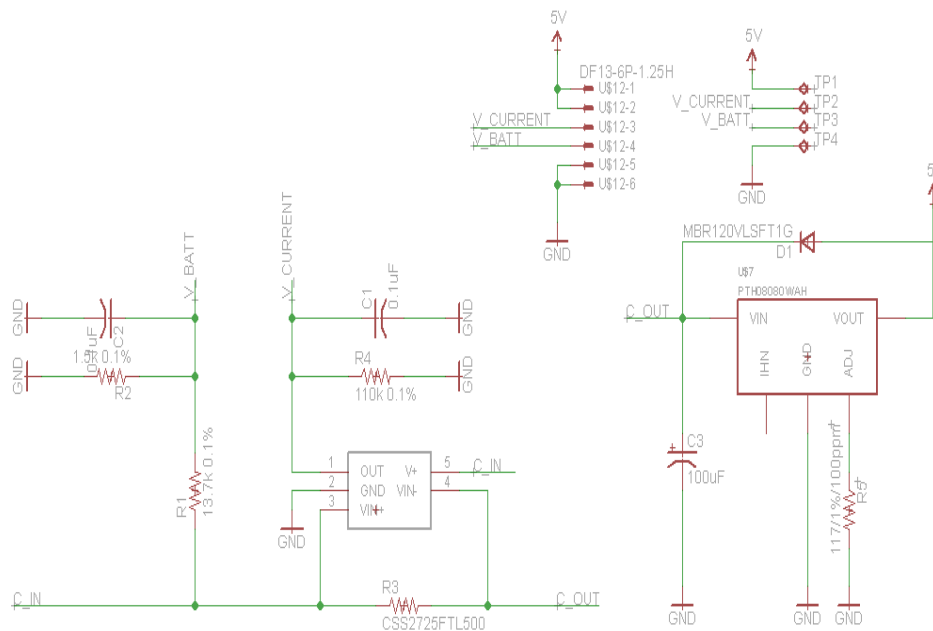
Now that we have everything that we will need to go into the flight controller, the next step is to choose an acceptable power supply that will be able to run our flight controller. We have a few options to look into for powering the flight controller. One method would be to have its own dedicated battery separate from that of the motors. This method would allow us to increase our battery life by not draining the main battery which is being drained significantly from the motors. To carry out this method we would need a battery that can supply the processor with five volts and can supply

a maximum of two amps. Using this method would be more expensive and would also take away from the available payload.

The other option that we can pursue is to power the flight controller from the main battery. Some of the pros of doing this are that it will reduce the weight of the overall payload of the copter since we will not be adding another battery, and it will also be cheaper. Since the adapters are not a battery, they will not go bad or need replacement. The battery that we are using to power the motors will be around 11 volts and we will have to step down the voltage. We could do this by creating a connector from the battery that allows us to break out the wires and connecting it to a simple switching regulator and then to our board. Alternatively, I have found a power module online from 3D Robotics that is designed to deliver power from a 4S LiPo battery, which will be used to power the motors to our board. It has an output of 5.3 volts and a max of 2.25 amp draw. Another use for this power module is that it also allows us to measure the current consumption and battery voltage through a six pin cable. The part is shown in Figure 45 (APM Power Module).



**Figure 45: 3D Robotics Power Module (reprinted with permission from 3D Robotics)**



**Figure 46: Schematic of the Power Module (reprinted with permission from 3D Robotics)**

The schematic in Figure 46 is the schematic drawing provided from 3D Robotics of the power module that we will be using to power the flight controller. The module breaks out the battery and runs the battery through a current sensor and then through the PTH08080WAH DC to DC regulator which steps the voltage down to five volts due to the value of the resistor connected to the ADJ pin. The switching regulator that they use is called a wide-input adjustable switching regulator. It allows for a maximum current draw of 2.25 amps and can supply an output voltage between 5.5 volts and .9 volts. This regulator operates from 4.5 volts to 18 volts input and boasts an efficiency of up to 93%. This efficiency rating alleviates some of the concern of running the flight controller board from the main battery, making this option a much more viable option than some of the others.

### 15.11 Speed controllers

The flight controller is also responsible for controlling the speed and direction of the motors for flight. In order to control the motors, there must be a bridge between motors and the flight controller called the electronic speed controller (ESC). This process of controlling the electronic speed controller is very similar to controlling a servo. The flight controller will be using the pulse width modulation outputs of the controller to send a pulse that will turn the speed controller on and send the correct power to the motors to adjust its speed and direction. Basically they turn the pulse input into a smooth linear response by opening and shutting MOSFET circuits that will throttle the motors, see Figure 47. The flight controller will send out a square wave that will have a certain frequency and duty cycle that the speed controller will interpret and subsequently open or close channels of MOSFETS to match the desired command.

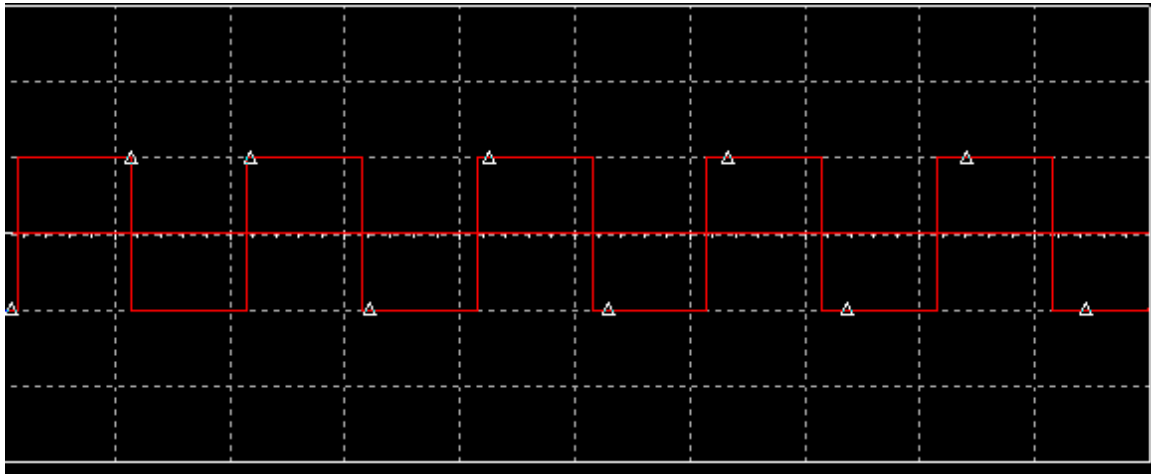
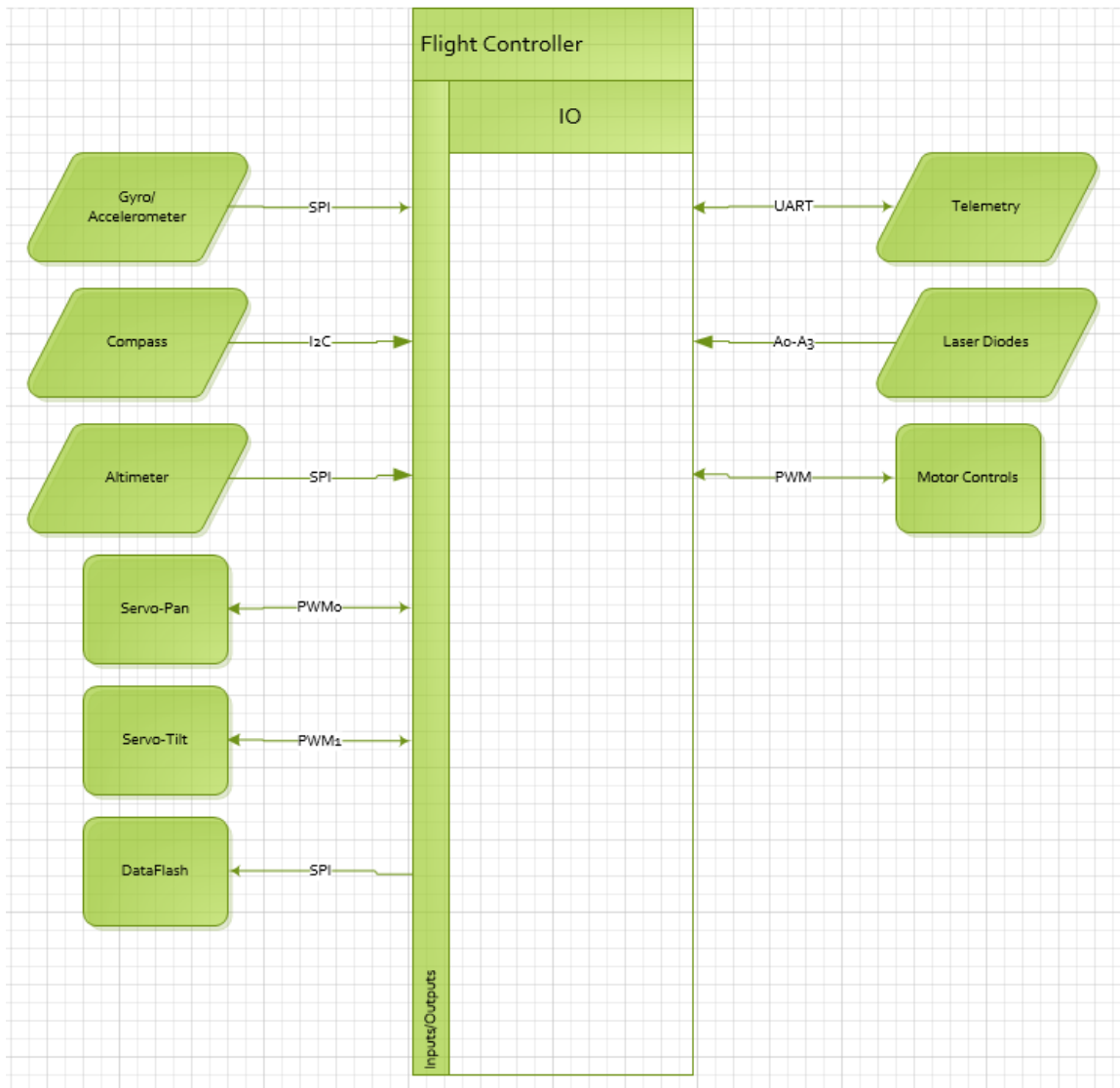


Figure 47: Output of Flight Controller

### 15.12 Layout

The flight controller layout is shown in Figure 48; it shows all of the components that will be run by the Atmega 2560. As depicted in the diagram, the flight controller will

be responsible for taking in data from our gyroscope, compass, altimeter, and telemetry system. Then the flight controller will send commands to the pan and tilt servos, the motor controls, the photocells, and send data back and forth from the telemetry system.



**Figure 48: Layout of the Flight Controller Components**



## 16.0 Quadcopter Power Subsystem

### 16.1 Power Sources and Storage

For the purposes of electrically powering a multi-rotor aircraft that could possibly be flying at altitudes over 100ft, a portable power source would be the only legitimate option for powering our quadcopters. There are a couple different options for portable power onboard an aircraft including, but not limited to: batteries, supercapacitors, and solar power.

We looked into solar-powered quadcopters and discovered that a team of master's students from the Queen Mary, University of London had been working on a solar-powered quadcopter (Borgobello). Pictured below, in Figure 49, is the world's first solar powered multi-rotor copter according to gizmag.com. From the article on gizmag.com we came to the realization that a solar-powered quadcopter would most likely require a custom-built frame to be extremely lightweight and it would also have a low limited payload weight. This was not ideal for our use of a multi-rotor aircraft.



**Figure 49: A solar-powered quadcopter built by master's students at Queen Mary University of London Permission Pending.**

Ultracapacitors and supercapacitors are up and coming technologies that we looked into as a source of stored energy to power our quadcopters. Unfortunately, the cost of these devices at the energy density we require for our aircraft was out of our price range for this project. Even if we could have afforded supercapacitors of this nature, they would not have been able to sustain flight for a long enough period of time to accomplish the goals we set for this project.

Ultracapacitors, however, are ideal for backup power, which we could use in future designs as an emergency reserve in case the voltage of the power source drops too

much for sustained flight (Buchmann). This would allow the quadcopter to return to the mobile landing platform safely. Below, in Figure 50 courtesy of Maxwell Technologies, is an Ultra capacitor produced by Maxwell Technologies that is rated at 2.7V and 3000 Farads (Maxwell Technologies).



**Figure 50: Maxwell Technologies' Ultracapacitor with a rated voltage of 2.7V and a 3000F capacitance (permission pending)**

## 16.2 Battery Technologies

Batteries looked like the most cost effective and efficient way to power our aircraft in order to accomplish our goals and achieve the flight time we desired. There are many different types of batteries and battery technologies and they come in a variety of different sizes, shapes, weights, voltages, capacities, and energy densities. All of these factors have to be considered in order to find the proper batteries for our applications.

Lead-acid batteries are used in most uninterruptable power supplies, automobiles, and other vehicles. They have been used commercially for since the 19<sup>th</sup> century and are still used heavily today because of their dependability and inexpensive cost-per-watt (Buchmann). However, there are several disadvantages to the lead acid batteries that would cause issues when using them as a power source for our aircraft.

Lead-acid batteries are very heavy compared to other battery options which would make getting off the ground a challenge without using massive motors and propellers. They are also fragile when it comes to wear-down from discharging and charging constantly; their capacities begin to shrink if discharged too low. These reasons deterred us from using lead acid batteries as our batteries of choice for our aircraft. However, we chose to use lead acid batteries in our mobile landing platform as discussed previously. Below in Figure 51, is a picture of a lead-acid battery we removed from a motorcycle.



**Figure 51: A lead-acid battery removed from a motorcycle**

Alkaline batteries are used in many consumer electronics such as television remotes, flashlights, calculators, wireless mice and keyboards, and just about anything that says batteries not included on the packaging. They are relatively inexpensive and are maintenance free. The most common are one-time use batteries, but they come in a rechargeable form as well. One-time use batteries will drive up the operating cost of our quadcopters and will not allow us to harness the energy of the sun in order to refuel our aircraft. Consumer rechargeable alkaline batteries have been around for a little over 20 years as an alternative to disposable batteries, but have a high cost per cycle (Buchmann). Every time a rechargeable alkaline battery is fully discharged and recharged (one cycle), the capacity is reduced permanently. This means we would not get many flights out of a set of these batteries. They are also limited in the amount of current they can supply and store. This makes these a poor choice for our aircraft, since the energy required to generate lift using electric motors is much higher than the limits of alkaline batteries.

Nickel based batteries have been used by hobbyists for several decades, so we decided to look into the possibility of using these batteries for our quadcopters. The two most commonly used nickel-based batteries for hobby applications are nickel-cadmium (NiCad) and, its close cousin, nickel-metal-hydride.

Nickel-cadmium batteries have been around since the 1930's and have a wide variety of size and performance options (Buchmann). They can be stored in a discharge state, have a high number of charge/discharge cycles (over 1,000 with proper maintenance), and they can be charged quickly with simple charging methods. They are also economically priced and have the lowest cost per cycle of all of the nickel-based battery technologies (Buchmann).

These are all great reasons for choosing NiCad's as our battery of choice. There are, however, a few drawbacks to using NiCad's that deterred us from going with NiCad's for our design. The main one being that they have a relatively low specific energy, or energy density, when compared to newer battery technologies and systems (Buchmann). This means we would require more batteries, which equals

more weight, in order to achieve the flight times we desire. This would also decrease the payload capacity of our aircraft, so we chose not to go with the time-tested nickel-cadmium batteries. Figure 52, below, features a NiCad battery from one of our 18V Dewalt cordless electric drills. These are used widely in the cordless drill industry, but are slowly losing market share to the lithium ion (Li-ion) batteries. We will speak more about Li-ion batteries later.



**Figure 52: A NiCad battery that we removed from an 18V Dewalt cordless drill**

Nickel-metal-hydride (NiMH) battery technology really took-off in the 1980's with the discovery of new hydride alloys (Buchmann). We just discussed the importance of a high energy density for our applications and the NiMH's provide a 40 percent higher specific energy than standard NiCad's (Buchmann). They are also far less toxic and more durable than NiCad's; this is a good thing for our project in case a quadcopter crashes and we are unable to locate and retrieve it. We don't want to destroy or pollute, but preserve our environment with our project.

Unfortunately, NiMH batteries have complex charging algorithms, limited cycle life, and a high self-discharge rate. The high self-discharge rate coupled with complex charging algorithms makes charging these batteries difficult and time consuming with a small power source on the mobile landing platform (Buchmann). These batteries lose 20 percent of their capacity within the first 24 hours without even being used and that would make flight preparation times much longer than we desire for our project (Buchmann). Pictured below, in Figure 53, is a NiMH battery that is sold by HobbyKing.com to hobbyists, courtesy of HobbyKing.com.





**Figure 53: AA cell 1.2V NiMH battery with a 2400mAh capacity (permission pending)**

Lithium is the lightest of all metals, has the greatest electrochemical potential and provides the largest specific energy per weight of all the batteries we have discussed. Lithium ion (Li-ion) batteries are a product of research that began due to instabilities of lithium metal (Buchmann). During the charging of the metal batteries, thermal runaway could occur and cause fires and/or burns to those handling them.

Li-ion batteries have twice the specific energy as the NiCad and a high nominal voltage at 3.60V (Buchmann). This is much higher than the 1.2V nickel battery systems. Li-ion batteries have become the universally accepted battery for portable rechargeable electronic applications. They are also the battery of choice for electric powertrains for electric car manufactures such as the up-and-coming Tesla Motors. The cost of these batteries has been dropping rapidly over the last decade to make them more affordable consumers.

One of the downsides for Li-ion, are the fact that they require a protection circuit in order to prevent abuse which can cause harm to users (Buchmann). Because of the lightweight design, high energy density, and high nominal voltage, Li-ion batteries are the most popular choice for consumer electronics. One such consumer electronic device would be a notebook computer, which contain these batteries as shown below, in Figure 54.



**Figure 54: A 14.4V Li-ion battery that we removed from an Asus notebook.**

There are several different varieties of Li-ion batteries that are made up of different materials that use Lithium Metal Oxide as the cathode in the battery (Buchmann). The choice of the cathode material gives them their unique features. Some examples of these materials include Lithium Cobalt Oxide (LCO), Lithium Manganese Oxide (LMO), Lithium Iron Phosphate (LFP), as well as Lithium Nickel Manganese Cobalt (NMC) and Lithium Nickel Cobalt Aluminum Oxide (NCA) (Buchmann).

Lithium Polymer (Li-Po) batteries are essentially made up of the same materials as Li-ion batteries, except that the electrolyte that is sandwiched between the anode and the cathode. The electrolyte found in Li-Po batteries differs in that it is micro porous and the one found in Li-ion is traditional (Buchmann). The gelled electrolyte in Li-Po batteries increases the electrical conductivity and offers a slightly higher specific energy (Buchmann). This allows Li-Po batteries to be made thinner than conventional Li-ion batteries. However, this benefit comes at a manufacturing cost increase of 10 to 30 percent (Buchmann). Because of the increased specific energy, as well as, the benefits of being lightweight and having a high nominal voltage, Li-Po batteries are the currently the most popular choice for aircraft hobbyists today. This is why we decided they would be the best choice for our project.

