

EEL 4914 Senior Design I

University of Central Florida
Department of Electrical and Computer Engineering



Senior Design I Project Documentation

SSQuaRe
Self-Stabilizing Quad-Rotor Helicopter

Group 7

Jared Rought
Daniel Goodhew
John Sullivan
Angel Rodriguez

Date: 8/4/2010

Table of Contents

1.0	Executive Summary	7
2.0	Motivation.....	8
3.0	Introduction.....	9
4.0	Risks	9
5.0	Goals and Objectives:	10
6.0	Team Management	11
7.0	Project Timeline	13
8.0	Specifications and Requirements	17
8.1	Microcontroller	18
8.2	Inertial Measurement Unit	20
8.3	Feedback Control	21
8.4	Camera.....	22
9.0	Research	22
9.1	Vehicle Body	22
9.1.1	Material Selection Trade Study.....	23
9.1.2	Vehicle Shape.....	24
9.1.3	Propeller Options.....	25
9.1.4	Hardware Placement.....	27
9.2	Power Research.....	28
9.2.1	Regulation	28
9.2.2	Power Source Comparison	30
9.3	Motor Research	32
9.3.1	Motor Variations	32
9.3.2	Motor Comparison	33
9.4	Motor Selection.....	34
9.5	Microcontroller Research	35
9.5.1	ATMega328	35
9.5.2	dsPIC30F4011	35
9.5.3	STM32F103CBT6.....	36
9.5.4	Microcontroller Selection.....	37
9.6	Development Board Research.....	38
9.6.1	STM3210B-EVAL	38
9.6.2	STM32 Development Environment	39
9.6.3	STM32-P103 and STM32-H103.....	40
9.7	Control Board Research.....	42

9.7.1	Schematic Editor	42
9.7.2	PCB Fabrication	42
9.8	Flight Dynamics	43
9.8.1	Outside forces	45
9.9	Directional Cosine Matrix:	45
9.10	Inertial Measurement Unit.....	48
9.10.1	Gyroscopes.....	48
9.10.2	Global Positional System.....	49
9.10.3	Accelerometer	49
9.10.4	Thermopile	51
9.10.5	Magnetometers	51
9.10.6	Control	53
9.10.6.1	Stabilization	53
9.10.7	Ultrasonic Range Finder.....	54
9.11	Sensor Data Filtering	57
9.11.1	Wireless Communication.....	58
9.11.2	Software Communication	59
10.0	Design	63
10.1	Vehicle Body Design	63
10.1.1	Material Selected	63
10.1.2	Vehicle Shape.....	64
10.1.3	Hardware Placement.....	65
10.1.4	Body Design Summary.....	65
10.2	Constructing the body.....	66
10.3	Motor Selection	68
10.4	Power Design	68
10.4.1	Regulator Setup	69
10.4.2	Power budget.....	70
10.5	STM32 Design	71
10.5.1	STM32 Power.....	71
10.5.2	STM32 External Clock.....	72
10.5.3	STM32 IMU Analog Interface	73
10.5.4	STM32 PWM Interface	74
10.5.5	STM32 SPI Interface	75
10.5.6	STM32 I2C IMU Interface	77
10.5.7	STM32 Software Development Tools.....	77
10.5.8	Firmware Configuration Design	78
10.5.8.1	STM32 Standard Peripheral Library	78
10.5.9	STM32 Clock Configuration	79
10.5.9.1	STM32 I/O Pin Configuration	80
10.5.9.2	STM32 DMA Configuration.....	81
10.5.9.3	STM32 ADC Configuration	82
10.5.9.4	STM32 SPI Configuration	83
10.5.9.5	STM32 TIM Configuration.....	83
10.5.9.6	STM32 I2C Configuration.....	85
10.5.10	System Control Firmware.....	85
10.5.10.1	Start Up and Calibration.....	86

10.5.10.2	Navigation and Stabilization	86
10.5.11	Control Board Design	88
10.5.11.1	PCB Component Layout.....	88
10.5.11.2	IMU, Wi-Fi Placement.....	88
10.5.11.3	Control Board Assembly	89
10.6	IMU Sensor board	89
10.7	IMU Sensor Board: Noise filtering	90
10.8	DCM Control Design.....	91
10.8.1	Rotational Matrix Calculation.....	91
10.8.2	Normalization	93
10.8.3	Drift Calculation: Roll, Pitch	94
10.8.4	Drift Calculation: Yaw	95
10.9	PI Controller.....	95
10.10	IMU Software	96
10.10.1	Directional Cosine Matrix	96
10.11	IMU Sensor Board Schematic.....	97
10.12	Camera Module Design	101
10.13	Ultrasonic sensor Design	102
10.14	Wireless Transceiver	103
	Software	104
10.15.....		104
10.16	iPhone Application.....	107
10.17	Design Summary	116
11.0	Testing.....	117
11.1	Testing the Vehicle Body	117
11.2	Motor Testing	118
11.3	Power Testing.....	118
11.4	STM32 Testing.....	119
11.4.1	Electronic Modules Testing	119
11.4.2	STM32 Firmware	119
11.4.3	Control Board	120
11.5	IMU Testing Methods:	120
11.5.1	Sensor Testing.....	121
11.5.2	IMU Board Testing and Verification	121
11.5.3	IMU Testing.....	122
11.6	Complete Design Testing	123
11.7	Communications Range:	123
11.8	Software:	124
11.9	User Interface:.....	124
11.10	Cold Start Test.....	125
11.11	Video Stream Delay	125
	Budget	126
12.0		126
Appendix A: Works Cited		128

Appendix B: Figure Permissions	132
---	------------

Table of Figures

Figure 1: Document Responsibilities	12
Figure 2: Milestone chart for the months of May and June.....	14
Figure 3: Milestone chart for the months of July and August.....	15
Figure 4: Milestone chart for the months of September and October.....	16
Figure 5: Milestone chart for the months of November and December	16
Figure 6: C3038-3620BW module showing the overall configuration (Permission pending from electronics123.com)	22
Figure 7: Alternating Propellers	24
Figure 8: Angular Momentum Canceled by Angled Motors.	25
Figure 9: Efficiency versus speed for the given ratios of power to propeller size when traveling through air. Image copyright of Martin Hepperle and reprinted with permission	26
Figure 10: Rotor Placement.....	27
Figure 11: Simple linear regulator.....	29
Figure 12: Hextonik 24 gram motor beside a measuring tape.....	34
Figure 13: STM32-H103, Left is the top side of board, Right is the bottom side of board, image copyright of Olimex Ltd and reprinted with their permission	41
Figure 14: Comparison of the USB-Tiny(left) with the ARM-JTAG(right), image copyright of Olimex Ltd and reprinted with their permission	42
Figure 15: Eclipse development environment accessing OpenOCD to write the firmware to the development board for debugging.....	42
Figure 16: Euler Angles. Reprinted with permission From Wikipedia under the GNU Free Document License.	44
Figure 17: DG500: Dual Axis Gyroscope Direction of Detectable Angular Rate (Reproduced with permission from InverSense)	48
Figure 18: LY530AL: Single Axis Gyroscope Direction of Detectable Angular Rate (Reproduced with permission of STMicroelectronics)	48
Figure 19: Sensor Axis of the ADXL335. (Permission pending from Analog Devices)..	50
Figure 20: HMC5843: Magnetometer Directional Assignments and Pins (Bottom View) (Reprinted with permission of Honeywell)	52
Figure 21: Beam width chart (Awaiting approval from MaxBotis)	55
Figure 22: MaxBotis Product line Features (Reprinted with permission from MaxBotis)	56
Figure 23: Sensor Blending	57
Figure 24: Prototype from Above/ Prototype from the Side	66
Figure 25: An example of the procedure for attaching a motor to the body of the quad-copter.....	67

Figure 26: Hextronik 24 gram motor beside a measuring tape.....	68
Figure 27: The small battery for the board is only about twice as wide and twice as tall as the battery sensor for the larger battery.....	69
Figure 28: Schematic for power supply and external oscillator.	73
Figure 29: Schematic showing the various IMU components and how they interface with the STM32	75
Figure 30: Schematic showing the SPI interface between the Wi-Fi module and the STM32	76
Figure 31: Schematic showing the I2C interface between the magnetometer and the STM32	77
Figure 32: Sample code showing the use of the STM32 Standard Peripheral Library to configure and access I/O pins	79
Figure 33: DCM Feedback Control	91
Figure 34: PI Controller.....	96
Figure 35: DCM.c Code Structure	97
Figure 36: IMU Sensor Board Schematic	101
Figure 37: Camera Module Software/Hardware Interface	102
Figure 38: Form Factor of ultrasonic sensor (Use with permission from MaxBotix Inc.)	103
Figure 39: Roving Networks Wi-Fi module pin layout.....	104
Figure 40: Wi-Fi transceiver to iPhone interface.....	104
Figure 41: Data Flow from Copter to Copter.....	105
Figure 42: Draft of iPhone Application GUI	107
Figure 43: UML Class Diagram of iPhone Application	109
Figure 44: UML Sequence Diagram	110
Figure 45: GUI Interface for the Main control screen.....	114
Figure 46: UIAlert View example	115
Figure 47: Single axis Testing setup (Conceptual Drawing. Not to scale).....	123
Figure 48: Rating Card for the user interface	125
Figure 49: Individual Item Difference	127

Table of Tables

Table 1: Team Member Responsibilities.	11
Table 2: General Requirements for the quad-copter.	17
Table 3: Common Material Properties	23
Table 4: Load factors for common propeller sizes (D = diameter, p = pitch).	27
Table 5: Lithium polymer, nickel metal hydride, lead acid, and nickel cadmium batteries compared side by side. Table courtesy of Cadex with permission [11].	31
Table 6: Motor comparison	34
Table 7: Microcontroller Comparison that summarizes the features that are useful for the project.....	37
Table 8: Translational Motion in Relation to Thrust	44

Table 9: Bandwidth as a function of capacitor value.	50
Table 10: Wi-Fi vs Bluetooth Chart.....	58
Table 11: Roving Networks Modules Comparison Chart.....	59
Table 12: Difference between iPhone Apps	60
Table 13: Weights	63
Table 14: IMU Sensor Listing	90
Table 15: PI Controller	96
Table 16: Budget	126

1.0 Executive Summary

Unmanned Aerial Vehicles have played a crucial role in modern warfare, reducing both cost and human liability. This reduction in cost has been relative and does not offer much for non-defense firms and the general public. The aim of the project is to create a lightweight, low cost, easy to use UAV for various applications. Such a device could have applications in areas such as warfare reconnaissance or for civilian uses such as for humanitarian aid, surveying missions, or amusement purposes.

The quad-copter will incorporate a medium to low power usage design running off of battery power. Two batteries will be incorporated into the design, one low power and one medium power, to maintain a clean source for the electronic boards. The main drain to the power source will be from the motors necessary to lift the aircraft off the ground and maintain stability. As with all aircraft, it is essential to reduce weight; therefore, all components will be lightweight to accommodate for less power consuming motors. The use of smaller motors will allow for increased flight time due to a reduction in power consumption. The main frame of the quad-copter will be constructed from lightweight aluminum tubing and will be designed to provide the largest possible stability to the aircraft.

In order to maximize the stability of the aircraft, an Inertial Measurement Unit (IMU) is necessary to regulate and compensate for aircraft movements. The IMU will track altitude, speed, position, direction and log the data. This information will then be used to stabilize the aircraft and support other features of the craft. Real-time video streaming will also be implemented and supported by the Image Processing Unit (IPU). This unit will consist of the camera module and the necessary processing elements. The IPU will then relay the information to the control board to be transmitted to the control interface.

The central controller will be the interface for all of the modules on the quad-copter. It will be capable of receiving the data from the IMU and data sent from the user to control the modules responsible for changing motor speed. It will process data from the IPU and transmit to the iPhone through a Wi-Fi connection. The phone user will then be able to view the images from the IPU in real time as the quad-copter flies. The central board, with the central processor mounted on it, will have pin headers so that the other modules can be connected by wire to the central controller. Connecting the modules like this will allow for damaged modules to be easily removed and replaced. This also allows for modules to be upgraded or changed based on user preferences. The central controller is required to be low power in order to conserve battery life while maintaining enough processing speed to send data in real-time so that the quad-copter can operate correctly. Additionally, the control board should be as small

2.0 Motivation

Modern militaries have utilized Unmanned Aerial Systems (UAS) for many years, offering both strategic and fiscal benefits. Most designs are geared towards supporting weapons payloads, thus providing an alternative to the use of fighter jets to neutralize targets. Using a UAS for such purposes has minimized casualties of military aviators. This technology however results in an aircraft that is quite large in size, to be able to carry weapons payloads. Functionality that is missing in the UAS arsenal is the ability to provide close surveillance support for platoons that are engaged in combat in locations where normal a UAS would normally not be of use. While there are surveillance drones available, very few work to minimize cost. The type of system that the group is envisioning would have the ability to provide surveillance at the street level. Such a system would aid in planning raids, rescue missions, and related operations. This type of system would be able to provide real-time reconnaissance for platoons engaged in combat.

Military applications are not the only possible areas that could benefit from the use of such an aircraft. Commercial and public sectors would also benefit from a low cost, ultra maneuverable surveillance aircraft. Rescue operations would benefit greatly from utilizing a UAS. Volatile environments are a great hazard to the individual that are involved in rescue operations. For example, anytime there is a natural disaster that search and rescue becomes a hazard for those involved, a small scale UAS could be utilized to search for survivors and map out the best way to reach them, without putting the rescuer in harms way. During hurricanes a UAS could be utilized to locate individuals that are trapped in places that where vehicles are unable to reach. Once the individuals are located the UAS can be used to find the best method to extract the rescued. In future space exploration, such as exhibitions to other planets, a UAS could prove to be useful. A UAS could be used to explore environments during manned exploration of new planets, thus mitigating the hazards of exploring new environments.

One of the centerpieces of this project is the Inertial Measurement Unit (IMU). When operating a quad rotor helicopter it is necessary to have some sort of method of controlling the four motors. Reaction time of the operator is simply not robust enough to be able to pilot a quad rotor helicopter. There are several approaches to solving this problem, some better than others. One of the motivations for this project was to develop a system that builds upon methods that are utilized for winged aircraft flight control and adapt such a technology so that it is able to stabilize a quad rotor helicopter.

A secondary motivation that leads the group to select such a device to design is the use of the device as a means of enjoyment. To put it quite simply, such a device could provide many hours of enjoyment. Some of the group members have dealt with RC airplanes and have learned from experience, and a couple damaged planes, that a self-stabilizing device would greatly benefit the operation

of model aircraft as a whole.

3.0 Introduction

Man's fascination with flight is seen throughout history, and the desire to conquer the skies is innate in humanity. The most common inventor to dream of flight by rotocraft was Leonardo Da Vinci. His designs for an "aerial screw" have inspired generations of engineers and inventors to aspire higher. With the wide-scale manufacturing of computers, aerospace systems that involved intense human interactions were replaced with "fly-by wire designs" and now require little human interfacing.

For group seven, the idea of building a quad-copter first came from looking for applications of imaging systems. The group's brainstorming sessions resulted in several possible projects, but in the end, the group decided to try the difficult task of creating a self-balancing, video transmitting flyer. The focus of the project begins with the basic yet complex task of getting the aircraft to hover and maintain stability. Ideally, the quad-copter will take off and land without human interference. Further features, such as video streaming and waypoint recognition, will be added as time allows. All added features and controls will be interfaced with the use of an iPhone application. It is the hope of the group to have a simple user interface that controls the quad-copter with ease and smoothness.

4.0 Risks

As with most experimental projects, there are associated risks with the assembly and operation of the device. The risks in implementing this project range from cuts and burns to death. To avoid bodily injury, care must be exercised when handling the unit as there are sharp edges. Soldering burns are also a concern during assembly. The unit is especially dangerous when the propeller blades are attached and the motors connected to the power source. It is essential to keep all body parts far away from the vehicle when it is about to be powered on. It is the operator's responsibility to make sure the unit is utilized only at a safe distance from spectators. The current to the motors is also very dangerous as the amplitude of said current traveling on the wires is enough to cause electrocution. Recharging the battery must also be done with caution as large amplitudes of current are flowing, and battery packs can explode if improperly connected to the charger.

Beyond bodily harm, the project may have adverse mental effects on the participants. A lack of sleep while working on a problem and long hours of frustrating results can lead to depression and possibly even insanity. Positive steps to avoid a mental breakdown include taking regular breaks, keeping hydrated, and meditating on a happy place from time to time. Humor does not

hurt either. Group camaraderie is essential to encourage all members to maintain a proper perspective.

5.0 Goals and Objectives:

The goal for this project was to create a highly functional, user controlled copter that can be accessed and controlled using an iPhone. The reason for the iPhone is that it is one of the leading smart phones and has one of the widest arrays of applications already built for the device, making it not only easier on the designer to build the program but also giving it a wider distribution opportunity for the public.

One of the main objectives is to effectively be able to fly a quad-copter from an iPhone using the Wi-Fi capabilities already built into the iPhone. The group will be sending data back forth between the copter and iPhone using the 802.11 technology that is widely used around the world for many applications. The group wants to be able to see where the copter is going without having a direct line of sight with the copter from the iPhone. To do this the group will be utilizing a camera that will be mounted onto the copter body. The group will have one main power supply that will drive all of the motors, processor, Wi-Fi module chips inertial measurement unit (IMU) and the other electronics on the copter. The batteries used to run the design will be rechargeable and give a total operating time of at least 15 minutes of run time. The group will have 4 motors that will each turn its own blade to give the maximum lift possible for the copter. There will be a motor controller connected to the microcontroller to make sure that the motors are receiving the proper amount of power for the controls that are being sent from the iPhone. The group will have a light body material composed of materials that give the copter structural strength to be crash resistant and also make it light enough for the copter to fly. The Electronics that will be housed on the copter should be protected from outside forces and be able to survive a crash if the copter begins to lose control.

The quad-copter will stream video wirelessly to the iPhone using a wireless transceiver module located on the copter and a custom designed user application. The application will be able to make the copter take off safely and attaining a certain height before the user is given controls of the copter. The application will also be able to land the copter safely on the ground once when the user has designated it is time or when the processor on the copter deems that the batteries are too low to continue flying in a safe manner. The iPhone application will give the user the ability to make the copter rise and fall and move the copter in any direction based on the user's controls. The iPhone application will also stream the video while the user has control of the copter giving the user real time information on the specifics of the copters location and signal status.

To better use the short amount of time that the group was given to research, design, build and test the project, the group created a Gantt chart to organize

how the group will attack this project. To start the group will research the respective areas that were split up among the different group members. From there the group will then move on to purchasing parts based on the researched the group conducted that will suit the needs for the copter. After the parts come in the group will begin the design stage of the project. Each member will work on their respective areas that they researched in the design stage to come up with the best way to communicate and efficiently work together as one project. After designing the project from the hardware to the software applications, the group will go into a testing phase that will test the full scope of the design. After testing the design and fixing the bugs the last phase of the project will be to clean up the software, hardware and physical parts of the project to make sure the design is running as efficiently as possible.

6.0 Team Management

The group initially identified the specifications and requirements that were necessary for the success of the project as a whole. The group further identified that certain specifications and requirements could be grouped together. The newly created subsections of responsibility could be easily assigned to individual group members. The next task taken on by the group was to identify the strengths and interest of each of the individual team members, thus allowing project responsibilities to be assigned. Each group member identified expressed interest in various areas and the responsibilities were divided accordingly.

Group Members Name	Responsibility
Jared Rought	Project Manager, IMU Design, Camera Implementation
Daniel Goodhew	Power Design, Frame Design
John Sullivan	iPhone interfacing and GUI Design
Angel Rodriguez	Microcontroller interfacing and controller software design, Control Board Design

Table 1: Team Member Responsibilities.

Modularizing the design allows each group member to be able to focus on a certain task, thus promoting the success of the overall project. This modularizing of the project by no means stipulates that each group member is not encouraged to work together to solve problems and troubleshoot each other's design. On the contrary, bi-weekly status meetings will encourage discussion of each other's design issues and concerns. If issues arise the workload of each individual will be addressed and adjust if need be. In addition the group decided to utilize a tracking method to better ascertain the status of the project documentation

throughout the design process. An example of the spreadsheet that was utilized is presented in figure 1.

Table of Contents	Pages	In Charge
Executive Summary	1	Daniel
Introduction	1	Daniel
Biographies	1	Angel
Motivation	2	Jared
Goals and Objectives	2	Angel
Specifications	1	Jared
Requirements	3	John
Risks	1	Daniel
Team Management	1	Jared
Budget	2	John
Project timeline	3	Daniel
Research		
Software - iPhone	6	John
Wireless Communications	4	John
Flight Dynamics	4	Jared
Controls and Filtering	3	Jared
IMU	3	Jared
Microcontroller	4	Angel
Microcontroller Software Architecture	4	Angel
Power Architecture	4	Daniel
Frame	4	Daniel
Design		
Software	6	John
Wireless Communications	3	John
Flight Dynamics	3	Jared
Controls and Filtering	3	Jared
IMU	3	Jared
Control Board (Microcontroller, Motor controllers, Wifi, ect)	4	Angel
Main Control Software Architecture	5	Angel
Power Architecture	3	Daniel
Frame	4	Daniel
Testing		
Software testing	2	John
Wireless Testing	2	John
User Interface Testing	2	John
Video Streaming Testing	2	John
Controls Testing	3	Jared
IMU Hardware Testing	3	Jared
Control Board testing	5	Angel
Microcontroller Software Testing	3	Angel
Power Architecture Testing and Power Budget	3	Daniel
Motor testing	4	Daniel
Appendix	3	Daniel
Total:	120	

Figure 1: Document Responsibilities

7.0 Project Timeline

The following milestone charts show the overall timeline for each month and the steps required to have success in the process of creating the senior design project. The information on the charts is first created by each member of the group individually and then assembled into a unified schedule to ensure all parts of the project are present. The beginning of the first semester focuses entirely on brainstorming and research. The group is formed early in the semester and has the opportunity to meet several times in the first two weeks to brainstorm and compare ideas before the decision of what project to undertake is finalized. Once the group is committed to constructing a quad-copter, the main goals are discussed at length. Dr. Samuel Richie approves the initial vision, and the group must immediately begin serious research into the implementation of the idea. With the short schedule of the summer semester, time is of the essence.

The duties associated with the project are divided according to each person's stated interest when selecting the project. Research in each person's section is conducted primarily by the responsible group member, but all information is shared at bi-weekly meetings. These meetings take place over Skype once in the evening during the week and once at the end of the weekend. The results from the preliminary research allow more detailed future agendas. The areas of research and the associated timelines are presented in Figure 2. Thanks to the division of labor among the group, many steps in the chart are worked on simultaneously. Once the research narrows down the available options, the parts or processes that will make up the system are selected. Some research and part selection is delayed for areas dependent on other decisions.

Any remaining choices regarding initial implementation are decided early in the month of July. Now that all major decisions are made, parts can be purchased. Some parts have been acquired in advance to test feasibility of use. The main goal for the rest of the semester is to document the project in full detail, but coding, assembling, and testing parts can be done on the side as the pieces arrive. Data from any such experimentation is useful in completing the documentation. All documents are reviewed by the group before submission to ensure the work is correct and to check for consistency from one author to another. After the final draft is submitted, the group will have a vacation period in between semesters. While continued work on the project is suggested, each member is free to do as they please during this vacation time. The tentative milestone chart for July and August reflects the above goals, as can be seen in figure 3.

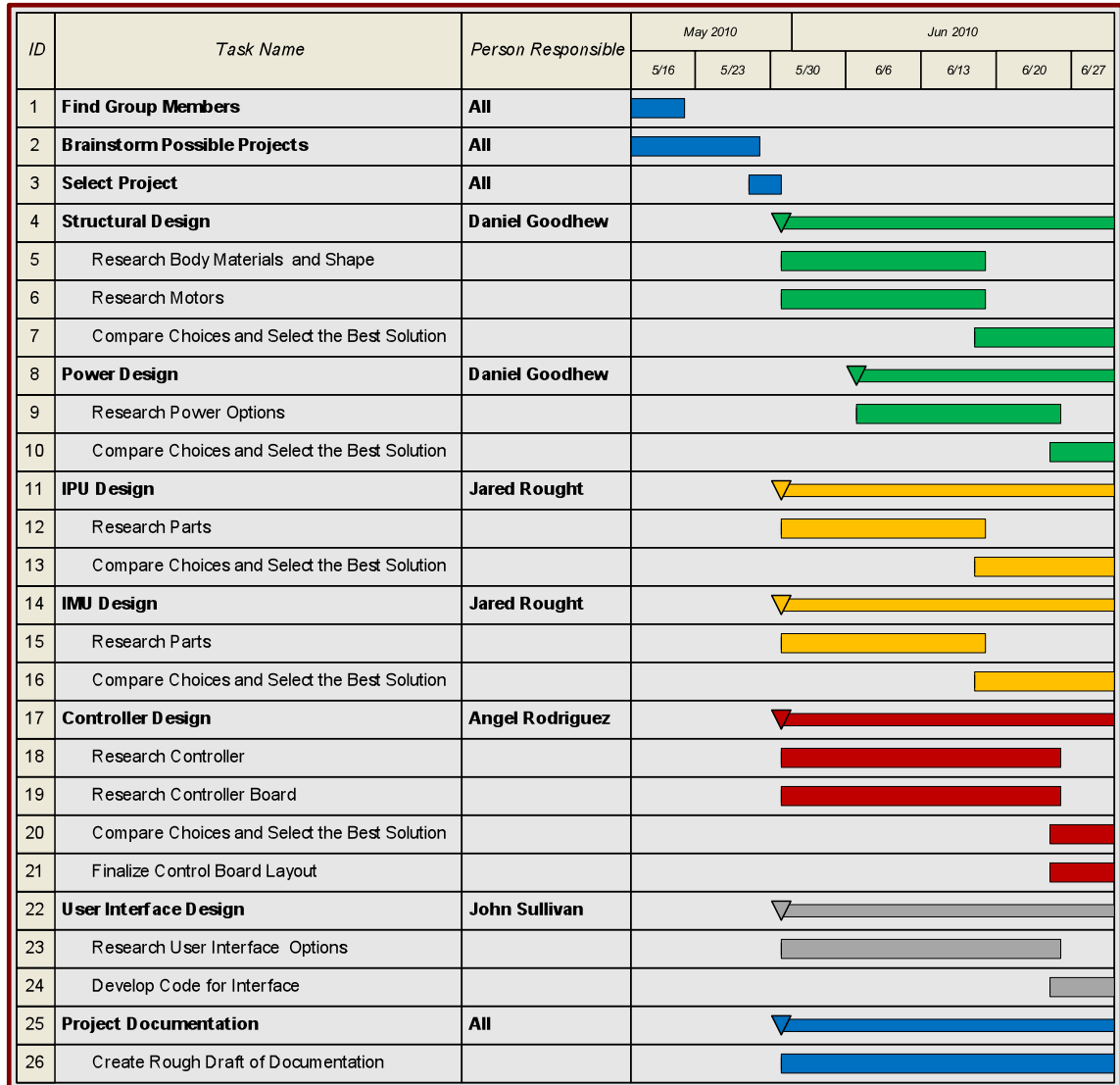


Figure 2: Milestone chart for the months of May and June

The beginning of the fall semester brings with it the anticipation of seeing how well the designed system performs. All parts should be ordered by this point, so any last minute purchases must be made quickly. The goals for the first two months of the semester will be testing and building the quad-copter. A detailed record must be kept of all changes made to the system as a result of testing. Each part will first be tested individually with as few components as necessary to have a functioning part. When all the parts show promise of working, smaller pieces will be combined and tested at a higher level. Eventually, the whole system will be operational. The iPhone interface will be the last segment to be incorporated into the system. The resulting milestone chart is seen in figure 4.

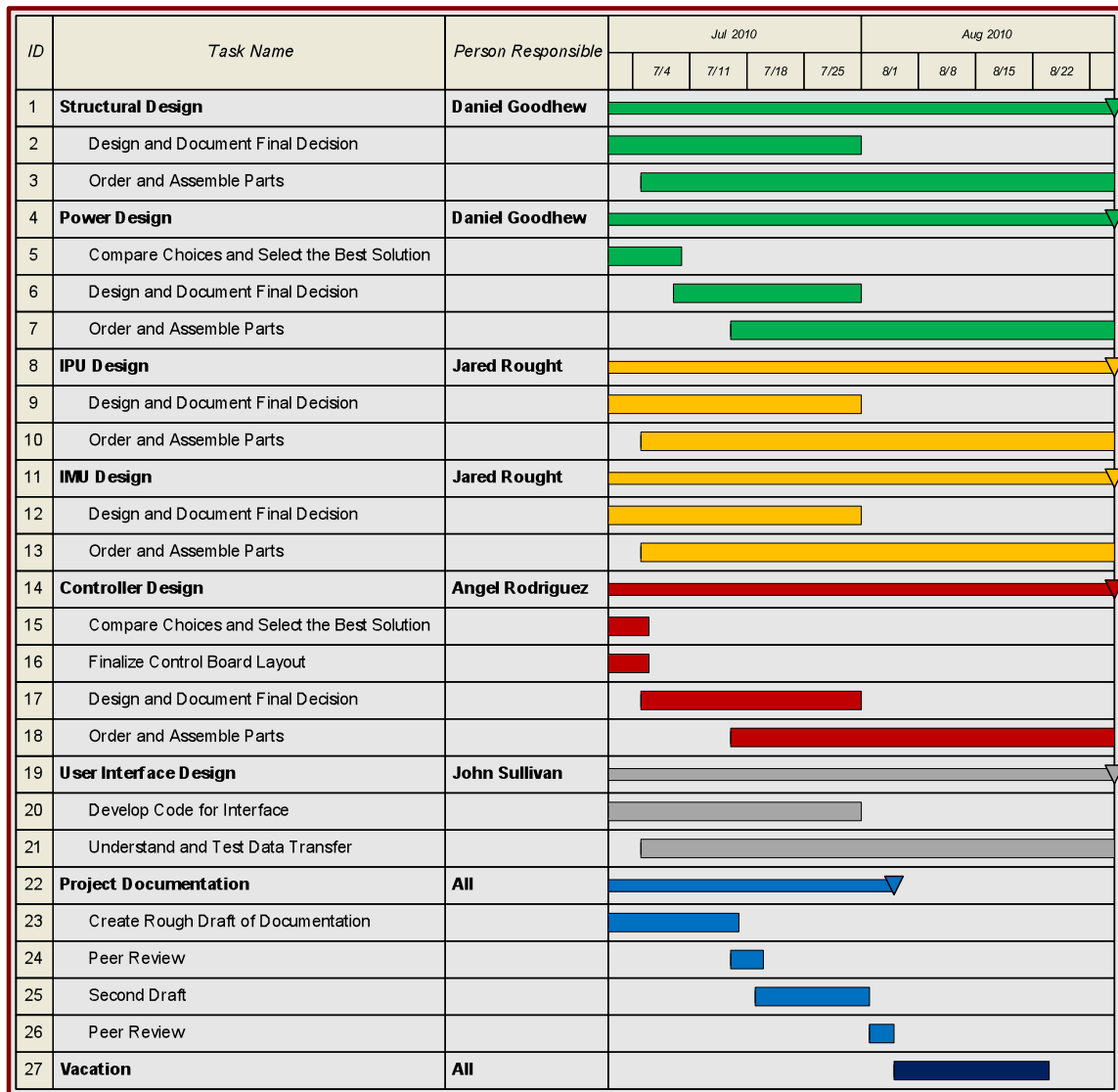


Figure 3: Milestone chart for the months of July and August

The last two months of the semester are reserved for final testing and documentation. The quad-copter will be the main focus, as the project is considered a failure if it does not work. Most of the documentation should remain the same as the original version with the exception of the changes recorded during testing. Once the vehicle can hover and land without an accident, further features will be added. First, the video streaming will be implemented. The ability to direct the aircraft to a given waypoint will be the next feature added. The project will most likely run short on time before very many features are working. The last two weeks of the semester are dedicated to the creation of the final presentation. The documentation will be finalized, the presentation will be perfected and several practice runs will take place. The final milestone chart in Figure 5 visually conveys these goals.

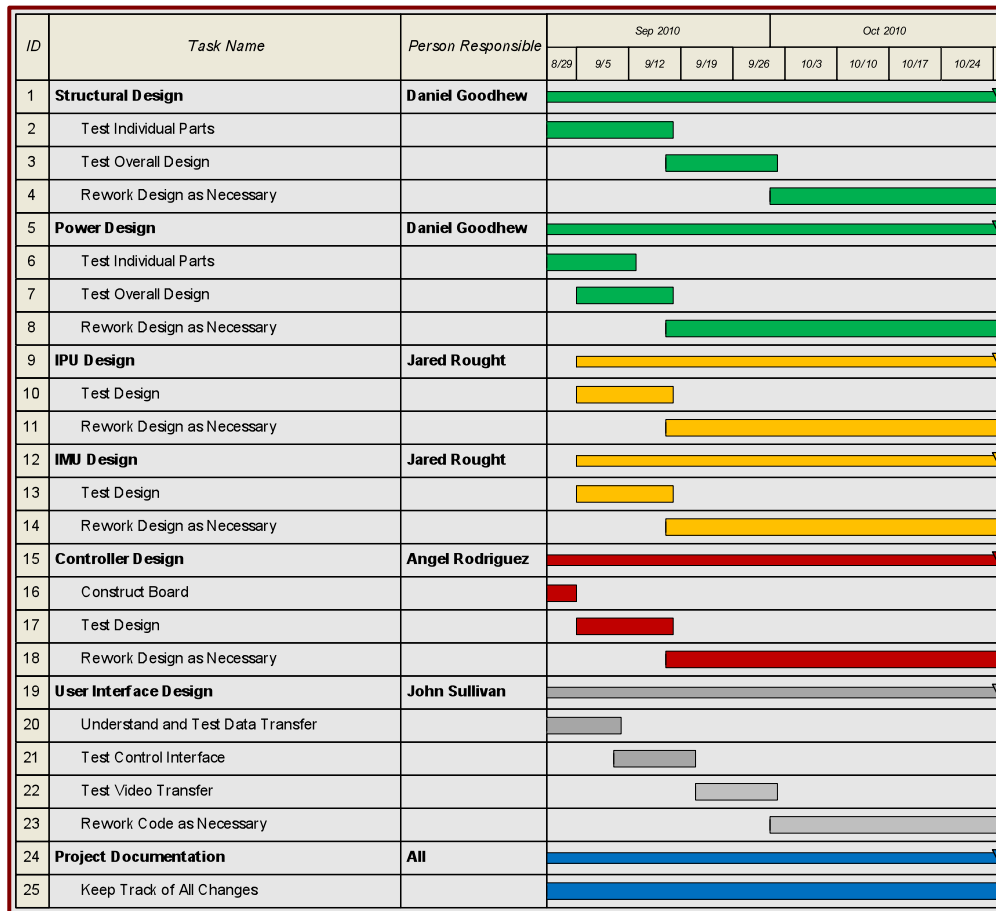


Figure 4: Milestone chart for the months of September and October.