Li-Po batteries for quadcopter hobbyists are categorized and sold using various criteria. The first would be the chemical makeup of the batteries; there are two major categories: Lithium Iron Phosphate (LiFePo4) and Lithium Polymer, which are generally Lithium Cobalt Oxide (LCO) (King). As stated above, we chose to go with the Li-Po batteries. The popular GoPro camera's use Lithium Polymer batteries, as shown below in Figure 55.



**Figure 55: A 3.7V Li-Po battery that we removed from a GoPro camera**

The second classification is the number of cells in the battery. LCO batteries have a nominal voltage of 3.7V per cell (Buchmann). For example, the nomenclature used to describe a battery with three cells connected in series would be “3S,” which would have a nominal voltage of 11.1V ( $3 \times 3.7$ ) (Bourke, Understanding Electric Power Systems Part 2). Some battery packs also come with a parallel classification such as “1P” or “2P”. The parallel number denotes the number of parallel series of cells. For example, a “3S2P” (6 cells total =  $2 \times 3$ ) battery would be able to provide twice the current and would last twice as long as a “3SP1” battery, consequently it would also weigh approximately twice as much.

The third major classification we needed to consider when choosing a battery for our quadcopters was the energy capacity. This is measured in milliamp-hours (mAh). This number represents the amount of milliamps (mA) the battery could supply at a constant rate for one hour (Bourke, Understanding Electric Power Systems Part 2). For example, a battery with an energy capacity of 2800 mAh could supply 2800 mA of constant current for one hour or 700 mA for 15 minutes.

The fourth and final specification used to categorize Li-Po batteries for quadcopters is the discharge rates. Because of the sensitive nature of Li-Po batteries, they have limited discharge rates. These rates are directly related to the capacities of the batteries. For example, a battery with a 2800 mAh capacity that can safely supply a constant rate of 28A would be specified with a 10C continuous discharge rate (RC Helicopter Fun). There is also a maximum burst discharge rating that indicates how long a battery can safely supply a short burst of current, usually for 10 to 15 seconds.

In order to correctly choose the correct battery size for our prototype aircraft, we needed to choose our propulsion system first. We first have to know how much current and voltage our motors will require.

### 16.3 Battery Charging

Li-Po batteries require special charging systems to extend battery lifetime, prevent damage to the battery, and balance the cell voltage throughout the battery. The ability for a charger to balance means it has the ability to monitor the voltage of each cell and charge or discharge individual cells; so that all of the cells will be at the same voltage (RC Helicopter Fun). We won't need to charge more than five cells simultaneously, which means we only needed look at chargers capable of charging six cells. We won't be using batteries over 300 grams, so our batteries will have a capacity of less than 3000 mAh. Most Li-Po batteries used in quadcopters now, support charging at 2C (King). This means our charger will not need the ability to charge at more than six amps. With these specifications in mind, we decided to go with the IMAX B6-AC Charger/Discharger from HobbyKing.com, as picture below in Figure 56.



Figure 56: iMax B6AC Li-Po battery charger/discharger (permission pending)

#### iMax B6-AC Features

- Max Charge: **50W**
- Max Discharge: **5W**
- Max Charge Current: **5A**
- Max Discharge Current: **1A**
- Li-ion/Poly Cells: **1-6**
- Individual cell balancing
- Time limit function
- Can store up to 5 packs in memory



## 16.4 Propulsion System

Before choosing the battery, motor, and propeller combination for a multi-rotor vehicle, there are several things that need to be considered. The first thing to be considered is the total weight of the aircraft, which would include the weight of the motors, propellers, batteries, wires, electronics, frame, and of course the payload of the quadcopter. Once the approximate weight is calculated or estimated, you can calculate the total amount of thrust you need to properly power the quadcopter. For our quadcopters, we estimated an approximate weight of 1.5 kg (3.3 lbs.) including our payload of 500 grams (1.32 lbs.). Below is a table of the weight budget for our quadcopter prototype, shown in Table 6.

<i>Part</i>	<i>Weight (grams)</i>	<i>Quantity</i>	<i>Total Weight (grams)</i>
Frame	270	1	270
Battery	231	1	231
Motors	55	4	220
Props	7	4	28
ESCs	25	4	100
Telemetry	15	1	15
GPS	20	1	20
APM	15	1	15
Receiver	18	1	18
Cables/Wires	35	1	35
Misc.	40	1	40
Payload	500	1	500
		Total	1492

**Table 6: Weight budget for the first prototype quadcopter**

The standard calculation for calculating the total amount of thrust (in kg of force) needed for a multi-rotor vehicle is two times the total weight of the aircraft. To find the amount of thrust required per rotor, we divided the total amount of thrust required by the number of rotors going on the aircraft, as shown below in Equation 2 (Bourke, Understanding Electric Power Systems Part 2).

$$\begin{aligned} \text{Total Thrust Required} &= 2 \times \text{Total Weight} \\ 3000g &= 2 \times 1500 \text{ grams} \end{aligned}$$

**Equation 2: Calculation for the total amount of required thrust**

$$\begin{aligned} \text{Thrust per rotor required} &= \text{Total Thrust Required} \div \# \text{ of rotors} \\ 750g &= 3000g \div 4 \end{aligned}$$

**Equation 3: Calculation for the amount of required thrust per rotor**

With four rotors being used on our aircraft, we calculated a minimum requirement of 750g of thrust per rotor. This meant that our motor, propeller, and battery combination would need to supply a minimum thrust force of 750g, as shown in the above calculation in Equation 3. This meant that the maximum thrust force per motor must be at least 750g for our planned payload.

Most of the motors used by hobbyist for flight are brushless electric motors to reduce friction and energy loss (Bourke, Understanding Electric Power Systems - Part 4). Besides choosing a brushless motor, there are many other factors that need to be considered when choosing the motors to be used. These would include the ratio of revolutions per minute (RPM) to voltage (Kv); voltage loss due to the motor's coil resistance (Rm); the motor's output power (Po); the motor's efficiency ( $\eta$ ); and the torque that can be provided by the motor. Below in Equation 4, are some of the calculation equations for figuring out what we can get out of a motor given certain variables (Bourke, Understanding Electric Power Systems - Part 4).

$$P_{out} = \text{Output Power (Watts)} = \text{Torque(Newtons * meters)} \times 2\pi \times \text{RPM}/60$$

$$P_{out} = \text{Output Power (Watts)} = (V_{in} - I_{in} \times R_m) \times (I_{in} - I_o)$$

$$\text{Torque(in. -oz.)} = K_t \times (I_{in} - I_o)$$

$$\text{RPM} = K_v \times (V_{in} - V_{loss})$$

$$K_t(\text{in. -oz./ampere}) = 1352/K_v$$

$$I_{in} = \text{Current drawn by the motor (Amperes)}$$

$$I_o = \text{Idle Current (Amperes)}$$

$$R_m = \text{Motors Coil Resistance (Ohms)}$$

$$V_{in} = \text{Voltage applied to the motor (Volts)}$$

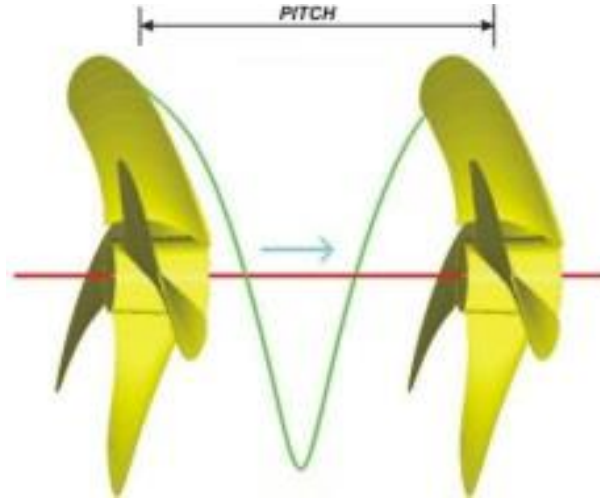
$$V_{loss} = I_{in} \times R_m$$

$$\eta(\%) = 100 \times P_{out}/P_{in}$$

$$P_{in} = \text{Power into the motor}$$

**Equation 4: Equations and variables used to calculate what we required from our motors**

Propellers are sold using several specifications. The first is based on the diameter of the propeller in inches. Most quadcopter propeller diameters range from 8 inches to 12 inches, with 10 inches being the most common (Bourke, Understanding Electric Power Systems Part 1). The larger the diameter of the propeller, the lower the Kv motor is required. Diameter and Kv are inversely related.



**Figure 57: An example of how a propellers pitch is measured (permission pending)**

The second specification is the pitch of the propeller, which is also measured in inches. A propellers pitch is given as the displacement of a propeller in one complete rotation of 360 degrees (Reyes). The greater the pitch, the higher the displacement of the propeller, which equates to more lift. This would also be a larger load on the motor. Diameter (D) and pitch (P) have to be calculated carefully in order to generate the proper amount of lift for a given engine and vehicle weight. For example, a 1045 size propeller has a diameter of 10 inches and a pitch of 4.5 inches. Shown above in Figure 57, is how a propellers pitch is measured; which is how much displacement happens in one revolution, courtesy of Propeller Pages.

The third specification that we needed to consider when choosing a propeller was the propeller constant, Kp. The propeller constant, unless given, is assumed to be approximately 1.25. The propeller constant varies depending on the brand of propeller. Below, in Figure 58, is an example of the calculation required to calculate the power absorbed by the propeller (Pa). The power absorbed is also the power output (Po) by the motor (Bourke, Understanding Electric Power Systems Part 3).

$$\begin{aligned}
 \text{Power Out(Watts)} &= \text{Power Absorbed(Watts)} \\
 Pa(\text{Watts}) &= Kp \times (\text{RPM(in thousands)})^3 \times \text{Diameter(ft.)}^4 \times \text{Pitch(ft.)} \\
 95.37 \text{ Watts} &= 1.25 \times 7.5^3 \times \left(\frac{10}{12}\right)^4 \times \frac{4.5}{12}
 \end{aligned}$$

**Figure 58: The calculation to find the Power absorbed by the propeller, given the propellers diameter, pitch, constant, and RPM**

The last specification is a classification based on the type of propeller application. Depending on the application, propellers need to be able to spin at certain speeds. Whether it's a single propeller model airplane or a multi-rotor quadcopter, there are different classes of propellers for each application. Below, in Table 7, are the different classes of propellers and their maximum RPMs per propeller diameter (Advanced Precision Composites).

Application	Max RPM (8" Prop)	Max RPM (9" Prop)	Max RPM (10" Prop)	Max RPM (11" Prop)	Max RPM (12" Prop)
<b>Glow Engine Speed 400</b>	23,750	21,111	19,000	17,273	15,833
<b>Thin Electric Folding Electric</b>	18,125	16,111	14,500	13,182	12,083
<b>Multi-rotor</b>	12,125	11,667	10,500	9,545	8,750
<b>Slow Flyer</b>	8,125	7,222	6,500	5,909	5,417
<b>Racing</b>	28,125	25,000	22,500	20,455	18,875

**Table 7: Maximum RPM's based on various propeller sizes for different classes of propellers**

Multi-rotor propellers have a maximum RPM that is equal to 105,000 divided by the propellers diameter in inches. This means a 10 inch propeller can handle a maximum of 10,500 RPMs or a 12 inch propeller would have a maximum RPM rating of 8750 (Advanced Precision Composites). To find out what kind of Kv rating our motors would need with a 10 in. propeller, we divided the maximum RPM (10,500) by a three cell battery nominal voltage (11.1V). This gave us a maximum Kv rating of 946 RPM/V. The calculation is illustrated below in Equation 5 (Bourke, Understanding Electric Power Systems Part 3).

$$Kv = \left( \frac{105,000}{\text{Prop Dia.}} \right) / 3S(\text{Nom. Voltage})$$

$$946(\text{RPM/V}) = \left( \frac{105,000}{10 \text{ in.}} \right) / 11.1V$$

Figure xxx:

**Equation 5: Calculations for finding the maximum Kv rating given a 10 in. propeller**

This meant we could look for a motor with a Kv rating in the middle to low 900's range. We searched the most popular website for purchasing quadcopter parts, Hobby King (King), for a motor that fell into this range of Kv. We found that the Turnigy branded motors had the best reviews and decided to go with Turnigy as a brand for most of our quad-related parts.

The Turnigy Multistar MT2213-935KV is a kit that came with a motor, the mounting hardware, and two propellers, one clockwise and one counter-clockwise. The propellers included in the kit were size 1045 (10 in. diameter and 4.5 in. pitch) as used in our example above. The motor included in the kit is a Multistar 2213 with a Kv rating of 935 RPM/V (King). The motor has a maximum thrust of 850 grams using a 3S Li-Po battery. With four motors and propellers, this would give us a total maximum thrust of 3.4 kilograms. This gave us more than we required, but was

within an acceptable range. Below are the specifications of the chosen motor/propeller kit.

#### Specifications for the Multistar MT2213-935KV Kit

- Kv (RPM/V): **935**
- Li-Po cells: **2-4s**
- Max. Power (Watts): **200**
- Max. Current (Amps): **15**
- No Load Current (Amps): **0.4**
- Internal Resistance (Ohms): **0.180**
- Number of Poles: **14**
- Motor Shaft: **3mm**
- Bolt Hole Spacing: **16mm x 19mm**
- Propeller Dimensions (inches): **10" x 4.5"**
- Max Thrust (1045 Propellers): **850g**

After selecting the MT2213-935KV kit, we needed a battery to match up with it. We needed a three cell (3S) battery that would supply 11.1V nominally, that also weighed less than 280 grams. Based on price, reviews, and product ratings, the Zippy Flightmax branded batteries appeared to be the prime choice for our project (King). We decided on the Flightmax 2800 mAh 3S1P 30C as our battery, shown below in Figure 59, courtesy of HobbyKing.com. The battery cost approximately \$18 on HobbyKing.com.



**Figure 59: The Zippy Flightmax 2800mAh 3S1P 30C Li-Po battery sold by HobbyKing.com (permission pending)**

We chose this battery because of its weight of 231 grams and its ability to support a constant discharge rate of 84A. This discharge rate equates to a constant discharge

rate per motor of 21A, which exceeds the maximum draw of the motors. This would make sure our battery was not overloaded, which could cause injury. The weight of the battery was well within our estimates as well. Our hovering flight time is estimated to be around six to nine minutes depending on our payload. This gave us enough time to complete our objectives and return to the mobile landing platform.

The last parts of the propulsion system are the power distribution and the electronic speed controls (ESCs). There are two main ways of distributing power to the motors from the battery. The first method is to use a power distribution board, which is a PCB that has one input and multiple outputs, as shown below in Figure 60, courtesy of HobbyKing.com. The input is for the battery connection and the outputs connect to the ESCs. Some of the distribution boards have monitoring devices for monitoring voltage levels and the batteries performance.



**Figure 60: A power distribution board for a quadcopter that is sold for \$4 on HobbyKing.com (permission pending)**

We chose to go with the second option, which is a wiring harness that connects straight from the battery to the ESCs. We went with this option to minimize the weight of the hardware. The power distribution board still requires about the same amount of wires and this would just add weight and cut down on our flight time and overall performance. We chose to make our own wiring harness using 3.5 mm bullet connectors that would fit the ESCs and the XT60 connector that would connect directly to our battery (King).

The electronic speed controls limit and control the amount of power that is allowed to be delivered to the motor. There needs to be one ESC per motor that connect to the flight controller for speed control. ESCs are categorized by one major specification, max current. We had the option of getting ESCs with battery eliminated circuits



(BECs), which is a 5V output for another device on the quadcopter, such as our flight controller (King).

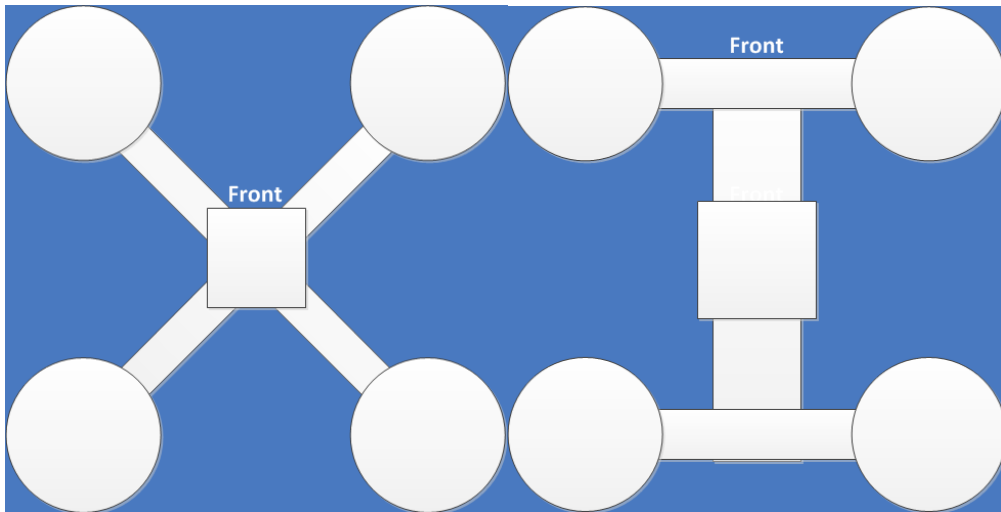
Since, we were planning on powering our flight controller in this way; we needed to calculate an additional max current rating. The way we calculated this was by taking the maximum current rating of the motors, 15A, and we multiplied it by 1.5 to get 22.5A as our max rating for our ESCs. We got the minimum rating by multiplying by 1.2 to get 18A. The ESCs are usually sold in increments of 5A, so we decided we needed four 18A to 22A ESCs. Since we decided on going with the Turnigy Multistar MT2213-935KV motor and propeller kit, we elected to go with the Turnigy Plush 18 Amp Speed Controllers, shown below in Figure 61, courtesy of HobbyKing.com. These came with a burst current option of 22A, which is exactly what we were looking for. We considered the Turnigy Multistar 20 Amp Multi-rotor Brushless ESCs, but they were only available in 20A and 30A versions, which wasn't ideal for our needs.



**Figure 61: The 18A Turnigy Plush Electronic Speed Controller we decide to use in our prototype (permission pending)**

## 16.5 Structural Subsystem

When choosing a frame for our quadcopters, we needed to consider what kind of configuration the rotors we would want the rotors to be in. There are two major configurations to choose from, shown below in Figure 62, the “X” configuration and the “H” configuration. We chose the “X” configuration based on research we did in multiple online forums (OpenPilot Forums) (RC Groups Forum). The discussions revealed that the programming for an “H” configuration wouldn't be as straightforward as would be for the “X.” This is due to the fact that the center of the frame on the “H” is sometimes stretched, which would cause the motors to not be equally distant. This causes more complex algorithms for the flight controllers. These complexities could be an issue if we have to modify the open-source code for the flight controllers.