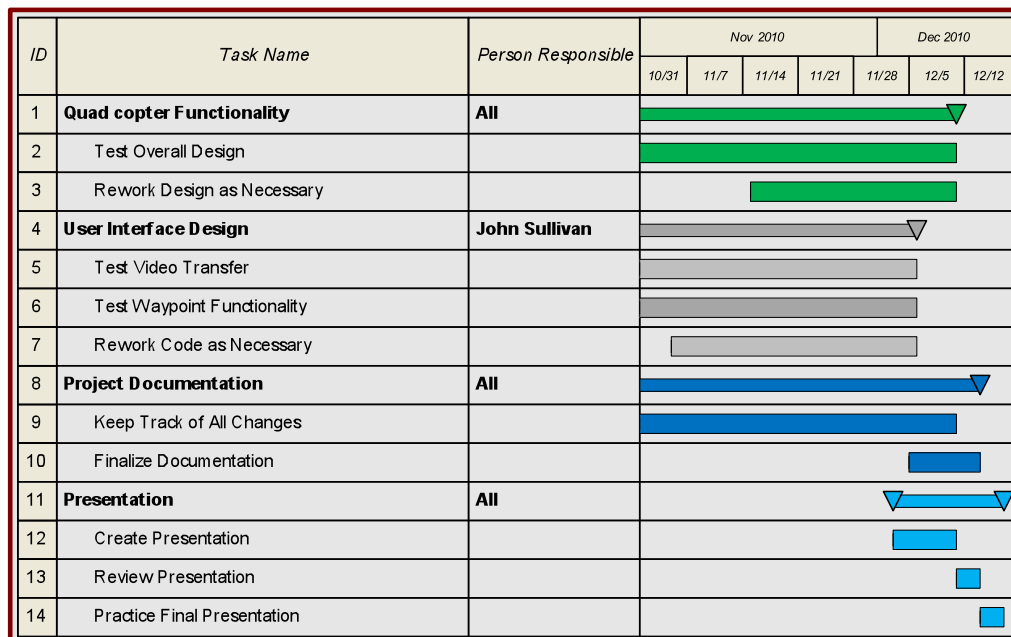


Figure 5: Milestone chart for the months of November and December

8.0 Specifications and Requirements

This section will describe the requirements the group have set for the quad-copter. All the requirements the group have selected were tested diligently and made sure they worked properly to ensure that the quad-copter performed up to the standards. For the thorough details of the testing procedures the group has created testing procedures in section 11.0 . In table 2 the group shows the main requirements for the project and how the group plans on testing the requirement.

General Requirements		
No.	Requirement	How To Test Requirement
1	The copter must be under 5 pounds	Put the copter on the scale and record its weight
2	The maximum current draw must be less than 20Amps	Add up all the currents being draw out of all the electrical components and record the amps
3	The copter must have a minimum flight time of 12 minutes	The group will time the copter from when it begins to hover to the time it stops hovering and lands on the ground
4	The copter must have 4 motors that generate 1Kg of thrust each	The group will buy motors that have a 1Kg of thrust
6	The copter must be able to transmit a live video feed to the iPhone in under 2 seconds	The group will perform a video test to measure the time difference between when the video is recorded and when it shows up on the iPhone screen
9	The control board should have an area of 16 square inches	The group will measure the control boards width and length to give the area of the control board
10	The copter will respond to the user's inputs to control the copter at 30 times a second	The group will monitor the data coming into the IMU from the iPhone and calculate how fast the data is coming into the IMU
11	The copter will have to provide live video to the iPhone by a camera located on the copter	The group will make sure the video shown on the iPhone is a live feed

Table 2: General Requirements for SSQuaRe

General Requirements		
No.	Requirement	How to Test Requirement
12	The vector the quad-copter is moving will be reported to the iPhone at least once every second	The group will monitor the data being sent to the iPhone and see how often the directional vector is being sent to the iPhone from the copter
13	The quad-copter must have a minimum communication range of 100ft	The group will fly the quad-copter away from the iPhone till the copter begins to hover which signifies it is outside the communication range
14	The User interface on the iPhone should be simple and easy to use	The group will have people control the device who have not been introduced to the controls of the iPhone and have them rate their experience with the user interface
15	The User Interface will show a live video feed of the copter that will update 15 frames per second	The group will monitor the frames per second that is being shown on the iPhone
16	The quad-copter will be able to land and take off automatically with a push of a button on the iPhone	The group will push the button on the iPhone and monitor if the quad-copter lands and take off after pressing the button

Table 3: General Requirements for SSQuaRe

8.1 Microcontroller

The microcontroller is one of the most important components of the quad-copter and there is no way that the quad-copter can even be flown steadily without it. It is responsible for taking in all the data that the other modules output and then processing the data into signals. Some of the signals will be used to control the flight characteristics of the quad-copter. The other signals meanwhile will be the output information on the status of the quad-copter and images from the camera to the Apple iPhone. The quad-copter will be composed of only one microcontroller to reduce the complexity of the design.

In order to reduce part count and the complexity of the design it was decided that the microcontroller that is chosen should have a built in ADC so that a separate ADC chip does not have to be acquired. The microcontroller should be able to convert the analog data from the IMU at a rate of 100Hz so that the quad-copter can stabilize itself every few milliseconds. The microcontroller must have enough communication ports in order to interface with the Wi-Fi and camera modules. The microcontroller should be able to handle receiving navigation instructions

from the iPhone many times per second to control the movement and height of the quad-copter. Be able to send status data to the iPhone, like the quad-copter's height and current position at least once per second. Filter the IMU and iPhone navigation data so that it can then be processed in a control loop to allow stable flight and navigation. The microcontroller also must receive video from a camera and then stream it to the iPhone through Wi-Fi. Finally, the microcontroller has to have an on board timer with 4 channels in order to generate PWM signals to control the speed of each of the motors and the microcontroller should be able to update the duty cycle of the PWM signals in microseconds.

From all the tasks the microcontroller has to do it is obvious that the microcontroller that is chosen for this project has to have a lot of processing power. So that means that an 8-bit microcontroller will not have the ability to do all that is required to get the quad-copter to operate. The quad-copter will be battery powered, which means that in order to maximize flight time the microcontroller that is chosen has to draw a reasonable amount of power because the microcontroller and all the other modules except for the motors will be powered by a 500mAh battery. Since the plan is for the quad-copter to be able to fly for 15 minutes all the modules except the motors can only consume an average of 1.5 A of current. The reason for the selected current limit is to allow the electronic modules to operate for 20 minutes before the battery runs out of charge. This run time is required because it allows enough time to get the electronic modules on the quad-copter to be started and calibrated before starting flight and it also allows enough time after the 15 minutes of flight to land the quad-copter safely. This means that the microcontroller has to consume less than 300 mA so that there is enough current for the other modules to operate for the required amount of time. The microcontroller can only have a maximum V_{cc} voltage of 5 V because of battery constraints. Another thing to consider when choosing a microcontroller is the cost of the microcontroller. Not only does the price of the microcontroller come into effect, but so does the price of the extra hardware and software required to develop for the microcontroller, like the development board and the software suite to write and debug the firmware. Since the funding for this project will come from only the group members the total cost of the microcontroller and its development tools should not exceed \$150.

The final consideration when picking a microcontroller is its popularity. If a microcontroller that is not used extensively is selected for the project it will make the project more difficult because there will be less information, helpful resources, and development tools available for the groups use when the firmware for the microcontroller is written. This is an important factor because the group has little experience in working with microcontrollers and the group does not have the time to learn very complicated microcontroller architectures.

The quad-copter microcontroller, IMU Sensor Board, and Wi-Fi module will be mounted onto a PCB control board that the group will design. The control board is going to be designed to be as light weight as possible because the lighter the

quad-copter the cheaper it will be to make. The control board will be made lighter by making it have as small an area as possible. Ideally the group would like the control board to be less than 16 in². Reducing the size of the control board also has the added benefit of reducing the price of having it fabricated. The control board will have to have connectors and pin headers in order to connect to the batteries and motor ESCs on the quad-copter frame. The control board PCB should be no more than two layers because this is going to be the first PCB design for the group and want to keep it simple.

8.2 Inertial Measurement Unit

The Inertia Measurement Unit (IMU) must provide the means for measuring the orientation of the aircraft. The measurements gathered by the IMU will then be used by the control board to implement a stabilization algorithm, thus delivering optimal control of the aircraft by the operator. Because the device is a mobile device power must be derived from an onboard battery. This characteristic of the aircraft makes weight of the device a crucial component to the successful flight of the aircraft. For this reason the IMU will implement electronic sensor components that are mostly derived from low power chips.

The IMU will encompass sensors that will be the means for deriving the necessary elements to describe the flight dynamics of the aircraft, specifically roll, pitch, and yaw. An electronic gyroscope will provide the central means of deriving the roll, pitch and yaw. Speed is essential to the smooth and fluid operation of the aircraft and thus the sensors must have a relatively quick response time. The response time of the gyroscope must meet the demands of the refresh rate that is described by the stability algorithm. In order to create a meaningful stabilization algorithm a minimum refresh rate of 2 Hz must be realized, with 5 Hz desired. Taking this into account the gyroscope to be used by the IMU must be able to handle a minimum of 100 rad/s measurement rate. This is easily achievable with today's MEMS based electronic gyroscopes.

Due to errors that can accumulate in the gyroscopic sensors, reference sensors will be necessary to alleviate said errors. An accelerometer will serve as a reference sensor for roll and pitch of the aircraft, while yaw will be measured by a magnetometer. The accelerometer must be sensitive to the g's that the aircraft will experience during the course of flight. For the purposes of the flight characteristics that the quad-copter will experience a minimum of 3g's of sensitivity must be achievable for the accelerometer. In order to derive the roll pitch and yaw angles necessary to implement the stabilization algorithm it is necessary for the accelerometer to be able to sense acceleration on each of the planes three axes within its reference frame. Environmental testing conditions also warrant a certain level of shock tolerance on the part of the accelerometer. Due to this constraint a minimum of 1000g-shock tolerance must be obtainable by the chip that is selected. Sensor stability and noise tolerance is also a must on

any chip that is selected. Noise experienced by the fluctuations in temperatures that the device may experience must be minimized through the selection of a high temperature stable chip. Accelerometers are also susceptible to noise that is introduced by the vibrations experienced during the flight of the device. This must be minimized in order to maintain accurate calculations.

While the accelerometer must measure the pitch and roll angles, an inability to sense yaw movement when the device is immobile, with respect to roll and pitch, will be present. A secondary sensor is necessary with the ability to measure yaw movement when the device is in a hover. Quick and accurate measurements are essential to the success of the IMU design. It is for this reason that reference vectors are necessary to be able to effectively update the rotation matrix. GPS units prove to be the most ideal choice for a reference vector but lack the ability to make yaw measurements when not engaged in forward motion. A magnetometer is better suited to provide a reference vector for yaw correction. Due to the nature of the flight of the aircraft it will be necessary for the magnetometer to make measurements of earth's magnetic field from every orientation. This is achieved by ensuring that the magnetometer is a three-axis device. Power is a constraint that determines the flight time of the device. Much care must be taken to ensure that the sensor selected consumes the lowest amount of power possible. As with the other sensor the magnetometer must be able to survive the flight conditions of the device. A significant shock tolerance must be required of the sensor selected. Measurement sensitivity is important to obtain accurate measurements. A sensitivity of microteslas is necessary to detect the earth's magnetic field, which can range from 30 – 60 microteslas. All of these measurements must be taken at a relatively rapid rate to ensure the overall quickness of the stabilization algorithm.

8.3 Feedback Control

Ensuring rapid and accurate measurements is paramount to enabling stable flight. Feedback controls will play an integral roll in mitigating errors that will accumulate due to sensor measurements and the numerical integration that is utilized.

The control loop must be able to utilize the sensor measurements in a fashion that optimizes the speed of the overall stabilization algorithm. To achieve this, the formation of the rotational matrix will rely solely on the inputs from the electronic gyroscopes. This is done because the measurements from the rotational sensor are slightly quicker than the other sensors. A PID controller will be utilized to mitigate the error that is present in the measurements between the different sensor measurements. The control loop must also be robust enough to mitigate any noise that is present in the sensor measurements, the PID controller will also aid in this matter.

8.4 Camera

Proper control of the aircraft is crucial to ensuring a safe flight. To aid in this endeavor real-time video will be streamed from the device to the controls, in this case the iPhone. Transmission of the video data must be relatively quick to mitigate lag that can be present in video feeds. The output of the Imaging Unit (IU) must have a digital output to aid in the transmission over Wi-Fi. The limit of the supported video resolution of the iPhone is 640x480; hence the video output of the camera is limited to an output of 640x480. Simplification of communications between the camera and the aircraft is also a desired characteristic. Thus a camera that outputs a digital signal is required, alleviating the need to sample and convert the signal. Weight is a constraint that is also important to the IU module. The selected camera must also minimize weight. Communications will be done through the microcontroller and sent directly to the iPhone over the Wi-Fi.

The camera that is being researched for use in the quad rotor helicopter utilizes the OV6630 imaging sensor to produce an image array. The array is then transmitted via a digital output of the C3038 module through the onboard ADC.



Figure 6: C3038-3620BW module showing the overall configuration (Permission pending from electronics123.com)

9.0 Research

9.1 Vehicle Body

When creating an aerial vehicle, one must make some key decisions about the type of vehicle to construct. As with all aerospace projects, one of the primary concerns is weight. Heavier vehicle designs allow for many parts and can provide greater towing capabilities, but require more powerful motors that consume large amounts of electricity and larger blades to maintain lift and stability. Major factors affecting flight time include the power consumption and availability of sufficient sized batteries. In contrast, lighter designs consume less electricity but require the use of as few components as possible in order to maintain the lower

weight.

9.1.1 Material Selection Trade Study

The key to successful flight of any rotorcraft revolves around stability, and stability starts with the frame of the quad-copter. One goal for the project is to develop a lightweight prototype for testing. Thus, the materials for the construction of the body are limited. The weight to strength ratio, material stiffness, cost, and availability are considered during the material selection process. While there are many types of wood, wood choices often suffer from poor strength, and even the best performers are not very stiff. Steel is an excellent choice for stiffness but suffers from an unfavorable strength to weight ratio due to its higher density. Most plastic material also suffers from low stiffness and strength. Some dense plastics have higher strength but the weight is too large for the support provided. All of the above materials are easily available for purchase and are relatively inexpensive as well.

There are some exceptions in the plastics category as certain polymers are combining better materials for their respective strengths, leading to light but strong final products. Possible choices for these polymer materials include fiberglass and carbon fiber. While carbon fiber is the material of choice, it is also the most expensive, in part due to lack of ease of availability. Additionally, carbon fiber has the possibility of cracking when being drilled or upon impact due to its non-homogenous nature. Fiberglass is not quite as strong as carbon fiber and is a more dense material. Other possible materials include lightweight, high strength metals. Aluminum is not much denser than carbon fiber, but is considerably cheaper while maintaining enough strength to withstand regular wear and tear. Additionally, aluminum is easily available. Titanium is another light metal with high strength to weight ratio and good stiffness, but it is more expensive due to decreased availability. Table 3 shows the values associated with each material.

Material	Tensile Strength (MPa)	Density (kg/m ³)	Young's Modulus (MPa)
Balsa	73	140	1130-6000
PVC	50-80	1390	2900-3300
Steel	2000	7860	200000
Aluminum	600	2700	69000
Titanium	1300	4510	103000
Carbon Fiber	4300	1750	125000-150000

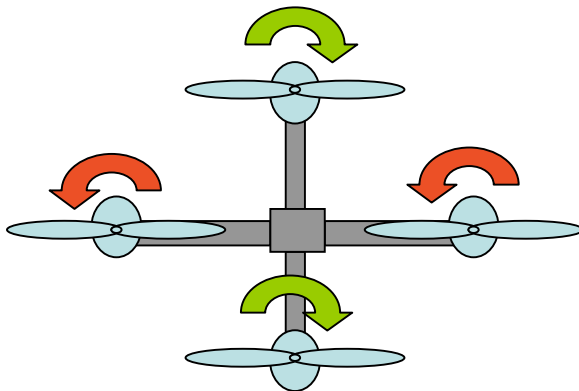
Table 4: Common Material Properties

9.1.2 Vehicle Shape

The shape of the body can take on many forms. Single bars can be mounted as arms extending from the center, thus reducing the amount of material used. Square or triangular lattices connecting the bars provide stability but increase the overall weight of the body. The bars can also be set at various angles with respect to the control board. Some popular configurations include the X and the plus shapes as seen in Figure 7:. Other designs create a square frame with the majority of the components outside of the central section. The bars can be made of square or circular tubes depending on the method of mounting the motors to the bars.

All designs also require the propellers to alternate spinning clockwise and counter-clockwise so that the two opposite motors spin in the same direction. The directions the propellers spin cause an angular momentum that must be offset. Another way to offset the momentum is to secure two opposing motors with their base at an angle to the body, and with the angle in opposite directions. The resulting forces are split between the vertical and horizontal directions. The more the angle is increased, the more the force acts in the horizontal plane. Figure 7: shows the X and plus configurations with the counter-rotating propellers, and Figure 8: demonstrates the approach of setting the motors at an angle.

a) Plus Arrangement



b) X Arrangement

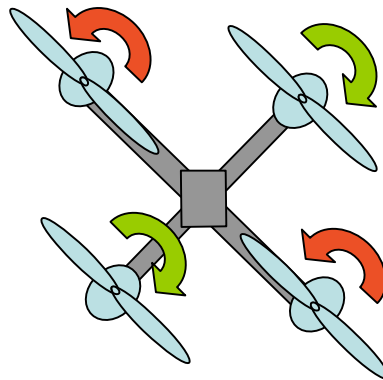


Figure 7: Alternating Propellers

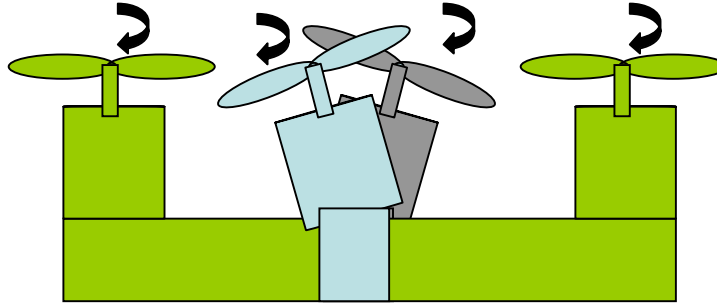


Figure 8: Angular Momentum Canceled by Angled Motors.

The shape of the arms that create the body can have a significant effect on the stability of the craft. Complete tubes are considerably more stable than brackets with an open side. The difference between a square tube and a round one, however, is not very big. The advantage of a square tube is the ease of mounting the motor to a flat surface, but they are more difficult to purchase. The round tubes are easier to find with parts made specifically for helicopters and can be purchased at a local hobby store.

9.1.3 Propeller Options

The propeller characteristics are vital to successful flight. Propellers are airscrews that create the force necessary to lift an aircraft based on Sir Isaac Newton's third law of motion. The force that the propeller puts on the surrounding air causes an equal and opposite force that lifts the copter. The thrust created by the propellers is dependent on three variables: length, pitch, and speed of rotation. The longer the propeller, the more thrust it will have and the more power it takes to rotate at the same speed. Similarly, a propeller will create more thrust but will be slower to turn with a larger pitch. Increases in the speed of rotation also increase thrust. The Abbott equations for approximating the static thrust and power are as follows:

$$T = p * D^3 * RPM^2 * 10^{-10}$$

$$P = p * D^4 * RPM^3 * 5.33 * 10^{-10}$$

where T is thrust in ounces, P is power in watts, p is the pitch in inches, D is the diameter in inches, and RPM are the revolutions per minute. The pitch is measured in inches because the angle changes significantly from the center to the tips. Also, if the ratio of the pitch to the diameter is greater than 0.6, the thrust does not improve with wider pitch values while the power calculations remain the same. The selection of an appropriate propeller is made easier by looking at the efficiency of the system in motion. The equation for thrust in a dynamic environment is expressed as the following:

$$T = (\pi / 4) * D^2 * (v + \Delta v / 2) * \Delta v * \rho$$

where T is the thrust in Newtons, D is the propeller diameter in meters, v is the velocity of the substance being traveled through in meters per second, ρ is the density of this medium in kilograms per meters cubed, and Δv is the change in velocity due to the propeller. Using the above equation, it is possible to calculate the power of the system by multiplying the thrust by the speed of the vehicle. The efficiency is then the ratio

$$\eta = (T * s) / P_{\text{motor}}$$

Given a maximum power rating for the motor, one can easily graph the efficiency at various speeds for the propeller size chosen. Figure 9: shows the efficiency versus speed calculated by combining the above equations. One can see from the figure that, for a given amount of power, the smaller propeller must move faster to get the same efficiency as a larger propeller. The faster the propeller tries to turn, the less ideal the forces and the more loss in the system. The Abbott equation for power can be simplified to get a close approximation of the load a propeller is placed under, with the only knowledge necessary being the dimensions the propeller. Since power directly correlates to load applied, various diameter and pitch propellers can be compared for equal power consumption. Table 5: shows the load factor for the most common propellers. Propellers with similar load factors have similar power consumption, but the one with the largest diameter is the most efficient. The simplified equation to calculate the load factors is as follows:

$$\text{Load Factor} = D^3 * \rho$$

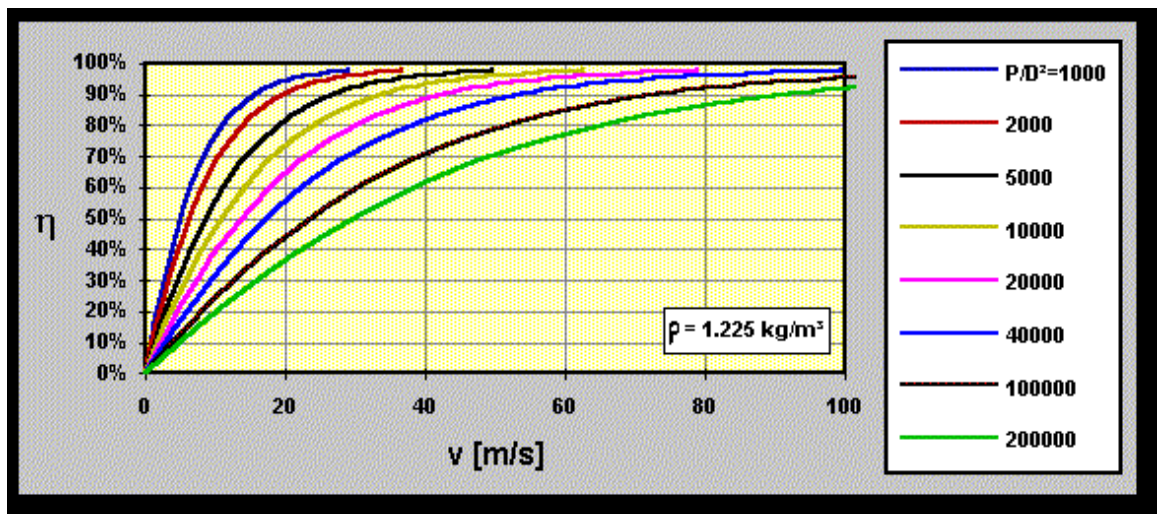


Figure 9: Efficiency versus speed for the given ratios of power to propeller size when traveling through air. Image copyright of Martin Hepperle and reprinted with permission

	p	3	4	5	6	7	8
D							
7		1029	1372	1715	2058		
8		1536	2048	2560	3072	3584	
9		2187	2916	3645	4374	5103	5832
10		3000	4000	5000	6000	7000	8000
11		3993	5324	6655	7986	9317	10648

Table 5: Load factors for common propeller sizes (D = diameter, p = pitch).

9.1.4 Hardware Placement

The distance between the opposing motors determines how quickly the quadcopter reacts. The force multiplied by the distance to the pivot point gives the torque. The pivot point in this case is the center of gravity of the vehicle, which requires careful balance with the various parts weighing down the craft. Thus, the closer the motors are mounted, the shorter the lever arm, and therefore the faster response time at the cost of stability. Conversely, distance between the motors is increased, the torque arm increases leading to slower response times but a more stable flight as seen in figure 3. L is inversely proportional to the response time of the system and proportional to the stability of the system. The motors must also be evenly matched in distance with the adjacent units to correctly counteract the angular acceleration.

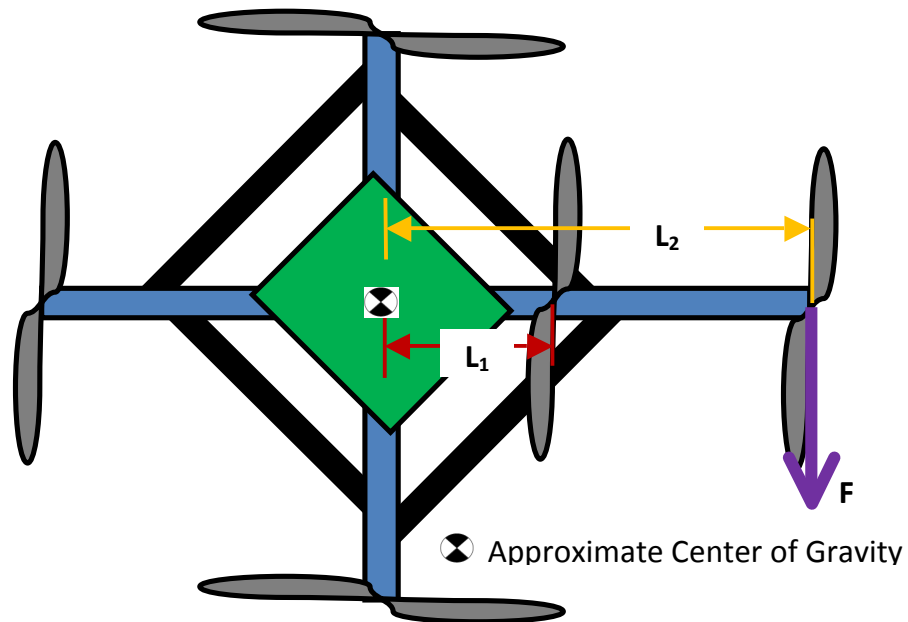


Figure 10: Rotor Placement

The center of the quad-copter will be the repository for all the boards as well as the battery pack. As stated previously, balancing the weight of all the components is crucial in obtaining the desired center of gravity to ensure stability. Having a central location for the majority of the weight allows for a more predictable center of gravity and therefore a more stable flight. Most quad-copters mount a box in the center to hold the hardware in place. Battery packs are also placed in the center and are strapped to the top or the bottom for easy access when the need to recharge arises.

Lastly, the landing gear for the quad-copter can be assembled with a variety of materials in even more arrangements. Possible options include standard radio controlled helicopter landing rails attached to the bottom center, tennis balls cut open and mounted below the motors, or a simple broad base made from spare metal. The most important aspect of the landing gear is that it protects the vehicle from damage when setting the copter down.

9.2 Power Research

The weight of the vehicle has been calculated and the respective motor has been selected to carry the load. The group can finally see what is needed to keep the system running. Digital and analog power should not mix as they can cause noise problems in the system. Noise problems could result in incorrect readings by the electronic speed controllers and the various analog components on the controller board. Therefore, the power supply can be divided or two separate power supplies can be used. The power supplies can be divided using several methods, some more robust than others. The simplest method requires several decoupling capacitors from the power supply to the ground. An alternate method is to insert a linear regulator into the system between the two lines. Care must be taken during layout to avoid coupling capacitance as well as to ensure the grounds only connect in a star formation and only when necessary.

9.2.1 Regulation

Regardless of the decision to have one power supply or two, a linear regulator will be necessary to limit the voltage for the components on the board. The power supply will need to drive a medium sized current, so a voltage reference will not be sufficient. There are several methods to create a linear regulator. One of the simplest circuit implementations is one that was taught in electronics II and can be seen in Figure 11:. At the heart of the circuit is an operational amplifier that drives the voltage on the base of a bipolar junction transistor and causes a constant output voltage value. The two diodes protect the circuit from breaking due to too much current or current trying to flow the wrong way. The output voltage is determined by the ratio of R1 and R2. The two resistors form a voltage divider with 1.2 V in the middle, so the output is $(R1 + R2) / R2 * 1.2$.

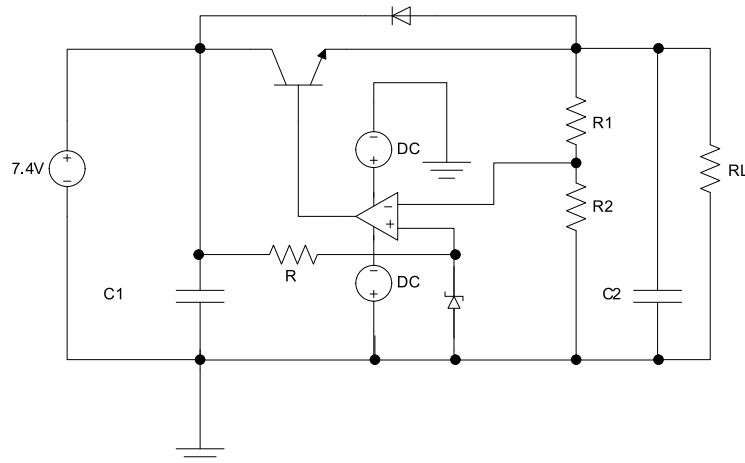


Figure 11: Simple linear regulator

Circuits made with individually created components are not extremely accurate. Conversely, integrated circuits allow the use of trimming techniques to match resistor ratios exactly. Thus, an appealing alternative to creating a linear regulator is to buy one. Many parts are available based on what the most important characteristic is to the consumer. One can choose to use an adjustable regulator or a fixed regulator with an output of 3.3 V for the project. The advantage of programmable regulators is the simplicity in changing the output voltage in the case of a change in input parameters for one of the parts. Fixed regulators will give more accurate constant voltages. Cost can vary for both types of parts and will increase with the accuracy of the part.

A third method of reducing the voltage for the flight hardware is to use a battery eliminator circuit or BEC. BECs are commonly found bundled with the electronic speed controllers so that the same battery can power both the board and the motors. However, the BECs are often simply voltage dividing resistors that dissipate the energy through heat [12]. The resulting thermal dissipation can have adverse effects on other circuit components and can destroy the BEC if it becomes too elevated. Once the BEC is damaged, the board components will likely be damaged next. Ultimate BECs use a switching power supply to reduce the voltage, but the potential for noise in such a system increases significantly. Regardless of the BEC chosen, the accuracy of the reduced voltage will not be very good.

The electronic speed controllers, or ESCs, will regulate the power received by the motors. Each ESC is controlled by the main board through a servo connection. It controls the motor speed by adjusting the rate at which the field effect transistors switch. DC ESCs control the duty cycle of the motor by pulse modulated signal. The signal is generally fifty hertz and contains one to two millisecond pulses. One millisecond pulses instruct the motors to turn off while two millisecond pulses direct the motor to full speed [13]. Variations in the

middle time period produce speeds around half power. The DC ESCs are used only with brushed motors. Brushless motors are more difficult as they require the signal to be changed into an AC signal than must be correctly timed to successfully change the magnetic field of the three-phase system. To find out at what phase the motor is currently running, the ESC reads in the counter electromotive force from the motor that pushes against the incoming current when the motor is in the breaking generator phase [13]. ESCs are rated based on the amount of current they can control. It is generally a good idea to have the ESC be rated higher than the motor so that the motor does not accidentally burn out.

9.2.2 Power Source Comparison

While an unmanned aerial vehicle could be tethered to a power source, it makes very little sense to restrict its movement in such a manner. All prototypes researched use battery power as the power supply. Methods to recharge the battery, on the other hand, do vary. Most people choose to recharge the battery with a store bought charger. Some prefer to mount solar panels to their aircrafts and have the vehicle sit in the sun to recharge. Other methods of powering and charging the system could be devised, but would probably merit time devoted as a second project. Therefore, only batteries will be considered in the present application. Batteries for most radio controlled machines are rated in milliamps per hour or mAh. This rating expresses the amount of current that can be discharged for one hour before the battery needs to be charged again. For example, a 1000 mAh battery can generate 1 amp for one hour. Careful attention must be paid to the level of charge and the evenness of discharge among cells, or the battery can be ruined. One good way to keep informed of one's battery status is to purchase a battery monitor that works while the system is in operation to reveal any potential problems. Battery monitors are very small and do not consume significant current. The total power that can be drawn from the system at one time depends on the capacity and voltage of the battery as well. All three numbers must be multiplied together to get the total maximum wattage that can be delivered to the system. Thus, if the 1000 mAh battery has a capacity of 20 C and is a 7.4 V battery, the maximum power output would be 148 watts. The maximum amperage is simply the capacity multiplied by the milliamps per hour.

Battery composition has changed significantly over the years. The technology keeps moving forward at a fast pace. Nickel metal hydride batteries are still in use, as are the slightly older nickel-cadmium batteries, but more and more people are switching to lithium batteries. Lithium provides a more constant current flow, holds a charge longer, holds more charge for a same sized unit, is a lighter material, and recharges faster. The main disadvantages of lithium batteries are their price, their inability to recover if a cell goes bad, and the increased risk of explosions if charged incorrectly. Lithium batteries have focused on lithium ion until recently, when new lithium technology has taken hold in the market. Many users are just becoming comfortable with the lithium polymer

batteries while many more are hailing the new technology and forgetting older ones. Table 6: compares the main features and drawbacks of the various battery technologies.

	Nickel-cadmium	Nickel-metal-hydride	Lead-acid sealed	Lithium-ion cobalt	Lithium-ion manganese	Lithium-ion phosphate
Gravimetric Energy Density (Wh/kg)	45-80	60-120	30-50	150 - 190	100 - 135	90 - 120
Internal Resistance in mΩ	100 to 200 ¹ 6V pack	200 to 300 ¹ 6V pack	<100 ¹ 12V pack	150 - 300 ¹ pack 100 -130 per cell	25 – 75 ² per cell	25 – 50 ² per cell
Cycle Life (to 80% of initial capacity)	1500 ²	300 to 500 ^{3,4}	200 to 300 ³	300 - 500 ³	Better than 300 – 500 ⁴	>1000 lab conditions
Fast Charge Time	1h typical	2 to 4h	8 to 16h	1.5 - 3h	1h or less	1h or less
Overcharge Tolerance	moderate	low	high	Low. Cannot tolerate trickle charge.		
Self-discharge / Month (room temperature)	20% ⁵	30% ⁵	5%	<10% ⁶		
Cell Voltage Nominal Average	1.25V ⁷	1.25V ⁷	2V	3.6V 3.7V ⁶	Nominal 3.6V Average 3.8V ⁶	3.3V
Load Current peak best result	20C 1C	5C 0.5C or lower	5C ⁹ 0.2C	<3C 1C or lower	>30C 10C or lower	>30C 10C or lower
Operating Temperature ¹⁰ (discharge only)	-40 to 60°C	-20 to 60°C	-20 to 60°C	-20 to 60°C		
Maintenance Requirement	30 to 60 days	60 to 90 days	3 to 6 months ¹¹	not required		
Safety	Thermally stable, fuse recommended	Thermally stable, fuse recommended	Thermally stable	Protection circuit mandatory; stable to 150°C	Protection circuit recommended; stable to 250°C	Protection circuit recommended; stable to 250°C
Commercial use since	1950	1990	1970	1991	1996	2006
Toxicity	Highly toxic, harmful to environment	Relatively low toxicity, should be recycled	Toxic lead and acids, harmful to environment	Low toxicity, can be disposed in small quantities		

Table 6: Lithium polymer, nickel metal hydride, lead acid, and nickel cadmium batteries compared side by side. Table courtesy of Cadex with permission [11].

9.3 Motor Research

Choosing a motor can seem like an overwhelming task as there are large amounts of options to choose from. The motors come in many selections made for specific purposes. Some of the more common variations in the types of motors are outrunner versus inrunner motors, brushless versus brushed, and a very large variety of kv ratings. The method in which the motors are mounted, and the components necessary to power them increases the complexity.

Motors are powered by magnetic fields caused by running electricity through a wire coil with a solid metal core. The motor has a stator that does not move and a rotor that powers the unit by being attracted or repelled by the stationary magnet [7]. The peak operating condition for a motor is based on a balance of torque and speed. The resistive torque of a motor increases with speed, while the torque generated by the output of the motor peaks at a certain speed and then drops[7]. The motors for model aircraft are DC motors, making the torque versus speed graph trapezoidal. Thus, operating above the critical point will actually slow down the aircraft instead of increase its speed. Due to this design, the motors can be divided into three different modes of operation. The device can be operating normally in the forward position, operating normally in the reverse direction, or in a regenerative braking state. During the regenerative braking state, the motor actually puts energy back into the system[7]. Thanks to the use of microcontrollers and increases in semiconductor technology, variable speed motors are readily available today.

9.3.1 Motor Variations

Brushed and brushless motors are the two main subdivisions in the types of electric motors. Brushless motors use magnetic rotors and a Hall effect device to drive the coils that make up the motor [8]. The lack of physical connectors reduces heat buildup, increases efficiency, allows for little maintenance, and does not suffer from mechanical limitations at increased speeds. However, they require electronic controllers to run constantly for proper operation, and the cost per part is higher. Brushed motors are less complex to control and are cheaper. The speed can be set and left at a given level, and the device does not malfunction in environments that are dangerous for sensitive electronic components. The technology is also older and much information is available on the operation of these motors.

Other variations in motor design involve the location of the wire coils. The outrunner motors are less noisy and are more resistant to damage in the event of a crash, but they can take longer to respond and use slightly more power. Conversely, inrunner motors are more efficient and can drive a larger propeller, but tend to be heavier and more complex [9]. This is because the drive shaft is controlled by gears. Thus, the inrunner motor can be better adjusted to the

application for which it is selected. Outrunner motors can also be adjusted to the application in which they are use, but require the user to do so by changing the number of coils in the system. They have an outer shell that rotates around the coils in the base inside; whereas, inrunners have the coils on the outside and a drive shaft on the inside.

The kv rating of a motor indicates the attempted revolutions per minute (rpm) the motor will try to turn based solely on the voltage received and regardless of the load applied. If the load is large, the current being drawn by the motor will increase in an attempt to maintain the given rpm. The motor will burn out if it draws a current greater than the maximum rated value; thus, it is important to get a motor that meets the criteria of the project. A lightweight plane can use a smaller propeller with high rpms, while a larger vehicle will require a larger propeller for the thrust and cause the rotating speed to decrease. The load applied to the motor refers to the force generated by the propeller, not the weight of the unit. Calculations based on the equations mentioned previously in the propeller options section can be used here to get approximate values. However, the only sure method of calculation is from measured values resulting from experiments.

9.3.2 Motor Comparison

The majority of motors suggested by do it yourself sites on the internet are brushless outrunners. Brushless motors are considered superior in almost all aspects except cost. The amount of variety in the numerous brushless motors is also impressive. The motors considered for the project include the Turnigy 2217, Hextronik DT750, and Hextronik 24 gram. Both the Turnigy 2217 and the Hextronik DT750 are heavier motors with very good thrust potential. Of course, the large thrust comes with an increase in power consumption. The Turnigy 2217 weights 71 grams and has an average thrust of over 950 grams. The low kv value of 860 rpm/V is good for larger propellers. The maximum current draw is 17 amps, which is about average for a motor of this size. Cost is somewhat higher than similar motors at \$14.04. Of the several powerful motors considered when the group first began to research the project, the Hextronik DT750 brushless outrunner motor seems the ideal for a quad copter that weighs over 2 kilograms. It has a weight of 78 grams and a thrust of over 1000 g per motor. This motor is ideal for lifting heavy loads as it has a kv of 750 rpm/V. The maximum current draw is large at 18 amps, but the efficiency is around 85% according to reviews at Hobbyking.com. Cost is not a factor at a price of \$8.99. The above motor would likely have been the top choice if the weight calculations for the quad copter had been higher. Unfortunately, the above motor is out of stock and therefore it is not guaranteed to arrive before the project is finished if placed on backorder. The search for a lightweight motor led to the Hextronik 24 gram motor. This motor is also a brushless outrunner. It has a kv of 1300 rpm/V, a thrust of over 400 grams, and only consumes a maximum of 7.5 Amps. The low weight of the unit coupled with decent thrust makes this unit very appealing

for lighter setups. Once again the price is a mere \$7.99 per motor. The main down side to this motor is the uncertainty of whether the team will be able to make a quad copter that is lightweight enough to get off the ground without overdrawing the current. Table 7 lists all the characteristics of the three motors. No inrunner motors are considered because the ones available are too overpowered for the current application. The few smaller motors found are also almost twice the price of an outrunner with similar characteristics.

Motor	Weight (g)	Thrust (g)	Kv (rpm/V)	Max. Current (A)	Cost (\$ US)
Turnigy 2217	71	950	860	17	14.04
Hextronik DT750	78	1000	750	18	8.99
Hextronik 24 gram	24	400	1300	7.5	7.99

Table 7: Motor comparison

9.4 Motor Selection

The total weight of the quad-copter according to the design is just over two pounds. The decision was made to conserve power over the guarantee of superior thrust. Few motors have qualities than compromised between the powerful and the lightweight, and the ones that do carry significant price increases. Thus, four of the 24 gram Hextronik motors will grace the final product. It is cheaper, draws less power, and is lighter than any of the competition. The cost is slightly cheaper as well. The only point of contention is whether it will be able to lift the aircraft off the ground and if it can do so without damaging itself or causing a current spike on the battery. Figure 1:2 shows the small size of the motor. The coils are clearly visible inside of the exterior casing.

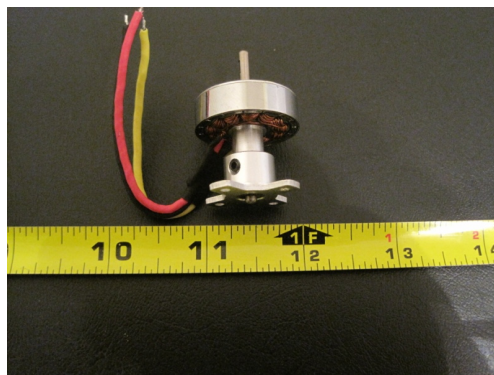


Figure 12: Hextronik 24 gram motor beside a measuring tape

9.5 Microcontroller Research

The method used for a microcontroller research involved searching the internet for UAV and quad-copter projects that others have done. The group would read about what microcontroller these various projects used and what features they require in order narrow down the amount of microcontrollers that were looked at. The two websites that gave the group the most information were the Aeroquad website and the DIY Drones website. When a microcontroller was found that was of interest for the project the group would skim the documentation available for it to decide if the microcontroller would meet the specifications and requirements that the group had decided on. The support and resources that are available for these microcontrollers would also be looked up by the group to see what kind of aid is available for the coding of the control firmware. After looking at many sources available on the internet the choice of microcontrollers to use for the project was narrowed down to the three choices discussed below.

9.5.1 ATmega328

The first microcontroller the group looked at for use on the project was the AVR ATmega328. The reason this microcontroller was considered was that it is what is used on the Aeroquad quad-copter kit that the group looked at to get some ideas for the project design. The ATmega382 is an 8-bit architecture microcontroller so it only operates on 8 bits of data at a time. It has a maximum processing ability of 20 MIPS at 20 MHz. There is a 16-bit timer and two 8-bit timers that can in total control 6 PWM channels which is enough to control the four motors on the project. There is a built in 10-bit ADC that has 8 channels which is enough for the 6 channels required by the IMU. The Atmega328 has three communication ports available. The ports are an USART, a SPI, and an I2C. These communication ports will be used to communicate with the Wi-Fi module and camera module on the control board. The operating voltage to run the ATmega328 at 20 MHz is 5 V which is within the limit of the 7.2 volt battery being used to power the electronics. The current consumption for the Atmega328 at its maximum operating frequency was not given by the documentation but should be reasonable given that it consumes a typical 5.2 mA at 8 MHz and 5 V Vcc. There is 32 KB of on board flash storage for firmware storage and 1KB of SRAM for storing data while running the firmware. The ATmega328 is available from online retailers in a price range of 4 to 5 dollars.

9.5.2 dsPIC30F4011

The next microcontroller that was looked at was the dsPIC30F4011 from Microchip. The dsPIC is intended for use in DSP applications but it was considered because it was used on the control board for a UAV project on the DIY Drones website. The dsPIC microcontroller is a 16-bit architecture controller

which means it can manipulate twice as many bits per clock cycle as the ATmega328. The dsPIC controller has a maximum processing ability of 30 MIPS at 40 MHz. The dsPIC has 5 16-bit timers that can control up to 6 PWM channels which again are enough since only 4 PWM channels are needed. There is a 10-bit ADC that has 9 channels just like with the Atmega328 this is enough to meet the needs of this project. The dsPIC controller has four communication ports which consist of two UARTs, a SPI, and an I2C. The dsPIC has 48 KB of on board flash and 2 KB of SRAM for storing and running the firmware. The dsPIC has an operating voltage of 5V when operating at 40MHz which is the same as the ATmega328. The current consumption of the dsPIC at the maximum level of performance is typically 132 mA which is a very high value compared to other microcontrollers. The price of the dsPIC is between 7 to 8 dollars at online retailers.

9.5.3 STM32F103CBT6

The final microcontroller that was researched was the STM32F103CBT6 produced by STMicroelectronics. The STM32 is based on the ARM Cortex-M3 processing core which is a 32-bit architecture microcontroller. The Cortex-M3 was designed by ARM to be used specifically for embedded applications. This leads to the Cortex-M3 being better for embedded application use compared to other ARM cores like the ARM7 for the following reasons; the Cortex-M3 features better interrupt handling, lower power consumption, better peripheral integration, and uses the Thumb2 instruction set to reduce the program memory footprint and increase instruction execution speed. The ARM architecture is one of the most used architectures in the mobile device market so there are a lot of resources available to help in developing the firmware required for the project.

While there are many manufacturers that make ARM Cortex-M3 based microcontrollers the STM32 was chosen because there was a UAV project using the STM32 and the source code is available for study. The STM32 variant chosen can run at 72 MHz and can process at up to 90 MIPS which is faster than other commonly used microcontrollers used in quad-copter designs like the Atmel Atmega328 that only operates at around 24 MIPS and the dsPIC30F4011 which operates at 30 MIPS. It has a reasonable current consumption in the 20 to 36 mA range, depending on how many of the on board peripherals the project ends up using, which is perfect for the needs of the project because it leaves a lot of the budgeted 1.5 A of current to operate the other electronic modules like the Wi-Fi or IMU modules for instance. It runs internal on 1.8 V but requires that the voltage source attached to it be between 2.4 to 3.6 V if the ADCs are going to be used. The STM32F103CBT6 has two 12-bit ADCs that can convert any of 10 available channels. This gives more channels than the project will need to convert analog signals from the IMU which only requires 6 channels. The STM32 also has 4 16-bit timers that each have 4 output channels. That means that with one timer all 4 motors in the quad-copter can be controlled. Even though there are four timers the only one the group can use for motor control is called the

advanced timer because according to the STM32 documentation it is the only timer that can generate PWM signals with duty cycles that can be changed between 0 to 100 percent. The STM32 has 2 USARTs, 2 SPIs, 2 I2C, and a USB as its communication ports. For the project only the first three types of communication ports are being considered for use. This specific variant of the STM32 also has 128 KB of on board flash memory for storing the quad-copter firmware. There is also 20 KB of on board SRAM to store data or instructions in. If the total size of the firmware can be kept below 20KB the firmware can run from within the SRAM instead of from the flash memory. Running the firmware from the SRAM has the advantage of making code execution faster because the SRAM runs at the same clock rate as the processing core while the flash does not causing the processor to have to wait for instructions. A great feature that STM32 has that most microcontrollers do not possess, is its 7 channel DMA. This feature can be used to reduce the amount of processing the STM32 has to do by sending the data from outside modules directly to the embedded memory on the STM32. Each of the seven channels are assigned to specific groups of peripherals. It is also used to transfer data from one memory location to another, from a peripheral to memory and from peripheral to peripheral without having to bother the processor on the STM32. This lets the processor work on more important things like running the control loops that keep the quad-copter stable. The price for a STM32 microcontroller varies between 8 to 10 dollars at online retailers.

9.5.4 Microcontroller Selection

Table 8 summarizes and compares the important aspects of the microcontrollers discussed above. The Table shows various parameters as well as the quantity of the peripherals in the microcontrollers important for the project. The parameters given in the table are for when the microcontrollers are operating at their maximum performance level.

While the ATmega328 has all the features needed for the project there was a concern that it did not have enough processing ability to keep the quad-copter running properly because it only operates at 20 MIPS and its 8-bit architecture limits the precision that can be achieved by arithmetic operations which is important for doing the various calculations required by the project firmware. Another concern about the ATmega328 was that the 32 KB of Flash might not be enough memory to store the firmware for the project. For the two reasons above the ATmega328 was not selected as the microcontroller for the project. Just like the ATmega328 the dsPIC has all the features required for the project but there is still concern that there is not enough processing power at 30 MIPS to run the firmware fast enough to achieve proper operation. The dsPIC also has a very high current consumption that will adversely affect the run time of the quad-copter. and this is the reason that contributed the most for not selecting it as the microcontroller for the project. For the microcontroller the group ended up deciding on the STM32F103CBT6. The STM32 was chosen above the other

microcontrollers for two reasons. The first reason is because it was superior to the other two microcontrollers in almost all the categories under consideration for the project, from processing speed to current consumption and peripheral set the other two microcontrollers could not compete. The second reason is that STMicroelectronics gives out free samples of the STM32. This was used to receive two free STM32s, so that a backup is available just in case the first one is damaged during design and testing.

	ATMega328	dsPIC30F4011	STM32F103CBT6
Data Bus Width	8-bit	16-bit	32-bit
Clock Speed	20 MHz	40 MHz	72 MHz
Execution Speed	20 MIPS	30 MIPS	90 MIPS
Operating Voltage	5 V	5 V	2.4-3.6 V
Current Consumption	Not Given	132 mA	36 mA
ADCs	10-bit, 8 ch.	10-bit, 9 ch.	2x 12-bit, 10 ch.
Communication	USART, SPI, I2C	2x(UART), SPI, I2C	2x(USART, SPI, I2C)
Timers	2x(8-bit), 16-bit	5x(16-bit)	4x(16-bit)
Memory	32 KB, 1 KB	48 KB, 2 KB	128 KB, 20 KB
Price	\$4-5	\$7-8	\$8-10

Table 8: Microcontroller Comparison that summarizes the features that are useful for the project

9.6 Development Board Research

In order to design and test firmware for the STM32, a development board will be needed. There three development boards the group was considered for the project were the STM3210B-EVAL, STM32-P103, and STM32-H103. These three development boards were looked at because they were recommended by the STMicroelectronics website as a starting point for low cost development. The development boards are discussed in more detail below.

9.6.1 STM3210B-EVAL

The STM3210B-EVAL development board is produced by STMicroelectronics. Its feature set includes the following items. This board has an LCD built in to it for applications that require a display. It has a full speed USB port, two RS-232 serial ports, and many other kinds of communication connections. There are also

many built in devices like a joystick and LEDs, which are used to evaluate the different features of the STM32. The pins on the STM32 on this development board are accessed by way of two male daughter board extension connectors. It can be powered in three ways, using a 5V power adapter, using its USB connector, or receiving power from another device that is connected to it. The board is connected to a PC through a JTAG interface. The price for this development board is around 250 dollars. This development board was not chosen for two reasons. The first reason is it has too many features that are useless for the project. The second reason was that its price is outside of what the group is willing to pay for a development board.

9.6.2 STM32 Development Environment

The development software being used for the project came with the JTAG programmer. The reasons this included software is being used are that the included software has a demo firmware programmed for use on the development board, the development software is free, and it is the recommended software environment for the JTAG programmer. The included demo firmware will be used as a reference for both configuring the STM32 when the firmware for the project is being developed and for learning the procedures that are required to compile a project successfully. Even though there are better development tools available out there they cost too much for the group to afford them and the JTAG programmer being used is only supported by two commercial development environments, the IAR Embedded Workbench EWARM and the Rowley Associates CrossWorks.

The free development environment supplied by Olimex is composed of the Eclipse IDE, GNU C compiler, and OpenOCD. The Eclipse IDE is an open source project manager and is where the C code for the project will be written. The GNU C compiler is what will convert the C code into instructions that the STM32 can execute. The GNU C compiler is not as efficient in optimizing C code to assembler as the C compilers available on commercial development environments but it has to be used because the commercial compilers have to be licensed, while the GNU C compiler costs nothing to use. The OpenOCD is open source software that is used for two functions. The first function is to write the firmware for the project to the STM32. The other function of the OpenOCD software, which is also its main function, is to debug the firmware while it is running on the STM32. OpenOCD has to be used as the firmware writer and debugger because it is the only software of this kind fully supported by the JTAG programmer. All the individual software parts above are integrated into the Eclipse IDE so all the parts of firmware development are accessed from within Eclipse. Figure 13 shows an image of the Eclipse development platform being used to develop the firmware and shows how OpenOCD is integrated into it so that debugging happens from within Eclipse.

development board. The development board is very small and has two extension ports on the bottom side to allow access to the STM32 pins. The board uses a JTAG interface located on the top of the board to allow the STM32 to be debugged and programmed. Power is delivered to the board through a USB cable connected between the development board and a PC. The board is very small and has dimensions of 2.4" x 1.3". Figure 14 shows the appearance and dimensions of the STM32-H103 development board and the locations of the JTAG, USB power connector, and extension ports.

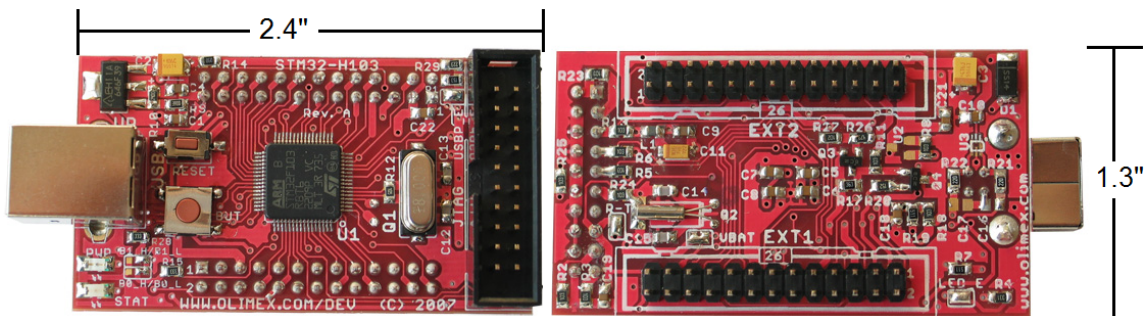


Figure 14: STM32-H103, Left is the top side of board, Right is the bottom side of board, image copyright of Olimex Ltd and reprinted with their permission

In order to use the development board a JTAG programmer cable has to be used. The JTAG programmer chosen for the development board was the ARM-USB-TINY which is also made by Olimex Ltd. The JTAG programmer allows the development board to be connected to a computer for programming and debugging. The reason the USB-TINY JTAG programmer was chosen was because it could be connected to a computer via USB port. This feature was needed because the group does not have a computer that has the serial and parallel ports that other JTAG programmers use. If a computer with serial or parallel port was owned by the group then a JTAG programmer that would have cost about half of the USB-TINY's price of 50 dollars could have been used instead. The other JTAG programmer that was under consideration was the ARM-JTAG that has a price of 21 dollars. The ARM-JTAG is also made by Olimex and would have been selected if the group had access to a computer with a parallel port. The figure 15 shows and compares how the USB-TINY differs in appearance compared to a more traditional JTAG programmer in the ARM-JTAG. The only real difference in appearance is caused by the way they interface to the computer the USB-TINY using a USB port and the ARM-JTAG using the parallel port.

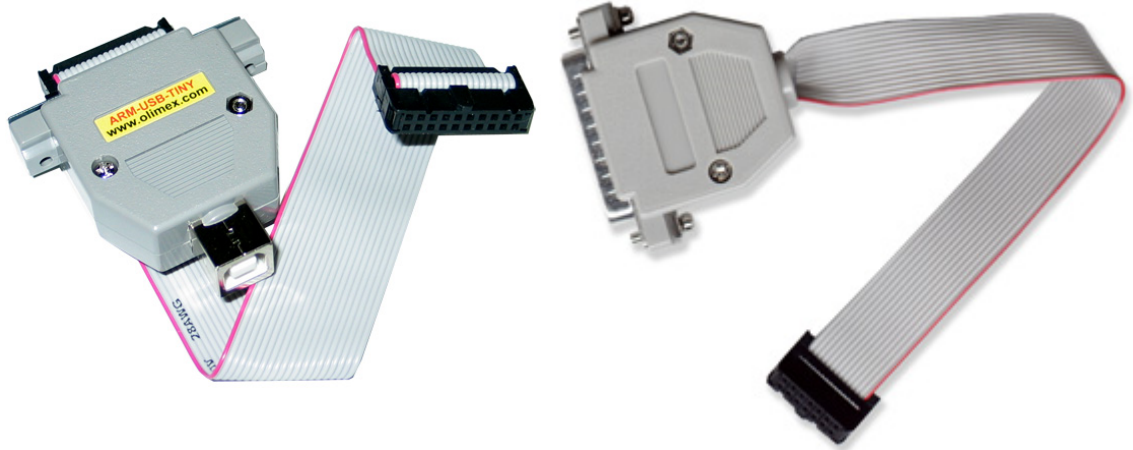


Figure 15: Comparison of the USB-Tiny(left) with the ARM-JTAG(right), image copyright of Olimex Ltd and reprinted with their permission

9.7 Control Board Research

9.7.1 Schematic Editor

The software the group decided to use to design the schematic and PCB CAD layout for the control board is called Eagle made by CadSoft. The group decided to check out Eagle because it is free and other senior design groups recommended it. Unlike a lot of other PCB design software eagle has a schematic capture mode so that the schematic is made first then after the schematic is done the PCB layout mode has all the parts and routes made in the schematic and all the user has to do is place the parts on the PCB. After experimenting with the software the group was pleased with its ease of use and features and decided to use it as the schematic creator. As for using it as the PCB design software the group will try to use it but the free version of Eagle is limited to designing PCBs with max dimensions of 4 x 3.2 inches and the PCBs can only have two layers. This PCB area should be large enough to for the needs of this project and the number of layers is perfect because there is no plan to use more than two. The backup plan just in case the Eagle PCB area is not a large enough area to place all the components is to either find a free PCB CAD or pay for a license for Eagle to remove its PCB size limitation.

9.7.2 PCB Fabrication

When it comes to having the control board fabricated there are two options. The first option is for the group to fabricate the PCB. The second option is to send the control board PCB design to a fabrication company. While there are many guides

on the internet on how to produce PCBs cheaply, the PCBs these methods produce look unprofessional and require that the proper tools, chemicals, and equipment be located and obtained. The group did not think it was worth the time and effort to fabricate a PCB to save some money when there are many PCB fabrication companies that have special offers and prices that make having a professional looking PCB affordable. To decide which PCB manufacturer will be used to fabricate the control board the PCB design files will be sent to each of the manufacturers to get a quote. The one that gives the best features for the money is the company that will be chosen.

9.8 Flight Dynamics

Flight dynamics play an integral role in the stabilization of the quad-copter. Deriving the proper equations ensure a model that behaves in a similar manner to a real life helicopter, thus allowing the control algorithms to stabilize the aircraft more efficiently.

In order to effectively describe the equations of motion, a proper reference frame must be selected. There are two types of reference frames of interest to model flight dynamics: translational and rotational. A three-dimensional spatial reference frame allows for the derivation of translational equation necessary to model the system.^[2] The translational frame of reference provides the linear position of reference.^[4] In addition the rotational frame of reference provides the orientation of the rigid body with respect to the inertial frame^[4].

The equations of motion were derived from the free body diagram of the aircraft. The aircraft has an upward thrust that is used to overcome the force due to gravity, enabling flight. Newton's laws of physics state the following.

$$F = ma$$

For a aircraft operating with a net upward force, neglecting drag and moments...

$$\sum F_{Thrust} - mg = ma$$

This representation of the forces acting on an aircraft are strictly in the Z-axis direction and do not take into account the orientation of the aircraft. The vector F_{thrust} represents the total thrust that is created by each of the four motors, to overcome the force of gravity.

In order to orient the aircraft, the individual thrust forces generated by the motors will be varied to create a net force in one of the XYZ directions, thus causing movement.^[2] Table 9: presents the variation in trust that is necessary to cause translation in the XYZ frame.

Translation	Motor 1	Motor 2	Motor 3	Motor 4
X-axis	$\downarrow F_1$	$\downarrow F_2$	$\uparrow F_3$	$\uparrow F_4$
Y-axis	$\uparrow F_1$	$\uparrow F_2$	$\downarrow F_3$	$\downarrow F_4$
Z-axis	$\uparrow F_1$	$\uparrow F_2$	$\uparrow F_3$	$\uparrow F_4$

Table 9: Translational Motion in Relation to Thrust

The control board will send the necessary pulse wave modulated signals to the motor controllers to adjust the speed of the rotors to generate the required lift.

The rotational direction of the rotors must be regulated in order to maintain proper heading. Without such a regulation, the aircraft would induce a spiral about the Z-axis. To alleviate such a movement the rotors are specified to spin in alternating opposite directions, with each adjacent rotor spinning opposite and each cross-sectional rotor having the same movement. Figure 16: depicts such a movement required to stabilize the aircraft, with regards to movement in about the Z-axis.

Figure 16 depict the rotations about the XYZ axis that are known as Euler Angles. Euler angles measure the orientation of one frame of reference with respect to the other.^[4] These angles aid in describing the orientation of the aircraft.

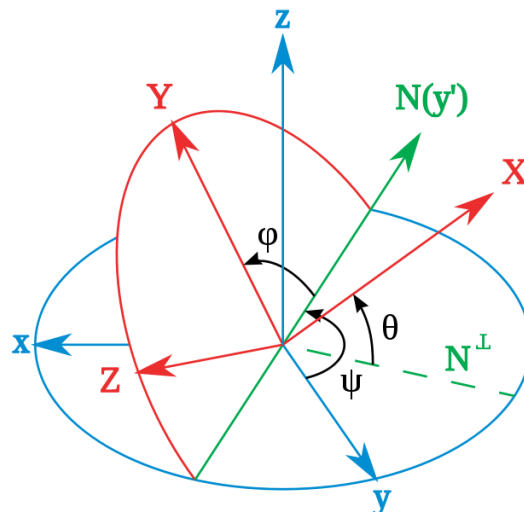


Figure 16: Euler Angles. Reprinted with permission From Wikipedia under the GNU Free Document License.

The axis XYZ as depicted in figure above is the earth's three dimensional reference frame. The axis XYZ is the three dimensional axis of the planes reference frame. Angles that are formed by sequentially rotating the plane's reference frame create Euler angles. Attitude can then be derived from the use of these angles.

9.8.1 Outside forces

Depending on the environment that the aircraft will be flying, it is necessary to take into consideration outside forces that can influence flight^[1]. As the group determines the best location for testing, and ultimately demonstrating, the device environmental factors must play a crucial role in the decision. The most general environment, and the one that will prove to have the most use in commercial applications, would be an environment that exist outside. Operation of the aircraft outside must take several factors into consideration. One of these factors being the influence that wind will have on the self-stabilization of the device. Winds can disturb the thrust that is to be generated by the rotational scheme that was discussed in the flight dynamics section.^[1]

9.9 Directional Cosine Matrix:

There are many ways in which to derive the orientation of the system, one of which is called a Directional Cosine Matrix (DCM). The method that the group are exploring builds upon a method that was developed by William Permelani that used a DCM for fixed winged aircraft. The design on the other hand would build upon the fixed wing design and adapt it to a quad rotor helicopter. The DCM presents a rotational matrix that can be used to describe the attitude of the system being analyzed^[6]. Directional cosine matrices are used to transform one reference frame to another^[6]. In the case of the helicopter, the reference frames are earth and the aircraft. Directional cosines are essentially a representation of a vector. For example, take a vector that is present in the XY plane. The vector \vec{x} can be represented by directional cosines.

$$r_x = \cos(\varphi)$$
$$r_y = \cos(\psi)$$

The most important aspect of the directional cosine matrix is the data that resides with in the matrix. The matrix is comprised of the angle that is formed between the vector and the coordinate axis^[1]. Rotational matrices can then be used to transform between two types of reference frames. Each row of the rotational matrix presents the unit vector as seen by the other reference frame, in the case the earths. To transform a vector to the other reference frame you must multiply that vector by the rotational matrix. Care must be taken when using matrix multiplication due to the fact that matrix multiplication is not commutative^[1], meaning...

$$y' = R * y \neq y * R$$

where...

$$R = \begin{bmatrix} r_{XX} & r_{XY} & r_{XZ} \\ r_{YX} & r_{YY} & r_{YZ} \\ r_{ZX} & r_{ZY} & r_{ZZ} \end{bmatrix}$$

$$y = \begin{bmatrix} y_X \\ y_Y \\ y_Z \end{bmatrix}$$

Each row of the rotational matrix describes the presents the aircrafts rotational vector as seen by earth's reference frame^[1]. The directional cosine representation for the vector with regards to the earth's reference frame, as seen by each individual axis. For example, take the vector \vec{r} . This vector is comprised of a component in the X-axis,

$$\vec{r} = r \cos(\psi) + r \sin(\zeta)$$

where...

$$\zeta = \psi - \frac{\pi}{2}$$

yields...

$$\vec{r} = r \cos(\psi) + r \cos\left(\psi - \frac{\pi}{2}\right)$$

Using this description the group are able to derive the directional cosine for the

$$r_{xx} = \cos(\psi) \quad r_{xy} = \cos(\psi - 90) \quad r_{xz} = \cos(0)$$

vector \vec{r} .

It is important to note that the magnitude of the vector \vec{r} is equal to 1. This vector is called a unit vector and is the fundamental reason that you are able to multiply any vector by the rotational matrix, to describe the orientation^[1]. Since the vector's magnitude is only one, the multiplication changes the orientation of the vector, without modifying the magnitude.

The order of rotation is also important to describing the orientation of a vector. Using the Euler angles (Φ , θ , and ψ), an order of rotation must be selected. For example if the order ZYX is followed for rotations corresponding to $\psi=45$ $\theta=90$ $\Phi=45$, the following rotation will result in a vector pointing in an equal but opposite direction to a rotation of $\Phi=45$ $\theta=90$ $\psi=45$. It is for this reason that the standard order of rotations is taken to be XYZ, for all calculations.

Directional Cosine Matrices can be easily transformed into the Euler angles by

utilizing the following property.^[1]

$$R = \begin{bmatrix} \cos \theta \cos \psi & \sin \phi \sin \theta \cos \psi - \cos \phi \sin \psi & \cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi \\ \cos \theta \sin \psi & \sin \phi \sin \theta \sin \psi - \cos \phi \cos \psi & \cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi \\ -\sin \theta & \sin \phi \cos \theta & \cos \phi \cos \theta \end{bmatrix}$$

The rotational matrix can then be multiplied by the ground reference vector to derive the aircrafts orientation.^[1] Conversely due to the orthogonality of the ground matrix and the rotational matrix; the matrix created by this multiplication can be multiplied by the inverse of the rotational matrix to realize the ground reference matrix.

Now that the group have discussed the preliminaries of the rotational matrix, the group are able to discuss how this is used to implement a DCM. The rotational matrix is updated by integrating the kinematic equation that describes the rotation of the ridged body being analyzed. The gyroscopes out put a voltage that is proportional to the rate of rotation that is being experienced by the aircraft.

The rate of rotation is equal to...

$$\frac{dr(t)}{dt} = \omega(t) \times r(t)$$

$\omega(t)$ = rate of rotation (gyro output)

This equation is then integrated to produce the following equation.

$$r(t) = r(0) + \int d\theta(t) \times r(t)$$

where...

$$d\theta = \omega(t)dt$$

The equation equates to the following numerical integration that is preformed on each row of the rotational matrix.

$$r(t + dt) = r(0) + r(t) \times d\theta$$

A generalized form yields...

$$R(t + dt) = R(t) \begin{bmatrix} 1 & -d\theta & d\theta \\ d\theta & 1 & -d\theta \\ -d\theta & d\theta & 1 \end{bmatrix}$$

9.10 Inertial Measurement Unit

Initial research for the inertial measurement unit (IMU) began by looking at the necessary sensors that are required to measure the directional characteristics. The first sensor that is needed to calibrate the aircraft prior to take off is a magnetometer. The magnetometer measures the strength and direction of the earth's gravitational field in order to detect heading. This device will give the IMU the initial heading of the aircraft to set the earth reference frame that can be used in the Directional Cosine Matrix (DCM). The magnetometer is also used for the calculation of yaw drift error created by the gyros. One downside to using magnetometers in such a way is that they are susceptible to stray magnetic fields and ferrous materials. Such instances introduce errors into the heading and yaw drift calculations.

9.10.1 Gyroscopes

Mechanical Gyros have been used in aviation for a number of years to maintain a desired heading. The same concept can translate to the electronic gyroscopes that are present in today's IMU's. To measure rotational movement Micro-electro-mechanical (MEMS) Gyroscopes will be used. The MEMS gyroscopes output the rate at which the aircraft is rotating in the XYZ axis. For the purpose of the device testing that was done prior to the prototyping of the system a dual axial gyroscope will be utilized to monitor X and Y-axis rotations. With the addition of a single axis gyroscope yaw rotations about the Z-axis will be monitored.

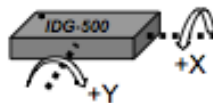


Figure 17: DG500: Dual Axis Gyroscope Direction of Detectable Angular Rate
(Reproduced with permission from InverSense)

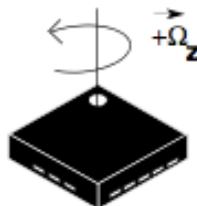


Figure 18: LY530AL: Single Axis Gyroscope Direction of Detectable Angular Rate
(Reproduced with permission of STMicroelectronics)

9.10.2 Global Positional System

Much thought has been put into how to go about compensating for the drift that is present when using gyroscopes. An alternative that is more widely used for the purposes of calculating yaw drift compensation in gyros, for purposes of IMU design, are the use of Global Positioning Systems (GPS). GPS allows a point of reference for the flight control systems to renormalize themselves to correct yaw drift that is introduced by the gyro. Global positioning systems provide the translational location of the device based on the data that is received from satellites that are orbiting the earth. GPS modules that are available provide a plethora of useful information, such as altitude, velocity, heading and position.

During the course of the research the group have decided that a GPS cannot be utilized to compensate for the yaw drift that is present in the gyros^[1]. The reason for this is the lack of forward motion. To derive the yaw from the GPS data you must utilize the velocity vector that is generated by the module. This works quite well if your aircraft is in constant forward motion. In applications where the device is operating in a stationary position there is no way to derive the yaw due to the absence of a velocity vector. It is for this reason that a GPS unit is inadequate for the design.

9.10.3 Accelerometer

Movement will be referenced by the measurements received by an accelerometer. The ADXL335 provides XYZ axis measurements of the perceived gravitational that the aircraft is experiencing. Accelerometers measure the gravitational force acting on the device minus the acceleration. To achieve these measurements MEMS technology is utilized, a capacitive plate rests on a pedestal that exists on the substrate. The capacitive plate has a cutout in the center that consists of a torsion bar crossing a void that exist in the center of the plate. The center of this torsion bar is where the capacitive plate rest on the substrate pedestal. A lower capacitance plate is present on the substrate, thus completing the capacitor. In the presence of acceleration the upper capacitance plate will create a moment around the pedestal that will cause a fluctuation in the capacitance existing between the upper and lower capacitance plates. This change in capacitance translates into a change in voltage that can be correlated to a specific value of gravity. The chip that the group are using in the preliminary test for the IMU design is the ADXL335. This chip is precise for up to 3 g's.

Based on the orientation of the accelerometer you are able to determine the gravity plus acceleration vector that the aircraft is experiencing. Figure 19 depicts the orientation scheme that is present on the ADXL335. The disadvantage of accelerometer is that they measure the gravitational force that is experience by the aircraft along with any other forces experienced.