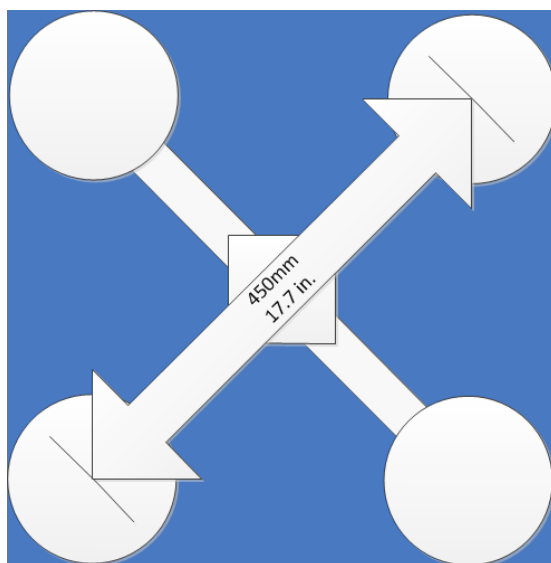


**Figure 62: The two major configurations for quadcopters the “X” (Right) and the “H” (Left)**

Next, we had to decide whether or not we would build the frame or buy it. Since, all of us are new to flying quadcopters and we will probably crash our prototype several times before we get the hang of flying, we should be able repair our frames if they should break. We can do this several ways. First, would be to build and frame using common inexpensive materials, so we would be able to readily fix our problems without having to order a new frame and we could save money on our budget. Unfortunately, we believed this project is going to be very time consuming and we did not think we had the time to learn to build a proper frame. We, instead, decided to buy a relatively inexpensive quadcopter frame and see how it performs before we decided on the final design.

Third, we had to decide what size frame we would want for our application. We needed a frame that was big enough to have ten inch propellers on each arm with enough clearance between them for any antennas and electronics we planned to have in the center of the quadcopters. For this calculation, we took the radius of two of propellers we choose (five inches for a ten inch propeller) and summed them with another six inches (for the electronics) to get 15 inches. “X” configuration quadcopter frames are generally sold by the diagonal length from one rotor to the opposite open and are measured in millimeters, as shown below in Figure 63.





**Figure 63: An example of how quadcopter “X” frames are measured and sized**

So, 15 inches is equal to 381 mm; this meant we were looking for a frame that had a diameter of at least 400mm. The smaller the diameter of the frame, the more agile the quadcopter; and the larger the diameter, the more stable the quadcopter would fly. According to DIY Quadcopters, a solid and stable quadcopter frame is between 500mm to 600mm from motor to motor (Quadcopters DIY).

We also needed to consider the weight of the frame versus the size based on the materials it would be made of. We had to decide on a frame that would be made of a material that was ridged, durable, lightweight, and relatively inexpensive. There are a few materials that most quadcopter frames are made out of including: carbon fiber, wood, aluminum, and fiber glass (Quadcopters DIY).

Carbon fiber is the material of choice for quadcopters in our size range. Carbon fiber is lightweight and ridged which is ideal for our quadcopter. Unfortunately, pre-built carbon fiber frames are expensive and out of our price range for this project. It would cost us over \$200 to buy a pre-built frame (King). If we were to continue this project with more time and money, we would elect to custom-build our own frame using carbon fiber using a CNC machine and possibly a laser for cutting. Shown below in Figure 64, is the Bumblebee Carbon Fiber Quadcopter Frame sized at 550mm (King). This size would make the quadcopter more of a stable flying quadcopter that weighs over 900 grams without a battery! In addition to costing too much for our budget, at \$144.99, this frame would be way too heavy with the motors and propellers that the frame comes with. The Bumblebee has plenty of ground clearance, which would make for easier landings and room for our camera.



**Figure 64: The Bumblebee Carbon Fiber Quadcopter frame (permission pending)**



**Figure 65: The all-wood Hobby King Quadcopter Frame V1 (permission pending)**

Wood is a very common material for building frames and it is also the lightest material for constructing them. However, wood is also the least sturdy and breaks much easier than some of the other materials (Quadcopters DIY). As stated before, this will be our first quadcopter we will be flying and we want something that will hold up to a little abuse in the beginning of prototyping while we get the hang of flying and

programming our copters. Wood frames are cheap; which would allow us to replace them for less than some of the other materials, but we wanted something more ridged and durable for our prototype. A prebuilt wood frame would easily cost less than \$10 for our needs. However, the wood frame featured above, in Figure 65 courtesy of HobbyKing.com, sold for approximately \$18 and weighed only 195 grams. It also had a wingspan and size of 550mm made out of plywood.

Aluminum is one of the most abundant elements in the universe and the most abundant metal in the earth's crust. This makes it an inexpensive and easy to obtain material for building frames. Aluminum is also a very lightweight and ridged metal, which makes for a good material candidate for building frames (Quadcopters DIY). Most aluminum parts for quadcopter frames are for do-it-yourselfers that want to build their own frames, so we elected not to go aluminum as a primary material for our quadcopter frame. Below, in Figure 66, is a picture of an aluminum frame that we were considering using and it weighs just approximately 300 grams (King). The X450 feature a folding design that would allow for easier transportation to and from the flight areas. It is sized 450mm and sells for approximately \$40. We were concerned with the amount of time it would take to arrive, since it was being shipped from China.



**Figure 66: The aluminum X450 CNC Metal MultiCopter frame listed on eBay (permission pending)**

Finally, we have fiber glass, which is more ridged than wood, lighter than aluminum, and cheaper carbon fiber (Quadcopters DIY). Best of all, it is almost as cheap as wood, but more durable. This is why we have chosen to go with fiber glass for the material of our prototype frame. The kits we looked at run from \$30 down to less than \$10. Because this is our first prototype and we are unsure about exactly how much agility versus stability we will need with our quadcopters, we opted to go with a size that would be a little more agile in order to get the hang of flying. We looked at

frames that were sized in the 450mm range that would weigh less than 280 grams to meet our weight budget. The frame we are going to use for our first prototype is shown below, in Figure 67. It is the Q450 450mm Glass Fiber Quadcopter Frame that is sold by HobbyKing.com.



**Figure 67: The Q450 450mm Glass Fiber Quadcopter Frame (permission pending)**

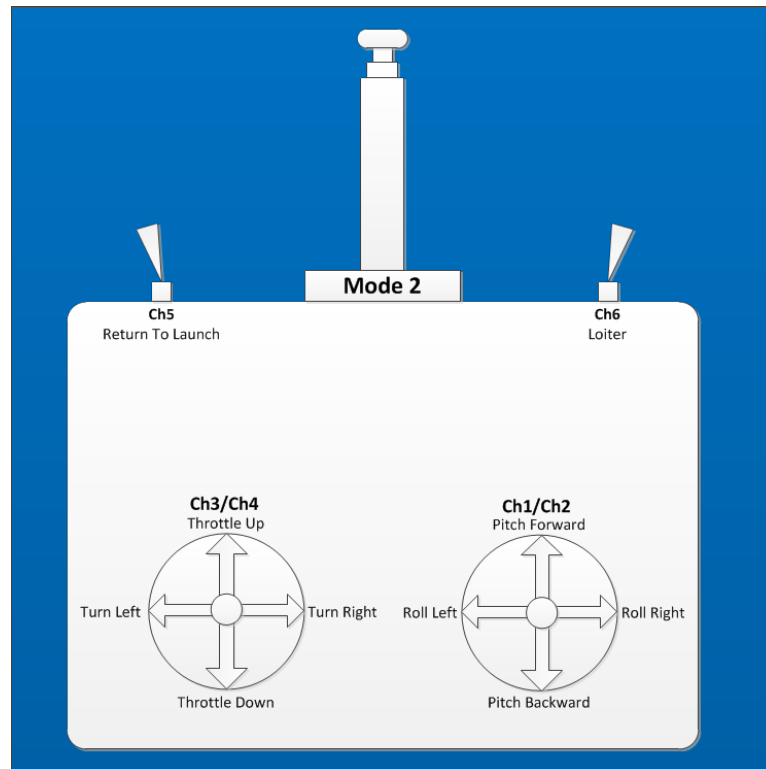
The Q450 has four arms that are made of ultra-durable polyamide nylon. They meet in the middle and are joined together on the top by a fiber glass frame (King). The arms are also two different colors to help the pilot know the orientation of the quadcopter. There is a larger bottom frame to hold the electronics and the rest of the payload. The Q450 weighs approximately 270 grams, which is less than our 280 gram budget. The motor mount holes also easily fit our selected motors without any modification, which helps keep our costs low.

## **16.6 Quadcopter Remote Control System**

For our first prototype quadcopter, we wanted to have the ability to have manual flight control, whether it is via a Remote Control (RC) Transmitter and Receiver or through a combination of the telemetry system and the Ground Control System (GSC), which we will discuss later. When deciding on which method we preferred, we had to consider the pros and cons of each system.

We needed to purchase and use the telemetry system regardless of whether we chose to use it as our sole method of control or not. If we decided on not using a RC

Transmitter and Receiver, we would save weight on the payload and spend less on our prototypes. However, if the telemetry system fails, we would not have any way of controlling the flight of our quadcopter for emergency maneuvering, which could result in damage or possibly a total loss of the aircraft. This is an unacceptable risk for us to take, so we elected to go with both the telemetry method and the RC Transmitter and Receiver method.



**When deciding on the Figure 68: Example of a Mode 2 6CH RC Transmitter**

RC Transmitter and Receiver for our prototype, we had to choose between two different control configurations, known as Mode one and Mode two (Liang). The difference between the two different configurations is which side of the controller the elevator control and the throttle control is situated. Mode one has the throttle controlled with the right hand and the elevator controlled on the left hand. Mode two is the most common for multi-rotor applications with the throttle on the left and the elevator joystick on the right. Mode 2 has the elevator joystick (right) that self-centers on both axes, while the throttle joystick only self-centers from left to right. Since this is our first time flying a quadcopter, we elected to go with the, traditional, Mode Two configuration, as shown above in Figure 68.

We also needed to know how many channels we would need to have on the system. We would require a minimum of six channels, recommend by the APM documentation, in order to properly control the quadcopter. The price range of a six channel transmitter and receiver combination ranges from \$25 to \$150 (King). The price range for an eight channel system, the maximum number of inputs the APM supports, ranges from \$50 to over \$1000 (King). Since this was our first prototype



we want to start with something that wasn't too expensive, but had good reviews. HobbyKing.com has the best selection of fairly priced multi-rotor parts and accessories, so we decided to look through their selection for our RC Transmitter and Receiver combination kits.

<i>Transmitter</i>	<i># of Channels</i>	<i>Mode</i>	<i>Frequency (GHz)</i>	<i>Modulation</i>	<i>Rx Included</i>	<i>Rx Weight</i>	<i>Price</i>
<b>Turnigy 6X FHSS</b>	6	2	2.4	FM	Yes	11.5g	\$30
<b>Turnigy 9X V2</b>	9	2	2.4	PCM/PMM	Yes	18g	\$54
<b>JR XG7</b>	7	2	2.4	DMSS	Yes	15g	\$248
<b>HK-T6A-M2</b>	6	2	2.4	FM	Yes	12g	\$25
<b>HK-6DF-M2</b>	6	2	2.4	FHSS	Yes	14g	\$31
<b>Futaba 6EX</b>	6	2	2.4	FASST	Yes	7g	\$170
<b>Futaba 7C</b>	7	2	2.4	FASST	Yes	7g	\$275

**Table 8: Controllers considered for our first prototype quadcopter setup**

The JR XG7 uses Dual Modulation Spread Spectrum (DMSS) technology to transmit its signal to the receiver. This means it operates using a wider spread signal than the Frequency-Hopping Spread Spectrum (FHSS) technology (Liang), which results in a link with the receiver that is more resistant to interference. Although the JR XG7 is highly rated on HobbyKing.com and would most likely perform the best out of the selection listed above in Table 8 (King); it is too expensive for our price range and budget. However, we would consider it if we decided to pursue flying RC quadcopters as a hobby.

Both the Futaba 6EX and 7C, shown below in Figure 69, feature a Futaba exclusive technology known as Futaba Advanced Spread Spectrum Technology (FASST). FASST continuously shifts between different channels hundreds of times per second, in order to reduce interruptions and signal conflicts (Hobbico). Both of the Futaba receivers weigh less than seven grams, which makes them the lightest receivers on the list. However, as appealing as both of these kits are, they are also more than we wanted to spend on our RC transmitter/receiver kits.



**Figure 69: The Futaba 6EX (Left) and the Futaba 7C (Right) (permission pending)**

The two six channel Hobby King transmitter/receiver kits are within our budget and price range. Both the HK-T6A-M2 and the HK-6DF-M2 are rebranded transmitter/receiver kits that are sold as HobbyKing.com's brand, as shown below in Figure 70 (King). They are the cheapest on our list of considered RC transmitter/receiver kits and also the considered to be entry level kits. Our restricted budget and entry-level starting point are the main reasons we considered these two kits.

The HK-T6A-M2 is the cheapest kit on our list and is a rebranded version of the Fly Sky FS-CT6A. This system must be programmed via a USB cable and requires a bit more time to configure than the other models. This model also uses very basic Frequency Modulation (FM) for transmitting its signal, which makes it very susceptible to interference. The HK-6DF-M2 kit uses a FHSS system to transmit, which is less susceptible to interference than the basic FM system in the HK-T6A-M2 (King). At a six dollar difference, this is a better value.



**Figure 70: The Hobby King HK-T6A-M2 (Left) and HK-6DF-M2(Right) (permission pending)**

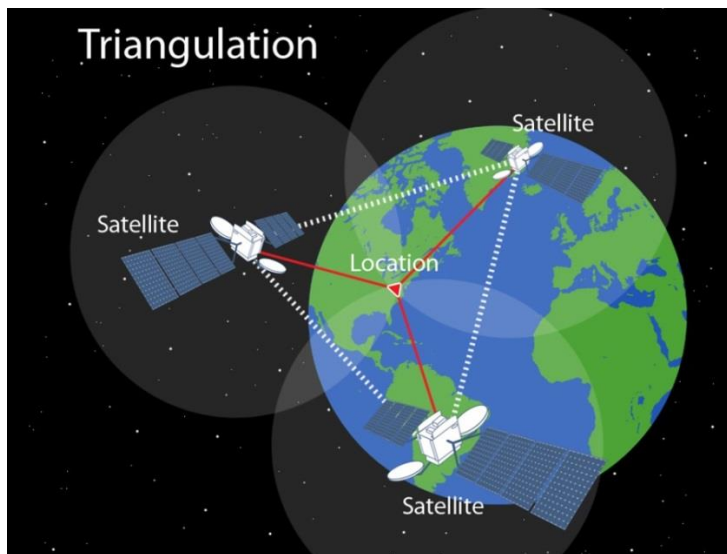
The Turnigy 6X is also in the entry level price range at \$30 and uses the FHSS modulation as well. The Turnigy 6X receiver weighs approximately 34% less than the Hobby King equivalent receiver (King). Turnigy also makes a nine channel receiver, the Turnigy 9X that is also sold on HobbyKing.com. The 9X features more three channels more than the 6X with a cost increase of only \$24. This looked to be the best value for our project and needs for our first prototype, so we elected to go with the Turnigy 9X transmitter/receiver kit, as shown right in Figure 71, courtesy of HobbyKing.com.



**Figure 71: Turnigy 9X 9 channel RC transmitter (permission pending)**

## 17.0 Navigation Subsystem

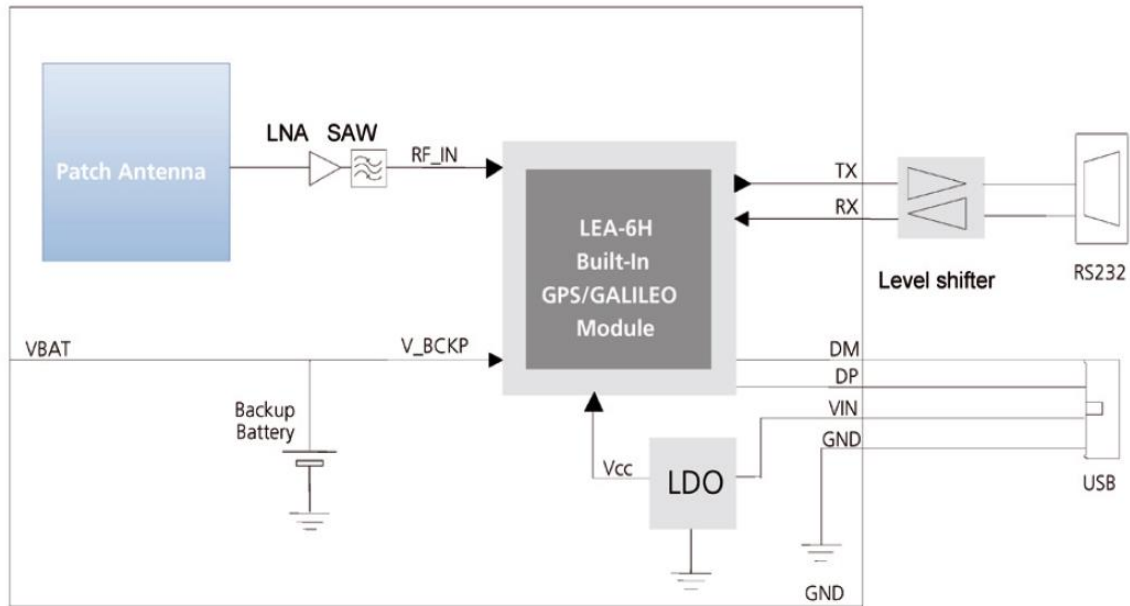
The methods and technology for autonomous navigation for a quadcopter, without highly sophisticated and expensive instruments, is limited the Global Positioning System (GPS). The GPS is a system of more than 24 satellites that orbit the earth and all transmit a one-way signal down to earth (U.S. Government). These signals can be pickup by GPS receivers on earth. These signals contain the time that the message was sent according to the atomic clocks located on the satellites. With these times, the receiver can measure the time delay between signals and triangulate their position with a minimum of three satellites, as shown below in Figure 72.



**Figure 72: An illustration of how GPS triangulation looks from space (courtesy of National Geographic under the Educational Use of Content agreement)**

The prototype navigation sub-system for the two aircraft receives positioning coordinates via their communication with the Global Positioning System by way of the U-blox C04-6H GPS Smart Antenna (Ublox). The C04-6H is a combination of a u-blox LEA-6H GPS module with a ceramic patch antenna, USB and UART interfaces, and an on-board backup battery all integrated together on single board. The architecture for the C04-6H board is shown below, in Figure 73.

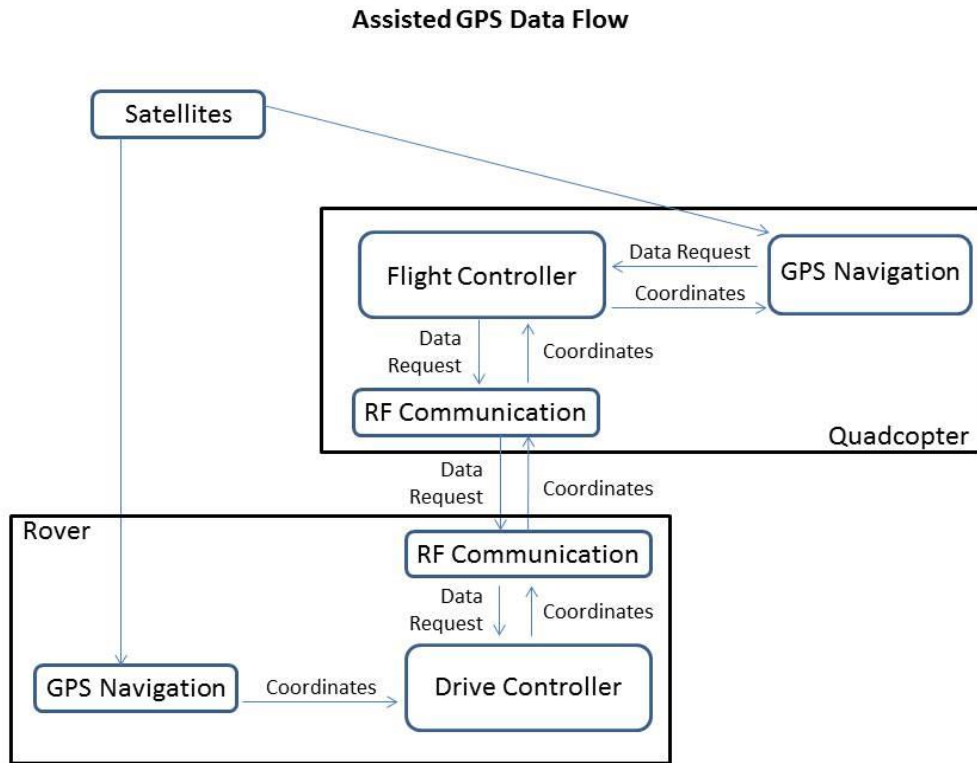




**Figure 73: C04-6H Block Diagram (permission pending)**

We wanted a GPS receiver that would be able to handle a weak signal in the case of inclement weather. This is why we chose the U-blox 6 GPS system, which provides -162 dBm tracking sensitivity. This would give us an accurate horizontal position down to 2.5 meters, without assisted GPS, according to the U-blox LEA-6H datasheet (Ublox). Assisted GPS uses previously acquired positions and times to help speed up the acquisition process for acquiring the first fix of satellites.

This can be done using an internet connection or previously logged location data. With assistance, our horizontal accuracy could increase to 2 meters. This can be accomplished using our communication system aboard the landing platform to the aircraft to read logs from previous flights and using its current position via its onboard GPS. A data flow block diagram, shown below in Figure 74: Assisted GPS flow of information, illustrates the flow of information when the quadcopter is acquiring its location for the first time.



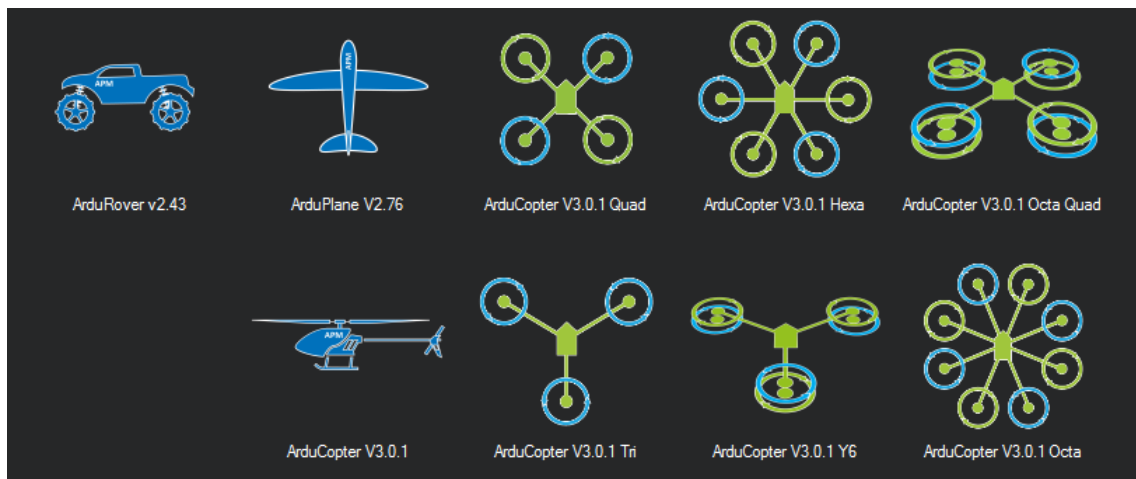
**Figure 74: Assisted GPS flow of information**

## 18.0 Mission Planning Software

Since our goal for our quadcopters is to be autonomous and hands off during missions, we needed a way to program mission parameters before takeoff. This type of software is known as the Ground Control Station (GCS). Since we have elected to use the APM2.6 flight controller based on the Atmega328p chip, we needed to use a supported software package for uploading the mission plans.

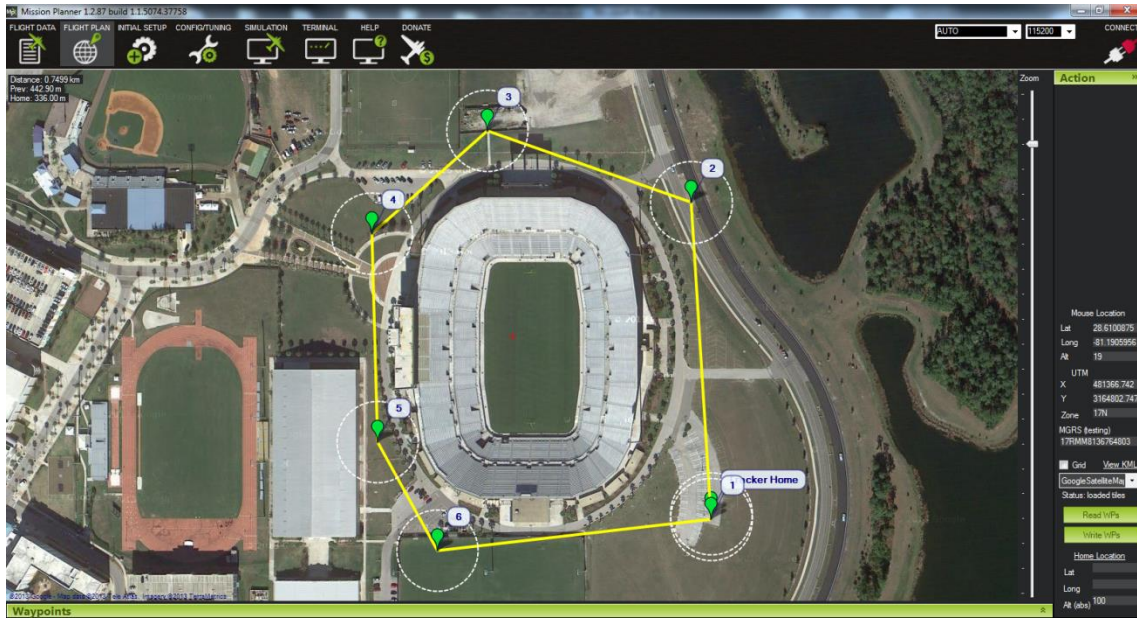
Michael Osborne's Mission Planner software is currently the most supported software for completing this task. Mission Planner is, by far, the most used open source software for configuring and planning autonomous missions. This results in a large online community with many resources for developing an autonomous vehicle. This also means we can add features, if we so desired, in the future.

Mission Planner is a software package that allows us to configure the APM's settings for our particular airframe setup, so we can achieve the best stability and control for manned and unmanned flights. Mission Planner supports many different types of configurations from boats to planes to multi rotor aircraft as shown below in Figure 75. This software package also supports calibration and customizable limits of all the onboard instruments in order to deliver the best results depending on our needs.



**Figure 75: Mission Planner's supported configurations for the APM**

Mission Planner will also allow us to plan missions with waypoints planned using a Google Maps interface within the software. Mission Planner makes this very easy with point-and-click waypoint entry, such as a quick flight around Bright House Stadium on UCF's campus as shown below in Figure 76. This flight plan only took five minutes to create using the software. We will also have the ability to download mission log files for analysis after our missions are completed and the copters have safely returned. This will give us the ability to fine tune the open-source code for the APM flight controller based on the feedback of we obtain from the analysis.

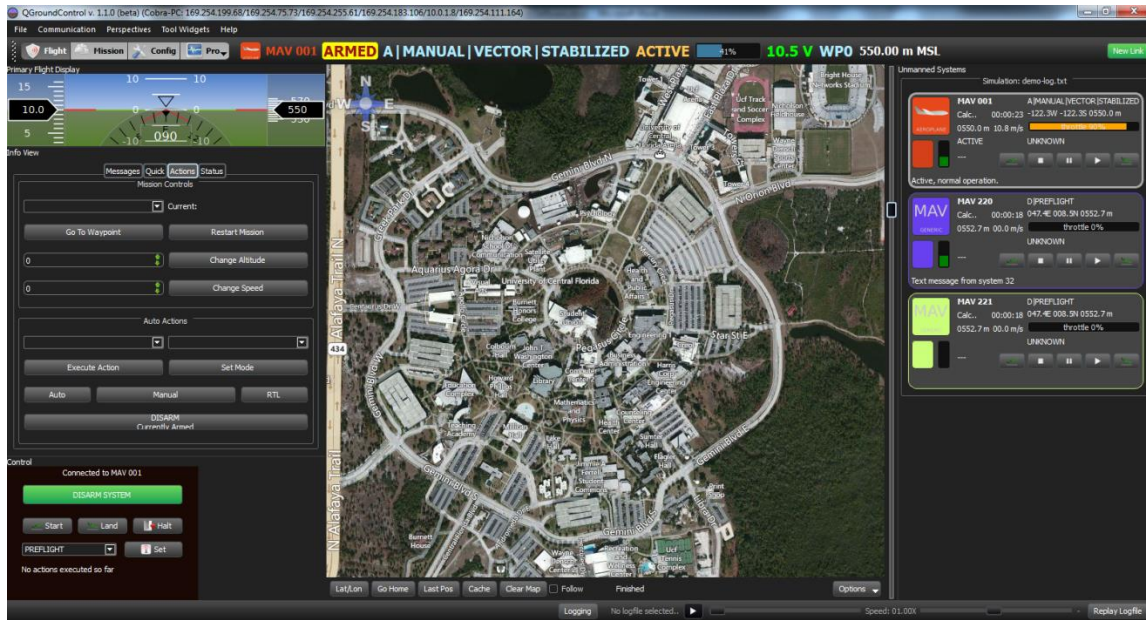


**Figure 76: Planned waypoints in Mission Planner using the maps interface.**

Another feature worth mentioning would be the ability to configure the function for each button/joystick on our remote transmitter through mission. This would give us the ability to assign a switch or button that would tell our quadcopter to return to launch (RTL). This is a major concern for all quadcopter pilots, because of the possibility losing sight of the aircraft in flight. With some button mapping and the flip of a switch, this problem is alleviated and our quadcopter would return to the location it originally launched from during the current mission.

The Mission Planner software also gives us the ability to navigate and fly in real-time via the embedded Google Maps interface (DIY Drones). These commands are transmitted using some telemetry hardware connects to the APM and the computer that has the Mission Planner software being used. This is a very useful and critical feature in the case of a remote controller failure that would normally render the quadcopter uncontrollable.

One of the major issues we are concerned about for our final design is the management of multiple quadcopters, which Mission Planner does not support. However, there is another open source GCS software package that has the capability of controlling multiple vehicles simultaneously called QGroundControl. QGroundControl is a newer software package for ground control and flight controller programming that is still in the beta stages of development (Meier). It supports the control and monitoring of multiple quadcopters simultaneously and is compatible with up to 255 vehicles IMU's; including the APM we have chosen for our prototype. In Figure 77, shown below, is a screenshot we took of QGroundControl simulating multiple aircraft being monitored simultaneously over UCF's campus.



**Figure 77: QGroundControl simulating the missions of multiple UAV's.**

QGroundControl interfaces with the IMU using the MAVLink Micro Air Vehicle Communication Protocol. According to the MAVLink webpage, it has been extensively tested on multiple platforms, including the ArduPilot-Mega platform (Meier). MAVLink was first released in 2009 by Lorenz Meier using a global public license and serves as the communication backbone for the IMU communication.

We plan to experiment with both Mission Planner and QGroundControl software packages with our first prototype quadcopter. After experimentation, we will make our decision of what software configuration we will use in our final design. We may even use both software packages to accomplish our goals. Both of the software packages have very similar features, except for QGroundControl supporting multiple vehicles simultaneously. We will begin working with Mission Planner until we are comfortable with piloted and unpiloted missions for our prototype. Then will begin experimenting with the capabilities of QGroundControl and formulate our plans for integration after becoming comfortable with it.



## 19.0 Systems Integration

This section will be further expanded once we start getting our systems tested and full prototypes built. Once a subsystem passes all of the systems testing, we will then begin integrating the subsystems together. Due to the amount of systems our project involves and since we designed it with a high level of feasibility and scalability, we will start by integrating the vital subsystems first. These subsystems include but are not limited to: Both basic control systems (quad and platform), platform motor controller, Bedini charging system, quad locking system, and the quadcopters with the basic APM 2.6 flight controllers on them. Once we have verified that all the subsystems have passed testing and have integrated well with each other we can then move onto adding additional features.

We currently are not finished with all of our schematics but have a few of the completed systems. Figure 78, Figure 79, and Figure 80 are our more integrated and completed schematics.

### 19.1 Schematics

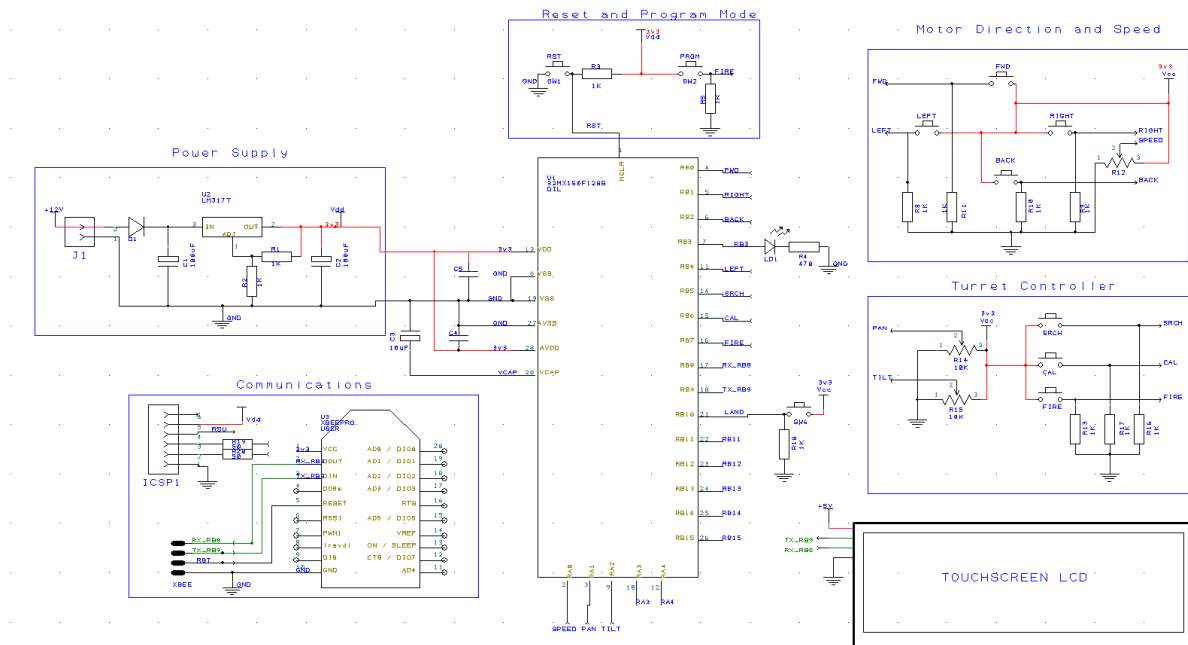


Figure 78: Platform Hand Controller

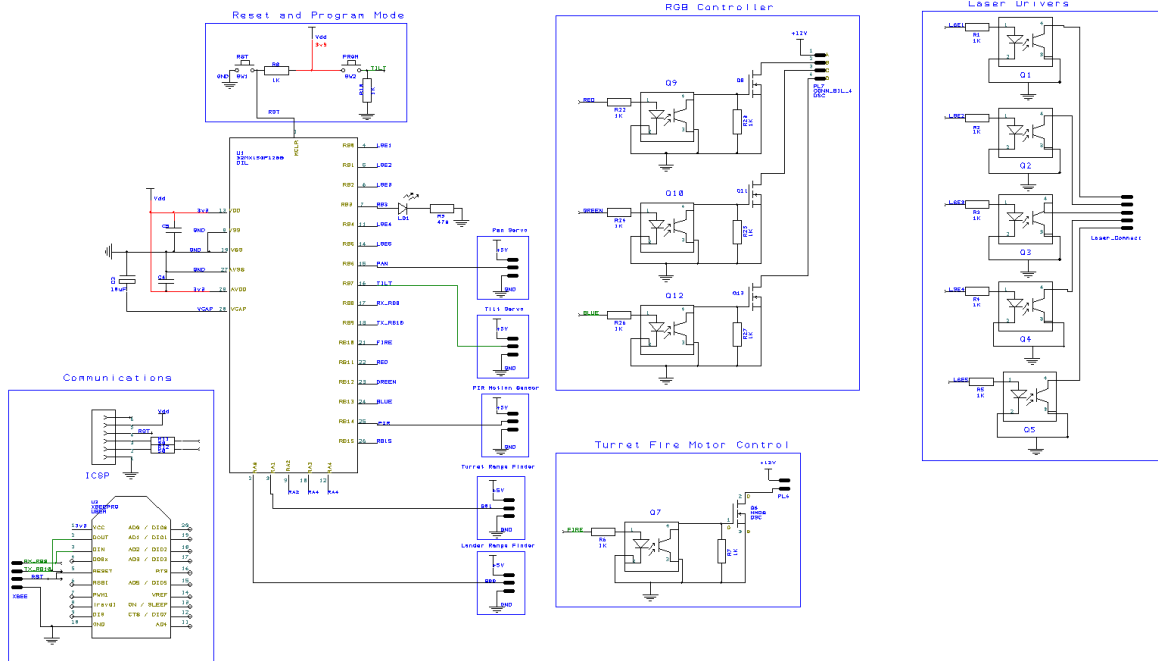


Figure 79: Platform Peripheral Controls

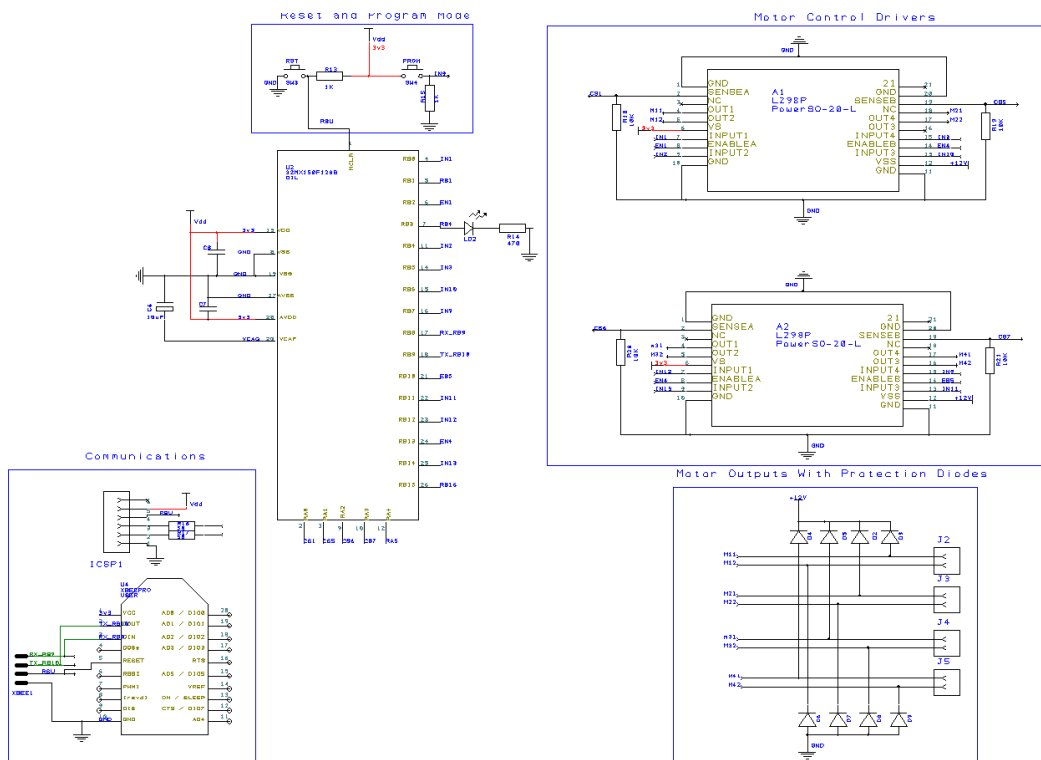


Figure 80: Platform 4-Wheel Drive Motor Controller

## 20.0 Systems Testing

### 20.1 Testing and Characterization Objectives

The goal for this project is to be able to reliably launch and land two quadcopters onto the mobile platform and connect to the charging system. In order to prove the success then, the following must be accomplished:

- Communications between the control terminal, mobile platform and both quadcopters.
- Mobile platform control and location updates.
- Quadcopter launching and landing.
- Charging of the quadcopters from the mobile platform.
- Combined system interaction

The following test procedure is written with the above goals in mind. The details however will be many, as each subsystem will be independently tested and proven prior to the overall system testing. Additionally, because of the interaction of the different subsystems, some testing will be done in parallel between two or more systems. Because these systems will not have been fully tested prior to this test procedure, extra care will be given to mitigate damage to any devices and personnel.

In addition to ensuring that the project works, there will need to be some characterization of the system to verify the range of features that are available with the project. This will allow us to determine additional functionality that may need to be implemented to make sure the system functions more smoothly. Also, any measureable or indicative parameters will be recorded during several of the initial tests so that historical data can be taken and expected parameters can be established for the operation of the project as a whole.

### 20.2 Testing Environment

There will be two primary test environments. The first environment will be indoors, most likely a garage, which will allow most of our preliminary testing. The intent of this is to perform most of the testing in a controlled environment with limited opportunity for any of the vehicles to move beyond our immediate control. This will help us gain the necessary confidence with the system to move out of doors for the duration of the testing and characterization.

The second environment will be a large open field where we can maintain direct line of sight to the mobile platform and either quadcopter in the air. This is where all of the system testing and characterization will take place. By maintaining a direct line of sight, we will be able to track and anticipate any change in the conditions of the environment or changes in the behavior of the vehicles. The hope is to keep as controlled an environment as possible while still giving us maneuvering room to fully



test the system features. With this in mind, weather conditions will apply and we will not test during rainy or overcast days; the former to mitigate unnecessary exposure of the electronics to moisture, and the latter to ensure a clear line of sight during testing.

An additional note concerning the testing environment: Due to the nature these vehicles and the distances that they can travel; walkie-talkies will be utilized for communications between the team. This will be especially important during the characterization phase of the testing, so that proper information is recorded. This is mentioned as part of the test procedure because the walkie-talkies may interfere with some functionality of the system communications, which should be kept in mind for this procedure.

### **20.3 Control Terminal and Testing**

The control terminal will consist of two or three devices that will provide the interface between the operators and both the mobile platform and the quadcopters. From this system, the location of all units will be monitored and real-time data will be displayed, providing feedback for the overall system. Manual control of mobile platform and the quadcopters will also be available. Each device and its function are briefly listed below to allow context for the procedure that follows.

### **20.4 Mobile platform control terminal**

This control terminal will provide control of the mobile platform and will also be the primary monitoring point of the overall system. The LCD display will be central to observing the health of the all components and will be used in all parts of testing and characterization.

### **20.5 Traditional Radio Controller**

This device will allow reliable control of the quadcopters for initial testing and will serve as a backup to the Ardupilot software. We anticipate the need for only one of these controllers; as our intention will be to have only one quadcopter in the air at a time. This device will be used primarily during the quadcopter testing and will used intermittently during system testing.

### **20.6 Ardupilot Software**

A laptop containing this open source software will be the secondary terminal in controlling the system. The purpose of having this as a secondary device is to be able to maintain continuous surveillance of each vehicle, relative to each other and the control terminal. From this software, we will be able to establish flight patterns and maintain the auto home feature for the quadcopters, in case of out-of-range situations or other malfunctions.

## 20.7 Mobile Platform Testing

In order to test the mobile platform, the mobile platform control terminal and LCD will be used and tested in parallel.

1. Set the LCD to the Bedini monitoring screen.
2. Using a DMM, verify that the proper power is supplied.
  - a. The electronics will be operated off of 12V rechargeable batteries. These batteries typically operate  $\geq 14V$  when new.
3. Verify and record all indicators on the mobile platform following power up.
  - a. For the initial power up cycle, all indicators will be recorded and stored for future reference.
4. Ensure that the rotor for the Bedini motor is free of obstructions, and that the power switch to the Bedini motor is “on”. By hand, rotate the rotor and verify that it accelerates its rotation.
5. Monitor the battery voltages and currents for the driving battery and the charging batteries.
6. Verify that the driving battery voltage displayed matches the voltage from the DMM.
7. Continue monitoring these parameters as the RPM of the rotor reaches steady state for 1 minute.
  - a. Record all of these voltages, currents, and RPM as well as the time it took to reach steady state.
8. Using a mechanical support, elevate the mobile platform so that the drive system is not in contact with the ground. Set the LCD to the Platform Motor Control screen.
  - a. Since the mobile platform uses a tank tread system for its mobility with all-terrain compatible features; lifting up the mobile platform permits testing without concern for the initial test environment, should a failure or unexpected event occurs.
9. With the speed slider set to minimum, apply a “forward” signal to the mobile platform. Verify the LCD indicates that the forward signal is being sent.
10. Slowly raise the speed slider to about  $\frac{1}{2}$  of the slide. Verify that the LCD shows the position of the speed slider and record the currents displayed for each motor on the LCD as well as the Target Speed.
  - a. For forward and reverse motion, both treads should move at the same speeds, relative to each other, regardless of the currents.
11. Raise the speed slider to the full length of the slide. Record the currents and Target Speed displayed.
12. Repeat steps 4-6 for the “reverse”, “left, and “right” signals.
  - a. For turns, the treads will be moving at different speeds. As such, we will observe that they are not moving at the same speed relative to each other, however, this will be mostly a binary observation.
13. Switch the LCD display to the Turret Control screen. Using the rheostats on the control terminal, verify that the Tilt and Pan knobs move the turret as expected and that the LCD display indicates the position of the rheostats.

14. Arm the turret and verify the LCD displays correctly.
  - a. There are two indications for this, the green box in the middle of the screen and the ARM button at the bottom.
15. Using a single, non-powered quadcopter; verify that the turret indicates Motion Detected and that numbers are displayed in the Range Values box.
16. Using a measuring tape, move the quadcopter to approximately 36" and record the range indicated on the screen.
17. Repeat step 11 for 72", 120", and 240" and so on, until the quadcopter is outside of the range of the turret. Be sure to record the range at which the Motion Detected indicator goes out.
  - a. These values will be used to verify the calibration of the motion detector and rangefinder.
18. Fire the turret and verify that the missiles launch. Record the distance from the turret that the missiles travel.
19. Switch the LCD display to the Platform Monitoring screen. Using a single, non-powered quadcopter; verify the quad-locking system functions properly. Verify the proper indication is given on the display.
  - a. While the quad-locking system is engaged, the quadcopters will have great resistance to being removed from the platform. This needs to be verified prior to field testing to ensure the quadcopters do not fall off of the platform while the mobile platform is moving locations.

**Note: If testing is being performed indoors, skip steps 20 & 21.**

20. Switch to the Ardupilot software and observe GPS indication of the mobile platform.
21. Move the mobile platform  $\geq 10$  ft. and verify the Ardupilot software updates as well. If the mobile platform indication in Ardupilot does not update, continue to move the mobile platform until the Ardupilot software recognizes the change in location and record the distance traveled.

This concludes testing of the mobile platform. This test should have stepped through powering up the Bedini system, testing the drive system, verifying proper communications with the control terminal, LCD display and Ardupilot software, tested the functionality of the turret, and ensured the proper operation of the quad-locking system. The next step will be to verify operation of each of the quadcopters.

## 20.8 Quadcopter Testing

Testing of the quadcopters will include verification on the control terminal and LCD in parallel. It is assumed that quadcopter testing occurs following the mobile platform tests.

**Note: Keep all limbs and loose articles clear of the quadcopter motors and propellers.**

### Pre-flight checks

1. Using a DMM, verify that the proper power is supplied. Verify these voltages are displayed on the LCD display as well.
  - a. The electronics will be operated off of 11.1V rechargeable lithium-polymer batteries. These batteries typically operate ~ 12.6V when fully charged.
2. Verify and record all indicators on the quadcopter following power up.
  - a. For the initial power up cycle, all indicators will be recorded and stored for future reference.
3. Secure the quadcopter to the mobile platform using the quad-locking system. Verify the quadcopter is secured on the LCD display.
  - a. This will allow for preflight testing to occur without concern for the quadcopter crashing.
4. Using the traditional radio controller, observe the direction of rotation of the props and motors at the lowest possible speed. Verify the direction versus expected.
5. Vary the speed of the rotors on the radio controller and verify that the quadcopter prop speeds follow.

**Note: If testing is being performed indoors, skip steps 6 – 9.**

6. Switch to the Ardupilot software and observe GPS indication of the quadcopter.
7. Verify that the Auto Home feature of the quadcopter matches the location of the mobile platform.
8. Move the mobile platform  $\geq 10$  ft. and verify the Ardupilot software updates the location of the quadcopter as well. If the quadcopter indication in Ardupilot does not update, continue to move the mobile platform until the Ardupilot software recognizes the change in location and record the distance traveled.
9. Verify that the Auto Home feature updates to the current location of the mobile platform.
10. Switch the LCD display to the Security screen.
11. Verify that the quadcopter and the mobile platform are properly paired.
12. Perform additional security testing, as needed (TBD).

## Low altitude flight test

13. Disengage the quad-locking system for the quadcopter in test.
14. Once verified, raise the quadcopter off of the platform about 4 ft. from ground level.
15. In the Ardupilot software, verify that the altimeter correctly displays ~4 ft.
16. Verify hovering stability by carefully grabbing a leg of the quadcopter and pulling it in any direction.
  - a. The quadcopter should resist this movement.
17. Using the traditional radio controller, move the quadcopter away from the mobile platform and land it in a clear area of ground.
18. Switch to the other quadcopter and repeat steps 1 – 17.

This concludes testing of the quadcopters. For each quadcopter; this test should have stepped through power up, verifying communication with the control terminal LCD and Ardupilot software, checking that the traditional radio controller took proper control of each quadcopter, checked the security link between the quadcopters and the mobile platform, and allowed a successful launch from the mobile platform. This test should also have stepped through minor flight functionality of the quadcopters, prior to flight above 10 ft.

## 20.9 System Testing

This test procedure will be divided into the main subsystems that remain to be tested. By treating each subsystem independently, it allows for easier scheduling of testing and isolates any potential issues with that system. It is assumed that the mobile platform testing and quadcopter testing have all been performed with satisfactory results; but those procedures will be referenced for powering up and continuing data collection.

**Note: Keep all limbs and loose articles clear of the quadcopter motors and propellers.**

### 20.10 Waypoint testing

1. Establish a testing area and update the Ardupilot software with that location.
2. Using the mobile platform and quadcopter tests as reference, power up the mobile platform and the quadcopter for the initial run.
3. Move the mobile platform to a fixed location and make sure the location is updated in Ardupilot. Also, verify the home location of the quadcopter matches the location of the mobile platform.
  - a. This location should be in the same vicinity as the control terminals.
4. Disengage the quad-locking system for the testing quadcopter.
5. Using the traditional radio controller, launch the quadcopter to a height of about 5 ft. from ground level.
6. First waypoint test

- a. In the Ardupilot software, establish a three waypoint route, with the ending waypoint being the mobile platform. Use an altitude of ~10 ft.
  - b. Have two team members go to the approximate locations using their cell phone GPS.
    - i. By placing a team member at each of the waypoints, if something bad occurs for the first waypoint test, a team member will be present to attend to the quadcopter.
  - c. Once everyone is in place, implement the route and verify that the quadcopter reaches each waypoint.
7. Second waypoint test.
  - a. In the Ardupilot software, establish a circular five waypoint route, with the ending waypoint being the mobile platform. Use an altitude of ~10 ft.
  - b. Have two team members go to the approximate locations of waypoints 2 and 4 using their cell phone GPS.
  - c. Once everyone is in place, implement the route and verify that the quadcopter reaches each waypoint.
8. Third waypoint test.
  - a. In the Ardupilot software, establish a five point, star pattern waypoint route.
  - b. Have two team members go to the approximate locations of waypoints 2 and 4 using their cell phone GPS.
  - c. Once everyone is in place, implement the route and verify that the quadcopter reaches each waypoint.
9. Land the quadcopter on the ground and manually place it on the mobile platform.
10. Switch to the second quadcopter and repeat steps 2 – 9.

## 20.11 Auto home testing

1. Establish a testing area and update the Ardupilot software with that location.
2. Using the mobile platform and quadcopter tests as reference, power up the mobile platform and the quadcopter for the initial run.
3. Move the mobile platform to a fixed location and make sure the location is updated in Ardupilot. Also, verify the home location of the quadcopter matches the location of the mobile platform.
  - a. This location should be somewhere other than the location of the control terminals.
  - b. A team member will accompany the mobile platform.
4. Disengage the quad-locking system for the testing quadcopter.
5. Using the traditional radio controller, launch the quadcopter to a height of about 5 ft. from ground level.
6. First auto home test
  - a. In the Ardupilot software, establish a two waypoint route, with the ending waypoint being away from the mobile platform. Use an altitude of ~10 ft.

- b. Have one team member go to the last of the two waypoints.
  - c. Once everyone is in place, implement the route and verify that the quadcopter reaches each waypoint.
  - d. Implement the auto home feature once the quadcopter reaches the last waypoint.
  - e. Verify that the quadcopter returns to the mobile platform, but maintains some altitude above it.
- 7. Second auto home test
  - a. In the Ardupilot software, establish a two waypoint route, with the ending waypoint being away from the mobile platform. Use an altitude of ~10 ft.
  - b. Have one team member go to the last of the two waypoints.
  - c. Once everyone is in place, implement the route and verify that the quadcopter reaches each waypoint.
  - d. Move the location of the mobile platform and verify that the Ardupilot software updates the home location of the quadcopter.
  - e. Implement the auto home feature once the mobile platform reaches its new location.
  - f. Verify that the quadcopter returns to the mobile platform, but maintains some altitude above it.
- 8. Land the quadcopter on the ground and manually place it on the mobile platform.
- 9. Switch to the second quadcopter and repeat steps 2 – 8.

## 20.12 Security and turret system testing

- 1. Establish a testing area and update the Ardupilot software with that location.
- 2. Using the mobile platform and quadcopter tests as reference, power up the mobile platform and the quadcopter for the initial run.
- 3. Move the mobile platform to a fixed location and make sure the location is updated in Ardupilot. Also, verify the home location of the quadcopter matches the location of the mobile platform.
  - a. This location should be in the same vicinity as the control terminals.
- 4. Disengage the quad-locking system for the testing quadcopter.
- 5. Using the traditional radio controller, launch the quadcopter to a height of about 5 ft. from ground level.
- 6. Switch the LCD display to the Security screen.
- 7. Manually fly the quadcopter away from and back towards the mobile platform. Observe the indications on the LCD screen indicate that the quadcopter is friendly.
  - a. Various indications on the LCD display will verify friendly target.
  - b. Turret Request to Fire should not light.
- 8. Un-pair the quadcopter from the mobile platform and repeat step 7.
  - a. The LCD screen should indicate hostile target.
  - b. The turret should track and fire its missile.



9. Land the quadcopter on the ground and manually place it on the mobile platform.
10. Re-pair the quadcopter to the mobile platform.
11. Switch to the second quadcopter and repeat steps 2 – 9.

### 20.13 Auto landing test

1. Establish a testing area and update the Ardupilot software with that location.
2. Using the mobile platform and quadcopter tests as reference, power up the mobile platform and the quadcopter for the initial run.
3. Move the mobile platform to a fixed location and make sure the location is updated in Ardupilot. Also, verify the home location of the quadcopter matches the location of the mobile platform.
  - a. This location should be in the same vicinity as the control terminals.
4. Disengage the quad-locking system for the testing quadcopter.
5. Using the traditional radio controller, launch the quadcopter to a height of 2 ft. above the mobile platform.
6. Switch the LCD display to the Auto Landing screen.
7. First auto landing test
  - a. Using the radio controller, rotate the quadcopter 360° and line-up the legs of the quadcopter with the lasers on the mobile platform and verify all 4 photocells are illuminated using the Auto Landing screen.
  - b. Initiate the auto landing sequence and observe that the quadcopter lands correctly and that the quad-locking system engages.
8. Second auto landing test
  - a. Disengage the quad-locking system.
  - b. Using the traditional radio controller, launch the quadcopter to a height of 5 ft. above ground level.
  - c. Manually fly the quadcopter approximately 50 ft. away and > 10 ft. altitude.
  - d. Initiate the auto home sequence and observe that the quadcopter returns to the mobile platform and is approximately 2 ft. above the mobile platform.
  - e. Initiate the auto landing sequence and observe that the quadcopter automatically aligns itself to the lasers and then lands correctly on the mobile platform. Verify that the quad-locking system engages.
9. Switch to the second quadcopter and repeat steps 2 – 8.