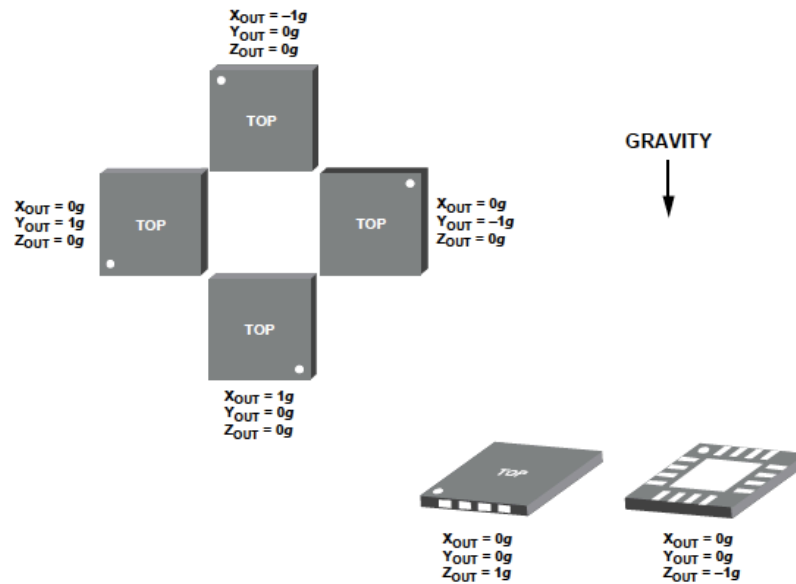


Figure 19: Sensor Axis of the ADXL335. (Permission pending from Analog Devices)

The ADXL335 chip that is being utilized for the IMU allows the user to band-limit the accelerometer to alleviate aliasing and noise. According to the datasheet utilizing the capacitor values specified in table 10 could choose the frequency.

Bandwidth (Hz)	Capacitor Value (μF)
1	4.7
10	.47
50	.10
100	.05
200	.027
500	.01

Table 10: Bandwidth as a function of capacitor value.

The correct capacitor values for the low pass filtering of the outputs is extremely crucial and will aid the software filtering that is chosen for the IMU. Further testing will be done to correctly choose the proper capacitor value.

For the purposes of the IMU design the accelerometer will be used as a reference vector that will correct any error in the XY plane calculations received from the gyroscope. This idea is further developed in the IMU design sections.

9.10.4 Thermopile

Thermopile inferred sensors were also investigated for possible use as an alternative to utilizing an inertial measurement unit. Thermopiles measure the temperature gradient that exists between the earth's surface and the sky. This gradient is then used to derive the orientation of the aircraft. There are several advantages and disadvantages to utilizing such a sensor. An advantage of such a sensor is the price. Thermopiles are, on average, much cheaper than implementing a complete inertial measurement unit. These types of sensors are also much easier to work with, with regards to programming the control algorithm, that being said, they are also not as accurate as IMUs in certain environments, such as colder weather. Thermopiles are also not as accurate in mountainous environments because fluctuations in the terrain can be interpreted as movement by the thermopile.

9.10.5 Magnetometers

Prior to the testing of the design the group realized through research of the gyroscopes that they are susceptible to yaw drift. There are several ways to alleviate yaw drift. In many applications GPS units are used as a reference vector to remove the yaw drift that accumulates in gyroscopes. Through the research of GPS modules the group quickly realized that using GPS data as a reference vector would not be possible due to the nature of the device. The device must be in constant motion in order to derive the necessary vector for comparison. Using a GPS module for yaw drift calculations involves these necessary steps:^[1]

1. Create an initial GPS vector.
2. After a small change in time, create a secondary reference vector.
3. Calculate the angle between the two reference vectors; this is the yaw angle.
4. Use the calculated yaw angle to compare with the gyro yaw angle
5. Calculate the error in the measured yaw angle

Once the calculated yaw error is realized it is fed through a PID feedback controller to provide the IMU with a correct yaw position of the aircraft. Through the research the group realized that the GPS module was insufficient and the search for an alternative turned the attention to the 3-axis magnetometer.

Magnetometers measure the strength of the earth's magnetic field in reference to the position of the aircraft.^[8] In addition to measuring the magnitude of the earth's magnetic field, magnetometers also measure the direction. There are many different types of magnetometers that are widely available. Due to the flight dynamics and nature of the device, it was realized that the group would need a 3-axis magnetometer. The reason for this is when the device is operating at an

angle with respect to earth's horizontal axis, the magnetic field vector that will be experienced by the magnetometer will vary according to angle formed. Therefore, utilizing a 3-axis magnetometer will allow you to derive the true orientation of earth's magnetic field.

For the reasons stated, the group have decided to utilize HMC5843. HMC5843 is a 3-axis magnetometer that utilizes Anisotropic Magnetoresistive. Magnetoresistance is a property of a material that causes a fluctuation in the electrical resistance based on the magnitude of magnetic field that is experienced. Specifically Anisotropic Magnetoresistive is the property of a material in which the electrical resistance depends on the angle between the direction of the magnetic field and the orientation of the device.^[8] The implementation of AMR in the HMC5843 allows for the both the calculation of the magnitude and direction of earth's magnetic field.

There are several different 3-axis magnetometers that were being evaluated for use in the system. HMC5843 was chosen for the features that were available. Some of these features included the ability to provide offset calculations. The ability to perform these calculations alleviates the need for software offset compensation. This chip also utilizes an I2C digital communication interface. Utilizing this communication protocol will greatly simplify communications with the central processor, the STM32 CORTEX processor. The HMC5843 also has sensitivity in the micro-gauss to 6 gauss.

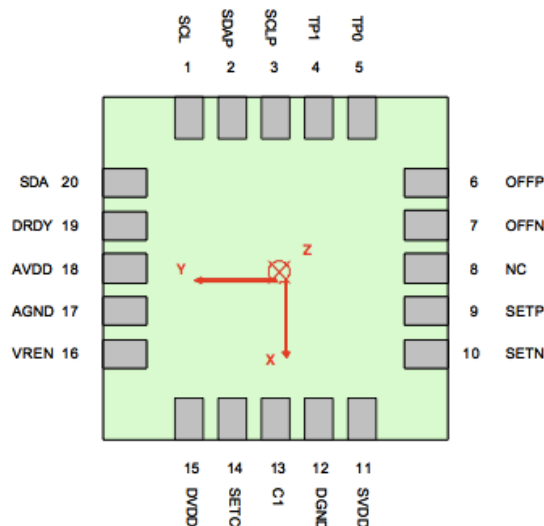


Figure 20: HMC5843: Magnetometer Directional Assignments and Pins (Bottom View) (Reprinted with permission of Honeywell)

9.10.6 Control

The implementation of a control loop to correct the error that is present in the gyroscopes is essential to the utilization of the rotational matrix. The rotational matrix relies on the accuracy of the elements of the matrix. These elements are the cosines of the angles between the ground reference matrix and the orientation of the aircraft. In order to increase the accuracy the rotational matrix must be compared to another matrix that measures the orientation, but is done by a different technology. The design that is used in the IMU utilized an accelerometer for XY measurements and a magnetometer for Z-axis measurements.

A Proportional, Integral, Derivative controller is used to further correct the error that lies in the drift calculation between the reference matrix and the rotational matrix. PID controllers utilize three different terms to try to adjust for errors that are present in sensor measurements.^[9] The proportional term is essentially a gain, correcting the output in proportion to the error value. A constant is used to tune the proportional that is multiplied by the error value. The integral term is used to decrease the rise time for the system response. The error signal is integrated and multiplied by a tuning constant. The final term is the derivative term that compensates for the overshoot that is caused by the proportional term. This term is also multiplied by a tuning constant. All of the tuning constants K_p , K_i , and K_d have several ways to realize the appropriate values for optimal system response. The method that will be selected initially will be derived through testing. Other methods will be explored during this phase.

9.10.6.1 Stabilization

There are several different ways to implement the stabilization controls that are necessary for flight of the aircraft. Each of these different methods comes with their own set of advantages and disadvantages.

The first method that was discussed was implementing a simple control loop that would simply maintain a hover by referencing the desired XY rotations for hovering ($X = 0$, $Y = 0$). Utilizing Proportional, Integral, Derivative controller the group would then be able to minimize the error between the sensor measurements and the desired position. Any movement command that is sent to the aircraft would be added to the output of the control loop and subtracted from the sensor measurements to allow for movement. There are advantages and disadvantages to such a simple method. Implementing such a method allows for extremely simple coding, thus alleviating mistakes that can be introduced into the code. Although this method is simple to implement the control loop is simply not robust enough to handle the errors that are present in the sensor measurements, specifically the gyroscopes. This method was immediately ruled out.

A second method that was explored was utilizing a Kalman filtering algorithm to maintain the orientation of the aircraft. Such an algorithm utilizes the averages and the errors in the sensor measurements to predict the outputs of the sensor. Each prediction is constantly changing based on the actual measurements recorded by the sensors. Many aircraft have been built utilizing such a method with quite favorable results. The disadvantage to this method is that it requires a significant amount of computing power due to the complexity of the code that must be run. The increase in the amount of computing power introduces many different hazards including an increase in power consumption and an increase in the overall cost of the device. Kalman filtering is discussed in further detail in the sensor data filtering section.

9.10.7 Ultrasonic Range Finder

The group will need a sensor to tell the quad-copter how far away the copter is from the ground. A reliable technology for this need would be ultrasonic technology. Within ultrasonic technology the group need to figure out the type of ultrasonic sensor the group need to meet the quad-copters goal of being able to hover between 1 and 10 feet. The sensor will help keep the copter will be able to tell the change in elevation and keep the copter safely away from the ground while the user is controlling the quad-copter from the iPhone.

There are many different companies that make high quality and resilient ultrasonic sensors that would meet the criteria for our application. The group has chosen to go with MaxBotix Inc. because of their readily available datasheets and also an easy to read selection guide to help pick the correct sensor to match the needs.

One of the main characteristics about the ultrasonic sensor is the width of the beam that the sensor sends out to measure and observe the surroundings. Wider beam widths are better to recognize and report for obstacle detection, people detection, collision avoidance, detecting small objects. The narrower and smaller the beam width is better used to detect acoustic noise, room mapping and clutter rejection. All beam widths can perform these tasks but the more specific the beam width for the application, the better the performance of the ultrasonic sensor will perform. The figure 21 shows how the width of the beam can detect the different sizes of the dowel.

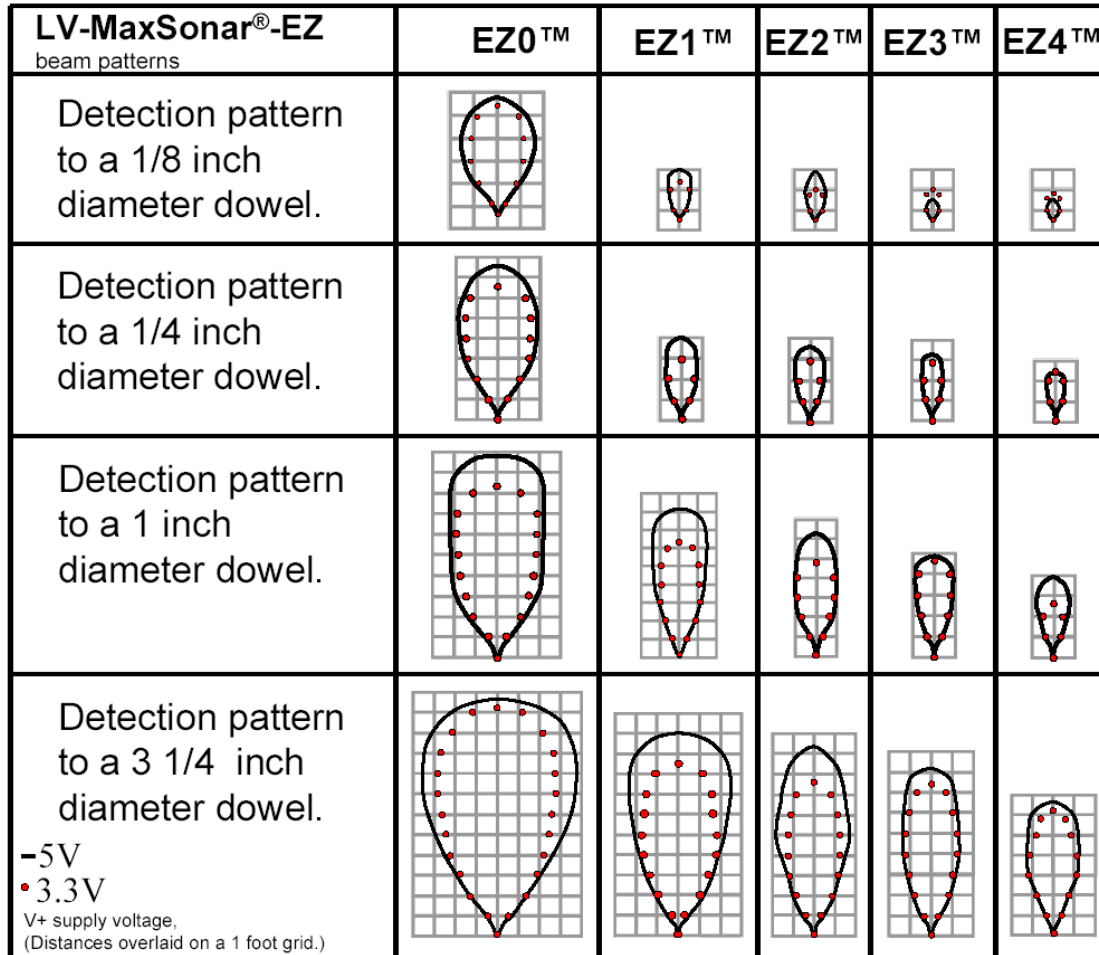


Figure 21: Beam width chart (Reprinted with permission from MaxBotix)

For the purposes, the group will need a wider beam width to better detect the ground and other objects that could possibly be under the quad-copter as it flies over. Along with having a wider beam to detect bigger objects, the wider beam will also allow for a better view angle and provide a greater range of vision than a narrower beam. This will make the chance of detecting objects that are not directly under the copter much greater than having a narrower beam. Other aspects of the wireless transceiver that need to be considered are clearly and nicely laid out in Figure 22 that was provided by the MaxBotix Selection Guide for their Ultrasonic Transceiver line.



					
Easy to use interface	Yes	Yes	Yes	Yes	Yes
Trigger or Free-run Operation	Yes	Yes	Yes	Yes	Yes
Stable Range Data	Yes	Yes	Yes	Yes	Yes
Serial Output	Yes	Yes	Yes	Yes	Yes
Analog Voltage Range Output	Yes	Yes	Yes	Yes	Yes
Pulse Width Output	Yes	Yes	No	Yes	No
Real-time Analog Envelope Output of the Acoustic Waveform	No	No	Yes	No	Yes
Outdoor Use (IP67 Rated)	No (can be mounted in a way that protects the sensor from exposure to the elements.)	No (can be mounted in a way that protects the sensor from exposure to the elements.)	No (can be mounted in a way that protects the sensor from exposure to the elements.)	Yes	Yes
Recommended for Industrial Use	Some	Yes	Yes	Yes	Yes
Recommended for Hobby Use	Yes	Yes	Yes	Yes	Yes
Automatic Calibration to Compensate for Changes in Temperature, Voltage, Humidity and Noise.	On power up only	Yes	Yes	Yes	Yes
Has noise canceling	Some	Yes	Yes	Yes	Yes
Resolution	1 inch	1 cm	1 cm	1 cm	1 cm
Maximum Rate Readings are taken	20Hz	10Hz	10Hz	10Hz	10Hz
3.3V Operation, Average Current Draw	2mA	2.1mA	2.1mA	2.1mA	2.1mA
5V Operation, Average Current Draw	3mA	3.4mA	3.4mA	3.4mA	3.4mA
Acoustic Frequency	42kHz	42kHz	42kHz	42kHz	42kHz
Minimum Object Detection Distance ⁽¹⁾	0"	0 cm	0 cm	0 cm	0 cm
Minimum Reported Distance ⁽¹⁾	6 inches	20 cm	20 cm	25 cm	25 cm
Maximum Range	254 inches (6.45 meters)	765 cm ⁽²⁾ 1068cm (25.1 feet)	765 cm ⁽²⁾ 1068cm (25.1 feet)	765 cm ⁽²⁾ 1068cm (25.1 feet)	765 cm ⁽²⁾ 1068cm (25.1 feet)
Semi-custom solution available to meet almost any need	Yes ⁽³⁾	Yes ⁽³⁾	Yes ⁽³⁾	Yes ⁽³⁾	Yes ⁽³⁾

Figure 22: MaxBotis Product line Features (Reprinted with permission from MaxBotix)

From the figure above, and also figuring the price into the equation, the best sensor for the needs is the XL-MaxSonar-EZ. From this sensor group the group will then choose the largest width beam which will be the EZ0 which has the largest beam according to Beam Width Chart in figure 1. This sensor has a low current draw, one of the lowest minimum reported distance and also has a maximum sensor range of just over 25 feet that will be well within the specifications that the group had put forth.

9.11 Sensor Data Filtering

The sensors that will be used in this project are susceptible to errors and interference. One such interference includes the accelerometers being susceptible to vibrations that can offset the data in give false readings. There are several ways to alleviate such an error.

One of the methods that is widely used, and also happens to be the most complex, is Kalman Filtering. Kalman filtering is an attempt to take data that is observed over a period of time and remove the erroneous components using mathematical filtering.^[3] The main attribute of such filtering is its ability to produce estimates that are closer to the true sensor measurement values. The Kalman filter uses the dynamic equations of a system, taking into consideration the controlling inputs, and produces an estimate of the varying outputs.^[3] The Kalman filter is a recursive estimator, meaning that only the previous state and the current sensor input are needed to produce an estimate. This type of filter proved to be the most complex out of the three that were explored and required the most computing power. Since the group are using a microcontroller more research is necessary in order to determine whether or not the group would be able to implement such a filter.

A widely used alternative to the Kalman filter for IMU control design is the Complementary Filter. Complementary filters make use of multiple sensors that measure the same event.^[10] The measurements are then compared to produce an output the filters out some unwanted component, in the case noise. A complementary filter is a filter that has a phase of zero and a magnitude of one. There are several ways to combine the sensor data to produce a filtered output. One technique for combining sensors is called blending.

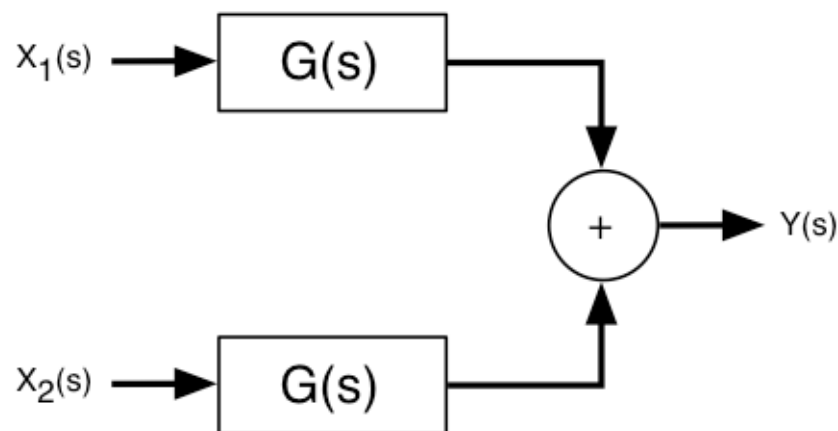


Figure 23: Sensor Blending

Typically for blending two sensors that sample the same event, operate in different frequency bands.^[10] For example a GPS can be used to specify an object's orientation. This data is then passed through a low pass filter and added

to an inertial sensor, such as a gyro, that is passed through a high pass filter.

9.11.1 Wireless Communication

The group will need wireless communication to talk between the iPhone and the quad-copter. There are many different technologies out there that allows for wireless communications between devices such as Zigbee, EnOcean, Bluetooth, Wi-Fi (802.11), ISA100, WirelessHART, IETF RPL, IEX 62591 and EnviroNet. If the group were to use any of these technologies that was not compatible with the iPhone, the group would then need to have to set up a host computer or relay point to transfer the data between the iPhone and the quad-copter. If the group were to use a host computer it would just add overhead costs to the project and also make the project less portable which was a big drawback to the initial plan. The group decided to go without having a host computer and allowing the iPhone and the quad-copter to directly communicate with each other.

By having the quad-copter and iPhone directly communicating between each other there were only 2 real choices to choose from to communicate with the iPhone and it was either Bluetooth or Wi-Fi network. Bluetooth and Wi-Fi both have their advantages, such as Bluetooth is low cost and low power and Wi-Fi has great range and greater data bandwidth. They also have their disadvantages as well, Bluetooth has a much smaller range than Wi-Fi and cannot transmit as much data as Wi-Fi can. Wi-Fi is open to everyone unless you give it security features and also takes a lot more power and has more overhead to get through that Bluetooth.

	Wi-Fi(802.11g)	Bluetooth
Bit-rate	54Mbps	3Mbps
Range	300ft(outdoors)	30ft(outdoors)

Table 11: Wi-Fi vs Bluetooth Chart

After going through and comparing the two technologies there was a clear front runner between the two. Bluetooth just did not have the data bandwidth and the range that the group were looking for in the project. The group wanted a range of at least 100ft and Bluetooth could not handle that amount of range efficiently enough. Wi-Fi was able to offer the range and bandwidth that the group needed to have for the design. Both of those factors, range and bandwidth were very important for the project especially the bandwidth in the event the group incorporated a video stream that would allow to show the “front” view of the copter to the user on the iPhone.

There are many different companies out there that produce a Wi-Fi transceiver module that would suit the means. The company that the group has chosen to choose from is called Roving Networks for they have readily available datasheets, Evaluation boards and walkthroughs and technical support. They

have 3 different modules that would fit the need for the design: RN-121, RN-131 and RN-134. The three different modules offer small form factors, low current draws both during receiving and transmitting and provides high data rates.

From table 12 you can see that the main differences between these modules are the Transmit Current and the Memory on the modules. The RN-131 has a bigger current draw then the RN-121 but it also has memory on the module that allows it to save the configuration files on the module to connect without having to have the microprocessor feed the transceiver with the configuration files on start up making it a quicker start up.

Roving Networks Modules Comparison Chart			
	RN-121	RN-131	RN-134
Transmit Current	110mA	140mA	140mA
Receive Current	40mA	40mA	40mA
Sleep Current	12uA	4uA	4uA
Standby Current	35mA	15mA	15mA
Supply Voltage	3.0V – 3.6V	3.0V – 3.7V	3.0V – 12V
Transmission Rate	54Mbps	54Mbps	54Mbps
Memory	N/A	128KB RAM, 2MB ROM	N/A
Hardware Interface	UART, RS232, RS485	UART	UART, RS232

Table 12: Roving Networks Modules Comparison Chart

9.11.2 Software Communication

To control the quad-copter the group decided that the group wanted to have a fun and widespread, easy to use portal to have the user control the quad-copter. The group decided that the iPhone would be a great portal for users to be able to control the copter and to do this the group needed to figure out how to have the iPhone communicate with the copter. The main data that needs to be sent between the iPhone and the quad-copter is the control signals that move the copter and the video feed from the copter to the iPhone.

The group decided that a Wi-Fi module was the best choice for the needs to send the video feed and the control signals back and forth between the copter and the iPhone. On the iPhone to control the copter the group will need an application to run the controls of quad-copter. There are two main types of applications that you can develop for the iPhone. There are web applications and

native iPhone applications and both have positives and negatives associated with them. With web applications you run the application inside a browser as in this case would be Safari. It is written in HTML and JavaScript and can be accessed by other phones such as the Palm and Android. Developing a web based Application is free and requires no apple membership. iPhone Native Applications run on the operating system of the iPhone, can use the hardware features such as the accelerometer and compass. Native applications are written in Objective C and can use the other frameworks from the OS. There are some setbacks for both web based and Native applications. For web based applications, web based cannot access the iPhone resources such as contacts, photos and calendar and the phone must be able to connect to the internet as well. They also cannot access any of the hardware features such as the accelerometer, camera or compass. For Native iPhone apps, they cannot run on multiple platforms which limit the number of users able to use the copter. Another setback for native iPhone apps is that to develop them it requires registering with apple and purchasing a license each year for a cost of \$99 per year to develop native iPhone apps. iPhone native applications also require that they be programmed and developed on Mac machine because the iPhone SDK is only available on the MAC OS X operating system. Based on the information above and table 13 the group have decided to choose the Native iPhone application because of the flexibility and the power that the Native app gave .

	Native iPhone App	Web App
Portable?	No	Yes, most times
Use hardware features	Yes	No
Store data on the iPhone?	Yes	No
Network connectivity required to access the app?	No	Yes
Speed	Faster	Slower
Technology/language written in	Objective C	HTML/JavaScript
User needs to install anything on the iPhone?	Yes	No
Updates need to be installed?	Yes	No

Table 13: Difference between iPhone Apps

After the group decided on using a Native app for the iPhone the group then had to decide what type of project template the group wanted to write the application in. There are 6 types of project templates out there for native iPhone applications: Navigation-Based Application, OpenGL ES Application, Tab Bar Application, Utility Application, View-Based Application and Window-Based

Application. Navigation-Based Application is an application that shows the data in a hierarchical way using a few screens to show the data, such as the contacts application on the iPhone. OpenGL ES Application is mainly used to show images and animation, video games for the iPhone are developed using this template. Tab Bar Application is an interface that lets the user choose between several screens through a radio interface. The Utility Application has a main view for the user to do the main operations of the application and then a flipside view to let the user make changes or modify the main view. View-Based Applications uses a multiple screens that the user can navigate through the user interface. Window-Based Applications is much like a blank space template where the developer can implement his own type of data hierarchy and use as many screens the developer wants to use to make his application.

The group chose to go with the view based application because of the power and the flexibility that the template offers to match the needs that the group need for the project. It gave the option to have multiple menus for the user to flip through and change different settings on different pages and then give the base view to allow for the user to control the copter from this view.

To control the copter the group needed to decide on how to set up the user screen to allow for an easy interaction and control method for the user. There are many ways to control the copter from the iPhone. The group first thought that the perfect solution to controlling the copter would be to use the accelerometer inside the iPhone. By tilting and rotating the iPhone the user will be able to move the copter in any direction based on the amount of tilt and rotation that the user applies. The accelerometer on the iPhone will be able to let the application know how much tilt the user is applying and then the group would be able to translate that into movement to the copter in the air. The group would then use the built-in movie player that comes with the Cocoa Touch library, MPMoviePlayer and then have the video coming from the copter and stream it live through an mp4 video stream. There were a few drawbacks to having the accelerometer control the copter. The accelerometer is extremely sensitive to any tilt that the user applies to the iPhone. If the user did not hold the iPhone very still the copter will always be moving in the direction that the iPhone is tilted in. Another drawback to only having the accelerometer control the iPhone and the MPMoviePlayer showing the video of the copter is that there is no way to allow the copter to rise and fall in height because the accelerometer is busy moving the copter left, right and forwards and backwards. Also the design was not a very fluid and steady way to control the copter as the controls were jerky and moved the copter too much. To be able to steady the copter you would have to keep the iPhone extremely still and this would be a difficult if the user is also trying to change the settings of the copter with one hand and keeping the iPhone stable at the same time.

Another way the group thought of controlling the iPhone is by having 3 different sliders on the phone. 2 of the sliders would be on the either side of the screen with the sliders being balanced out right in the middle of the screen and a third

slider on the bottom section of the screen to adjust for the height and it would indicate the current height of the copter. The user will then be able to control the copter by moving the sliders on either side of the screen. To move forward the user would just move both the sliders on either side of the screen forward at the same rate and it would smoothly move the copter forward, backwards would be the same way but moving the sliders backwards. There would be many ways for the user to turn the device using the sliders on the side of the screen. The user could just use one of the sliders to turn the copter in either direction or use both of the sliders moved in opposite directions to turn the copter. To make the copter rise and fall all the user would have to do is adjust the slider on the bottom of the screen to the desired height for the copter. The quad-copter will have a safety device not allowing the copter to get too close to the ground in fear of crashing. The copter will use an ultrasonic sensor to sense how far away the copter is from the ground and when it reaches a certain threshold it will not allow the user to lower the copter anymore. The group would be able to have the sliders be overlaid on the MPMoviePlayer allowing for a crisp smooth stream of the video beneath the sliders.

With this design the group would have all the functions to control the copter on the main view and not have to rely on the user keeping the iPhone steady all the time. Without using the accelerometer, this relieves a big safety threat that would have been present if the group would have used the accelerometer. If the user accidentally moved the iPhone too fast or had trouble accessing the menu on the iPhone it could accidentally move copter in a way the user did not intend for it to move.

The group will also have other menus besides the main screen to host other options and show the status of the helicopter. The information displayed would be how high off the ground the is, the signal strength between the copter and iPhone, and other statuses that the group come across that will let the user know what is going on with the copter.

To start the application the group will have a start up screen that will ask the user if it wants to connect to the quad-copter. When the user connects to the copter, the iPhone will send over a request to connect to the transceiver chip located on the quad-copter and also send over the security codes that will be embedded in the iPhone application to make sure that no other individuals will be able to connect to the copter.

10.0 Design

10.1 Vehicle Body Design

The final quad-copter prototype will be a lightweight design of under two and one half pounds. A list of parts with the associated weights is given in table 14. By reducing the weight, the size of the motors needed to lift the craft is reduced. Motors will need at least three hundred grams of thrust, with a preference of up to four hundred grams. Thus, less power will be consumed by the smaller motors than for the average size motors used in such applications. The group feels it is within reason to attempt a flight time of approximately 15 minutes in length. The largest gauge wire that is thick enough to deliver the necessary current will be used in order to reduce weight. Total current draw should remain below 35 A at peak. Power must come from an internal source on the aircraft so that it is not limited to a certain area and is free to fly to multiple locations on command. Overall dimensions of the copter should remain within three square feet and will likely be less than half as tall as it is wide.

Item	Weight (g)
4400 mAh LiPO Battery	322
610 mAh LiPO Battery	35
Brushless Motor (x4)	20 (x4)
Battery Monitor	17
Electronic Speed Controller (x4)	6 (x4)
Propeller (x4)	22.7 (x4)
Four-Way Connector	62
Landing Base	90
Bracket (x4)	5 (x4)
Frame	319.5
Total	1060.3

Table 14: Weights

10.1.1 Material Selected

The requirements established for the team design allow only the strong lightweight materials to suffice. Within this category, the next factors considered are cost, weight, and availability. The most available materials for the craft body are aluminum and carbon fiber. However, the cost per unit of carbon fiber is considerably higher than that of aluminum, while the decrease in weight is minor. Thus, the body will be made of aluminum hollow tubes. The tubes will be round as opposed to square as they are easier to join together at the center. The tubes are 21.5 millimeters in diameter and fit snugly into a half inch PVC connector.

Circular tubes are also easier to find in the correct dimensions without having to buy excess parts. The tubes have slots already cut out that will allow for easy motor mounting. C brackets will ensure that the motors remain in the proper position and do not slide on the curved surface. The bracket will be made by cutting up a curtain rod.

10.1.2 Vehicle Shape

The group intends to build the body in the shape of an X unless it proves too difficult to control the aircraft. The motors will be mounted on the sides so that a camera will have increased visibility in the forward direction. The difficulties lie in making the copter move in the correct direction. Forward motion will require equal changes in power to both of the forward motors. If the system proves too difficult to implement, the body design will then move to the plus shape. The plus configuration has one motor on each of the four sides of the board. Movement is controlled by the adjustment of a single motor. Both designs have the motors attached at right angles to one another. The angles do not have to be ninety degrees, but can cause potential stability issues if the motors are closer to each other.

With increased access to hobby shops using the internet, the counter-rotating propellers are readily available and cost only a couple dollars more than the regular propellers. Therefore, the design will be kept simple and the vehicle will be stabilized by the propellers alternating their direction of rotation. Mounting two motors at an angle requires a more complex feedback loop. The horizontal and vertical vectors would have to be calculated based on the degree of the angle, and the angle would have to be readjusted so that the compensation for the angular momentum does not result in elevation of the unit as well. If for some reason the quad-copter experiences serious problems, such as rotating in circles, the group may decide to try the alternate approach. The propeller size will be the APC 8x3.8SFP slow flyer counter rotating propellers suggested by the manufacturers. The slow flyer designation refers to the slower than normal rotation of the blade to create more thrust at the expense of flight speed. According to reviews by users on the online store, the propellers will generate approximately 425 grams of thrust and consume 8 amps of current. The current is slightly higher than the recommended value by the manufacturer, but there have been no complaints and several users who have followed this design. Calculation based on the equations previously provided yield significantly higher values.

10.1.3 Hardware Placement

Stability of the aircraft is much more important than maneuverability for the scope of the project; therefore, the motors will be mounted farther apart at a distance of approximately twenty-five inches. After initial testing, the distance between motors may be changed to optimize performance. Because the motors will not be bolted directly to the round tube but to a bracket, the fixture will keep the unit aligned and the propellers pointed in the correct direction.

The center of the craft will be comprised of a flat platform with a one half inch four way polyvinyl chloride connector that connects the four arms together, holds the control boards on top, and holds the battery in place below. The platform will most likely have edges going upward to protect the boards from damage, although consideration will be given for the effects on stability. The boards will be bolted in place to prevent any shifting of weight or possible disconnections. The battery will be strapped onto the bottom with a strip of hook and loop fastener running perpendicularly to each other for easier access when the battery needs to be recharged.

The group will make the landing gear from a metal clothes hanger shaped into a large trapezoid. The smaller end of the trapezoid will be attached to the bottom of the center platform, and the larger side will contact the ground. The square touching the ground will measure sixteen inches on each side. The hanger is chosen because it is free and should not significantly affect the weight of the craft. The alternate landing gear proposed is to buy the standard rails from a hobby shop. The rails will cost about twenty dollars with standard shipping included.

10.1.4 Body Design Summary

In summary, the design of the quad-copter body is relatively simple. A one half inch four way polyvinyl chloride connector will be at the center of the vehicle and will connect the four twelve inch aluminum arms at right angles to each other. Figure 4 illustrates a top view of the vehicle from the above description. On top of the connector will be a flat platform to which the controller boards are bolted. The platform will form forty-five degree angles with the arms. Below, the battery will be attached with hook and loop fasteners. Also attached to the bottom of the platform will be a wide square landing base made from a coat hanger that is eight inches on each side, as seen in figure 24. The motors will be bolted onto the ends of the round tubing using c brackets for stability. The propellers will alternate between each motor rotating clockwise and counter-clockwise. The propellers are eight inches in diameter with a 3.8 inch slow flyer pitch.