### 20.14 Overall system test

1. Establish a testing area and update the Ardupilot software with that location.
2. Using the mobile platform and quadcopter tests as reference, power up the mobile platform and the quadcopter for the initial run.
3. Move the mobile platform to a fixed location and make sure the location is updated in Ardupilot. Also, verify the home location of the quadcopter matches the location of the mobile platform.

- a. This location should be in the same vicinity as the control terminals.
4. Establish two different six waypoint routes, starting and ending at the current location of the mobile platform.
  - a. The first waypoint for each route should be at the current location, but some altitude above the mobile platform (i.e. >5 ft. above ground level).
  - b. Waypoints 2 and 4 should be in the same vicinity for each route, to minimize relocation of each team members between both routes.
5. Have two team members go to the approximate locations of waypoints 2 and 4 using their cell phone GPS.
6. Disengage the quad-locking system for the first quadcopter.
7. Implement the first of the two routes and verify that the first quadcopter goes to each waypoint.
8. Once the first quadcopter returns to the mobile platform, initiate the auto landing sequence and verify that the quad-locking system engages.
9. Disengage the quad-locking system for the second quadcopter.
10. Implement the second of the two routes and verify that the second quadcopter goes to each waypoint.
11. Once the second quadcopter returns to the mobile platform, initiate the auto landing sequence and verify that the quad-locking system engages.
12. Disengage the quad-locking system for the first quadcopter.
13. Implement the second of the two routes and verify that the first quadcopter goes to each waypoint.
  - a. Switching which quadcopter performs which route allows us to test the dynamics of the waypoint system for each quadcopter.
14. While the first quadcopter is running through the waypoint routes, move the mobile platform some distance away (>20 ft.).
  - a. Send a third team member with the mobile platform for verification of the quadcopter returning and landing properly.
15. When the first quadcopter completes the second route, implement the auto home feature and verify that it returns to the current location of the mobile platform.
16. When the first quadcopter returns to the mobile platform, implement the auto landing sequence and verify that the quad-locking system engages.
17. Disengage the quad-locking system for the second quadcopter.
18. Implement the first of the two waypoint routes and verify that the second quadcopter goes to each waypoint.
19. While the second quadcopter is running though the first route, move the mobile platform some distance away (>20 ft.), but not back to its original position.
20. When the second quadcopter returns to the mobile platform, implement the auto landing sequence and verify that the quad-locking system engages.

This concludes all system testing. Once these tests have been completed and passed, the project will have been proven a success. However, there is still more

work to be done. What follows will be the part of the project that will take considerable more time to fully understand what this project is actually capable of.

## 20.15 Characterization

During the development cycle of any new product, most of the time spent on development is characterizing the product. In many cases, ranges of values have already been anticipated for the product, based on the needs of the client or consumer. Many of these values are attained through market research or directly from the client. For this project, we are our own clients and have established some desired ranges of various features, based on components we chose or expectations considered during the design phase. This section will explore those features.

Because of the dynamic nature we aspired for this project, there will not be a firm procedure to characterizing the entire system. As such, the plans that follow will be guidelines on the various aspects we hope to understand better about our project, and subject to change.

## 20.16 Range

The intended range is limited to the communications of the Xbee component. The anticipated range is 1 mile.

1. Establish a testing area and update the Ardupilot software with that location.
  - a. The ideal area for this characterization should be an open field to maintain line of sight to the mobile platform as well as the quadcopters.
2. Using the mobile platform and quadcopter tests as reference, power up the mobile platform and the quadcopter for the initial run.
3. Using handheld GPS; divide the team into two groups and separate the mobile platform and quadcopters from the control terminal to locations that are  $\frac{3}{4}$  of a mile apart.
  - a. The traditional remote controller will be with the team that is away from the control terminal, but will remain powered down.
4. Launch each of the quadcopters and land them on the ground next to the mobile platform. Leave one team member with the quadcopters.
5. Coordinating between each team, increase the range of the mobile platform incrementally and record the distance achieved for each increment. Make note of when the mobile platform no longer responds to commands from the Ardupilot software.
  - a. Also record other environmental conditions such as wind speed and direction, temperature, cloud cover, and so on.
6. Once the mobile platform is outside of the communication range of the control terminal, switch to one of the quadcopters and repeat step 5.
7. Repeat step 5 again with the second quadcopter.

## 20.17 Speed and battery life

Characterizing these features will help us to understand the life cycle that each quadcopter will be limited to for each mission. It will also allow us to understand the max duration with which the mobile platform can sustain in the field. Speed will be characterized in parallel, because the higher the sustained speed, the shorter the battery lifetime will be.

1. Establish a testing area and update the Ardupilot software with that location.
2. Using the mobile platform and quadcopter tests as reference, power up the mobile platform and the quadcopter for the initial run.
  - a. Make note of time to start recording the flight duration.
3. Move the mobile platform to a fixed location and make sure the location is updated in Ardupilot. Also, verify the home location of the quadcopter matches the location of the mobile platform.
  - a. This location should be in the same vicinity as the control terminals.
4. Have two team members relocate to a significant distance away ( $\geq \frac{1}{2}$  miles).
  - a. Use GPS and/or Ardupilot software to calculate the actual distance.
5. Using the traditional radio controller, raise the quadcopter to a height of 10 ft.
  - a. The traditional radio controller will be used to maximize the speed that the quadcopters can attain.
6. Coordinating with the remote team, start timing the quadcopter as it moves between teams and make note of the travel times. Keep track of the quadcopter battery voltage as well.
7. Perform these runs several times to acquire lots of data points and to run the battery down.
8. Once the quadcopter battery attains a minimum voltage (~11.1V), record the time and determine the duration of flight.
9. When finished testing the first quadcopter, return it to the mobile platform and prepare the second quadcopter for launch.
10. Establish a two waypoint route in the Ardupilot software that matches closely the manual run that was just completed.
  - a. Both runs should be a simple back and forth pattern that the quadcopter will run through, once at high speed and then again at a more moderate speed.
11. Run the second quadcopter along this route repeatedly until the battery voltage drops to the minimum voltage (~11.1V). Record the time to get to this point as well as the voltage. Make note of any other observations that may affect the speed and/or flight duration of either quadcopter.
12. Use this data to calculate the speeds for each run and average the speeds for each quadcopter.
13. Lastly, move the mobile platform in a straight path to the remote team and record the time to cover the distance.
  - a. For the first part, operate the mobile platform at full speed for about 1 minute and record the change in distance on the Ardupilot software. Do this at least 3 times.

- b. Once the top speed is complete test, operate the mobile platform at half speed for the rest of the distance and record the time it takes to get to the remote team. Use this to calculate 50% speed.

## 20.18 Quadcopter load capacity

Understanding the load capacity for this configuration of quadcopters will be useful to determine what other devices might be connected to it, including additional battery packs to extend its life. There will be a trade off in battery duration due to this extra weight. However, characterizing this will be useful for understanding how to optimize the system for different loads.

1. Use the speed and battery life guidelines for this characterization, focusing solely on the quadcopters.
2. First determine the weight capacity of the quadcopter and compare to the designed weight capacity. Use known or measureable weights to accomplish this. The quadcopter should attain a height of at least 10 ft. for each weight.
3. Once the weight capacity has been determined, run the back-and-forth route with the each of the weights used and determine the voltages, times and duration of the flights.
  - a. Use a full charge for each run to determine maximum flight duration at each weight.
4. Graph this data.

## 20.19 Bedini motor characterization - Duration of battery cycles

This will be a long-term, continuous characterization through the duration of all testing. Understanding the power being supplied to the system, including charging of the other batteries, will be instrumental in calculating the efficiency and regenerative ability of the Bedini motor. Prior to each day of testing, the Bedini motor will charge overnight using a 12V, plug-in power supply will be used to operate the Bedini motor and charge all four of the batteries.

1. Whenever the rotor starts spinning, make note of the time it starts and the ID number of the battery that is connected to the drive portion of the circuit. (ID #: 1, 2, 3, and 4).
2. Whenever the Bedini motor changes the driving battery (determined by voltage), record the time that the battery changed. Record the ID of the new battery.
3. Each time the rotor is started, a different battery should be used as the initial driving battery for the system. This ensures that the batteries will get cycled through and have a more even usage time.

Auto landing during platform movement

The last feature that we would like to characterize is the capability of landing a quadcopter while the mobile platform is changing its location. This is intended as a

superfluous feature, but it would be interesting to see how well (or not) our system would be able to accomplish this.

1. Use the auto landing testing procedure as the guideline for this characterization.
2. Modify the procedure so that the quadcopter auto landing feature tries to land while the mobile platform is moving at 10%, 25%, 50%, and full speeds, as applicable.
3. Record success or failure (or not attempted) at each of these speeds.

## **20.20 Test Summary and Recording**

The following charts will be used as a summary of the above procedures and a place for recording the various parameters to be measured. During any testing, additional copies will be made as needed to ensure that data is being recorded during all trials.

Mobile Platform Testing		Test Date:	
	Expected Values	Measured Values	
Driving battery voltages and ID	$\geq 14V$	Start: ID:	
		Finish: ID:	
LEDs and indicators	List all		
Bedini monitoring	2200-2400 RPM Input: 12.5V @ 0.6A Output: varies	RPM:	$V_{out}$ :
Time start:		$V_{in}$ :	$I_{out}$ :
Time finish:		$I_{in}$ :	$t_{ss}$ :
Drive system (half-speed)	Current? Speed?	Fwd:	Rev:
		Left:	Right:
Drive system (full-speed)	Current? Speed?	Fwd:	Rev:
		Left:	Right:
Turret system (~36")	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail Measured vs. displayed ranges	Pan: <input type="checkbox"/> Tilt: <input type="checkbox"/> Motion: <input type="checkbox"/> Armed: <input type="checkbox"/> Fire: <input type="checkbox"/>	Range(s):
Turret system (72", 120", 240", etc...)	Measured vs. displayed ranges	Range(s):	
Quad-locking system	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	platform: <input type="checkbox"/> LCD: <input type="checkbox"/>	
GPS	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	Ardupilot pre-movement: <input type="checkbox"/> Ardupilot post-movement: <input type="checkbox"/>	



(This page intentionally left blank.)

Quadcopter Testing I / II		Test Date:
	Expected Values	Measured Values
<b>Pre-flight checks</b>		
Li-po battery voltages and time	$\geq 11.1V$	Start:  Finish:
LEDs and indicators	List all	
Quad-locking system engaged	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	platform: <input type="checkbox"/> LCD: <input type="checkbox"/>
Prop rotation	Rotation will be CW or CCW	Front-Lt:                  Front-Rt:  Back-Lt:                  Back-Rt:
Prop speed	RPM change as function of steering direction? <input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	Fwd: <input type="checkbox"/> Rev: <input type="checkbox"/>  Left: <input type="checkbox"/> Right: <input type="checkbox"/>
GPS	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	Ardupilot pre-movement: <input type="checkbox"/> Ardupilot post-movement: <input type="checkbox"/>
Auto home follows mobile platform	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	Ardupilot pre-movement: <input type="checkbox"/> Ardupilot post-movement: <input type="checkbox"/>
Security		
<b>Low altitude flight test</b>		

Quad-locking system disengaged	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	platform: <input type="checkbox"/> LCD: <input type="checkbox"/>
Ardupilot altitude	4ft. from ground level	Altitude:
Hovering system	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	Hovering: <input type="checkbox"/>

Additional Notes:

System Testing		Test Date:
	Expected Values	Measured Values
<b>Waypoint Testing</b>		<b>Quadcopter Testing I / II</b>
Driving battery voltages and ID	$\geq 14V$	Start: ID: Finish: ID:
Bedini monitoring Time start: Time finish:	2200-2400 RPM Input: 12.5V @ 0.6A Output: varies	RPM: $V_{out}$ : $V_{in}$ : $I_{out}$ : $I_{in}$ : $t_{ss}$ :
Route 1 3 waypoint route	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	First waypoint: <input type="checkbox"/> Second waypoint: <input type="checkbox"/> Third waypoint: <input type="checkbox"/>
Route 2 5 waypoint route, circular	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	First waypoint: <input type="checkbox"/> Second waypoint: <input type="checkbox"/> Third waypoint: <input type="checkbox"/> Fourth waypoint: <input type="checkbox"/> Fifth waypoint: <input type="checkbox"/>
Route 3 5 waypoint route, star	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	First waypoint: <input type="checkbox"/> Second waypoint: <input type="checkbox"/> Third waypoint: <input type="checkbox"/> Fourth waypoint: <input type="checkbox"/> Fifth waypoint: <input type="checkbox"/>

Additional Notes:

System Testing		Test Date:	
	Expected Values	Measured Values	
<b>Waypoint Testing</b>		<b>Quadcopter Testing I / II</b>	
Driving battery voltages and ID	$\geq 14V$	Start: ID:	
		Finish: ID:	
Bedini monitoring	2200-2400 RPM	RPM:	$V_{out}$ :
Time start:	Input: 12.5V @ 0.6A	$V_{in}$ :	$I_{out}$ :
Time finish:	Output: varies	$I_{in}$ :	$t_{ss}$ :
Auto home test 1	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	Moves to mobile platform: <input type="checkbox"/>	
		Altitude:	
Auto home test 2	Move mobile platform > 50 ft.	Mobile platform moved: <input type="checkbox"/>	
		Distance traveled:	
Auto home test 2	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	Moves to mobile platform: <input type="checkbox"/>	
		Altitude:	

Additional Notes:

System Testing		Test Date:
	Expected Values	Measured Values
<b>Security and Turret Testing</b>		
Driving battery voltages and ID	$\geq 14V$	Start: ID: Finish: ID:
Bedini monitoring Time start: Time finish:	2200-2400 RPM Input: 12.5V @ 0.6A Output: varies	RPM: $V_{out}$ : $V_{in}$ : $I_{out}$ : $I_{in}$ : $t_{ss}$ :
Friend or Foe? (Friend)	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	Motion sensor activated: <input type="checkbox"/> Request to fire (not lit): <input type="checkbox"/>
Friend or Foe? (Foe)	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	Motion sensor activated: <input type="checkbox"/> Request to fire (lit): <input type="checkbox"/> Turret fires: <input type="checkbox"/>

Additional Notes:

System Testing		Test Date:	
	Expected Values	Measured Values	
Auto Landing Testing		Quadcopter Testing I / II	
Driving battery voltages and ID	$\geq 14V$	Start: ID:	
		Finish: ID:	
Bedini monitoring	2200-2400 RPM	RPM:	$V_{out}$ :
Time start:	Input: 12.5V @ 0.6A	$V_{in}$ :	$I_{out}$ :
Time finish:	Output: varies	$I_{in}$ :	$t_{ss}$ :
First landing test	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail Range when all lasers are aligned	Laser alignment: <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> Rangefinder: Quad-lock engaged: <input type="checkbox"/>	
Second landing test	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail Range when all lasers are aligned	Laser alignment: <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> Rangefinder: Quad-lock engaged: <input type="checkbox"/>	

Additional Notes:



System Testing		Test Date:	
	Expected Values	Measured Values	
<b>Overall System Testing</b>			
Driving battery voltages and ID	$\geq 14V$	Start: ID:	
		Finish: ID:	
LEDs and indicators	List all		
Bedini monitoring	2200-2400 RPM	RPM:	$V_{out}$ :
Time start:	Input: 12.5V @ 0.6A	$V_{in}$ :	$I_{out}$ :
Time finish:	Output: varies	$I_{in}$ :	$t_{ss}$ :
Route 1 Quadcopter 1	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	First waypoint: <input type="checkbox"/> Second waypoint: <input type="checkbox"/> Third waypoint: <input type="checkbox"/> Fourth waypoint: <input type="checkbox"/> Fifth waypoint: <input type="checkbox"/> Sixth waypoint: <input type="checkbox"/> Auto landing sequence: <input type="checkbox"/> Quad-lock engaged: <input type="checkbox"/>	
Route 2 Quadcopter 2	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	First waypoint: <input type="checkbox"/> Second waypoint: <input type="checkbox"/> Third waypoint: <input type="checkbox"/> Fourth waypoint: <input type="checkbox"/> Fifth waypoint: <input type="checkbox"/> Sixth waypoint: <input type="checkbox"/> Auto landing sequence: <input type="checkbox"/> Quad-lock engaged: <input type="checkbox"/>	
Route 2 Quadcopter 1	<input checked="" type="checkbox"/> if pass <input checked="" type="checkbox"/> if fail	First waypoint: <input type="checkbox"/> Second waypoint: <input type="checkbox"/> Third waypoint: <input type="checkbox"/> Fourth waypoint: <input type="checkbox"/> Fifth waypoint: <input type="checkbox"/> Sixth waypoint: <input type="checkbox"/> Auto home feature: <input type="checkbox"/> Auto landing sequence: <input type="checkbox"/> Quad-lock engaged: <input type="checkbox"/>	

Route 1	<input checked="" type="checkbox"/> if pass	First waypoint:	<input type="checkbox"/>
Quadcopter 2	<input checked="" type="checkbox"/> if fail	Second waypoint:	<input type="checkbox"/>
		Third waypoint:	<input type="checkbox"/>
		Fourth waypoint:	<input type="checkbox"/>
		Fifth waypoint:	<input type="checkbox"/>
		Sixth waypoint:	<input type="checkbox"/>
		Auto home feature:	<input type="checkbox"/>
		Auto landing sequence:	<input type="checkbox"/>
		Quad-lock engaged:	<input type="checkbox"/>

Additional Notes:

## 21.0 Milestone Discussion

Developing the milestones started in the very beginning when we started discussing features for our project. Along with those features, we determined that there were going to be some easy and other more difficult features that would take time to research and develop. Based on the schedule we had at the time, we figured that the research aspect needed to be completed by the first half of the semester. At the same time, we felt that defining our project would need to be refined as we performed our research. Similarly, we allowed room for redefining our project through the first half of the semester.

Scheduling the design portion was a relatively easy task, once the block diagrams were developed. When we first started discussing our project, we talked about what were the different subsystems that comprised our project. We determined that our project comprised of 3 main subsystems: quadcopters, the mobile “battlestation” and our control terminal. There is a fourth subsystem, but the technical details were fairly minor; the wireless communications system; the details of which were being covered by the research. Realizing that some of the research would spill over into the design, we planned for about five weeks to get the design for the three main subsystems completed. The critical aspect of this was to ensure there would be plenty of time for putting all of our documentation together.