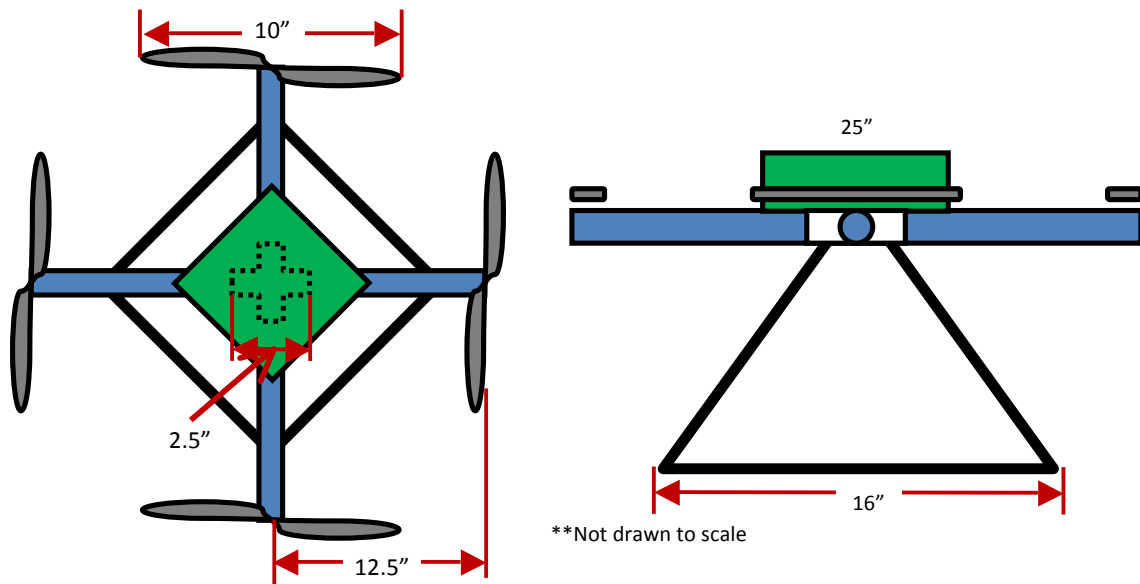


Figure 24: Prototype from Above/ Prototype from the Side

10.2 Constructing the body

The body will be built in the senior design lab of the University of Central Florida main campus. The central part of the body consists of four circular aluminum tubes fastened together by the polyvinyl chloride connector. The four tubes are created by sawing in half two Align 600 helicopter tail booms with a hacksaw. The tubes are measured with a reliable tape measure, and a pencil mark is made at the midpoint. The tubes are then clamped to prevent sudden movement and sawn through. The sawn ends are roughly sanded before they are inserted into the connector. An adhesive compound is applied to the inside of the connector before insertion. Before the adhesive solidifies, the arms of the body are adjusted so that they form right angles to each other and protrude the appropriate distance from the center. The tubes also come with slots and holes at the far ends that must be directed face up for proper motor mounting. An opening is created in the center of the connector to allow wiring to run inside the tubes. The opening begins with a drilled hole and is widened by a rotary tool. An old square curtain rod made of metal, most likely tin, is cut into one and one half inch sections using the hacksaw. One of the larger sides of the rod is sawed off to form a c bracket, and the edges are bent to make a tight fit with the aluminum tubes. The motors are permanently bolted to the brackets with three millimeter bolts, a washer to absorb vibrations, and two nuts to ensure the setup remains intact. The brackets are initially held in place by a matching bracket on the opposite side. Once the exact motor placements are established, the brackets are permanently bolted directly to the body. It is necessary to drill two new holes in each arm to get the brackets properly mounted. The figure 25 demonstrates how all the pieces fit together on one branch of the body.

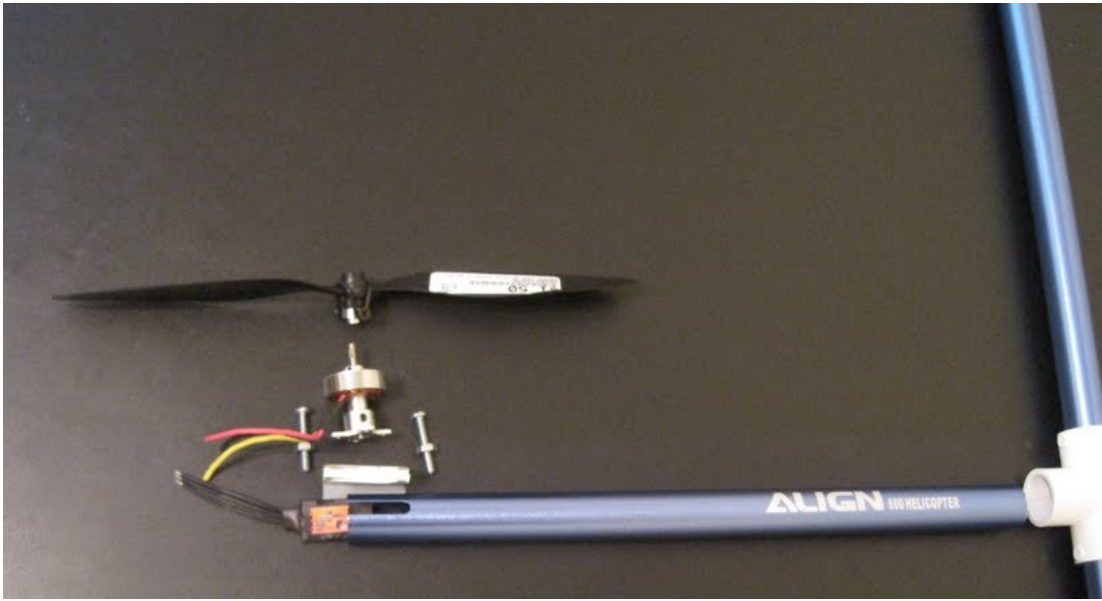


Figure 25: An example of the procedure for attaching a motor to the body of the quad-copter.

The landing gear is constructed by straightening out a coat hanger and cutting off the twisted ends. The hanger is then to be bent into a square with four lengths extending back to the connector. The leads are curved to fit the tubes and are attached with duct tape until the group is certain that the copter can land safely. Once the landing gear is finalized, the parts are glued together with epoxy. The batteries will be weighed to ensure uniform density across the entire volume. The center of gravity will then have a hook patch attached. A matching loop fastener patch will be placed at the center of gravity of the main frame. Finally, the controller boards are bolted to the box being attached to the frame. The box will be made of tin and will have an insulating layer of protection. Once again, the centers of gravity must be overlapping. Care must be taken to mount the board with the gyros in the correct alignment so that the correct motors activate when reacting to stimuli that will unbalance the system. Now that the frame is entirely built, the boards, motors, and batteries must be connected. Between each motor and the batteries is an ESC. They will also fit inside of the tubes. The apertures in the connector should allow for easy wiring, but lines should be run before installing the box to ensure easy access. The propellers are the last parts to be attached and will be added after testing. All initial construction must allow for small changes until the design is finalized and approved by all members of the group.

10.3 Motor Selection

The total weight of the quad-copter according to the design is just over two pounds. The decision was made to conserve power over the guarantee of superior thrust. Few motors have qualities than compromised between the powerful and the lightweight, and the ones that do carry significant price increases. Thus, four of the 24 gram Hextronik motors will grace the final product. It is cheaper, draws less power, and is lighter than any of the competition. The cost is slightly cheaper as well. The only point of contention is whether it will be able to lift the aircraft off the ground and if it can do so without damaging itself or causing a current spike on the battery. Figure 26 shows the small size of the motor. The coils are clearly visible inside of the exterior casing.

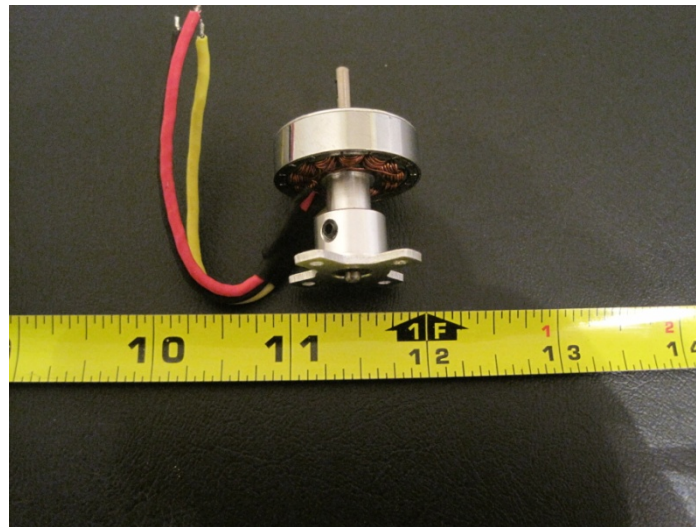


Figure 26: Hextronik 24 gram motor beside a measuring tape

10.4 Power Design

The controller boards and the motors are going to be using separate power supplies to keep noise from the digital side isolated. Noise spikes can cause erroneous readings in the electronic speed controllers and cause crashes. The 4400 mAh 15 C lithium polymer battery will be used to power the motors as it provides the suggested 11.1 volts and should have enough power to keep the aircraft airborne for up to fifteen minutes. A 610 mAh 7.4 V LiPo battery with 20 C will be powering the boards and will provide power for much longer than the battery for the motors. Both batteries will be recharged every time one of them needs to be refilled, and the larger battery will be equipped with a monitor to advise the user of a low power condition. Therefore, all components should be receiving power unless the user is told otherwise by the battery sensor. The decision to use an entirely separate power source for the board was made with

the knowledge that the cost is minimal as is the weight contribution. The small size of the battery can be seen next to the battery sensor and measuring tape in figure 27.

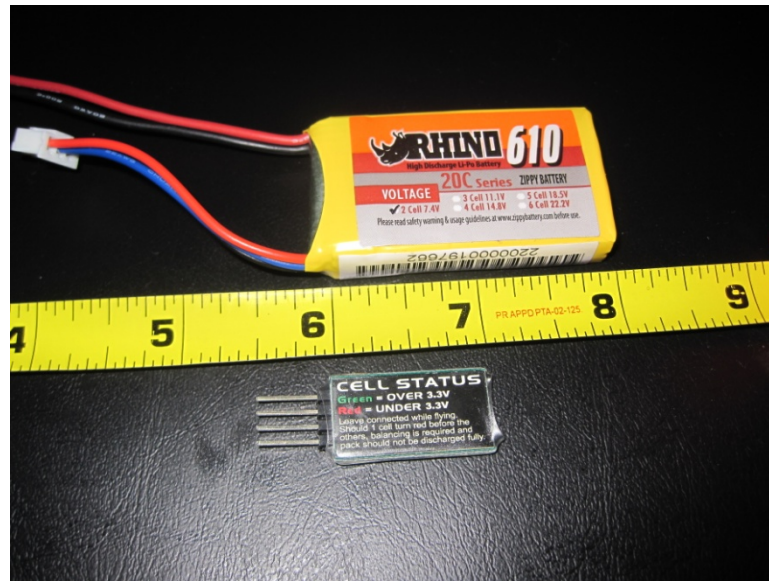


Figure 27: The small battery for the board is only about twice as wide and twice as tall as the battery sensor for the larger battery.

Because of the sensitivity of the batteries, a special charger is used to charge both packs. The charger is made for lithium polymer batteries with up to six cells, and is equipped to charge nickel metal hydride batteries as well. The LDC screen displays the levels of charge on each cell and balances cells as necessary to ensure a uniformly charged battery. The charger is a small, portable unit that can be powered by any twelve volt DC power supply. Thus, it can be used in the car through the cigarette lighter while traveling. Unfortunately, the unit did not come with a standard AC adapter, so a power supply must be created if a car battery is not available or convenient to use. Suggested makeshift power supplies include taking an old computer power supply and shorting the safety circuitry so that it powers on without being connected to a motherboard. Once operational, any of the wires used for powering drives will have live current at the desired voltage. It is very important to be careful whenever handling circuitry will large current draw, as imprudent handling can result in death.

10.4.1 Regulator Setup

All wiring will be run inside the hollow tubing that makes up the body of the craft. Wire size will be eighteen gauge to allow enough power to flow freely, but not weight more than necessary. The ESCs are also adequately sized to fit in the

tubes, but caution will be taken to ensure the units do not overheat. All wiring will be soldered and sealed to prevent shorts and wire separation. Once the ESCs are attached to the power supply, they must be calibrated so they can recognize the limits of the current passing through the system. The units are calibrated by sending the peak value of current until a beep is heard. The ESC being used for the Hextronik 24 gram motor has a rating of ten amps.

The power supply to the control board will be regulated using Texas Instrument's TLV1117-33. It is a simple voltage regulator that will take in 7.4 volts and give out 3.3 volts to within three millivolts of accuracy. The board requires 3.3 volts to operate and has its own regulators for any parts that need a lower voltage. The regulator is free as it is a sample from Texas Instruments. It is also much smaller and much more accurate than the regulator designed by the group. Furthermore, the regulator is significantly more reliable than a BEC both in lack of noise and thermal stability.

10.4.2 Power budget

The expected power consumption of the parts in Table 14 will be compared to tested values to make sure the power supply meets the demands of the circuit.

Part	Current from 4400 mAh 11.1 V LiPo	Current from 610 mAh 7.4 V LiPo
Hextronik motor	8 A (x 4)	0
ESC	2.4 mA (x 4)	0
Battery monitor	3 mA	0
Cortex M3 processor	0	36 mA
Wi-Fi Transceiver	0	180 mA
Ultrasonic sensor	0	2.1 mA
IDG500	0	7 mA
LY530AL	0	5.5 mA
ADXL335	0	350 uA
HMC5843	0	9 mA
Camera sensor	0	20 mA
1.8 Voltage regulator	0	150 mA
2.5 Voltage regulator	0	150 mA
3.3 Voltage regulator	0	150 mA
Total	32.0126 A	709.95 mA

Table 14: Power consumption estimates for each component.

10.5 STM32 Design

The connections between the STM32 and the various other components in the project were designed with the help of the documentation available for the STM32 on the STMicroelectronics website as well as the schematics in the development board manual from Olimex. The documentation of the other modules was used in designing the interfaces between the STM32 and the other modules on the quad-copter. The stm32 we are using is a 36 pin QFN package as shown on Figure 28.

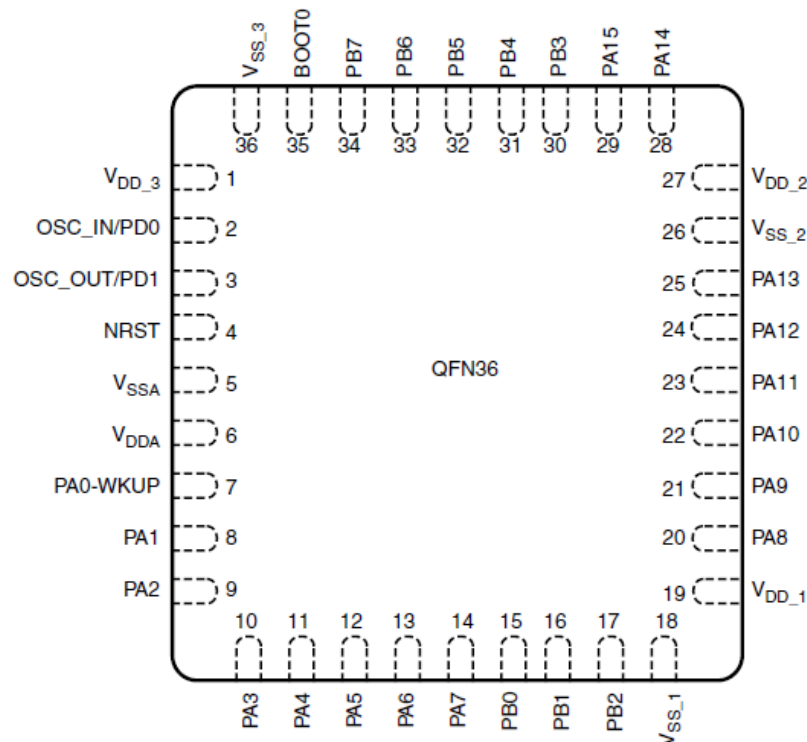


Figure 28: Schematic for power supply and external oscillator.

10.5.1 STM32 Power

To provide power to the STM32 a 3.3 V line from a regulator on the control board will be used. The STM32 has three VDD pins that will be tied together and connected across decoupling capacitors. The ground line on the control board is connected to the three VSS pins that are also tied together and connected to the VDD pins across the decoupling capacitors. The ADCs have their own power pins called VDDA and VSSA, these pins will be connected to the same 3.3 V and ground lines as VDD and VSS and will require some more decoupling capacitors. The final power pin that needs to be connected is the pin for battery backup

called VBAT. The project does not need the battery backup features of the STM32 but the pin cannot be left floating so it will be connected to the 3.3 V line across another decoupling capacitor. The documentation for the STM32 advises that there should be 1 decoupling capacitor for every pair of VDD and VSS pins plus an extra higher capacitance capacitor. The suggested values for the capacitors are 100 nF for the small capacitors and 10 μ F for the larger capacitor. The ADC power supply pins and the battery backup pin also need decoupling capacitors of values 100 nF and 1 μ F for VDDA and VSSA and another 100 nF for the VBAT pin. To reduce the effect of power supply noise on the STM32 ADCs so that more precise values can be obtained during conversion, a ferrite bead will be placed on the line connecting VDDA and VDD at the suggestion of the STM32 documentation. Refer to Figure 88 below the STM32 external clock section to see a schematic that shows how the power supply pins on the STM32 are connected.

10.5.2 STM32 External Clock

Even though the STM32 has a built in high frequency oscillator an external 8 MHz oscillator will be used for the main clock source. One reason for not using the internal oscillator is that the STM32 core can only clock to 64 MHz with the internal oscillator, while with an external oscillator the full 72 MHz clock can be used. The other reason for using an external oscillator is that it will be more accurate and stable than the built in one according to the documentation from STMicroelectronics. The project needs the system clock to be consistent so that the period of the PWM signals are always the same length of time. The external crystal oscillator will be connected across pins 6 and 7 on the STM32. The plan is to use one of the recommended crystal or ceramic resonators given by the STMicroelectronics oscillator design guide. The design guide also gives equations to select the proper size for the load capacitors needed for proper operation.

Figure 29 is a schematic showing the implementation of the power and oscillator design discussed in the above two sections. It shows the number of the STM32 pins being used for the various connections as well as all the connections between the pins and the external components.

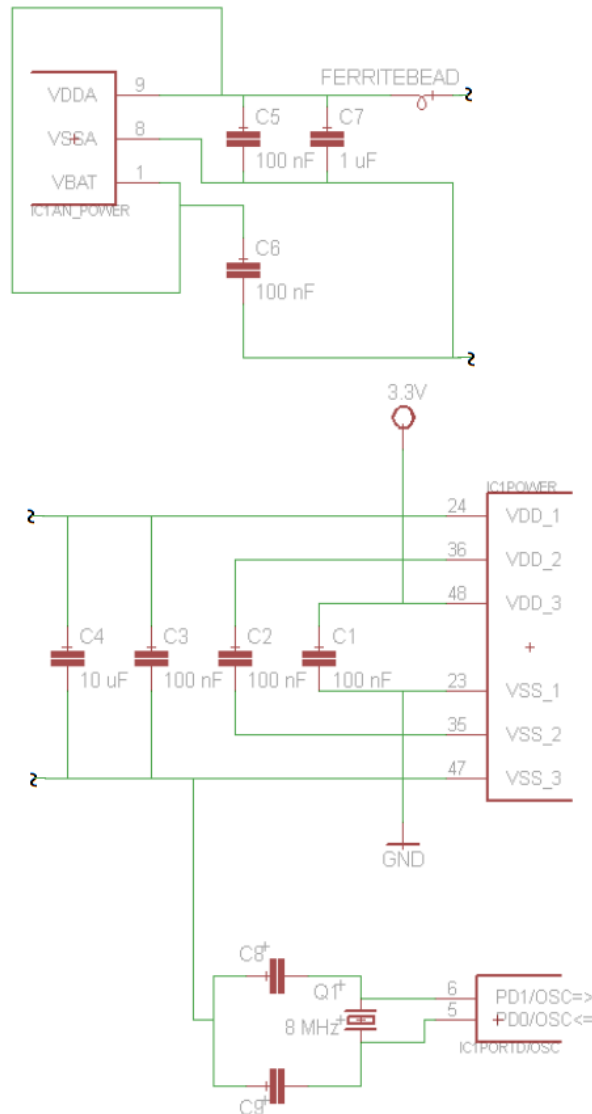


Figure 29: Schematic for power supply and external oscillator.
The STM32 block shows the pin name followed by the pin number

10.5.3 STM32 IMU Analog Interface

To interface the analog components of the IMU, six pins on the STM32 will be configured as analog channels. The IMU is composed of multiple analog devices which are a two axis gyroscope chip, a one axis gyroscope chip, and a three axis accelerometer chip. Six of the STM32 pins will be used to interface with the gyroscope and accelerometer one for each degree of freedom available on the devices. The 6 pins that will be used are channels on the built in ADC of the STM32. The pin numbers of the ADC channels are 10-13, and 18, 19. The STM32 variant that is being used has two ADCs available to convert the IMU values. The STM32 allows the two ADCs to convert data synchronously and continuously. So the plan is to convert half the channels on one ADC and the

other half in the other ADC. By using both ADCs the conversion time for all the IMU data will be cut in half allowing the IMU to be sampled more often. The DMA channel 1 will be used to transfer the converted values from the ADC data register to SRAM. The raw data will then be converted to usable values by firmware based on information given by the IMU spec sheets. The values from the IMU will then be passed through a software filter that smoothes out the data to factor out readings caused by the vibration of the motors and not the changing orientation of the quad-copter. The filtered data is then used by the software control loops that keep the quad-copter flying stable.

10.5.4 STM32 PWM Interface

The four motors on the project will be controlled using PWM signals generated by the advanced timer on the STM32. The STM32 does not directly send the PWM signals to the motors. The PWM signals are actually sent to the ESC modules which use the PWM information to control the speed of the motors. The advanced timer will output a PWM signal for each of its four available channels. These channels are located on pins 29-32 of the STM32. Each channel will generate the PWM signal for one of the four ESCs. After reading information at the DIY Drones website it was learned that having a PWM signal with a period of 1-3 ms is what is recommended for controlling the motor's speed. Changing the duty cycle of the PWM signal is what allows the motor speed to be adjusted. The duty cycle and period of the PWM signals will be adjusted by the control firmware to achieve stabilization and to also allow navigation of the quad-copter. The duty cycle of the PWM signals will be adjusted every period which means that all the on board processing must be completed within 1-3 ms so that the PWM can be updated every period.

Figure 30 shows a portion of the schematic for the control board of the project. The figure shows how the two of the IMU components are interfaced with the STM32. The STM32 ports are shown on the left side of the schematic while the three analog IMU components the accelerometer and the gyroscopes are shown on the upper right and lower right side of the schematic respectively. The capacitors C10-C12 are the capacitors used to select the band limit for the accelerometer outputs. The values of these capacitors will be determined by experimentation. The values of the other capacitors are the recommendations given by the respective devices spec sheets.

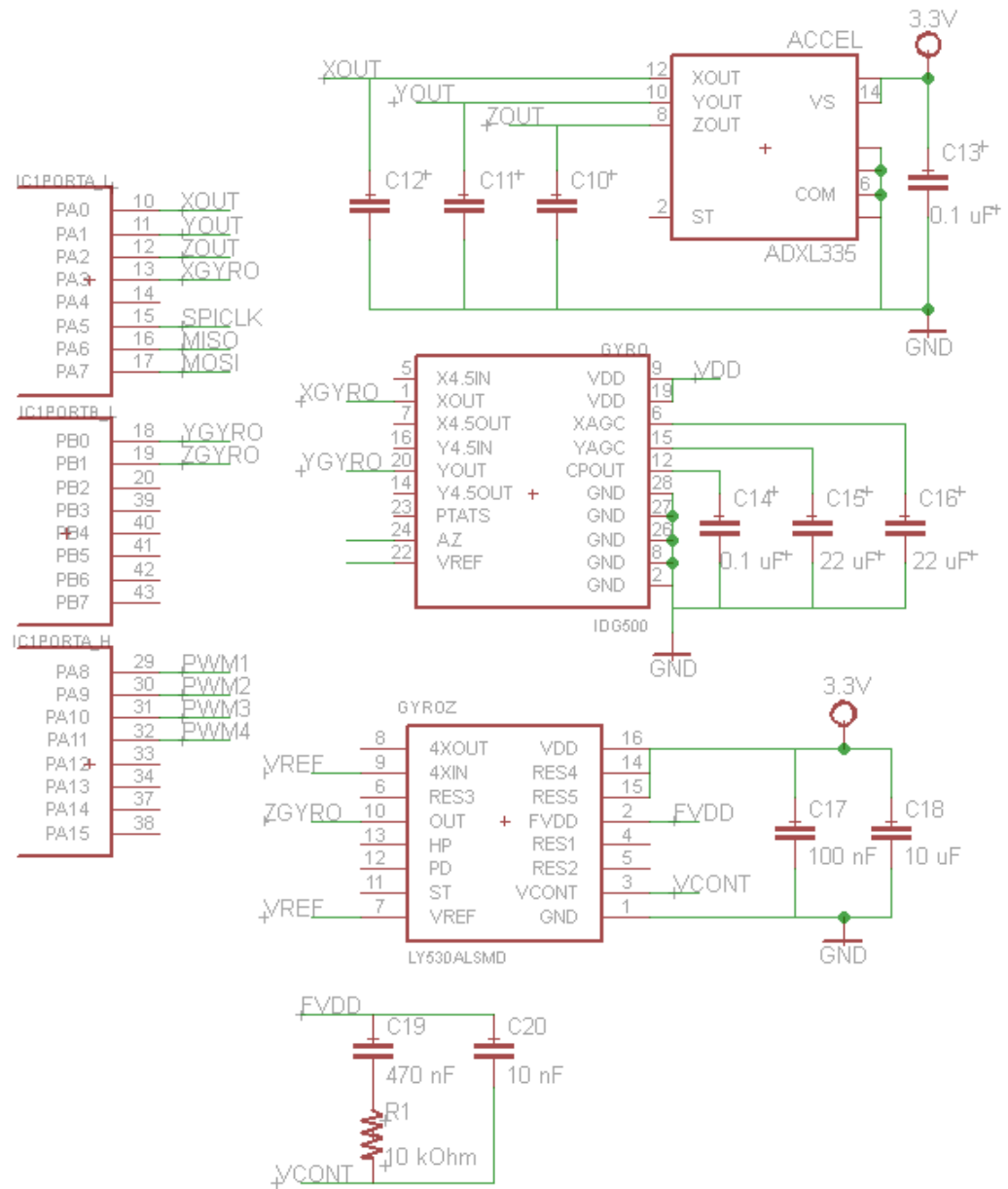


Figure 30: Schematic showing the various IMU components and how they interface with the STM32

10.5.5 STM32 SPI Interface

The microcontroller will communicate with the Wi-Fi module by using one of the communication peripherals on the STM32. The STM32 has three different types of communications ports available; USART, I2C and SPI. Given all these choices in communications ports on the STM32 the group decided to use the SPI peripheral because it provides the most bandwidth of any of the other ports at 18

Mbits/s, compared to the 4.5 Mbits/s available for the USART and the 400 kbits/s available for the I2C. The pins on the STM32 that are used for the SPI are number 15-17. These pins are connected to the SPI pins on the Wi-Fi module as follows. Pin 15 on the STM32 is connected to pin 15 on the Wi-Fi module, pin 16 on the STM32 is connected to pin 16 on the Wi-Fi module, and pin 17 on the STM32 is connected to pin 14 on the Wi-Fi module. In order to save processing cycles the DMA will be used to transfer the Rx and Tx data from the SPI data registers directly to the STM32 SRAM. DMA channel 2 is used to store data that is received, while DMA channel 3 is used to fetch data that is to be transmitted. Figure 30 shows a schematic for the connection between the SPI pins on the STM32 and the Wi-Fi module.

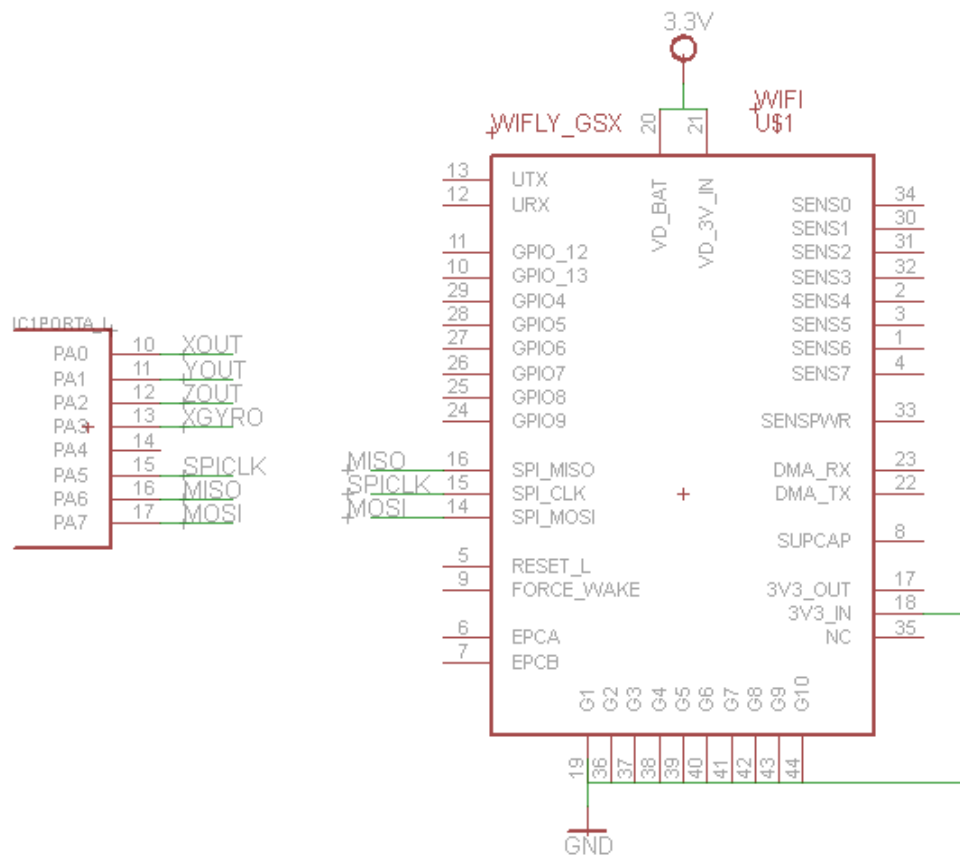


Figure 31: Schematic showing the SPI interface between the Wi-Fi module and the STM32

The camera module will also be connected to one of the communication ports on the STM32 so that its video information can be buffered before being sent to the iPhone. Video requires a great deal of bandwidth in order to have smooth playback. Taking this into account he preferred port that the group would want to use is the SPI port that the Wi-Fi module is connected to because it is the communication port that has the most bandwidth. If this configuration is used the Wi-Fi and camera modules will have to share and take turns using the SPI interface, so some kind of scheduling subroutine will have to be written to allow

the SPI to be shared by the camera and Wi-Fi in such a way that navigation and video data can be communicated between the STM32 and the iPhone efficiently.

10.5.6 STM32 I2C IMU Interface

The I2C1 communication port on the STM32 is going to be used to interface with the last IMU component. This last component is the magnetometer that is used to set the reference axes for quad-copter navigation. The I2C protocol is being used because this the only communication protocol that the magnetometer uses to transmit its values. The I2C communication port only uses two pins on the STM32 for communication. These pins are 42 and 43 on the STM32. Pin 42 is the data clock and pin 43 is the serial data transfer. Pin 42 is connected to pin 1 on the magnetometer while pin 43 is connected to pin 20. The standard I2C data transfer rate of 100kbts/s will be used because the magnetometer has built in pull up resistors that can be used with this transfer rate which saves having to buy and place more parts to go onto the control board. Just like with the ADCs and SPI interface the I2C1 port will use the DMA to transfer the magnetometer sensor readings data directly into memory without bothering the STM32 processor. Figure 32 is a schematic displaying the design of the magnetometer being interfaced with the STM32.

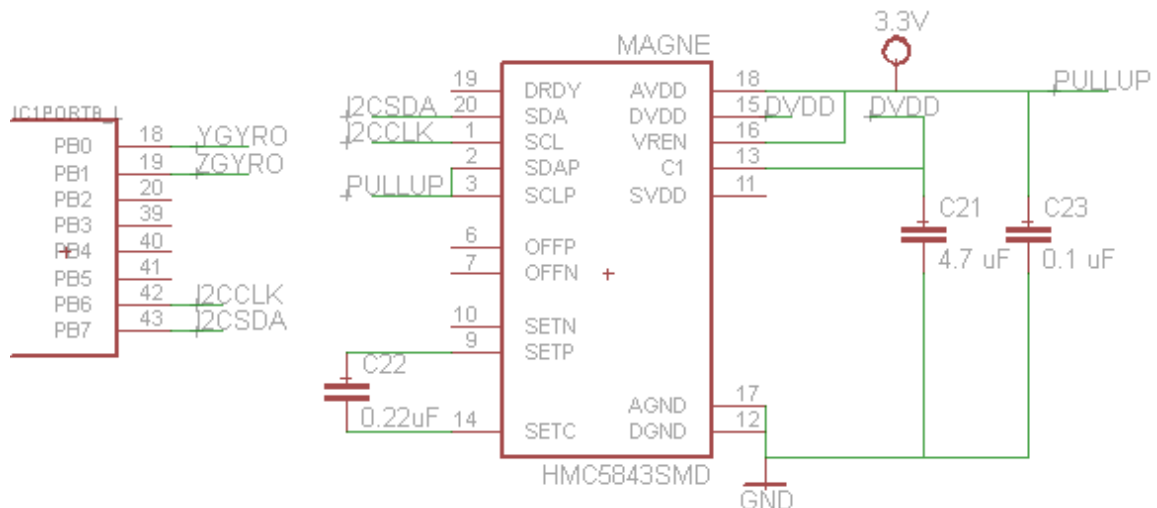


Figure 32: Schematic showing the I2C interface between the magnetometer and the STM32

10.5.7 STM32 Software Development Tools

The firmware for the STM32 will be coded in the C programming language because it is the programming language the group members are most familiar with, it is the programming language used by most ARM Cortex-M3 development tools, and it takes less time to make a program than using assembler. The

Eclipse IDE will be used to write the source code for the STM32 firmware . The source code written on Eclipse will then be compiled by the GNU C compiler. Then OpenOCD will be used to allow communication between the development board and the computer through the JTAG programmer and is what writes the firmware onto the STM32 flash memory and allows the communication required for the firmware to be debugged while running on the development board. STMicroelectronics has created a C library that is freely available that makes configuration of the STM32 simpler. This library is called the STM32 standard peripheral library and is intended to cut development time. The standard peripheral library cuts the development time of the firmware because the library provides all the functions, structures, and prototypes necessary to initialize and command any peripheral on the STM32, so that the functionality of this library does not have to be developed by the group. Since the group is composed of novices when it comes to programming for microcontrollers, other people's source code will be examined to help with the writing of the subroutines for several components of the quad-copter firmware.

10.5.8 Firmware Configuration Design

The development of the firmware for the quad-copter will be comprised of two stages. The first stage is the development of the code that will configure the STM32 and the peripherals that will be used for the project. The configuration for the peripherals will be comprised of setting up the peripherals to collect and transmit data to and from the various components on the quad-copter. The second stage is writing the sections of code that will use the data collected by the peripherals to generate signals to stabilize and allow control of the quad-copter. The firmware for the quad-copter will be designed one subroutine at a time starting with the subroutines that initialize the STM32 and continuing down until the subroutines that sends out the PWM signals to the motors and transmit status information to the iPhone. Once one subroutine is finished and operates correctly, the subroutines that depend on its results will be the next ones to be written. This allows the subroutines to be tested one at a time before being integrated into the final firmware.