### Milestones for Senior Design I

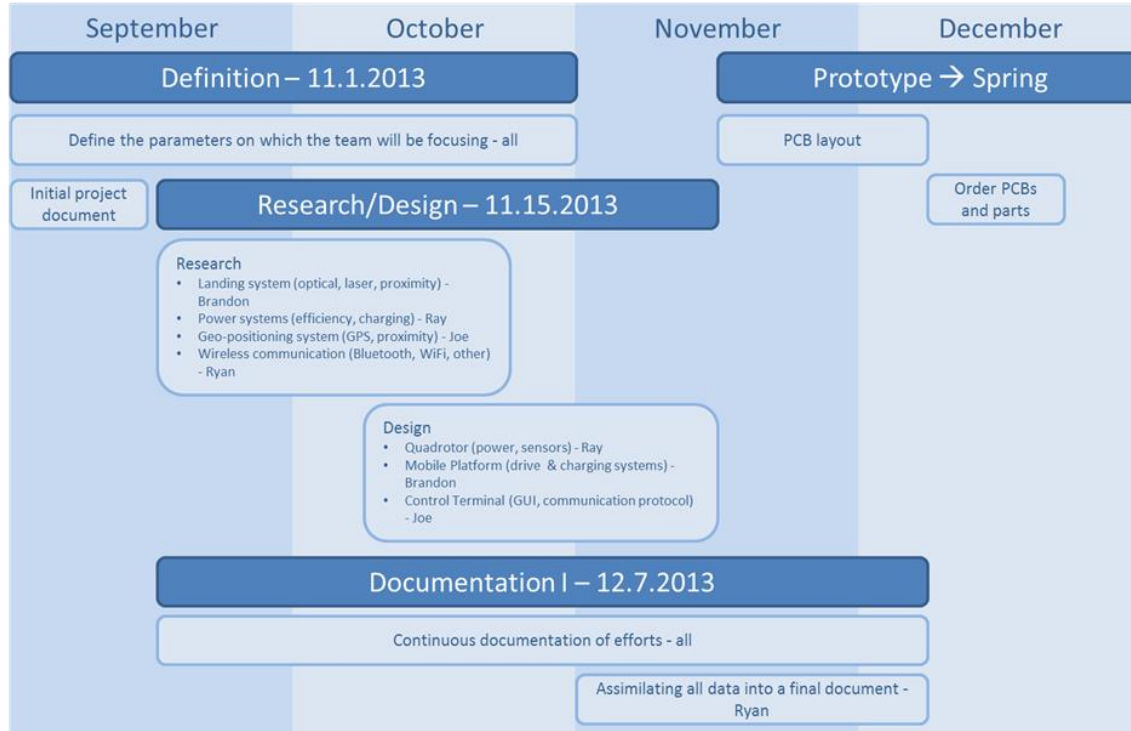


Figure 81: Milestone diagram for fall semester.

Documentation was going to be a combined effort. As each of us completed our respective parts for the project, it was our responsibility to make sure we documented all of our information. Because the final document must be at least 120 pages, we also needed to ensure there would be sufficient time to go through the entire document and make sure all relevant data was provided, check over the grammatical content, and allow time to get any missing data that needed to be added to the document.

In the context of the overall project, we didn't want to wait until Senior Design II to order PCB's and parts. We wanted to hit the ground running come the following semester, so we included PCB layout and ordering in the fall. The intent was to have our prototypes built within the first couple of weeks of the following semester and get testing almost immediately. We also allowed some room to have revisions printed of those PCBs, as needed.

Prototype and testing would comprise the entirety of Senior Design II. As such, laying out the schedule for spring was simple. Understanding that there would be some changes to that schedule, we intended to have most of our testing done by the middle of the semester. Then we would evaluate if any changes would need to be made to any aspect of the project and retest everything again.

## Milestones for Senior Design II

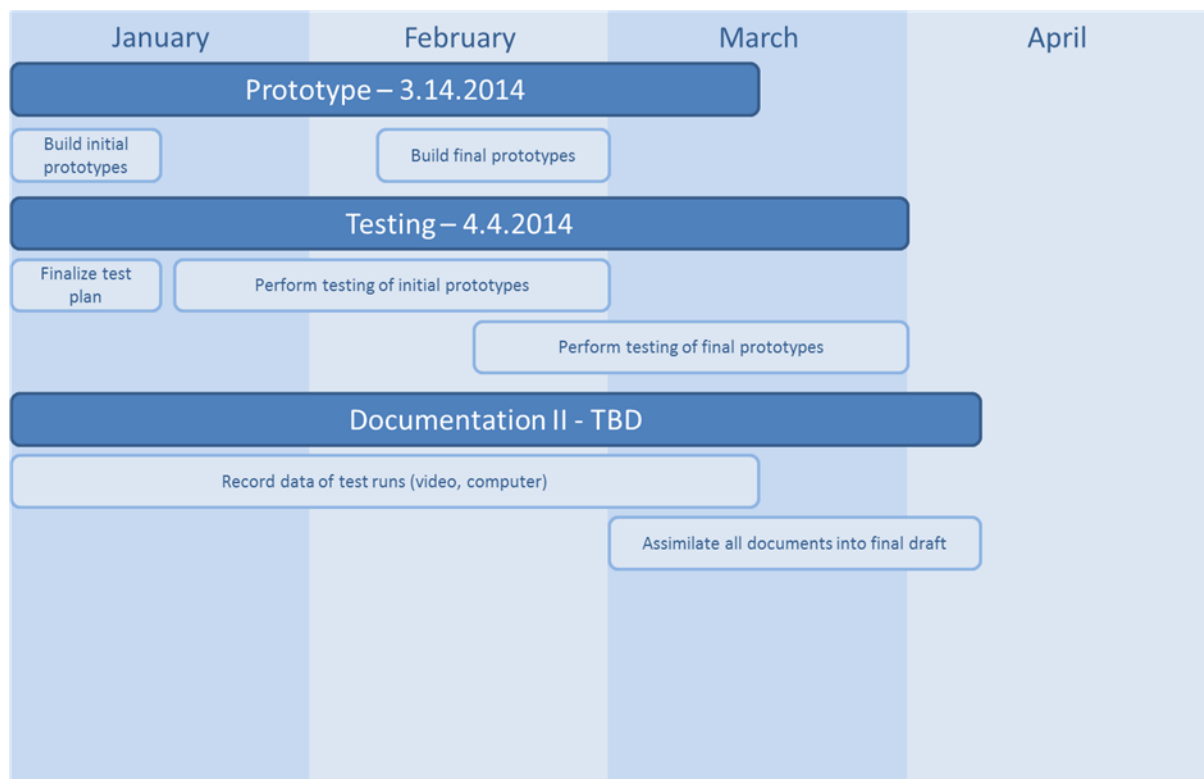


Figure 82: Milestone diagram for spring semester.

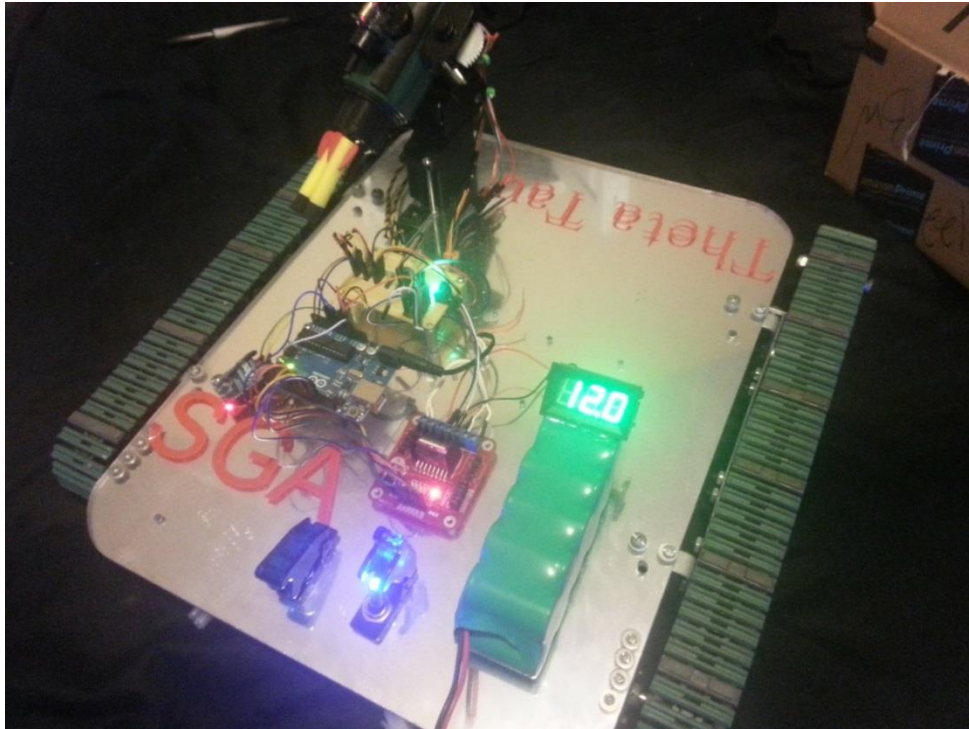
We intended to modify the milestones for spring once we had a better idea of what the schedule should be. We figured that having prototypes developed so early, it would give us plenty of time to adjust as necessary. However, we knew that there were going to be meetings and a final review board that would need to be scheduled. It would have been nice for that information to be available, but there would be time for adjustment for those items later.

Documentation for spring semester was going to be much more data oriented. Part of the documentation for fall would be to write up a preliminary test plan. This would help us to understand how much data needed to be taken and what our goals were to prove that the project was a success. We anticipated that there would be a lot of data, as we had 4 subsystems that needed to be proven out, as well as the overall system working together. Once the new semester started, we would re-evaluate the test plan and make any modifications during the building phase.

The last aspect of the documentation would be preparation for the final board review. This would include presentations, videos, and demonstrations. A lot of this preparation would be done in parallel to the rest of the documentation, which is why it wasn't listed as a separate task on the milestone chart.

## 22.0 Current Prototypes

Figure 83 and Figure 84 display the current progress of our prototypes. These prototypes are not to scale and only being used as development platforms. The platform will end up being much larger and several more features on it such as the Bedini motor, four lead acid batteries, the quad locking system, and our custom PCB's along with other small devices. The controller is a larger than our final design will be and also does not have the same layout. These are being used for functional testing to provide quick changes during the testing and prototyping stage.

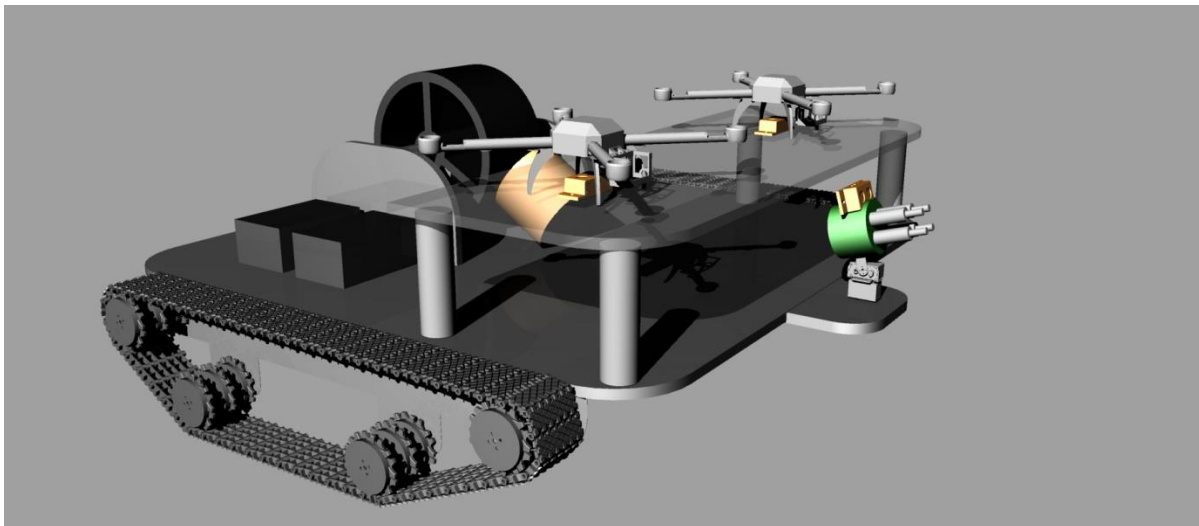


**Figure 83: Small Scale Platform Prototype**



**Figure 84: Platform Controller Prototype**

When designing the platform we realized that there is an endless amount of ways we could layout the physical structure of the platform. The size of the platform also became a concern once we started laying out the measurements. In order to aid us in exploring all possible outcomes of the structure of the platform we started doing 3D renderings in Rhino3D to model it. After several revisions and scaling the platform we came up with the design shown in Figure 85.



**Figure 85: 3D Model of Quadcopter Platform**



## 23.0 Budget and Finances

### 23.1 Funding

We chose a very ambitious project with a lot of features and subsystems. Our project also included several different mechanical features that we did not want to spend the whole semester trying to build ourselves. We will therefore require purchasing most of the mechanical parts of the project versus building them. We soon discovered that with the amount of motors and development parts we would need for this project, the cost increase rapidly. We have designed our project such that it is very modular and if the funds did not work out or the subsystem does not work it will not deter the whole project.

We ended up becoming very fortunate and have received outside funding. Our main sponsor will be Duke Energy through the University of Central Florida providing us financial support. We were able to gain the support from Duke Energy due to the energy conservation needs of our project. We are displaying a form of alternative energy and performing a sustainable system.

We also have a few smaller sponsors that have helped us through the development phase and getting started. These sponsors include:

- *Monica Bertram*: Mechanical Engineer for Siemens Energy
- *Matthew Harrison*: Mechanical Engineer for 3D Medical Manufacturing
- *Sean DeVecchio*: Mechanical Engineer for Lockheed Martin MFC
- *Steven Darrow*: Aerospace Engineering Graduate Student at UCF
- *Theta Tau Professional Engineering Fraternity*: Rho Gamma Chapter

These sponsors have provided financial support as well as advice for some of our flight considerations and mechanical designs.

We wanted to see the actual overall cost of the project including all of development and everything it would take to actually achieve this project. This included things that are often overlooked such as soldering irons and accessories necessary for building the components. We felt that if we would ever need to present this to a company to display the true cost of the project, this would depict a more accurate representation of the project as a whole. Table 5 below shows our current Bill of Materials.

<i>Project Section</i>	<i>Cost PU</i>	<i>Amount</i>	<i>Total Cost</i>
<b>Quadcopters (x2)</b>			
Rotary Motors	\$ 10.53	8	\$ 84.24
Speed Controller	\$ 8.20	8	\$ 65.60
Flight Controller	\$ 59.95	1	\$ 59.95
Propellers	\$ 3.49	10	\$ 34.90
Wireless Receiver	\$ 10.06	2	\$ 20.12
Servo Connector Wires	\$ 1.42	8	\$ 11.36
Futaba Servo Terminals (10/set)	\$ 2.25	4	\$ 9.00
LIPO Battery Pack	\$ 5.65	4	\$ 22.60
Turnigy Silicone Wire /meter	\$ 0.60	10	\$ 6.00
Altimeter	\$ 14.95	2	\$ 29.90
Self Tapping Screws (100)	\$ 1.95	1	\$ 1.95
Magnetometer	\$ 17.99	2	\$ 35.98
Xbee Pro (wireless communication)	\$ 29.00	2	\$ 58.00
GPS	\$ 59.95	2	\$ 119.90
Ultrasonic Range Finder	\$ 29.95	2	\$ 59.90
SD Reader/Writer	\$ 6.95	2	\$ 13.90
Photocells (for laser guidance)	\$ 14.95	1	\$ 14.95
Solar Panel	\$ 24.95	2	\$ 49.90
<b>Quad Copter(s) Total:</b>			<b>\$ 698.15</b>
<b>Mobile Platform</b>			
Planetary Geared Motors	\$ 36.95	4	\$ 147.80
Motor Hub Mounts	\$ 7.65	4	\$ 30.60
Motor Mounts	\$ 7.95	4	\$ 31.80
Vex Tank Tread Kit	\$ 29.95	4	\$ 119.80
Laser Diode (x30)	\$ 10.00	8	\$ 80.00
Plexi for Platform (24"x24")	\$ 23.99	1	\$ 23.99
2800 mAH 12V NiMh battery	\$ 39.99	1	\$ 39.99
Vex Angle Brackets(for motors)	\$ 17.99	1	\$ 17.99
GPS	\$ 59.95	1	\$ 59.95
Xbee Pro (wireless communication)	\$ 29.00	1	\$ 29.00
Nerf Turret Launcher	\$ 29.99	2	\$ 59.98
Servos for Turret Launcher	\$ 44.90	2	\$ 89.80
Servo Brackets	\$ 14.99	2	\$ 29.98
4-Wheel Drive Motor Controller	\$ 24.95	1	\$ 24.95
Locking Landing System	\$ 31.49	1	\$ 31.49
Range Finder	\$ 29.95	2	\$ 59.90
SD Reader/Writer	\$ 6.95	1	\$ 6.95
LCD	\$ 14.95	1	\$ 14.95
Solar Panel	\$ 24.95	1	\$ 24.95
<b>Mobile Platform Development Total:</b>			<b>\$ 923.87</b>
<b>Control Station</b>			
4D Systems Touch Screen LCD	\$ 149.95	1	\$ 149.95
Joystick	\$ 41.99	1	\$ 41.99
Buttons/Switches	\$ 9.99	6	\$ 59.94
Variable Drive Encoders	\$ 4.99	8	\$ 39.92
LED Indicators	\$ 2.99	10	\$ 29.90
Enclosure and Materials	\$ 40.00	1	\$ 40.00
Xbee Pro (wireless communication)	\$ 29.00	2	\$ 58.00
<b>Control Station Total</b>			<b>\$ 419.70</b>
<b>Additional Development Tools</b>			
Breadboards	\$ 8.00	4	\$ 32.00
Chipkit32 Uno MCU Dev Kit	\$ 84.85	1	\$ 84.85
chipKIT PMOD Dev Board	\$ 21.99	1	\$ 21.99
ChipKIT DP32 Dev Board	\$ 23.99	1	\$ 23.99
chipKIT CMOD Dev Board	\$ 29.00	1	\$ 29.00
Cerebot MX7cK Dev Board	\$ 99.99	1	\$ 99.99
Motor Control Dev Board	\$ 29.99	4	\$ 119.96
ABS Filament for 3d Printer	\$ 90.00	2	\$ 180.00
Additional Parts(screws, wires, etc)	\$ 150.00	1	\$ 150.00
PCB Manufacturing	\$ 66.00	6	\$ 396.00
PCB Components	\$ 150.00	1	\$ 150.00
Bedini Motor Deveopment	\$ 200.00	1	\$ 200.00
Soldering Iron	\$ 83.99	1	\$ 83.99
Solder	\$ 6.22	2	\$ 12.44
Solder Paste	\$ 10.45	2	\$ 20.90
Solder Sucker	\$ 5.99	1	\$ 5.99
Solder Wick	\$ 5.00	2	\$ 10.00
<b>Development Tools Total:</b>			<b>\$ 1,621.10</b>
<b>Total Project Cost:</b>			<b>\$ 3,662.82</b>

**Table 9: Bill of Materials**

## 24.0 Bibliography

- 3DR Radio Telemetry. n.d. <<http://store.3drobotics.com/products/3dr-radio>>.
- 78M12 Data Sheet. n.d. <<http://www.alldatasheet.com/datasheet-pdf/pdf/260981/KEXIN/78M12.html>>.
- Advanced Precision Composites. APC Suggested RPM Limits. n.d. 13 October 2013 <[http://www.apcprop.com/v/html/rpm\\_limits.html](http://www.apcprop.com/v/html/rpm_limits.html)>.
- APM 2.6. n.d. <<http://store.3drobotics.com/products/apm-2-6-kit-1>>.
- APM Power Module. n.d. <<http://store.3drobotics.com/products/apm-power-module-with-xt60-connectors>>.
- AT45DB161D-MU Data Sheet. n.d. <<http://www.mouser.com/ds/2/590/doc3500-219593.pdf>>.
- Atmega 2560 Data Sheet. n.d. <<http://www.atmel.com/Images/2549S.pdf>>.
- Bedini Motors. 2 July 2010. September 2013 <<http://bedinimotors.blogspot.com/2010/07/bedini-motor-clarification.html>>.
- Borgobello, Bridget. 27 March 2013. 15 September 2013 <<http://www.gizmag.com/solarcopter-solar-helicopter/26645/>>.
- Bourke, Jim. Understanding Electric Power Systems - Part 4. 1 December 1998. October 2013 <<http://www.rcgroups.com/forums/showthread.php?t=393252>>.
- . Understanding Electric Power Systems - Part 5. 1 January 1999. October 2013 <<http://www.rcgroups.com/forums/showthread.php?t=337735>>.
- . Understanding Electric Power Systems Part 1. 1 September 1998. October 2013 <<http://www.rcgroups.com/forums/showthread.php?t=333326>>.
- . Understanding Electric Power Systems Part 2. 1 October 1998. October 2013 <<http://www.rcgroups.com/forums/showthread.php?t=393253>>.
- . Understanding Electric Power Systems Part 3. 1 November 1998. October 2013 <<http://www.rcgroups.com/forums/showthread.php?t=393251>>.
- Buchmann, Isidor. Battery University. 2013. September 2013 <<http://batteryuniversity.com/>>.