10.5.8.1 STM32 Standard Peripheral Library

The STM32 standard peripheral library will be used to aid in the writing of the subroutines that initialize and control the STM32 and its peripherals. The library is composed of many files that contain functions, structures, and prototypes that initialize, configure, and control the various peripherals and features of the STM32. The library documentation provides the instructions of how to use the various library files to setup the various peripherals and features on the STM32. The only problem with this library is that the documentation is not as thorough as

it can be causing issues like confusing definitions of functions and variables and trouble finding out what arguments need to be passed into the various functions. If the library is not used as an aid in programming, the programmer would then have to setup the STM32 by knowing the memory address of the various peripheral configuration and data registers and set the corresponding option bits in them. This method is very tedious and can take months to complete because the programmer has to keep searching through the STM32 reference manual for things like the memory addresses of the various registers and which bits in the registers configure the feature that is needed. Then which bits in the registers control which functions have to be looked up and finally, what the bits have to be set to also has to be looked up. The first step in setting using the library is to create a configuration structure. This structure is composed of the various options that can be modified for the given STM32 peripheral or feature. This configuration structure is then passed into a function that initializes the STM32 feature or peripheral. Once the initialization is done other functions related to the peripheral that was initialized can be used to accomplish things like data transfers and reads. Figure 33 shows a simple piece of example code using the peripheral library to configure port A pin 0 as an input. The first three lines is the configuration structure that is used to select the settings for the pin. The fourth line is the function that takes in the configuration structure and what port the pin is on and initializes the pin. The last line is a function used to read the logic level on the pin and stores it for use in later parts of the code.

```
// Configure Port A pin 0 as input

// GPIO configuration structure
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0;
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;

GPIO_Init(GPIOA, &GPIO_InitStructure); // Initialize pin with struct

LEDON = GPIO_ReadInputDataBit(GPIOA,GPIO_Pin_0); //read configured pin
```

Figure 33: Sample code showing the use of the STM32 Standard Peripheral Library to configure and access I/O pins

10.5.9 STM32 Clock Configuration

The first subroutines that have to be written for the firmware are the ones that initialize the STM32 clocks. The initialization is handled by the Reset and Clock Control(RCC) registers. The first thing that must be done is to modify the RCC registers to set the system clock as well as enable and configure the clocks of all the peripherals that are used like the ADCs, and the advanced timer TIM1. The system clock is the highest clock signal and all the other clocks on the STM32

are based on divisions of this clock rate. The clocks are set by first selecting the clock source for the STM32. The clock source to be selected is the PLL on the STM32. This source is selected by first telling the library function called `RCC_HSEConfig` to turn on the external 8 MHz crystal. The crystal signal is then passed through the PLL so that it can be multiplied to the full speed of 72 MHz. The PLL is configured by passing the crystal clock source and a multiplication factor of 9 to the `RCC_PLLConfig` function. After the PLL is configured it has to be enabled by telling the `RCC_PLLCmd` function to enable the PLL. The other clocks on the STM32 are configured just as the PLL was but using functions with names that are slightly different to the PLL configuration function. Once all the clocks are configured the system clock can be enabled by telling the `RCC_SYSCLKConfig` function to use the PLL as the main clock source for the STM32. Now the clocks for the peripherals that are being used by the project can be enabled. Enabling the clock of a peripheral is also how a peripheral is turned on for use. By only enabling the clocks of the peripherals needed for the project and leaving the other peripherals disabled the current consumption of the STM32 can be reduced because if a peripheral does not have its clock enabled it consumes a negligible amount of current thus reducing the power expenditure of the STM32.

10.5.9.1 STM32 I/O Pin Configuration

The next subroutine will configure the STM32 I/O pins. The configuration for the I/O pins is handled by the General-Purpose IO(GPIO) registers on the STM32. The subroutine will use the standard peripheral library to set the mode of each pin that is to be used. The pins are grouped into ports that have 16 pins each. The STM32 variant chosen for this project only has two ports. These ports are called port A and port B, pins from both ports are being used in the project. The first step in configuring a pin is to create and fill out the GPIO initialization structure. The information that is placed in this structure is the pin number, pin output speed, and pin mode. The pin number is to select which pins out of the 16 in the port is to be configured. The pin output speed is only used to select the maximum rate at which data is sent out a pin this option is not needed for pins that are set as inputs. All the pins that are outputs will be set to to operate up to the maximum output speed of 50 MHz to avoid having to keep changing the initialization structure when configuring every output pin. The pin mode is to set the function of the pin. There are 8 different pin function modes but not all of them are used for this project. The pin functions that will be used are analog mode, input floating mode, and alternate function push-pull mode. The 6 pins that will be used for the accelerometer and gyroscope IMU components will be set into analog mode to convert them to analog value reading channels. The TIM1 pins that output PWM signals will be configured to the alternate function push-pull mode. The SPI port pins will be configured depending on their function. The data receiving pin will be configured as an input floating mode pin, while the clock and data transmitting pin will be configured the same as the TIM1 pins. The two I2C interface pins will be configured to both be alternate function open drain.

Once the initialization structure is complete it is passed on to the GPIO initialization function that configures the pins to the specifications defined in the initialization structure. Once the pins are configured they are ready to be used by the peripherals.

10.5.9.2 STM32 DMA Configuration

The DMA implementation is the next code section to be written. The DMA will be used to transfer the data between peripheral registers and memory. The DMA is configured by first creating an initialization structure that is used to modify the DMA settings before the structure is passed on to the DMA initialization function. The DMA structure will be given the direction the data of the memory transfer is to be transferred, either peripheral to memory or memory to peripheral. The next items that the DMA has to be configured with are the address of the memory location where the peripheral data is located and the starting memory address in SRAM where the data will be transferred to. Next the size in bits of the data to be transferred is specified to the DMA channel for both the starting and ending destination. Then the DMA structure will be told how many items of data are to be transferred before the DMA transfer is over. If the amount of data transfers is more than one then the data will be stored in an array that will have as many elements as the numbers of data transfers that will be performed. The DMA automatically increments the memory location it is storing data into as it decreases the count of how many items it has left to transfer, this is useful because loops that increment the array indexes after every DMA transfer will not have to be coded. The DMA structure will be configured so that when the DMA reaches the end of the item transfer count it generates an interrupt telling the firmware that the data the DMA placed in memory are ready to be accessed. Once the data the DMA transferred is used by the firmware the firmware will reset the interrupt which in turn causes the DMA channel to reset the item count and set the starting memory address where the data is being transferred into back to their initial values and start the transfer over again, overwriting the values from the previous transfer. Once all the settings for the DMA channel have been filled in the channel is initialized using the DMA initialization function and is finally enabled by using the DMA command function.

Since multiple DMA channels are being used each one has to be configured with a priority value to determine which channel will have its request completed first if multiple DMA channels request a transfer. Each DMA channel that will be used will be configured separately and only be used for one task so that the channel's configurations do not have to be readjusted. Only five of the seven DMA channels are going to be used for this project. Channel 1 will be configured to transfer the converted ADC values to an array in the STM32 memory. Channel 2 will be configured to transfer the data received from SPI1 into memory. Channel 3 will be configured to transfer the data in memory to SPI1 so that it can be transferred out to the Wi-Fi module. Channel 6 will be configured to transfer the

magnetometer register addresses stored in the STM32 memory to I2C1 so that the magnetometer registers can be read. Finally channel 7 will be configured to store the magnetometer readings coming in on I2C1. Channel1 will be given the highest priority because the ADCs are constantly converting and changing which ADC channel the value is from so if the DMA is delayed it will mess up the array for the ADC values because the entries will be out of order. Channels 6 and 7 will be given the second highest priority because data from the magnetometer is very crucial for proper quad-copter stabilization. Channel 2 will have the third highest priority because data from the Wi-Fi module contains instructions for navigating the quad-copter, while the data being transferred by DMA channel 3 will have the lowest priority because it only contains data on the status of the quad-copter for the users reference which is not as important as the other data that is to be transferred.

10.5.9.3 STM32 ADC Configuration

The code to control the ADCs is the next code section that will be written. The first thing is to fill in all the configuration data into ADC initialization structure. The STM32 has two ADCs that are capable of working together to speed up the conversion of a group analog channels. So the initialization structure will be told to configure the ADCs to work in dual mode so that both ADCs are used to convert data. The initialization structure will then be given the number of channels each ADC will scan and convert which is three. The ADC1 will convert the three-axis gyroscope and ADC2 will convert the three axis accelerometer that are on the IMU. Next the structure is filled with the setting that tells the ADCs to convert continuously, this means that after the three channels on each ADC are converted the conversion starts over again from the first channel in the group. Since the ADC is 12-bit but the data size it is stored to is 16-bit the ADC structure has to be given how to align the data into the sixteen bits. The data will be aligned to the right of the 16-bit storage space and the remaining 4-bits are used for sign extension. The initialization structure is passed into the ADC initialization function. After initialization the ADC channel configuration is used to set what three channels the ADCs are converting, what order they are to be converted in, and the sampling time for the channels. Then the ADC DMA enabling function activates DMA transfer for the conversion values. The ADCs are then enabled by using the ADC command function. Once the ADCs are enabled they have to be calibrated by using the ADC calibration function to get the most precise and accurate data out of the ADCs. When the ADCs have finished their calibration conversions of the channels will begin.

After every channel conversion the conversion is stored in a 32-bit register. The upper 16-bits are the ADC2 conversion while the lower 16-bit are the ADC1 conversion. The DMA channel 1 will transfer the data in the register after every conversion into an array in memory. The ADCs are also configured so that when they finish converting the six channels they generate an interrupt. The interrupt will be used to signal that the IMU data is ready to be accessed. Once the data is

used and the interrupt is cleared the ADCs begin converting the channels again from the beginning automatically. The individual array elements will be named after the source of the data. For example, if the first two converted values are the X axis acceleration rate and the X axis rotation rate, then the first two array elements would be named Xacc and Xrot.

10.5.9.4 STM32 SPI Configuration

The next peripheral to be configured will be the SPI1 port. Like all the other peripherals, the SPI1 will be configured by setting the proper settings into its control registers by using the functions of the STM32 Standard peripheral library. First the SPI initialization structure will be set to operate at full synchronous duplex. This allows the SPI to receive and transmit data at the same time. The next configuration item to be set in the structure is to set the STM32 as the SPI master that controls all aspects involved during SPI transmission. SPI1 will be configured to operate at its maximum possible data transfer speed of 18 Mbits/s by setting the Baud rate to its maximum setting in the configuration. The transmission frame size, which is the how many bits are transferred every clock period, will be set to 16-bits in the initialization structure, in order to send the most amount of data per transmission. Now that the initialization structure is told how the data bits are to be sent, this will be set so that the MSB of the data transfer is sent or received first. The bits are sent in this order so that they are stored in the proper order in memory once they arrive at their destination. Now that the SPI is configured it is initiated by passing the initialization structure to the SPI initialization function so that the SPI can begin transmitting data. The SPI will be shared by the Wi-Fi and camera modules so that two slave select lines will be needed to switch between the two modules.

The data that is to be transmitted through the SPI will first be loaded into an array that will serve as a data buffer. The elements in the array will be sixteen bits in size to match the size of the SPI transmission frame. The number of elements in the array will be based on how much data is required to fit all the information that will be sent through the iPhone through the Wi-Fi. This array will be passed through a function that transfers each element in the buffer array using both the DMA and the SPI transfer function of the standard peripheral library. After the array buffer has been transmitted it will be refilled with the next set of data to be transferred. While the Wi-Fi data is being transferred the camera data will be loaded into a buffer to be transferred after the SPI is finished transmitting data to the Wi-Fi and while the camera data is being transmitted the Wi-Fi buffer will be refilled with data.

10.5.9.5 STM32 TIM Configuration

The next peripheral on the STM32 that will be configured is the advanced timer TIM1. Unlike the other peripherals that have been configured so far the TIM1

requires that two initialization structures be configured in order to enable the timer and configure it to operate in PWM mode. The first structure that is filled out is the one that sets up the timer base, which contain the main timer settings. The first thing that this structure configures is the maximum integer value that the timer counts to. The second value placed into this structure is the value of the frequency that is used to update the timer's value to the next integer in the count, the higher this frequency the shorter the amount of time between counts. This frequency value along with the integer value that the timer is set to count to is what determines the period of the PWM signal. Once the count is finished it restarts from the beginning to start the next period. The last setting that is placed into the main timer settings structure is the direction the timer should count in, either upward to the maximum value or downward from the maximum value to zero. The PWM mode requires that the timer count upward to its final value so this the setting that will be written into the structure. When this first configuration structure is finished it is passed through the timer base initialization function to setup the main timer settings.

The second configuration structure that is used configures the four output channels of the timer. The first setting placed in this structure is the operation mode for the timer channels. The operation mode will be set to PWM output mode so that the channels generate PWM outputs. The next setting that is placed into the structure is the one that sets whether or not the four timer channels are allowed to output signals. This setting will be configured so that the output from the four channels are enabled, so that the PWM signals can be sent out to the ESCs. Now the pulse setting is the next setting to be defined in the channel configuration structure. This setting is an integer value that is stored in a comparison register. After every count the timer makes the value in the comparison register is compared to the count value the timer is currently holding and is used to set the duty cycle of the PWM signal. If value in the compare register is greater than the current value of the timer count the output of the PWM channel will be set to a logic high, once the count is higher than the value in the compare register the PWM channel will be set to a logic low. The next value that is set in the configuration structure is the one that chooses what logic level will be outputted on the PWM channels when the pulse setting value is greater than the value of the timers count. Since the pulse value is going to be the duty cycle this setting will be set to output a logic high when the pulse value is less than the counter value. If the value in the compare register is greater than the current value of the timer count the output of the PWM channel will be set to a logic high, once the count is higher than the value in the compare register the PWM channel will be set to a logic low. By adjusting the comparison register with different values every period the duty cycle of the PWM can be altered. The timer is then enabled using the timer initialization function from the peripheral library. The timer pre-configuration function from the peripheral library will be used in order for the duty cycle value to be changed to a new state at the end of every PWM period. The duty cycles are generated by the navigation function

10.5.9.6 STM32 I2C Configuration

The final peripheral on the STM32 that will be configured is the I2C communication port. The first step in configuring the I2C for data transmission is setting the configuration values into the peripheral library I2C initialization structure. The first value that will be configured is in which mode the I2C will operate. This will be set to I2C mode as the other modes have to do with the SMBUS protocol for computers. The second value that is configured is the duty cycle of the transfer clock. This value will be left at its default to reduce the complexity in controlling the I2C transfers. The next setting to configure is to select an address for the I2C interface if it is a slave. Since the magnetometer is configured to be the slave device that means that the I2C on the STM32 is the master so it does not need an address to be set to it. The next configurable item is if the acknowledgment of a transfer is enabled or disabled. The magnetometer always sends an acknowledgment after every transfer so this setting must be enabled. The next configuration setting is the addressing mode of the I2C. This will be set to 7-bit addressing as that is what the magnetometer uses for the length of its address. The final configuration setting for the I2C is the clock rate for data transfer. This setting will be set to 100 kHz to give a data bandwidth of 100 Kbits/s. The I2C is then initialized by passing the configuration structure into the peripheral library I2C initialization function so that it can be then be used to get readings from the magnetometer.

The first step in receiving the magnetometer readings is to load the address of the magnetometer from the STM32 memory into the I2C shift register using the DMA. When the magnetometer receives the address it starts clocking out the value of its readings one after the other. The magnetometer sends the readings for its three axes as six 8 bit I2C transfers, two 8-bit transfers per axis. The DMA stores the readings as they are coming into the STM32 into an array that has three 16-bit elements. The first element is for the x-axis, the second for the y-axis and the last element is for the z-axis measurement. Once the magnetometer has finished clocking out its readings the I2C on the STM32 will generate an interrupt so that the values in the array can be read in by the DCM function. Once the values have been used the interrupt will be cleared and the next set of magnetometer readings will be clocked in to replace the old readings in the array.

10.5.10 System Control Firmware

This is the firmware code that actually implements all the features of the quad-copter by using the data collected from the modules mounted onto the quad-copter. This part of the firmware does not begin to execute until all the settings and peripherals being used on the STM32 are configured and initialized. To write this code examples from other quad-copter projects on the internet will be referenced to help speed up the development time of the firmware.

10.5.10.1 Start Up and Calibration

The first thing that has to be written is the start up and calibration procedure for the control board to prepare it for operation after it is turned on. First a function will be created that takes in data from the IMU to set the reference coordinate axes for navigation. The function will use the IMU accelerometer data to locate the direction of gravity and set it as the reference z axis. The x and y reference axes will be set by the function by setting one arm of the quad-copter as x-axis and the arm adjacent to its right as y-axis. The magnetometer on the IMU will measure the direction of north at start up and this initial measurement will be used as a reference point of the initial position of the quad-copter x and y axes. The next function that will be created will start communication with the Wi-Fi module. This function will wait for the Wi-Fi module to send a signal when it has established a connection to the iPhone and is ready to transfer data. The last control board start up function to be written is the one that will check to see if the camera is ready to send video. This function just like the Wi-Fi checking function will wait for the camera to send a signal when it is ready to send video information.

Then the subroutine to start up the motors on the quad-copter will be written. The motors will be started one at a time after calibration is over. The motors will be started one at a time because starting all the motors at the same time could potentially cause the power system to be overloaded because the motors consume the most power when starting up. Then the motors will be sent initial PWM signals with duty cycles that make the motors produce just enough thrust that the quad-copter will hover just barely off the ground. The values of these duty cycles will be found through experimentation. Once the quad-copter is in this state it will be ready to be controlled by the iPhone.

10.5.10.2 Navigation and Stabilization

Another algorithm that might need to be written is a filtering algorithm for the IMU data. The IMU data might need to be filtered because the components of the IMU are very sensitive and pick up on the slightest vibrations of the quad-copter. These readings caused by the vibrations might need to be eliminated so that only the information pertaining to the actual change in quad-copter orientation is left. A Kalman filter will be the algorithm implemented to filter the data. The Kalman filter will take the IMU data and smooth out the spikes caused by vibrations leaving behind a good approximation of how the quad-copter orientation is changing. This algorithm will only be implemented in the project if the data coming from the IMU contains too many inconsistent readings that cause difficulty in achieving stabilization of the quad-copter.

The IMU data will then be used in the next section of the code. This will be the flight dynamics code. The flight dynamics code will implement the direction

cosine matrix (DCM) to keep track of the orientation axes of the quad-copter as well as the rate of change of the orientation and the velocity the quad-copter is moving at. The velocity of the quad-copter will be used to estimate the position and height of the quad-copter compared to where it was calibrated at. This positional information will then be sent to the iPhone control software to give the user data on how far away and how high up the quad-copter is. The DCM will be a function that takes in the rotation data from the IMU gyroscopes and takes in the orientation data from the accelerometer and magnetometer and outputs information on the position and orientation of the quad-copter that can be used to both inform the user of both position and orientation and sent to the navigation control loops that control the stability and orientation of the quad-copter in space.

The next code section will be the control loop algorithms. The control loop is responsible for keeping the quad-copter level and steady during flight. The control loop function takes in as its input the rate and position data that is the output from the DCM function and compares them to the reference axes. If they do not match the control loop will produce a correction signal in the form of a change in the duty cycles of the PWM signals so that the motor speeds change allowing the axes to be adjusted until they match the reference axes set during calibration.

The quad-copter navigation subroutine will be a function that takes in for its input the PWM duty cycles that are output from the control loop functions that keeps the quad-copter hovering stably. It also takes in the user control data from the iPhone that is used to control how the quad-copter flies. Modifies the PWM signals coming out of the control loop algorithm. It will add and subtract from the duty cycle of the PWM for each motor causing the quad-copter to tilt in a certain direction. The direction it tilts towards will be the direction the quad-copter will move in. For instance to move left the two motors on the left side will have their duty cycles decreased while the two motors on the right side will have their duty cycle increased this will tilt the quad-copter causing it to move left. The amount to change the duty cycles to induce a certain speed will be found through trial and error. It must be made sure that there is a limit to the amount the quad-copter can be tilted by the user or else control of the quad-copter will be lost and it will crash to the ground. Another part of the navigation subroutine is to control the height of the quad-copter. The height will be controlled similarly to how the translation is controlled above except that all the motors will have their duty cycle changed by the same amount. If the user wants to gain altitude the duty cycle for all motors is increased, similarly if the user wants to lose altitude the duty cycles will be decreased. The duty cycles will only be changed just enough so that the quad-copter only gently gains or loses altitude and does not shoot up into the sky or plummet straight down to the ground.

10.5.11 Control Board Design

10.5.11.1 PCB Component Layout

For the design of the control board the idea was to place the STM32 and IMU components toward the middle of the top layer of the control board PCB, all connectors and switches would be placed toward the out side portion of the control board, and all the small simple components like capacitors and LEDs would be placed in between. For the components that connect to the control board like the motor ESC's and camera, connectors placed on the outside perimeter of the control board would be used. Any small component that is difficult to place and route on the top layer will be moved to the bottom layer. The empty areas of the bottom layer will be covered with a ground plane to provide easy access for the ground pins of all the ICs through vias near every corner there will be a connector that connects to the ESC on the arm of the quad-copter nearest to the connector. The connector for the control board power source will be in the middle of a side of the control board. The camera connector will be on the opposite side of the battery connector. A slide switch will be used to power the control board on or off. The slide switch will be placed somewhere on the control board where it will not be interfering with other components and easily accessible by hand. An LED will be placed on the control board that will light up when the quad-copter has finished its start-up and calibration routine and is ready to be flown. A 20-pin ribbon cable connector will be placed on the control board for a JTAG interface so that the STM32 on the control board can be programmed and tested. The connector will be the same kind as the one on the development board so that one can use the same JTAG programmer that is used on the development board. The group will try to acquire as many surface mount components as possible for the control board because surface mount components are smaller than the equivalent through hole size component thus reducing the PCB area needed for the project and surface mount parts are usually cheaper than through hole parts reducing the amount of money spent on parts for the project. The traces on the PCB will have a minimum width equal to the width of the smallest surface mount pin soldering pad on the board. The smallest soldering pad belongs to the STM32 at 0.3 mm (0.019 in.) per pin, so all traces on the PCB will be of this width. The maximum trace width will be 2.54 mm (0.1 in) which is the width of the soldering pad for the crystal oscillator.

10.5.11.2 IMU, Wi-Fi Placement

There are two components that require special consideration in how they are to be mounted onto the control board. The first of these components is the Wi-Fi module. The Wi-Fi module needs be placed so that the portion of the module that contains the on board antenna is off the edge of the control board PCB until the grounding pins are partially exposed on the PCB edge. This is done to reduce

radio interference to the antenna from the components on the control board and improve the quality of the wireless link to the iPhone. Furthermore the documentation for the Wi-Fi module states that the PCB area underneath where the Wi-Fi module is mounted on the control board should be kept clear of components, traces, vias, and other things that can cause radio interference problems. The second module that requires special consideration is the three components that make up the IMU; the accelerometer, gyroscope, and magnetometer. These three chips measure information based on the change of their axes' orientations. These axes are measured parallel to a specific side of each respective chip. The IMU chips will be mounted so that the x-axis is perpendicular to the top and bottom sides of the control board, this also automatically places the y-axis on the chips to be orthogonal to the x-axis. The z-axis requires no special placement on the control board it only requires that the control board be level with the quad-copter frame. When it comes time for these IMU chips to be mounted onto the control board the chip packages have to be aligned with the proper sides on the control board because if the chips are mounted in the wrong orientation the chips will show a change in the axes that is not the change the project firmware is expecting.

10.5.11.3 Control Board Assembly

The soldering that will be required to mount the components onto the control board will be done by the members of the group. The surface mount components for the control board will be soldered with the aid of the surface mount equipment in the radio club section of the senior design laboratory. The group will be taking the training session given by the radio club for using the surface mount equipment. Soldering of any through hole components will be done with a soldering iron to be acquired by the group. The control board will be mounted onto the frame of the quad-copter in such a way that each of the four sides of the control board will be perpendicular to each arm of the quad-copter. The finished control board will be mounted on top of a flat hard mounting surface that is centered on the middle of the quad-copter. The control board will have four screw holes, one near each corner that will be aligned to four stand-offs on the mounting surface. The stand-offs are used to provide some separation between the control board and the mounting surface. Once the control board is mounted onto the quad-copter it will be covered by a well ventilated hard but light weight plastic enclosure. The plastic enclosure is to provide protection to the control board if there should ever be an accident like the quad-copter falling to the ground from a great height.

10.6 IMU Sensor board

The IMU sensor board will encompass all of the necessary components to sense pitch, roll and yaw. In addition the sensor board must include the sensors necessary to be able to create the reference vectors to compensate for the errors

discussed in the DCM Controls Design section. The overall design of the IMU sensor board will utilize the following components in Table 15 to realize a self-stabilizing aircraft:

Component	Type	Use
IDG500	Dual Axis Electronic Gyroscope	Pitch and Roll Measurements
LY530AL	Electronic Gyroscope	Yaw Measurements
ADXL335	Triple Axis Accelerometer	Pitch and Roll Reference Measurements
HMC5843	Triple Axis Magnetometer	Yaw Reference Measurements

Table 15: IMU Sensor Listing

A dedicated PCB will be created to hold all of the sensors that are listed in the above table. This is done mainly to mitigate noise that can be induced through the presence of other devices, particularly with regards to the magnetometer. Electric current flowing through a wire can induce a magnetic field that can influence the magnetometer. While this effect is minimal, proper operation of the magnetometer is crucial to the success of the reference vectors to mitigate gyro drift. A schematic for the IMU sensor board can be found in the IMU Sensor Board Design section.

10.7 IMU Sensor Board: Noise filtering

Noise can greatly affect the measurements that are taken by the different sensors that are utilized by the IMU. The noise can be both environmental and due to the influence of other electrical components. Each component handles the noise differently and each component is susceptible to different types of noise. In the case of the accelerometers vibrations can influence the measurements taken. As discussed in the research section the MEMS accelerometer uses the capacitance that is experienced between plates that move based on the perceived gravity that is present. This perceived gravity is earth's gravity with the addition or subtraction of the experienced acceleration, depending on the direction of motion. The vibrations that are experienced during the course of flight can induce fluctuations in the perceived gravity measured by the accelerometer. There are several ways to correct this error signal. One way is to utilize an advanced software filtering technique such as the Kalman filter. As discussed in the research section the Kalman filter is quite CPU intensive and is thus not going to be used.

Another technique is to utilize a PID controller to correct for the fluctuations in the accelerometer measurements. This method is the preferred method and the method that is most likely to be realized. Further testing is needed to determine whether or not a PID controller alone will be acceptable to fulfill the needed vibration filtering. Quite often, specifically in applications where sensors are

exposed to vibrations that are not the process being measured, a protective coating is used to absorb some of the vibrations being experienced by the sensors. This method is to be used in addition to the use of a PID controller. The combination of a PID controller and a protective coating should produce measurements with minimal errors.

There are some negatives that may be introduced by using a protective coating to absorb vibrations. If the entire board is coated this may hamper the heat dissipation of the components present on the board. In addition once the board is coated it will be difficult to remove if troubleshooting of component electrical connections is necessary. Of course further testing will be needed to verify such claims. Testing methods for the IMU sensor board protective coating are presented in the Design Testing section.

10.8 DCM Control Design

The directional cosine matrix is the approach that has been chosen to represent the orientation of the helicopter. The gyroscopes will provide the rotational data for roll pitch and yaw, while the accelerometers and the magnetometer will provide the reference vector that will determine the error in the gyro measurements. Figure 34 depicts the top-level control loop that will be implemented in code to stabilize the helicopter.

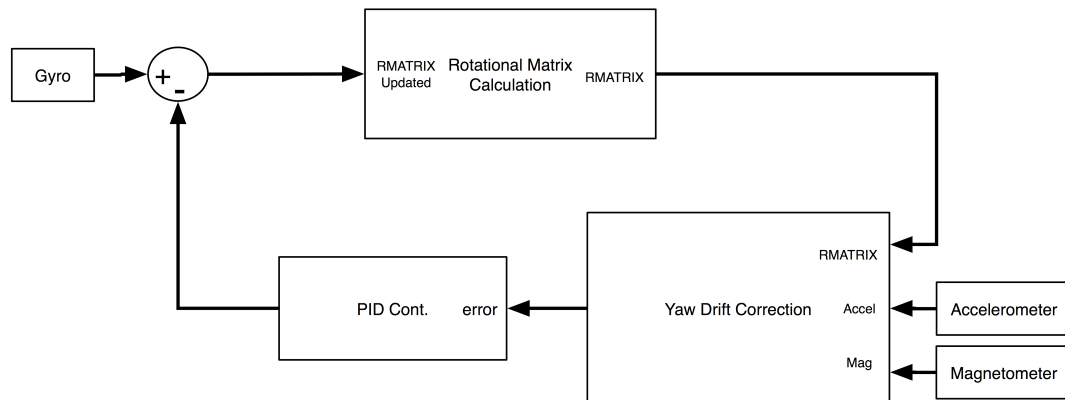


Figure 34: DCM Feedback Control

10.8.1 Rotational Matrix Calculation

There is a certain initialization routine, with regards to the IMU, that must be followed while the device is on the ground to ensure proper measurements can be realized during flight. Before the rotational matrix can be formed from the electronic gyroscope measurements, the rotational matrix must be first initialized while the device is on the ground. Rmat is first initialized to the following...

$$Rmat = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

This matrix is an identity matrix that sets up earth's reference frame for all subsequent measurements to be derived from. Once this initial Rmat is created the code will progress to update Rmat to the orientation information derived from the accelerometer and the magnetometer. This updated Rmat is the aircrafts initial heading while on the ground. Each subsequent iteration of the rotational matrix update is set equal to RmatOld, which provides a reference to derive the rotational angles from the gyro measurements.

The rotational matrix calculation is derived from the rotational rates that are measured by the gyroscopes. One gyroscope measures roll and pitch, the IDG500; yaw rotational rate is then measured by a dedicated single axis gyro. The measurements that are recorded by the gyroscopes are voltages that correspond to rotational rates that are experienced by the aircraft in relation to a given axis. These values must then be converted to their corresponding rotational rates to be able to acquire the angular measurement to be able to set up the rotational matrix.

$$\omega = \frac{GyroVoltageOutput - GyroZeroRateOut}{Sensitivity}$$

The dual axis gyroscope that is being used for the IMU sensor board, IDG500, has two sensitivity modes. Each mode has its own zero rate voltage and sensitivity range. The sensitivity range that has been chosen for the purposes of this design corresponds to ± 500 °/s. Taking into account the given sensitivity range of the sensor and the nominal zero rate output, the rate of rotation can be derived using equation XX. Given that the group now have the rotational rate, a relation to the angle of orientation can be made. The angle of rotation, with respects to the gyro measurements can be made by relating measurements over a specified time period, T.^[5]

$$AngleX = RgyroX * T$$

$$AngleY = RgyroY * T$$

$$AngleZ = RgyroZ * T$$

Taking the equations from above the group can now form the preliminary rotational matrix. Recall from the research section that...

$$r(t + dt) = r(t) + \int r(t) \times d\theta(t)$$

where...

$$d\theta(t) = \omega(t)dt$$

This property can be applied by each element of the rotational matrix to update the matrix each specified interval, T.

$$RmatUpdate = RmatOld + RmatOld * Rmat$$

where...

$$Rmat = \begin{bmatrix} 0 & -d\theta_y & d\theta_z \\ d\theta_x & 0 & -d\theta_z \\ -d\theta_x & d\theta_y & 0 \end{bmatrix}$$

The negative sign is derived from the fact that the gyroscope measurements are in the reference frame of the helicopter and must be converted to the earth's reference frame to be of any use.^[1] Realizing the symmetry that exists between the reference frame the group can convert to the earth's frame by simply reversing the sign of each measurement. Also, in an effort to reduce the amount of addition that is seen by the microcontroller the first term of equation of RmatUpdate can be brought into the Rmat to realize the following equation and thus removing one matrix addition step and three subsequent additions to be performed.

$$RmatUpdate = RmatOld * Rmat$$

where...

$$Rmat = \begin{bmatrix} 1 & -d\theta_y & d\theta_z \\ d\theta_x & 1 & -d\theta_z \\ -d\theta_x & d\theta_y & 1 \end{bmatrix}$$

10.8.2 Normalization

Proper measurements are crucial to the success of the IMU's stabilization algorithm. In order to maintain the integrity of measurements and calculations the group must recognize the errors that can occur in the measurements and calculations. Due to the numerical integration that is utilized in the DCM algorithm errors tend to accumulate in the rotational matrix calculation that can offset the orthogonality of the XYZ identity vectors. To alleviate the accumulation of errors the rotational matrix must be periodically renormalized. Each axis is first separated into a matrix as follows.

$$X = \begin{bmatrix} r_{xx} \\ r_{xy} \\ r_{xz} \end{bmatrix} \quad Y = \begin{bmatrix} r_{yx} \\ r_{yy} \\ r_{yz} \end{bmatrix}$$

The matrix property of the dot product is utilized to determine the error that may

exist in the orthogonality between the X and Y vectors. To be orthogonal the product must be equal to zero and any deviation otherwise is an error that has accumulated. Taking the X matrix and the Y matrix presented previous, the dot product is performed to derive the error. The error is then apportioned to evenly to each vector and subsequently subtracted from the vector to derive the orthogonal updated version of the vector.

$$\begin{aligned} X_{orthogonal} &= X - \frac{error}{2} X & \text{where ...} \\ Y_{orthogonal} &= Y - \frac{error}{2} Y & error = X \cdot Y = X^T Y \end{aligned}$$

The updated X and Y matrices are now orthogonal and can be used to calculate the orthogonal Z matrix. This is done by taking the cross product of $X_{orthogonal}$ and $Y_{orthogonal}$, yielding $Z_{orthogonal}$.

$$Z_{orthogonal} = X_{orthogonal} \times Y_{orthogonal}$$

To complete the renormalization process the each of the vectors must be verified to have a magnitude of one. In order to find the scaling factor that is necessary to adjust each row vector, several properties of matrices must be identified.

$$X \cdot X = |X| = \sqrt{x_x^2 + x_y^2 + x_z^2}$$

The cross product is then subtracted from three (the number of vectors) and subsequently multiplied by $\frac{1}{2}$, thus yielding a orthogonal matrix who's vector have magnitudes of one.

10.8.3 Drift Calculation: Roll, Pitch

As discussed in the research section the electronic gyroscopes that are being used in this project are susceptible to drift in the measured values. The drift is also compounded by the errors that can accumulate in due to the fact that the numerical integration is approximation, albeit a rather acceptable one. To alleviate the accumulation of errors in the gyro's measurements and subsequently the rotational matrix the group must perform some sort of drift correction. The chosen method is to utilize a separate reference vector that is determined by alternative measurement technology to the gyroscope. An accelerometer will be used to generate a reference measurement to compensate for the roll and pitch drift.

To achieve the correction that is necessary the group realized the properties of the accelerometer. Accelerometers measure gravitation that is experienced by a

device plus the acceleration. To create the reference vector the group will exploit the gravity measurements that are taken by the accelerometer. From these measurements the group can take the cross product with the estimated gyro measurements, the rotational matrix. The cross product of two vectors yields a magnitude that is proportional to the sine of the angle that is formed between the vectors. It also produces a direction that is perpendicular to both vectors, thus it is a rotational axis.

$$Accel = \begin{bmatrix} ref_{ZX} \\ ref_{ZY} \\ ref_{ZZ} \end{bmatrix} \times Rmat_{ZZ}$$

10.8.4 Drift Calculation: Yaw

Equally important to the proper operation of the IMU is the compensation of the yaw drift that can accumulate in the gyro measurements. As discussed in the research section, a magnetometer was chosen as the sensor of choice to generate the yaw reference matrix. In particular a triple axis magnetometer is necessary to be able to correctly derive the matrix. The magnetometer produces

10.9 PI Controller

The PI controller is responsible for reducing the error that is present between the reference vectors and the rotational matrix's previous calculation. The Proportional Integral controller was chosen over the Proportional, Integral and Derivative controller because of the simplicity of code and use of resources on the microcontroller. The derivative term would be much more difficult to implement on the microcontroller, both in use of resources and coding the algorithm. The controller will be utilized to minimize the error that is present between the two values. Once the vectors have been sent through the PI controller the vectors will then be fed back to the rotational matrix update algorithm. Tuning of the PI controller is also an essential element to determining the response of the controller. Tuning methods will be addressed in the Testing section of the document.

The each error that is calculated by the pitch, roll and yaw drift correction algorithm is fed into the PI controller, along with the new measurement generated from the electronic gyro sensors. The measurement matrices are added together to form the error measurement.