ChipKIT. MPIDE. Ed. Under the GNU General Public License. 2012. 2013 <[www.chipkit.org](http://www.chipkit.org)>.

CSAW. "Embedded Systems Challenge under the GNU General Public License." 2009.

Cytron. Cytron Technologies Rotary Encoder: License: Use of Cytron Technologies's products as critical components in life support systems is not authorized except with express written approval by Cytron Technologies. No licenses are conveyed, implicitly or otherwise. 2009. 2013 <<http://www.cytron.com.my/datasheet/sensor/RE08A%20User's%20Manual.pdf>>.

Dana, Peter H. Global Positioning System Overview. 1 May 2000. 12 October 2013 <[http://www.colorado.edu/geography/gcraft/notes/gps/gps\\_f.html](http://www.colorado.edu/geography/gcraft/notes/gps/gps_f.html)>.

Digilent. ChipKIT PIC32. Ed. License: Creative Commons — Attribution-ShareAlike. 2012. 2013 <[www.digilentinc.com](http://www.digilentinc.com)>.

DIY Drones. The Mission Planner utility. 2013. 21 November 2013 <<https://code.google.com/p/ardupilot-mega/wiki/Mission>>.

Electronics, Spark Fun. n.d.

Foscam. Amazon Network Camera. 2010. 2013 <[http://www.amazon.com/Foscam-FI8910W-Network-Camera-Two-Way/dp/B006ZPWS4U/ref=sr\\_1\\_1?ie=UTF8&qid=1385609687&sr=8-1&keywords=foscam+network+camera](http://www.amazon.com/Foscam-FI8910W-Network-Camera-Two-Way/dp/B006ZPWS4U/ref=sr_1_1?ie=UTF8&qid=1385609687&sr=8-1&keywords=foscam+network+camera)>.

Friis Equation. n.d. <<http://www.antenna-theory.com/basics/friis.php>>.

Gladman, Brian. "AES Implementation." 2009.

GNU. "GNU Linux Operating System." 1984. GNU. 2013 <[www.gnu.org](http://www.gnu.org)>.

GoPro. GoPro Hero3. 2013. 2013 <<http://gopro.com/products/>>.

HMC588L Data Sheet. n.d. <<http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Sensors/Magneto/HMC5883-L-FDS.pdf>>.

Hobbico. The Futaba 2.4GHz Story. 2013. 11 November 2013 <<http://www.futaba-rc.com/technology/fasst.html>>.

Hobye, Mads. "GUINO." 2012.

Instruments, Texas. Quadruple Half H-Driver. 1995. 2013  
<https://www.sparkfun.com/datasheets/IC/SN754410.pdf>.

Jin, Dr. Yier. Alpha System Project 2: Hardware Security and Trusted Circuit Design - University of Central Florida. Orlando: CSAW, 2013.

King, Hobby. Hobby King Store. 2013. 2013  
<http://hobbyking.com/hobbyking/store>.

KJMagnetics. n.d. <http://www.gwlink.com/design-guide.aspx>.

KJMagnetics Magnetic Field Calculator. n.d.

KK2 Flight Controller Board. n.d.  
[http://hobbyking.com/hobbyking/store/\\_\\_24723\\_\\_Hobbyking\\_KK2\\_0\\_Multi\\_rotor\\_LCD\\_Flight\\_Control\\_Board.htm](http://hobbyking.com/hobbyking/store/__24723__Hobbyking_KK2_0_Multi_rotor_LCD_Flight_Control_Board.htm).

Liang, Oscar. How To Choose RC Transmitter For Quadcopter. 3 October 2013. 10 November 2013 <http://blog.oscarliang.net/choose-rc-transmitter-quadcopter/>.

Linear Actuator. n.d. <http://www.firgelli.com/products.php>.

Linear Solenoid Design Guide and Technical Information. n.d.  
<http://www.gwlink.com/design-guide.aspx>.

LM 2576 Data Sheet. n.d. <http://www.ti.com/lit/ds/snvs107c/snvs107c.pdf>.

LM 2931 Data Sheet. n.d. <http://www.ti.com/lit/ds/symlink/lm2931-n.pdf>.

Meier, Lorenz. QGroundControl Overview. 2013. 12 November 2013  
<http://qgroundcontrol.org/>.

MPU 6000 Data Sheet. n.d.  
<http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Components/General%20IC/PS-MPU-6000A.pdf>.

MS5611. n.d. <http://www.meas-spec.com/downloads/MS5611-01BA03.pdf>.

OpenPilot Forums. 23 October 2012. 29 October 2013  
<http://forums.openpilot.org/topic/14422-h-frame-or-stretched-x-frameqav500ov3rquad/>.

Pi, Raspberry. Raspberry Pi Specs. 2010. 2013 [www.raspberrypi.org](http://www.raspberrypi.org).

Pistoia, Giangranco. Industrial applications of batteries: from cars to aerospace and energy storage. Elsevier, 2007.

Quadcopters DIY. Frame. 2013. 30 October 2013  
<<http://www.quadcoptersdiy.com/information/flight-components/frame/>>.

RC Groups Forum. 12 March 2013. 29 October 2013  
<<http://www.rcgroups.com/forums/showthread.php?t=1850602>>.

RC Helicopter Fun. Understanding RC LiPo Batteries. 2008. October 2013  
<<http://www.rchelicopterfun.com/rc-lipo-batteries.html>>.

Reyes, Edgr. What is Propeller Pitch? n.d. 28 October 2013  
<[http://www.propellerpages.com/?c=articles&f=2006-03-08\\_what\\_is\\_propeller\\_pitch](http://www.propellerpages.com/?c=articles&f=2006-03-08_what_is_propeller_pitch)>.

SFE. Motor Controller. 2008. 2013 <<https://www.sparkfun.com/products/9571>>.

ST. ST L298. 2013. 2013  
<[http://www.st.com/web/catalog/sense\\_power/FM142/CL851/SC1790/SS1555/PF63147](http://www.st.com/web/catalog/sense_power/FM142/CL851/SC1790/SS1555/PF63147)>.

U.S. Government. Space Segment. 2013. 17 October 2013  
<<http://www.gps.gov/systems/gps/space/>>.

UBD5- Pic UAV. n.d. <<https://www.sparkfun.com/products/11703>>.

Ublox. C04-6H. 2013. 20 October 2013 <[http://www.u-blox.com/images/downloads/Product\\_Docs/C04-6H-ReferenceDesign\\_ProductSummary\\_\(GPS.G6-CS-10000\).pdf](http://www.u-blox.com/images/downloads/Product_Docs/C04-6H-ReferenceDesign_ProductSummary_(GPS.G6-CS-10000).pdf)>.

Xbee Pro Data Sheet. n.d. <[www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf](http://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf)>.

## 25.0 Appendix A: Copyright Permissions and Authorizations

### Permission to use Figure 33

Ryan,

The content on our website is copyrighted, and legally belongs to K&J Magnetics. That said, it's also public information. You are free to use content, images and calculator output in your paper, just as long as you're not trying to pass it off as your own work. A footnote about where the info comes from should really be all you need.

Best Regards,

Michael Paul  
K&J Magnetics, Inc.  
[www.kjmagnetics.com](http://www.kjmagnetics.com)

On 11/27/2013 1:22 AM, Ryan Borden wrote:

Greetings,

I am using the KJM magnet calculator in my senior engineering project, I would like to have permission to use the graphs and images in my paper. Giving KJM credit of course.

Sincerely,  
Ryan Borden

### Permission to use Figure 35

**Product Photos:** SparkFun product photos may be used without permission for educational purposes (research papers, school projects, etc.). However, permission must be granted for commercial use and proper credit to SparkFun must be given. For inquiries about the use of our product photos or permission to use them, please contact [marketing@sparkfun.com](mailto:marketing@sparkfun.com).

### Atmel Corporation

*"Materials from this website [www.atmel.com](http://www.atmel.com) and any other website owned, operated or controlled by Atmel and/or its affiliated or subsidiary companies (together, Atmel) are owned and copyrighted by Atmel. Unauthorized use of such Materials (e.g., information, documentation and software), including these Terms, may be a violation of Atmel's intellectual property rights or other applicable laws. If you agree to these Terms, you may download (on a single computer), copy or print a single copy of all or a portion of the Materials for informational, non-commercial, lawful purposes only."*

<http://www2.atmel.com/About/legal.aspx>

### Microchip

*"If you use Microchip copyrighted material solely for educational (non-profit) purposes falling under the "fair use" exception of the U.S. Copyright Act of 1976 then you do not need Microchip's written permission. For example, Microchip's permission is not required when using copyrighted material in: (1) an academic report, thesis, or dissertation; (2) classroom handouts or textbook; or (3) a*



*presentation or article that is solely educational in nature (e.g., technical article published in a magazine). Please note that offering Microchip copyrighted material at a trade show or industry conference for the purpose of promoting product sales does require Microchip's permission."*

[http://www.microchip.com/stellent/idcplg?IdcService=SS\\_GET\\_PAGE&nodeId=487&param=en023282](http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=487&param=en023282) <http://tinyurl.com/microchip-com>

#### Permission to use Figure 41, Figure 45



**Francisco (3Drobotics)**

Oct 28 04:24 pm (PDT)

Hello Ryan,  
Thank you for contacting 3DRobotics Tech Support.

I just went to weight a quadcopter similar to the description you gave and its around 2.687435 pounds.  
No problem in using the images or tables on the website, just mention and give credit to 3DRobotics please.

Thank you,  
Francisco.

#### Permission to Use Figure 16 and Figure 17

TI further grants permission to non-profit, educational institutions (specifically K-12, universities and community colleges) to download, reproduce, display and distribute the information on these pages solely for use in the classroom. This permission is conditioned on not modifying the information, retaining all copyright notices and including on all reproduced information the following credit line: "Courtesy of Texas Instruments". Please send us a note describing your use of this information under the permission granted in this paragraph. Send the note and describe the use according to the request for permission explained below.

## 26.0 Appendix B: References for Figures

Figure 1: Platform Motor Control.....	7
Figure 2: Platform Control Terminal Layout .....	8
Figure 3: <i>Cytron RE08A Rotary Encoder Kit</i> .....	9
Figure 4: Motor Control Feedback System .....	10
Figure 5: LCD Main Menu Screen.....	11
Figure 6: Platform Motor Control Screen .....	12
Figure 7: Platform Monitoring Screen .....	12
Figure 8: Security Systems Screen.....	13
Figure 9: Turret Control System Screen .....	13
Figure 10: Illustrative schematic of a simple Bedini motor circuit. ....	17
Figure 11: Development of the Bedini motor.....	19
Figure 12: Model of the rotor holding the permanent magnets. ....	20
Figure 13: One half of the spool holder for the dual coil.....	20
Figure 14: Bedini motor circuit including monitoring system. ....	21
Figure 15: Schematic of the charging and switching circuit connected to the Bedini motor circuit. ....	22
Figure 16: LM2576 Efficiency (reprinted with permission from Texas Instruments) .	25
Figure 17: LM2576 Configuration (reprinted with permission from Texas Instruments) .....	26
Figure 18: Power Supply Schematic.....	27
Figure 19: Security Initialization Procedure.....	29
Figure 20: Request to Fly Security Protocol.....	30
Figure 21: Request to Fire Protocol .....	31
Figure 22: Request to Land Protocol .....	32
Figure 23: Laser Landing System Layout .....	33
Figure 24: Platform Controller Laser Driver .....	35
Figure 25: Laser Landing System Software FSM .....	36
Figure 26: Wireless Camera System .....	37
Figure 27: Targeting System Component Layout .....	41
Figure 28: Target Acquisition FSM.....	42
Figure 29: Platform Controller Turret Control System .....	44
Figure 30: Development GUI for Platform Controls.....	46
Figure 31: Linear Actuator (permission pending) .....	48
Figure 32: Neodymium Magnet.....	49
Figure 33: Magnetic Pull With and Without a Gap (reprinted with permission from KJMagnetics) .....	50
Figure 34: Small Solenoid.....	50
Figure 35: Push-Pull Solenoid (reprinted with permission from Sparkfun) .....	51

Figure 36: Relay and Solenoid Configuration.....	52
Figure 37 : Side View Model of the Quad Locking System.....	52
Figure 38: SPI Configuration .....	55
Figure 39: Definitions of roll, pitch, and yaw .....	55
Figure 40: I2C Setup .....	56
Figure 41: 3D Robotics Telemetry System (reprinted with permission form 3D Robotics).....	60
Figure 42: Telemetry Diagram.....	61
Figure 43: Xbee Pro Modules (permission pending) .....	62
Figure 44: X-CTU Modem configuration (permission pending) .....	63
Figure 45: 3D Robotics Power Module (reprinted with permission from 3D Robotics) .....	64
Figure 46: Schematic of the Power Module (reprinted with permission from 3D Robotics).....	64
Figure 47: Output of Flight Controller .....	65
Figure 48: Layout of the Flight Controller Components .....	66
Figure 49: A solar-powered quadcopter built by master's students at Queen Mary University of London Permission Pending.....	67
Figure 50: Maxwell Technologies' Ultracapacitor with a rated voltage of 2.7V and a 3000F capacitance (permission pending).....	68
Figure 51: A lead-acid battery removed from a motorcycle .....	69
Figure 52: A NiCad battery that we removed from an 18V Dewalt cordless drill .....	70
Figure 53: AA cell 1.2V NiMH battery with a 2400mAh capacity (permission pending) .....	71
Figure 54: A 14.4V Li-ion battery that we removed from an Asus notebook.....	72
Figure 55: A 3.7V Li-Po battery that we removed from a GoPro camera .....	73
Figure 56: iMax B6AC Li-Po battery charger/discharger (permission pending).....	74
Figure 57: An example of how a propellers pitch is measured (permission pending) .....	77
Figure 58: The calculation to find the Power absorbed by the propeller, given the propellers diameter, pitch, constant, and RPM.....	77
Figure 59: The Zippy Flightmax 2800mAh 3S1P 30C Li-Po battery sold by HobbyKing.com (permission pending).....	79
Figure 60: A power distribution board for a quadcopter that is sold for \$4 on HobbyKing.com (permission pending).....	80
Figure 61: The 18A Turnigy Plush Electronic Speed Controller we decide to use in our prototype (permission pending).....	81
Figure 62: The two major configurations for quadcopters the "X" (Right) and the "H" (Left).....	82
Figure 63: An example of how quadcopter "X" frames are measured and sized .....	83

Figure 64: The Bumblebee Carbon Fiber Quadcopter frame (permission pending).	84
Figure 65: The all-wood Hobby King Quadcopter Frame V1 (permission pending) .	84
Figure 66: The aluminum X450 CNC Metal MultiCopter frame listed on eBay (permission pending) .....	85
Figure 67: The Q450 450mm Glass Fiber Quadcopter Frame (permission pending) .....	86
When deciding on the Figure 68: Example of a Mode 2 6CH RC Transmitter .....	87
Figure 69: The Futaba 6EX (Left) and the Futaba 7C (Right) (permission pending)	88
Figure 70: The Hobby King HK-T6A-M2 (Left) and HK-6DF-M2(Right) (permission pending) .....	89
Figure 71: Turnigy 9X 9 channel RC transmitter (permission pending).....	89
Figure 72: An illustration of how GPS triangulation looks from space courtesy of National Geographic under the Educational Use of Content agreement.....	90
Figure 73: C04-6H Block Diagram Permission Pending .....	91
Figure 74: Assisted GPS flow of information.....	92
Figure 75: Mission Planner's supported configurations for the APM.....	93
Figure 76: Planned waypoints in Mission Planner using the maps interface.....	94
Figure 77: QGroundControl simulating the missions of multiple UAV's. ....	95
Figure 78: Platform Hand Controller .....	96
Figure 79: Platform Peripheral Controls.....	97
Figure 80: Platform 4-Wheel Drive Motor Controller .....	97
Figure 81: Small Scale Platform Prototype .....	125
Figure 82: Platform Controller Prototype.....	126
Figure 83: 3D Model of Quadcopter Platform .....	126

## 27.0 Appendix C: Table References

Table 1: Motor Control Logic.....	9
Table 2: Microcontroller Research Comparison Specifications .....	15
Table 3: Switching Regulator Comparisons .....	25
Table 4: Atmega Specs.....	57
Table 5 : Telemetry Comparison Chart.....	61
Table 6: Weight budget for the first prototype quadcopter .....	75
Table 7: Maximum RPM's based on various propeller sizes for different classes of propellers .....	78
Table 8: Controllers considered for our first prototype quadcopter setup.....	88
Table 9: Bill of Materials.....	128

## 28.0 Appendix D: Equation References

Equation 1: Received Power.....	59
Equation 2: Calculation for the total amount of required thrust .....	75

Equation 3: Calculation for the amount of required thrust per rotor .....	75
Equation 4: Equations and variables used to calculate what we required from our motors .....	76
Equation 5: Calculations for finding the maximum Kv rating given a 10 in. propeller	78