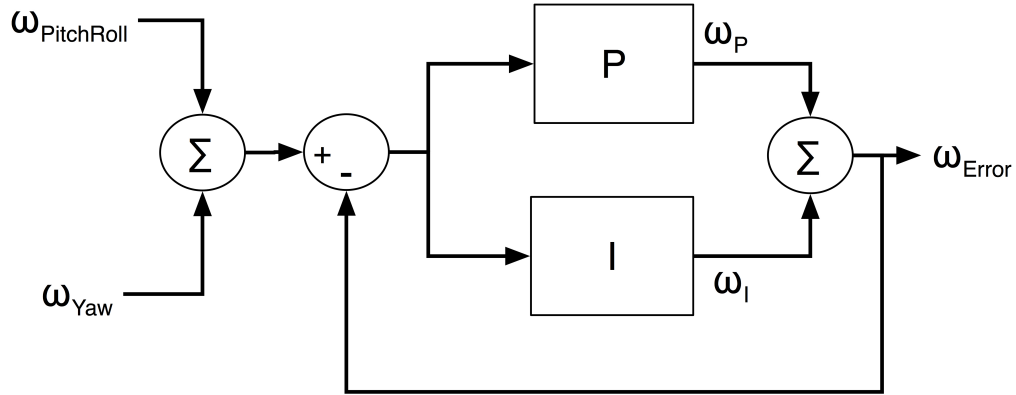


Figure 35: PI Controller

$$CorrectedTotal = PitchRollCorrection + YawCorrection$$

$$\omega_{error} = \omega_I + \omega_p$$

Subsequently the corrected total is multiplied by a term that is proportional to the corrected total multiplied by the proportional coefficient. The proportional term adjust the output proportionally to the current error value. The corrected total is also fed through the integral term of the PI controller. Integration is realized by maintaining an accumulation variable (ω_I), essentially the previously determined Integration correction result, which is added to the integration coefficient multiplied by dt and the CorrectedTotal.

Term	Equation
Proportional	$\omega_p = k_p CorrectedTotal$
Integral	$\omega_I = \omega_I + k_I dt CorrectedTotal$

Table 16: PI Controller

10.10 IMU Software

10.10.1 Directional Cosine Matrix

The directional cosine matrix forms the basis of the overall IMU design that is being implemented. In order to create this matrix measurements first need to be sampled from the gyroscopes. The signals that are being sent from the gyroscopic sensor to the STM32 microcontroller are analog signals and need to be converted to a digital representation to be of use to the microcontroller and subsequently the control algorithm. The STM32 contains two 12-bit analog-to-digital converters that share 16 external channels. Both of these ADCs will be

utilized to sample the measurements that are produced by the sensors. These values will then be utilized to update the DCM. The ADC module code specifies how the sensors are sampled and filtered. Values that are produced by the ADC that originate from the gyro sensors will then be sent to the DCM update algorithm. First the several steps need to be followed to initialize the DCM. The initialization module, IMUinit, will handle sensor measurement matrix initialization and DCM initialization. Figure 36 depicts the overall software architecture of the DCM.c module that will handle the IMU calculations and provide the orientation of the aircraft to the main.c module.

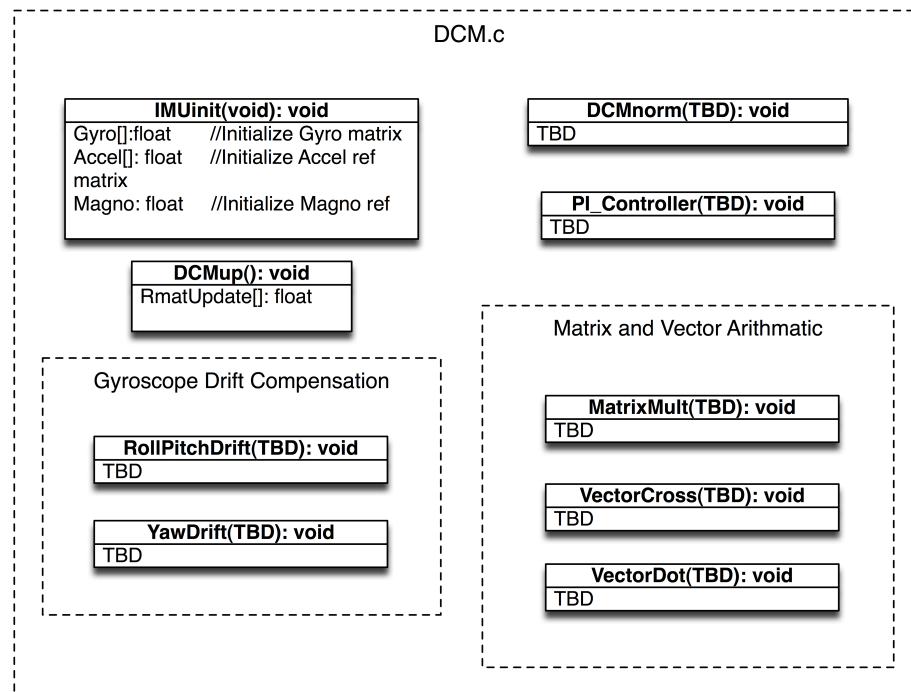


Figure 36: DCM.c Code Structure

The DCM module will provide the functionality that has been discussed through out this document. While the DCM module will not handle the stabilization functionality directly, this will be determined by main.c, DCM will provide the orientation of the aircraft. Based on the calculated orientation information and the current command that is being sent to the aircraft, main.c will be able to determine the signal that is necessary to send to the servos to achieve the desired movement.

10.11 IMU Sensor Board Schematic

The IMU schematic provides the necessary circuitry for the sensors that are required by the DCM method that is implemented to provide orientation information to the quad-copter. The circuit currently encompasses a two axis electronic gyroscope for pitch and roll measurements, a single axis gyroscope for

yaw measurements, a three axis accelerometer for that will provide pitch and roll drift correction and lastly, a three axis magnetometer for yaw drift correction. There are three different voltages that are required for the operation of the IMU sensors: 3.3V, 2.5V and 1.8V. A main power signal will be sent to the IMU sensor board via the power circuitry on the control board that is derived from the dedicated electronic battery. This power signal is designated 3.3V_POWER in Figures 37 through Figure 44.

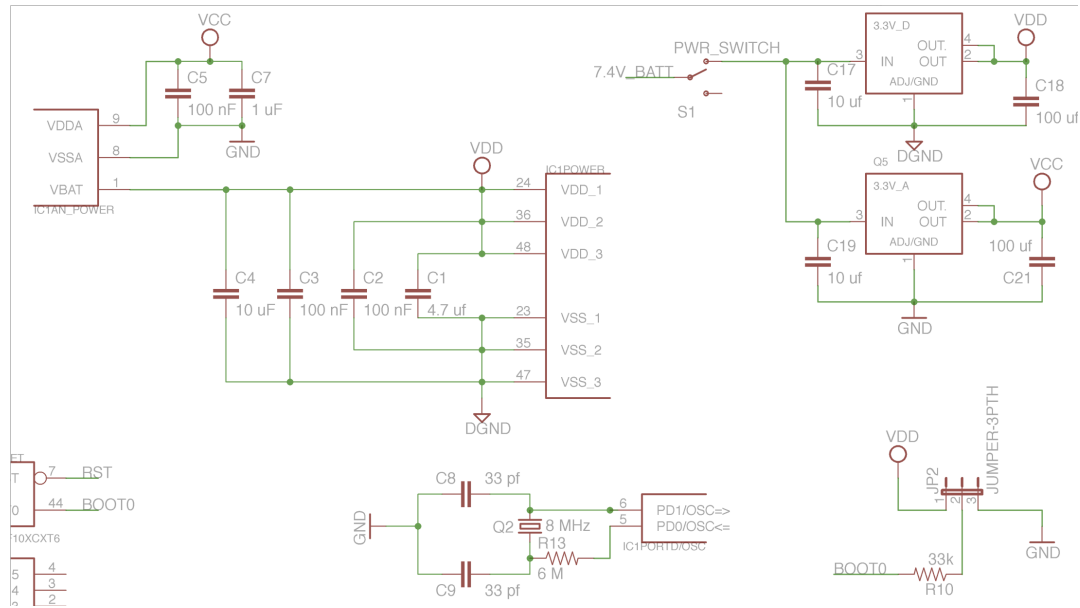


Figure 37: Control Board Power Schematic

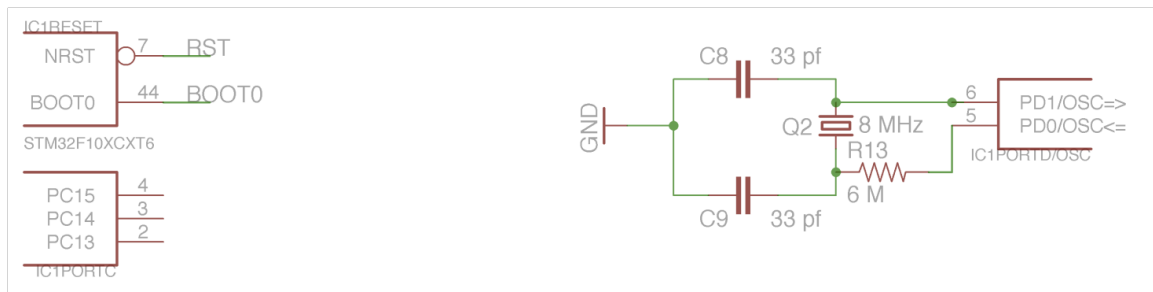


Figure 38: Control Board Crystal Schematic

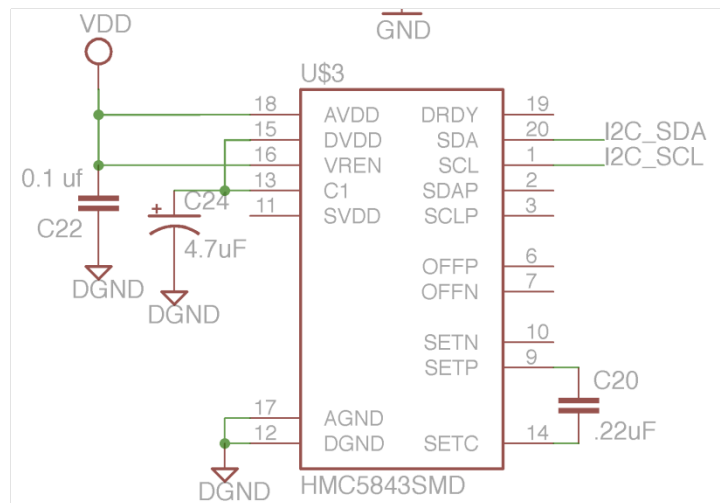


Figure 39: Control Board Magnetometer Schematic

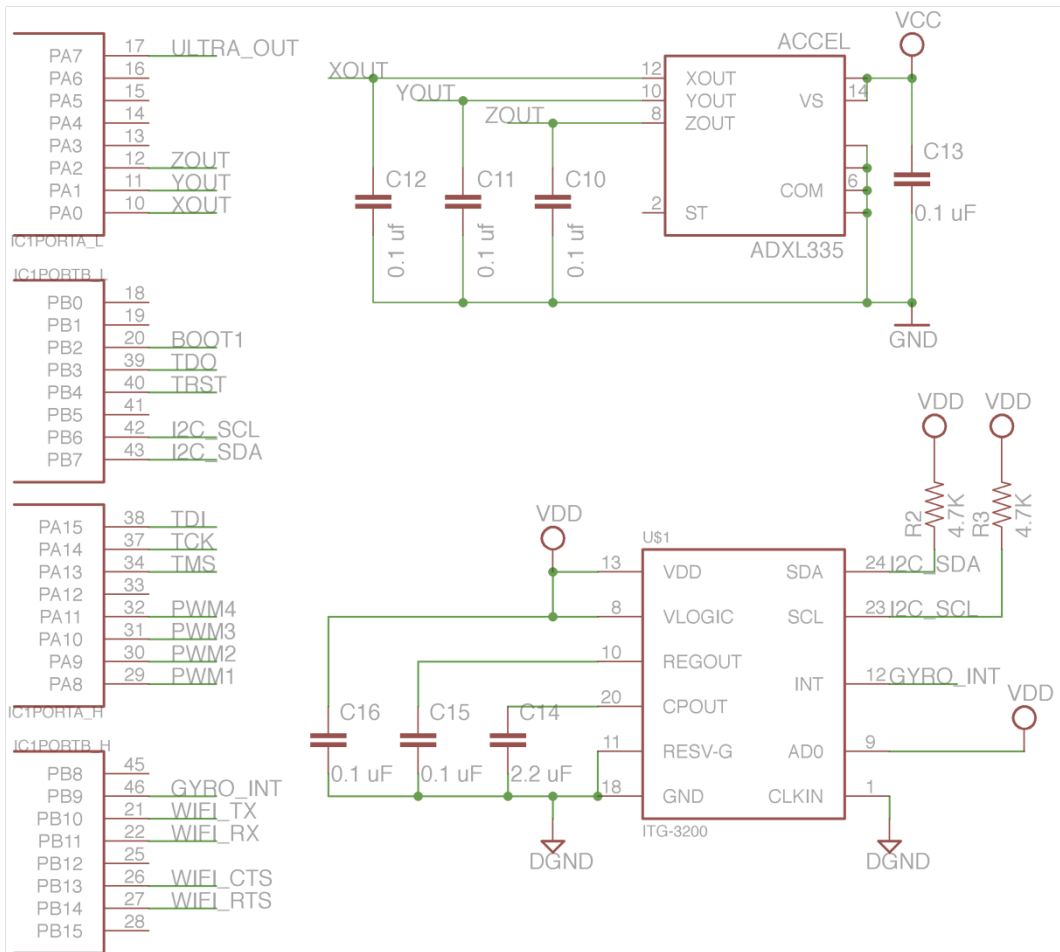


Figure 40: Control Board: Microcontroller Interface Schematic

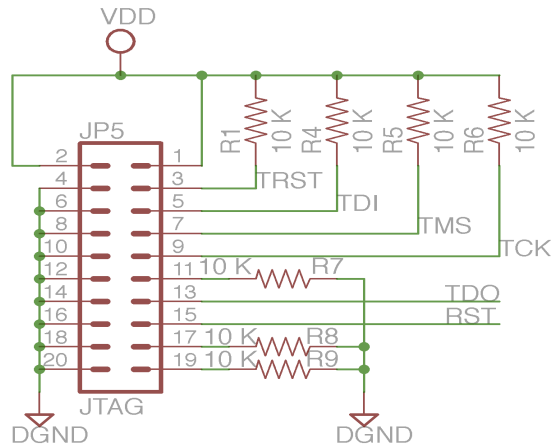


Figure 41: Control Board: JTAG port Schematic

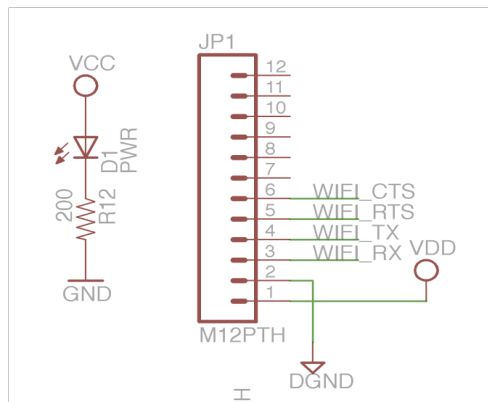


Figure 42: Control Board: Wi-Fi connection Schematic

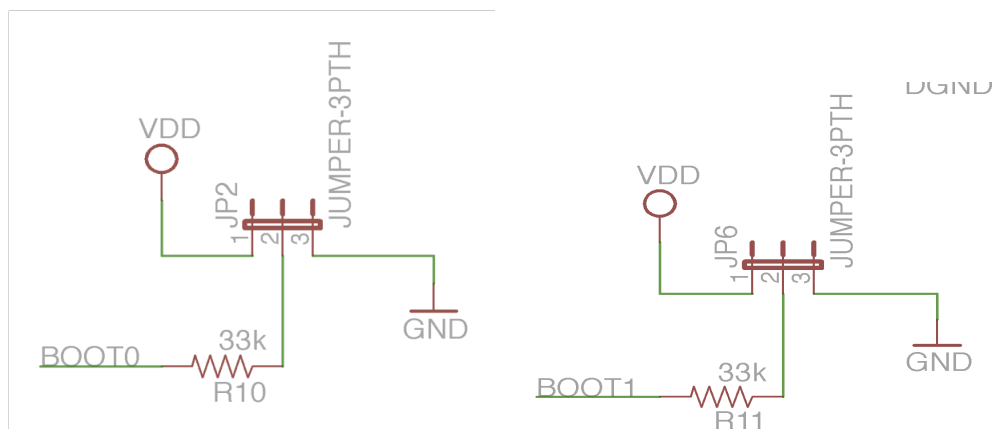


Figure 43: Control Board: Boot Mode Jumper Schematic

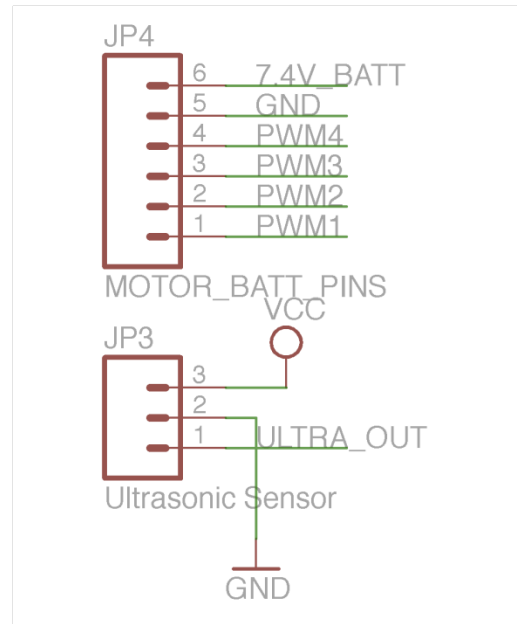


Figure 44: Control Board: Motor Signals and Range Finder Schematic

10.12 Camera Module Design

A camera module will be utilized in this design to transmit real time video to the iPhone screen. The module that was selected was the C3038 from electronics123.com. This module was chosen for several reasons. The first reason being the ability to produce a digital output, thus alleviating the need for an ADC for the camera output. The module produces a progressive 356 X 292 pixel format. This is a bit less than the native resolution of the iPhone but the image will be suitable for the needs. The device also has the ability to output analog video.

The data out utilizes I²C communications. Thus the module will be able to interface with the STM32, which has an I²C interface. Subsequently, the video will be analyzed by the STM32 and sent to the WI-FI module to be transmitted and received by the iPhone for viewing.

A function that will reside in the main.c, which is the main control program for the STM32 microcontroller, will handle the tasks necessary to gather the camera data and is responsible for setting the interrupt to transmit the data to the WI-FI module.

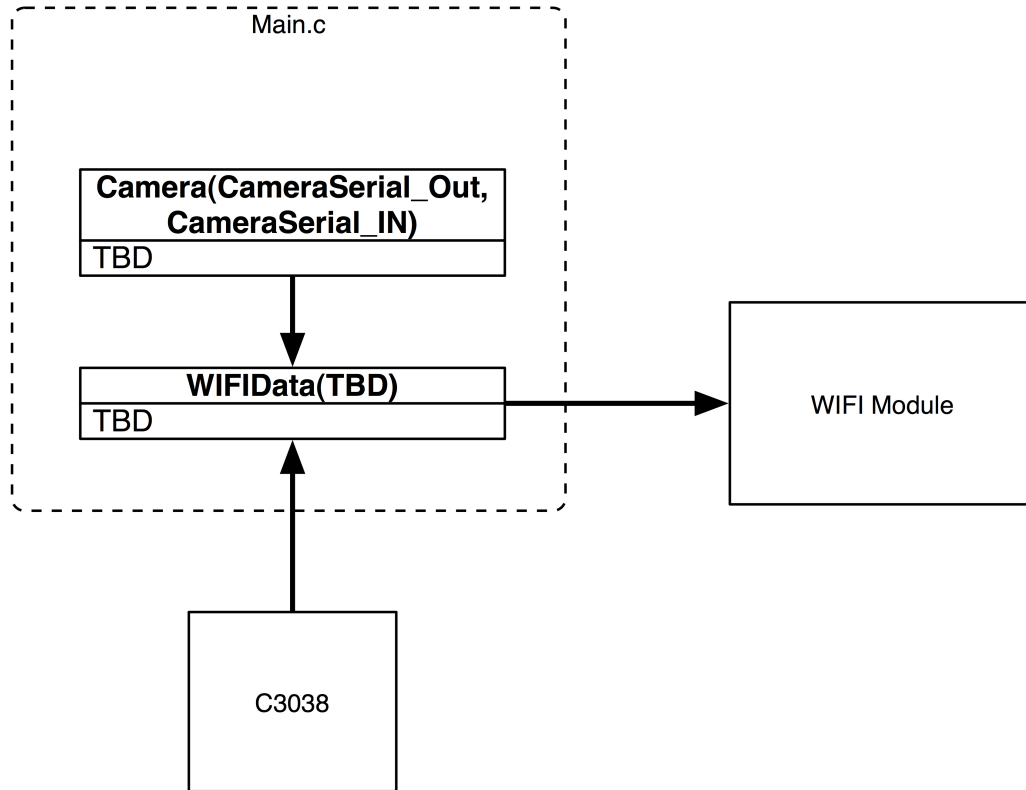


Figure 45: Camera Module Software/Hardware Interface

10.13 Ultrasonic sensor Design

The ultrasonic sensor we purchased has easy access to the pins it offers to power the sensor, ground the sensor and also transmit and receive the sensor information as well. The sensor has a small form factor that makes it easy to place anywhere on the copter. For our purposes we will need to have the sensor located on the bottom of the copter to transmit the height at which the copter is flying at. The sensor takes in 3.3V to 5V voltage source to power the sensor. The bigger the supplied voltage to the sensor the larger the maximum range the sensor can report. With 5V of supplied voltage, the ultrasonic sensor can detect items from 25.1 feet away and with 3.3V of supplied voltage, the sensor can pick up on items all the way up to 19.6 feet away. There are multiple ways for the sensor to link to the processor the sensor can hook up to the processor using an analog, serial or PWM port. To measure the distance using the analog port, the processor will have to monitor the voltage coming from the analog port and divide the voltage coming out by 1024 ($V_{cc}/1024$) to give the measured distance. The serial interface is formatted closely to an RS-232 port and has a baud rate of 9600 with 8 bits of data, no parity bit and 1 stop bit. The final way to read the sensor is through the pulse width modulation (PWM) pin which outputs a pulse width representation of the distance to the object with a factor of 58 microseconds per centimeter.

For the aspect of our project, the easiest solution for our design is to mount the sensor on the bottom of the copter. We will be using the serial port to connect to the processor and monitor the height of the copter. This will give us an easy and stable connection to our processor. We will be using a 3.3V source to power the sensor from the battery located on the copter and since it has a very small current draw of only 2.1mA when connected to a 3.3V source. Figure 46 shows the small form factor of the sensor and how it can virtually be placed anywhere without interfering with the design of the copter.

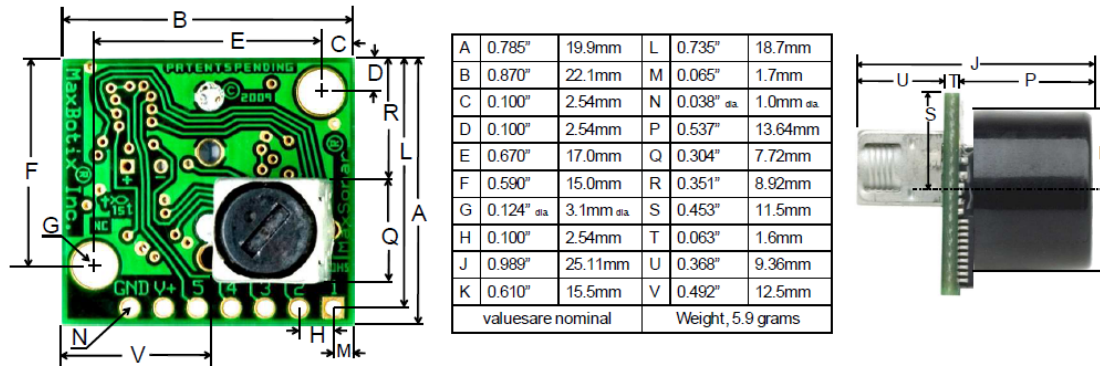


Figure 46: Form Factor of ultrasonic sensor (Use with permission from MaxBotix Inc.)

10.14 Wireless Transceiver

The Wi-Fi module the group purchased has easy access to the pins that the group need to transmit, receive and power up the Wi-Fi module. The module has 3 LED's on the board to show the current status of the Wi-Fi module, which allows for easy debugging and knowing what is currently going on in the module. The module accepts power from a regulated 3.3-3.7V source or from 2.0-3.0V batteries. The group plan on using the SPI ports on the module to allow for fast connections between the Wi-Fi module and the micro-controller. If the SPI ports do not give a fast enough connection the group plan on using the UART pins to connect to the micro-controller, this configuration is a very easy connection as you only need to have 4 pins connected to start receiving and transmitting data (Power, Transmit, Receive and Ground Pins). Figure 47 shows the pin layout for the RN-131 module that the group will use to connect to the battery and the microcontroller. This module will allow the iPhone and microcontroller on the quad-computer to communicate and send the necessary data between each other.

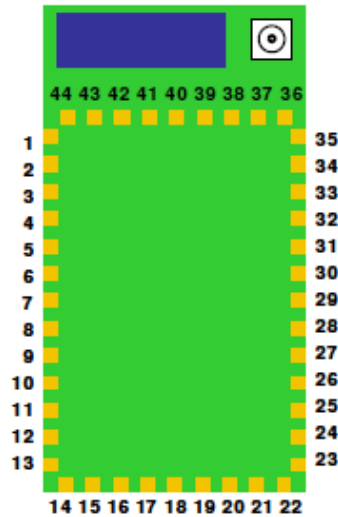


Figure 47: Roving Networks Wi-Fi module pin layout

Figure 48 shows how the iPhone and the quad-copter will communicate with each other. The video and data that is coming into the Wireless Transceiver will be coming from the microcontroller located on the copter and the transceiver module will then wirelessly send the information on over to the iPhone where the iPhone will then display the video and status information from the copter. The iPhone will then in turn send the information that the user is inputting into the iPhone and transmit that to the wireless transceiver which in then will communicate these control messages to the microcontroller for the movement of the quad-copter.

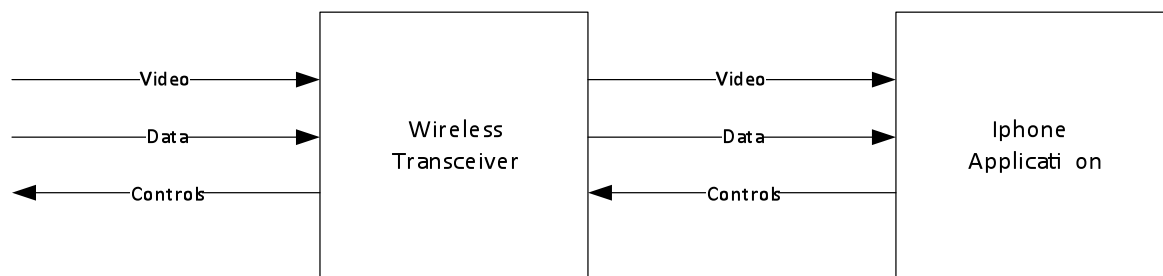


Figure 48: Wi-Fi transceiver to iPhone interface

10.15 Software

The software is split into two different sections: the iPhone and the quad-copter. The two sections communicate between each other using a Wi-Fi connection that is set up through a Wi-Fi transceiver that is on the quad-copter. Figure 49 gives an overview of the software system communicating with the quad-copter

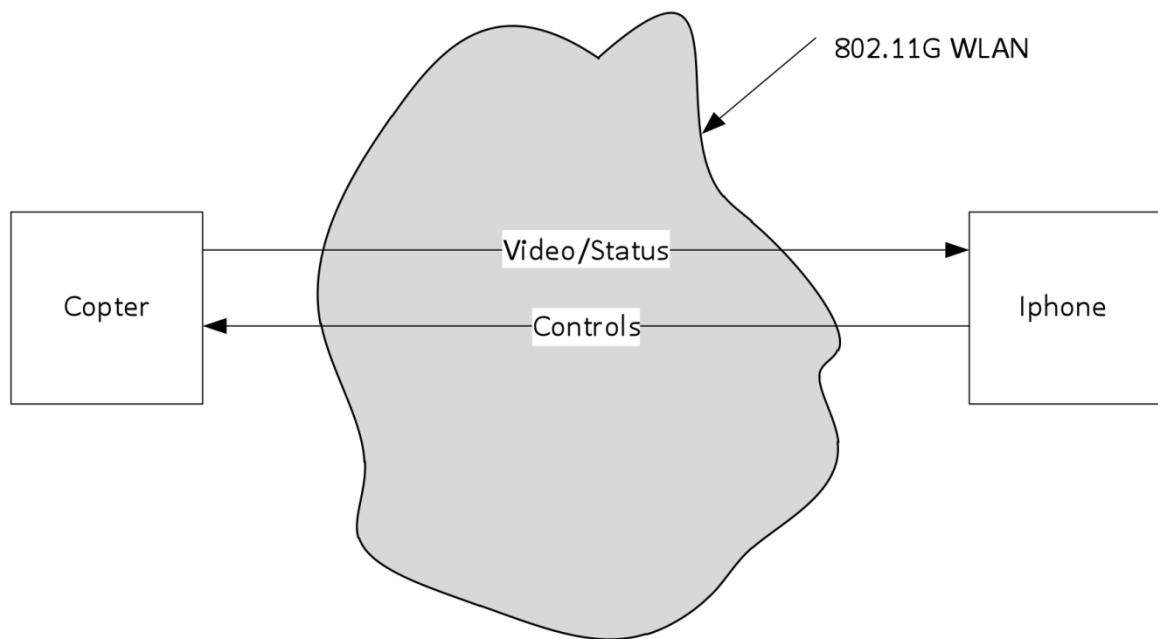


Figure 49: Data Flow from Copter to Copter

The application on the IPhone will be the main control method to control the quad-copter. The IPhone app will have two main screens, one to show a video of where the copter is facing and control the copter and the other give details of the current status of the quad-copter. The screen that houses the video of the copter will have three sliders and one menu bar. The sliders on the left and right hand side of the screen will tell the control the copters left, right, forward and backward movements. The slider on the bottom of the screen will control the height of copter. These messages will be sent over an 802.11G protocol from the IPhone on over the Transceiver that is on the quad-copter. The messages for the left and right sliders had a default format of XX.XX where each “X” represents a value from 0 to 9 and had a max value of 99.99. This gave the user too much control over the copter making even the slightest change to the sliders would send control messages to the copter and moving it in the direction that the sliders were moved to. This also made it harder for the user to completely keep the copter still.

There were two fixes that were made to keep the copter steadier and also give the user an easier way to control the quad-copter as well. To keep the copter steady the group made the sliders move back to a steady state in the middle when the user has lifted their fingers off of the sliders. This fixed two problems; it didn't allow the copter to be stuck in a permanent state of moving in one direction because of a slider becoming stuck and also allowed the copter to come to a

hovering state when the user was done applying the sliders. The other fix that was made was to fix the sensitivity of the sliders. Instead of having a format that was XX.XX which was very sensitive to any small amount of movement of the sliders, it was decided that the decimal points be dropped off of the format for the value of the sliders.

The next change to the slider value format is that the user would have to choose between three different levels of control for the quad-copter control. The three different levels would be: beginner, trained, and expert. Each level would give the user a different amount of control over the copter. The beginner level would take the slider values and divide it by 20. This would only send a max movement to the quad-copter of ± 5 for each slider movement. For the trained level the slider levels would be divided by 10 giving a maximum movement for each slider of ± 10 . For the Expert level the slider levels would be divided by 5 giving a maximum movement for each slider of ± 20 . Giving the user this amount of freedom to choose between the different levels allows the user to pick the controls that are most comfortable for them. It also allows the user to pick the amount of fine control that is needed sometimes in closed areas.

To control the height of the copter there will be a slider on the bottom of the screen that will adjust the height to the user's specification. Unlike the sliders that control the side to side and forwards and backwards movement, this slider will not move back to a central position. Instead this slider will stay wherever the user has last moved the slider to. Upon start up of the copter, the copter will move to a height of 5ft and the slider will accurately portray this as the copter begins to rise. The quad-copter will have a safety feature in it to not allow the user to bring it all the way too close to the ground by use of the slider. To monitor the height of the quad-copter the group will be using an ultrasonic sensor that will be able to monitor the height. The maximum height for the copter to accurately fly within control of the ultrasonic sensor will be of 10 feet. This slider also has a default format of XX.XX which would give the user too precise of a control that would not be useful for the quad-copter. To give it a better feel for the height slider, the group will divide the slider level by 5 giving 20 equal levels of unit to use to control the height. Each level will represent a half of foot giving a range from 6 inches all the way up to 10 feet.

The packets that will be sent to the quad-copter from the iPhone will be a string that will only have 3 characters. The packets will have one of three different leading characters either "L", "R", or "H" representing which slider was moved and by how much. The characters following one of those characters will be a number from -10 to 10, besides the bottom the slider which will be from 0 to 20.

To be able to see the direction and where the copter is facing the group will be using a video camera to allow the user better control. The video is captured by a sensor on the copter and then is processed through an Image Processing Unit (IPU). The Image Processing Unit then sends on the video data onto the

microprocessor where it then sends the data onto the wireless transceiver. The Wi-Fi transceiver then sends the video data to iPhone via an 802.11 WLAN connection. The iPhone then plays the video in the application.

10.16 iPhone Application

The iPhone Application will be the primary method to control the quad-copter. To write this application the code had to be written in the XCode IDE developer in Objective-C using the Cocoa Touch library which is the standard library for iPhone applications. The main function of the iPhone application will be to control the quad-copter using the sliders that were previously talked about and to show the video that is being captured by the camera on the copter.

The GUI for the application will be made up of different view controllers and a navigation controller that will push the different views onto the iPhone screen. The different views will lead the user through logging into the quad-copter and securing a connection with copter. Once the user gets through the connection screen the user then reaches the main screen where they can control the robot and change the different setting of the robot as well. Figure 50 shows the draft version of what the GUI would look like.

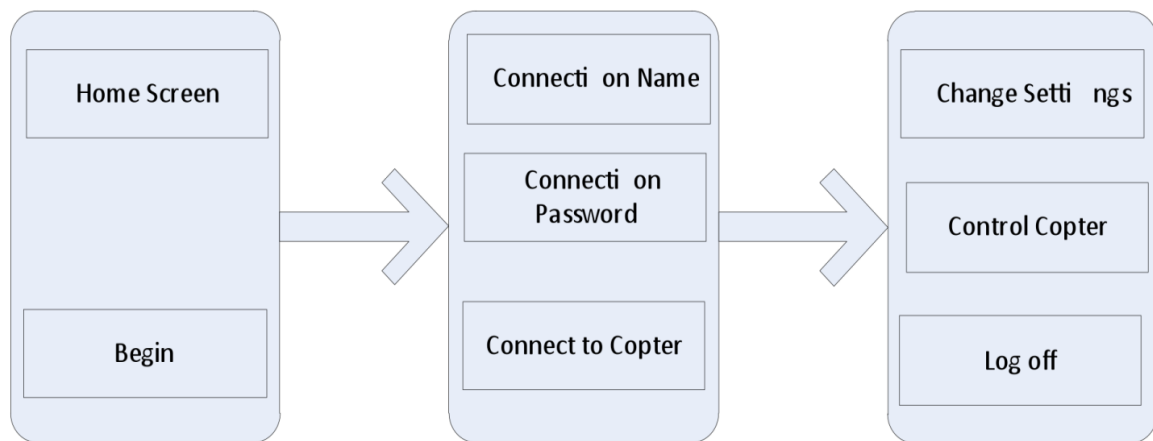


Figure 50: Draft of iPhone Application GUI

The home screen for the iPhone application will be a screen that displays a picture in the background with a button on the bottom of the screen that will say “begin”. When the user clicks on the “begin” button the user will be brought to a new screen where it has the connection name and connection password fields. The user will then have to input the connection name and password and then hit the “Connect to copter” button. The iPhone will then try to connect to the quad-copter using the Wi-Fi capabilities of the iPhone. The iPhone will try for a certain amount of time and if the connection cannot be confirmed in the allotted time the iPhone will display a message saying the “Connection has timed out”.

After the iPhone has been connected, the iPhone will then send a packet over to the quad-copter and the quad-copter will then interpret that packet and make sure the information in that packet is correctly formatted. Once the quad-copter has made sure the packet is correct, the copter will then send a return packet that will verify that original packet has been received and is valid. If the copter receives a packet that is incorrectly formatted the copter will then send a message back to iPhone that will in turn display a message on the iPhone screen saying “The connection is not secure, the connection has been disconnected” and then disconnect the connection between the iPhone and quad-copter. If the iPhone receives a packet from the quad-copter that is not in the correct format, the iPhone then will disconnect from the connection between the iPhone and the quad-copter and display a message to the user saying “The connection is not secure, the connection has been disconnected”.

Once the connection has been secured and confirmed by the iPhone and copter, the user is shown a screen with the 2 sliders on either side of the screen and a third slider on the bottom part of the screen. The sliders will be overlaid on top of a video screen that will show the video that the copter is sending over to the iPhone. These controls will be coming from the Cocoa Touch libraries from the iPhone SDK. The sliders that will be used will be coming from the UISliders class. The video will come from the MPMoviePlayer class that will stream the live video feed coming from the quad-copter.

The Model-View-Controller pattern is used to organize the iPhone application for the quad-copter. Every screen that is created using this pattern creates two files from the iPhone SDK, an XIB file and a controller file. The XIB file contains all of the GUI components and how the screen is laid out on the iPhone and also the graphical elements of the screen. The controller file has all of the objective-C code that is passed through to the XIB file. The controller file reacts to all of the inputs that occur to the GUI from the user and outputs the corresponding event.

For every screen that the group need, the screen will need its own UIViewController class that is linked to the XIB file. All of the classes for the GUI, controller and the event callback methods are included on Figure 51. The UML diagram below shows there is a Quad-copterAppDelegate class that implements the UIApplicationDelegate protocol. This is the root class for the application. The navigation Controller located in the Quad-copterAppDelegate class shows the various screens that the group created on the screen.

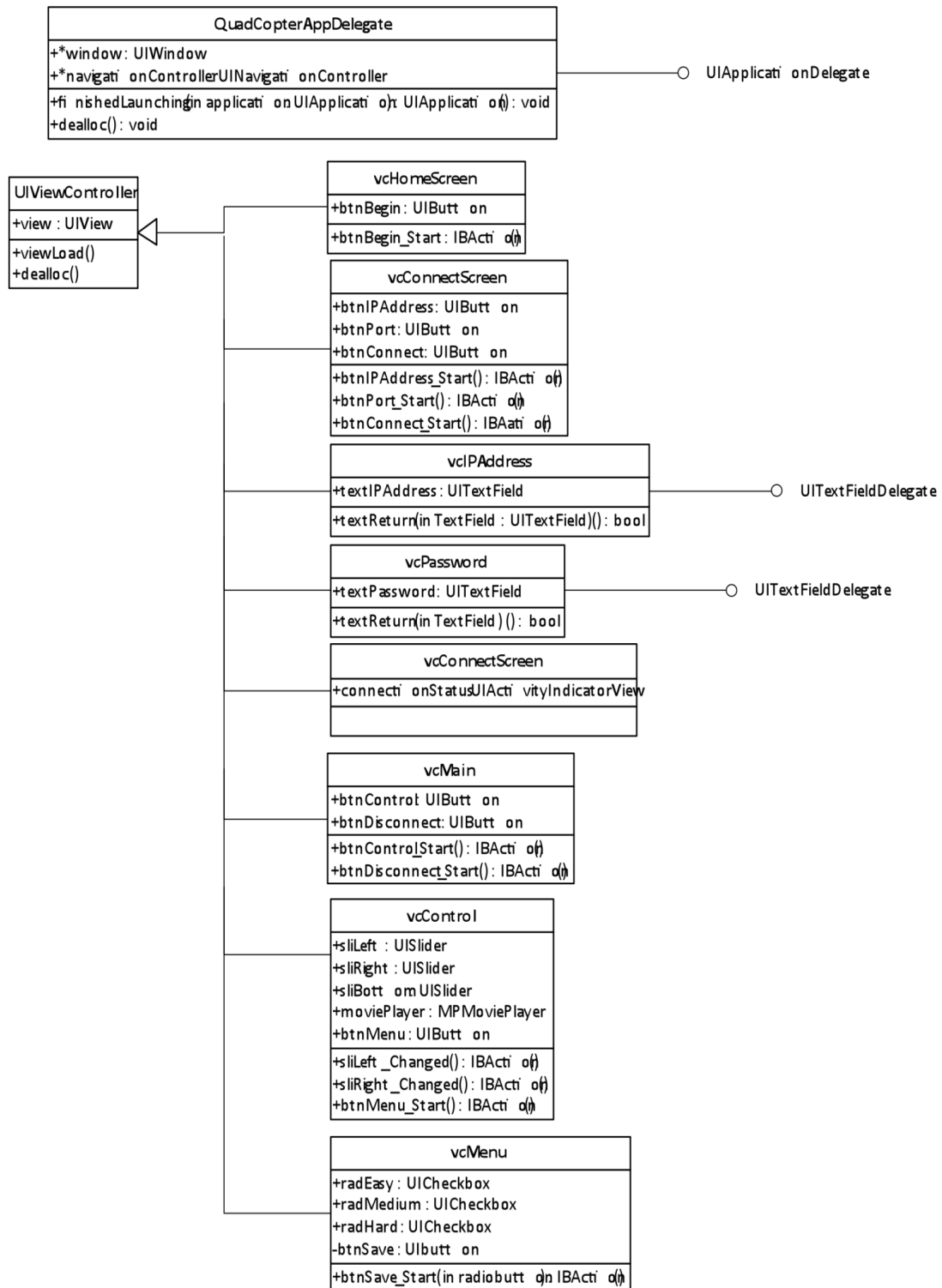


Figure 51: UML Class Diagram of iPhone Application

iPhone applications are not like that of a computer in the sense that multiple applications cannot be running at the same time where as on a computer or other operating systems multiple applications are allowed to run at the same time. On the iPhone once when the application is started the iPhone operating system lets the application take over and runs it its own environment.

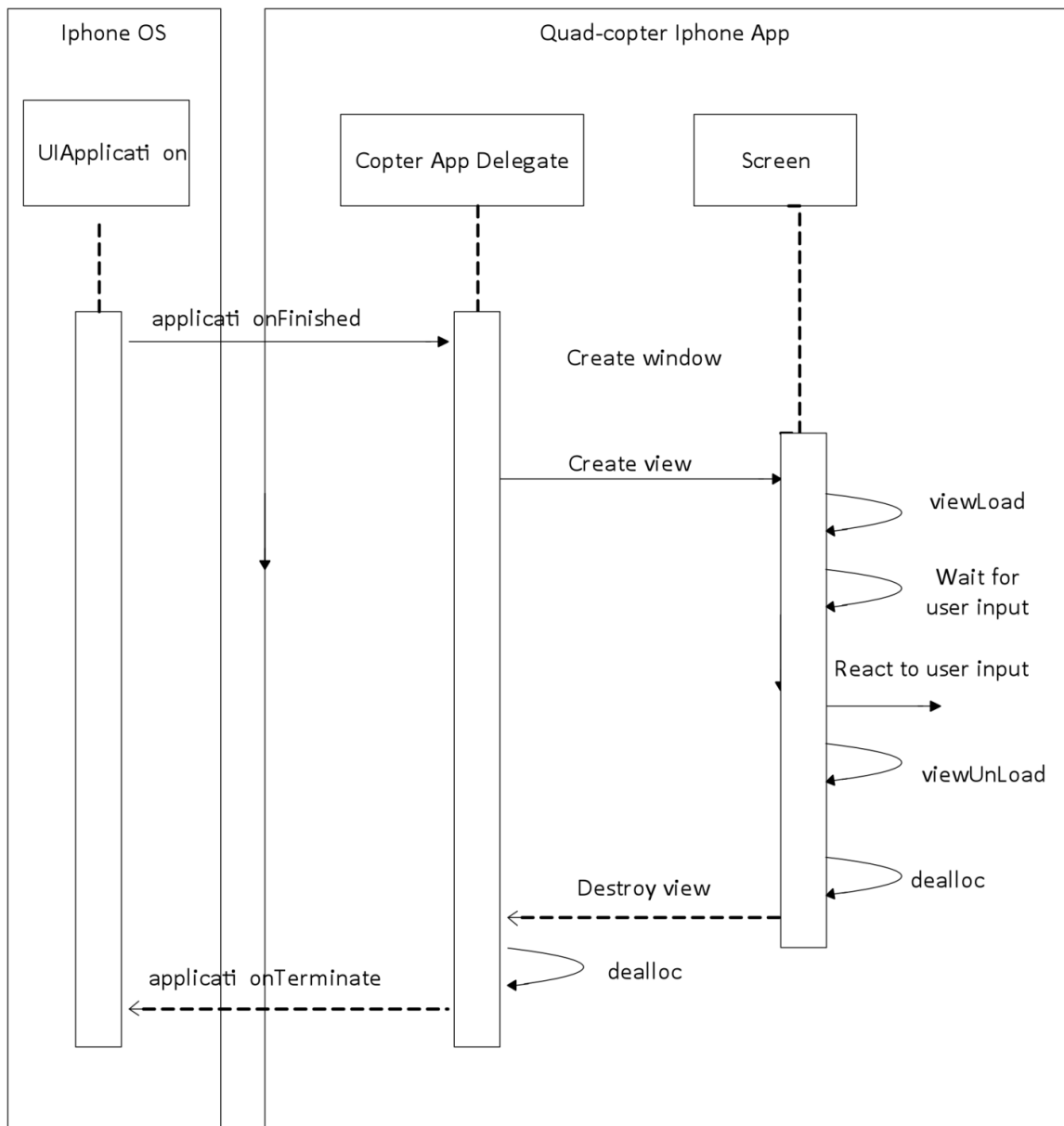


Figure 52: UML Sequence Diagram

Each screen that is loaded by the application has a viewLoad method that gets called to display the view and a viewUnLoad method that is called to destroy the current view. This allows for the view controller to initialize the screen and also

close it down reallocate the memory that was used to run the screen. For initialization the methods can set the fields up already for the different view such as the connect screen for the user name and password.

To push a view controller through onto the navigation controller, the following piece of code is needed.

```
vcSettings *vc = [[vcSettings alloc] init];  
[self.navigationController pushViewController:vc animated:YES];  
[vc release];
```

Figure 53: Make view controller visible

This piece of code sets up the view controller and initializes it. The init command ties it to the XIB file to set up the GUI components elements. The self.navigationController pushViewController:vc animated:YES command pushes the view controller onto the screen and passes the animated Boolean of YES which is equal to True in Objective C to make sure that the animated images between views is smooth and not jagged or grainy. The vc release part of the code gives up the view controllers local copy and puts the screen onto a stack. If the group wanted to clear the screen the group would then just have to make a call to the instance method of “popViewControllerAnimated” which would pop off the navigation controller for that screen. This will then return the screen to either the main screen or wherever the method points it to.

For text and other fields that will need to populate by the user, this last way of pushing code onto the screen will not work for . If the group did run it with the last method inputting the connection name and password would take up the whole screen and leave no room for the user to see the rest of the screen they were currently on. To fix this, the group used another method to push the view onto the screen. The method presentViewController instance method provides a quick and easy way to bring up a new screen to set up configurations and other options of that nature that do not need a full page all to themselves. An example from the project would be when the user clicks on the connection name or password text fields, a keyboard will appear with a textbox on top of the keyboard along with a “done” button. The user will enter the information using the keyboard and then when finished putting in the necessary information the user would just have to bit the “done” button and the temporary view of the keyboard and textbox will disappear and the information that the user has typed would be saved in the textbox the user had previously selected. This type of view would be have to be called from a parent view controller because modal views which is the type of view that the keyboard works on, are considered to be children to the main view of a screen.

A feature so that the user would not have to remember the connection name and password every time, the group will have the fields pre-loaded with values that

the last user had used to make the user's entry simpler in the case it is the same returning user. The OS for the iPhone allows the applications to save files in their own directory in the address <Application_Home>/Documents folder. In here the files can be saved and they will not lose their contents once when the application is exited. This gives the application a way to save information that is outside of its source code. The group used a property list file to store the connection name and connection password. Property list files are similar to XML files in being that they only store basic type of data such as strings, integers, dates and arrays. The group are able to load this file into the NSDictionary where the group have easy access to read and write from this file. After the group have loaded the information of the file into the dictionary, it is easy for to re-write the property list and save it in the applications document folder.

The property list file that the group will save in the applications/Documents folder will contain only 2 different values. The first value will be the connection name and the second value will be the connection password. The file values will be read in by line to make the distinction between the values as clear as possible. The path of the document will be loaded by the application by the use of the NSSearchPathForDirectoriesInDomain method that is provided by the framework of the Apple's Foundation. To use the method, you input the parameter for the documents folder and it will in turn give out the path to the documents folder. After having the path returned, the group can then in turn see if the property list file exists and if it does not have the file the group then create it. Once the file is located or created, the data gets loaded out of the file and back into the dictionary.

One of the main features of the quad-copter application is the video playback for the video that is being streamed from the copter to the iPhone. The movie player is available due to the Apply Foundation's Framework and is under the class MPMoviePlayerController. The movie player class gives the flexibility to stream a video from a URL and play it. The video must be in an mp4 format for the MPMoviePlayerController to be able to play the video. Once when the video is loaded the group can simulate the user pressing the play button to start playing the live video from the quad-copter.

The movie player normally takes over the whole screen of the iPhone when the video begins to play. To control the copter the group need to be able to overlay the sliders on top of the video to give the user the necessary the ability to move the copter in all directions including rising and falling. To overlay the sliders onto the movie player the group need to capture the window that is playing the video. To do this the group grabs the "keyWindow" which is always the window currently in view.

Once the window is grabbed, the group now has the ability to add the sliders to the frame. The sliders were created using the UISliders class which is created in the XIB file and the group has the necessary references to the left, right and

bottom sliders. The left and right hand side sliders need to be positioned on the top and bottom side of the iPhone since the screen is launched in landscape mode due the movie player. The group then needs to position the bottom slider for the height to the left hand side of the screen to make the slider appear as if it was on the bottom of the screen in landscape mode. The code snippet below allows positioning of the sliders.

The code below shows that the group declares a CGRect frame, which is used to be the variable to modify where the sliders are placed in the window. The SliderGap is the space needed to give the slider the amount of room necessary to fit within the screen comfortably. Once the group obtains the amount that the group wants to have the gap for the slider, the group set the leftSliderOverlay's frame to the modified frame that the group had just changed. The right slider is undergoing the same procedure as the left slider with the only difference being that the group wants to have the slider on the opposite side of the screen. To do this the group has to subtract the height of the iPhone's resolution in pixel's which is 480 pixels, in addition to the SliderGap and the height of the slider. For the bottom slider, the group set the frame to the gap that the group specified for the bottom slider with xSliderGap. The difference with this slider is that the group use the x axis as the point of reference instead of the y axis as the group did with the first two sliders.

```
CGRect frame;  
  
frame = [leftSliderOverlayframe];  
frame.origin.y = SliderGap;  
leftSliderOverlay.frame = frame;  
  
frame = [rightSliderOverlayframe];  
frame.origin.y=480–SliderGap–rightSliderOverlay.frame.size.height;  
rightSliderOverlay.frame = frame;  
frame = [bottomSliderOverlayframe];  
frame.origin.x = xSliderGap;  
bottomSliderOverlay.frame = frame;
```

Figure 54: UISlider set up

Once the group have modified the sliders to what are needed, the sliders now need to be over laid on top of the movie player. To do this the group just need to do one simple method call for each slider called addSubview. This takes the overlay the group just created for each of the sliders and applies it to the main window. The final format of the screen will have the movie player in the back with 2 sliders on either side of the screen and one slider on the bottom of the screen.

The code snippet below shows how the addSubview method is called and applied to the main window.

```
[moviePlayerWindow addSubview: self.leftSliderOverlay];  
[moviePlayerWindow addSubview: self.rightSliderOverlay];  
[moviePlayerWindow addSubview: self.bottomSliderOverlay];
```

Figure 55: Movie Player Class

Another element that was used from the UI library to better control the quad-copter application is the UIButton to have the quad-copter land and take off. This feature is a crucial step in making the quad-copter an easy to use device by taking the difficulty of having to take off and land out of the user's hand. The button will be between the left hand side of the bottom slider that controls the height of the copter and the slider that controls the left movements of the plane. This button also had to be overlaid on top of the movie player and used the same method as the sliders, the addSubview method. Figure 56 is how the preliminary view of the application will look like when the user is flying the copter.

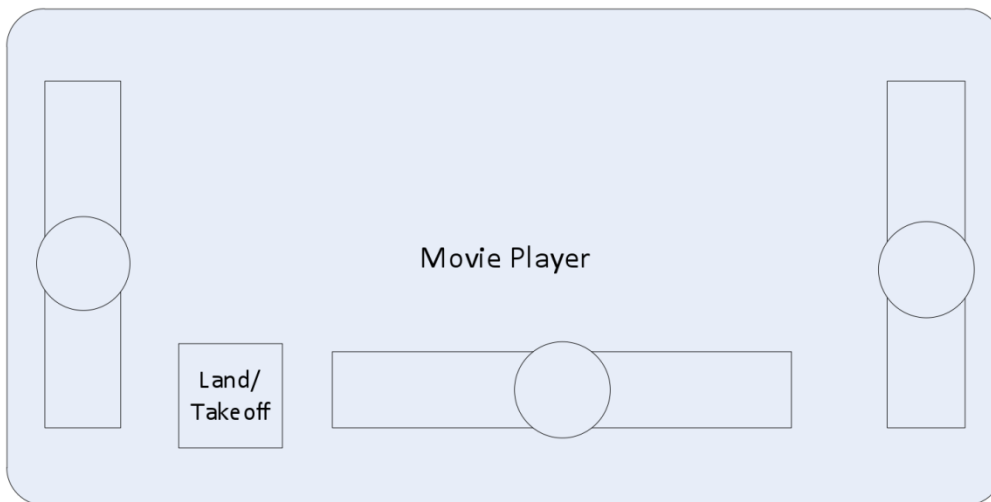


Figure 56: GUI Interface for the Main control screen

Another element used from the UI library to alert the user better is the UIAlertView. This element was useful in letting the user know when an error had occurred or another important event had taken place. The UIAlertView makes a message box pop up that can give the user a message that the user has to read because the class blocks the rest of the screen. You can see from Figure 46 that the user can no longer touch any other part of the screen besides the alert message that is displayed.

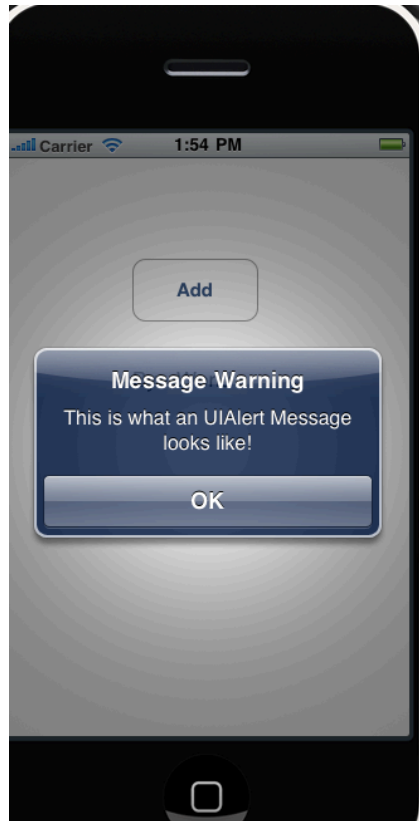


Figure 57: UIAlertView example

To implement this feature, the group had to create an `UIAlertView` object and then create an `init` method which can then run the entire Alert message from the `init` method. The group would be able to set the title, message and the visible properties from the `init` method as well without having to create many different methods to do all of these variable settings. After all the variables have been set up the group can then display the message by calling the `show` method that is built into the `UIAlertView` class. This message was used to indicate if the connection between the iPhone and copter cannot be connected, or if the connection is lost or if the data coming between the copter and the iPhone is too corrupted and forcing itself to terminate the connection.

Another UI element used from the UI library that gave the application a better look and feel is the `UIActivityIndicatorView`. This was a class that just helped make the application look like something is going on while the iPhone is trying to connect with the quad-copter. The class shows a an activity meter that is spinning to show that work is being done. To create the activity indicator, you had to declare an object an `UIActivityIndicatorView` and then call its predetermined `init` method called `initWithActivityIndicatorStyle`. The group then had to place the object on a view. After it was created and been placed on a view, the group was

able to start and stop the indicator by calling the two functions, `startAnimating` and `stopAnimating`. This class was used whenever the iPhone application is trying to connect with the quad-copter in the project.

This wraps up all of the design that was included for the iPhone application. To design the iPhone application though the group needed to have the proper tools to build and design and implement the code. The only way to build an iPhone application is by using XCode and the iPhone SDK that Apple has the rights to. To gain these valuable resources, the group had to sign up with apple and pay a \$99 developer fee. This fee has to be paid for every year that you want to design and put applications on the iPhone. The developer fee gave a lot of information and useful resources to help the development and also it gave the rights to be able to put the quad-copter application onto a physical iPhone.

10.17 Design Summary

The design of the quad-copter is broken down into several main components. Each component has its own purposes and is vital for the overall performance for the quad-copter. The iPhone is important for the design of the copter for it gives the user the control to fly the copter and see where the copter is and heading without always having being in the direct line of sight of the quad-copter. The Wi-Fi transceiver chip on the quad-copter is an integral part of the copter. It is responsible for connecting to the iPhone interface and receiving and transmitting the necessary information through the wireless connection. The transceiver chip must make sure that the data coming through between the two, iPhone and copter, be is genuine and in the correct format so that the copter does not lost control from the user or that an incorrect signal be sent and send the copter in the wrong direction. The back bone of the design is the processor on board of the quad-copter. All the data from the system will be sent to the processor so it can interpret the data and spit out the necessary information. This receives the information that's coming from the iPhone through the wireless chip. It also receives information coming from the IMU and the camera module located on the quad-copter. And lastly it receives information coming from the power sensors from the battery and the motor controllers. The processor must process all of this data and then in very timely manner output information going to the iPhone through the wireless transceiver chip and send the necessary power regulations to the motor controllers. The IMU is another important part of the design for it must be able to locate where the quad-copter is located in relation to its last position and figure out if it is upside down or has moved since its last update. The IMU make sure the copter stays steady and does not wobble. It sends all of the information that it gathers on over to the processor. The camera module on the quad-copter is one of the most important features from the user's interface aspect for it allows the user to see the copter without having to be in direct line of sight. The camera has its own image processing unit (IPU) to process the image that is coming from the sensor and put the video into the mp4 that the group

needs for the iPhone movie player class. The last part of the design is the motors and motor controller. This part of the copter is important for the entire design for it is what actually gives the copter its ability to fly. The motor controller takes in the outputs from the processor and changes the motor speeds for each motor according to the data that the processor fed to it. The motors in turn then change according to the outputs from the motor controller.

11.0 Testing

11.1 Testing the Vehicle Body

Once the parts for the body are procured, the assembly will be divided into a series of steps. Each step will be constructed, tested, evaluated, and then either accepted as satisfactory or redone with the necessary changes. The body will be one of the last parts to really be tested, as all other parts must work first. The first test will be to determine that the motors are functional. Power will be applied to each unit to ensure it turns on. Note that testing the reaction of the motors to various stimuli is part of the power and feedback testing. Next, the correct placement of the motors on the arms will be determined. Two motors will be placed across from each other at a distance of twenty-five inches. The motors will be controlled by the electronic speed controllers. The test will prove successful if the feedback loop is known to be working and if the motors balance each other. The aircraft will be tethered to ensure the safety of the people nearby as well as protect the structural integrity of the body. If the test is not successful, the motor location should be adjusted. Each motor must be adjusted separately after multiple tests. Changing too many variables at one time can lead to frustrating circles of operation. If the motors are not balancing properly despite multiple adjustments, testing should be continued on the feedback loop.

The next test requires the feedback loop to be operational for the motors in the X formation. The landing base should be attached and tested for sturdiness. A controlled flight with a soft landing will demonstrate the design is sufficiently sturdy. Now that the landing gear is attached, all parts must be secured to the platform and attached to the center of the craft. Weight should be adjusted until the quad-copter flies smoothly. The only thing remaining is to further protect the boards on the platform. Test all additions to make sure nothing changes for the worst.

Failure to produce the expected results in the x formation will result in the implementation of the plus setup. The plus arrangement is easier to test as each motor across from the other offsets in a cardinal direction as referred to the board. Testing can therefore be done on two motors at a time, and if a motor is causing instability, the guilty party will be easier to identify. A change to the plus shape will require the board to be remounted to the body and the weight must

once again be centered for an acceptable center of gravity. Most other attachments will remain the same.

11.2 Motor Testing

The main concern when testing motors is current draw. The first test requires a current meter to be put in series with the power supply to verify the motor is using the correct amount of power. The tests will begin with no load attached to the motor and will continue with the propeller attached. Care must be taken when the propeller is attached to avoid injury. Testing will begin with the smallest propeller suggested by the manufacturer and will be increased until the current readings begin to get within ten percent of the maximum value. The minimum propeller suggested is 7035 and the maximum advised is 9047. The test will then shift to ability to fly. The last propeller to be under the designated value will be used to see if the thrust is sufficient to lift the quad-copter. The unit will be tied down loosely with rope to avoid any harmful accidents. If the test fails, larger sized propellers will be tested with the current under close monitoring. Several sites do suggest using propellers that cause the current draw to go over the intended value specified by the manufacturer. If the motors lift the vehicle with ease, the propeller size may be decreased in order to conserve power. Once the current levels are found to remain constant and safe to all people and equipment, the motors are considered to have passed the first functionality test. Motors can burn out quickly, especially under strenuous conditions, so tests should be repeated if someone suspects the unit of inaccurate operation. The next test for the motor is to check that the speed decreases and increases appropriately with manual changes to the power supplied. Power supplied will be changed by using a potentiometer to change the current running through the system.

11.3 Power Testing

Power must be the first element of the system to be tested. Problems with incorrect loads being applied to various components can easily destroy the system. Initial measurements must be made before any components are attached. The group will first test the charge on the battery by plugging it into the charger. Once the battery is fully charged, the voltage across the terminals will be measured with a voltmeter. Any problems with the battery will be fixed using the balance feature of the charger. If an issue cannot be corrected properly, a new battery will be purchased. When the group is satisfied that the battery is functioning correctly, it will be connected to the controller board in series with an ammeter to see the current draw. The voltage on the output of the linear regulator will also be measured to ensure it is at 3.3 V. If the current values read match the theoretical values expected by each group member, the board can be connected to an ESC and the ESC will be calibrated. If the current draw is too large, the board will be inspected for shorts. The motor will be attached next with

no propeller mounted. The current will once again be monitored by an ammeter in series and the ESC will be used to regulate the speed of the motor. Propellers will finally be added when the system is tied down for safety purposes, and the current being drawn by the motor will give a good estimate of the battery life that can be expected. The current with 8x3.8 slow flyer propellers installed should be no more than 8 amps. If all of the above criteria are met to the group's satisfaction, the power ought not to be a problem and further error lies with the individual components.

11.4 STM32 Testing

11.4.1 Electronic Modules Testing

The development board extension pin headers will be connected to the other modules by wiring the individual header pins to the corresponding pins on the other modules. The wiring will be routed through the use of a breadboard. The oscilloscope will be used as the first step in troubleshooting connections between the development board and the other modules if the connections are not operating as expected.

11.4.2 STM32 Firmware

The testing of the firmware on the STM32 will first involve writing the code for a function and testing and debugging the function on the development board until the function does its task correctly. The IMU code will be tested by first connecting the IMU to the development board. Then the IMU subroutine code will be loaded onto the STM32 on the development board. Next the OpenOCD debugger will be started from the Eclipse IDE and the memory locations where the IMU values are stored will be observed. The subroutine will be known to work correctly if when the IMU is moved or rotated the memory values change appropriately. The PWM subroutine will be tested by first connecting the advanced timer channel pins on the development board to an oscilloscope. The next step of the test is using the debugging software on the computer to change the duty cycle on each PWM output and checking that the period and duty cycle of the PWM signal appear correctly on the oscilloscope. If the initial PWM testing is successful then the PWM signals will be connected to the ESCs to test if they cause the appropriate changes in motor rotational speed when the PWM duty cycle is changed. The SPI communication firmware will be tested at the same time the Wi-Fi is to be tested. The testing will involve connecting the development board to the Wi-Fi module and uploading the SPI code to the STM32. The test will involve sending data from the STM32 to the iPhone. The test is successful if the data gets transmitted correctly to the iPhone. Once all the firmware is tested and operates correctly it will then be flashed onto STM32 that

is mounted on the control board.

11.4.3 Control Board

To test the control board for proper operation the equipment available in the senior design lab as well as testing equipment that the group will acquire will be used. The pieces of equipment that will be used the most are the oscilloscope and the multimeter. The control board PCB will be tested when it arrives from the fabrication company by first visually inspecting the PCB for any obvious defects. Every discrete part that will be placed onto the control board will be tested to operate correctly before being soldered onto the control board. The simple parts being the parts that are not IC chips like the capacitors and switches. The simple parts will be retested once soldered onto the PCB to make sure they are still operating correctly. Once all the parts are soldered onto the control board PCB the control board will be powered on to write the firmware onto the STM32. The JTAG connector on the control board will be connected to a PC in order to write the firmware to the control board. The final test of the control board and firmware will be picking up the control board and tilting and rotating it in all directions so that the board generates correction signals to the motors via PWM to try to get back to a level orientation. The signals coming out of the control board will then be observed on an oscilloscope to make sure that they are changing in a way that will cause the control board to become level and to make sure that the period of the PWM signals are not fluctuating and are staying constant. The control board will then be mounted onto the mounting platform on the quad-copter frame and be connected to the 4 motor ESCs, the camera, and the power supply to begin flight testing of the quad-copter using the iPhone as the source for navigation commands.

11.5 IMU Testing Methods:

Testing of the IMU is necessary to verify the proper functionality of the sensor measurements and DCM algorithm. There are several areas of the IMU that need to be tested to verify correct functionality. Testing for the IMU will consist of several parts:

1. Sensor Testing: Testing the functionality and proper measurements of the sensors. This testing is necessary to verify the measurements produced by the sensors to verify that the components are operating correctly.
2. IMU Board Testing and Verification: Testing and verification of the IMU PCB, essentially this will be continuity checks of the PCB. The testing will be concluded by soldering the chips to the IMU PCB and testing the connections.
3. IMU Testing: Testing of the complete IMU with the DCM algorithm. Essentially this testing will consist of connecting the IMU to the STM32

microprocessor and verifying the functionality of the design of both the IMU hardware and algorithm.

11.5.1 Sensor Testing

Preliminary testing will be conducted once the sensors are ordered and have subsequently been received. This testing will involve two phases. The preliminary design will be done with development sensor boards to verify functionality. Testing of the development sensor will be necessary to verify the outputs are correct. To do this the group will power the sensor boards that have been acquired from SparkFun. A linear regulator will provide filtered 3.3V power to the board will power each sensor that is being used. The board's outputs will be connected to a digital oscilloscope to determine the waveforms that are produced by the sensors. Measurements will be taken at several different instances:

1. +/- Pitch
2. +/- Roll
3. +/- Yaw
4. Stationary

Each iteration of the test will be conducted to ensure that the device's outputs are valid within reason. The most important iteration of the test is the one evaluating the output when the sensor is stationary. This will allow to evaluate the drift that may be present in the measurements. The test will also be preformed on the other sensors; the accelerometer and the magnetometer. For both of these sensors the outputs will be evaluated against the manufactures specifications. Paying particular attention to the stationary measurements to evaluate any transients that may be present when the device is not in motion.

11.5.2 IMU Board Testing and Verification

PCB verification and testing is necessary to ensure the proper connections are present to mitigate any continuity errors that may be present prior to software testing. The continuity testing that is to be conducted will be done in two phases. Phase one involves testing the PCB by itself, with out any components soldered to the board. This testing is done strictly to test that the connections are correct and that there were no manufacturing errors or defects. The second phase of the testing will involve soldering the components to the board and more or less verify the solder connections are valid. Phase one will involve a continuity check that will involve using a multimeter that is set to measure resistance. The two probes are then set on either end of the traces that are to be evaluated. To determine the continuity the following rules are followed: if the multimeter reads an infinite resistance then there is an open. If the multimeter measures a resistance other than infinite then the connection is valid.

11.5.3 IMU Testing

IMU testing is necessary to verify device design and coding methodologies that were utilized. As with the other aspects of testing that has been discussed there are multiple phases. The first phase consist of testing and simulating the software that is being utilized to orient the aircraft. Mathlab's simulink has been chosen as the tool of choice to simulate the software's feedback control loop. This will also aide in the design process by testing and subsequently modifying the system. Simulink allows the user to specify the plant that is being used to model the system and test the frequency response of the modeled system. This will greatly aide in optimizing the system. The PI controller that was discussed in the design section utilized coefficients for the proportional and integral terms. These terms may be derived theoretically and tested using simulation. The testing methodology that is going to be used is called the Ziegler–Nichols and is used to derive the PI terms are as follows. First the group will derive the coefficients by using simulink to acquire the frequency response of the P term, as it is adjusted. The group then continuously adjust the proportional term until it has reached its maximum gain, which corresponds to oscillation at the output. This point of maximum gain can be used to derive the integral term through the following equation.

$$K_p = 0.5 * K$$

$$K_I = 1.2 * \frac{K_p}{T}$$

Where...

K = Ultimate Kp gain

T = Period of oscillation at K

Upon selecting the proper coefficients for the PI controller, The software will be evaluated in two phase, phase one will utilized the STM32 development board that was purchased, phase two will be utilizing the IMU sensor board and the Control board. Software testing in both phases will be conducted in a similar fashion. The software will be loaded onto the board, development board and designed board, the sensor board will then be subjected to variations in the pitch, roll and yaw of the board. Movements will be done both in positive and negative directions to verify measurements. The simulation will store the DCM information in an output file that will be used by Matlab to determine the orientation of the sensor board and the corresponding development or control board. The output that is created by the DCM will be displayed in graphical format in Matlab to provide an easy to verify output. An M-file that is to be determined will manipulate the data produced allowing to experiment with different ways of representing the data.

11.6 Complete Design Testing

Further testing is also necessary to verify that the entire system is able to function. This testing will be conducted in two phases. Phase one will consist of testing the system in a controlled environment and only testing two axes of the system at a time, this is done to mitigate damage that can occur. The quad-copter will be restrained during this phase of testing. Several methods of restraint have been devised in years past to allow for testing of the control algorithm and the rest of the quad system. The version will consist of operating the system with only two motors connected at a time to test the stabilization of a single axis. This will mitigate damage to the system, while allowing use to verify each axis. Noted is the absence of the ability to test all axis together, this will be conducted in the second phase of complete design testing. A single axis of the helicopter will be tethered to a test platform that is depicted in the figure below.

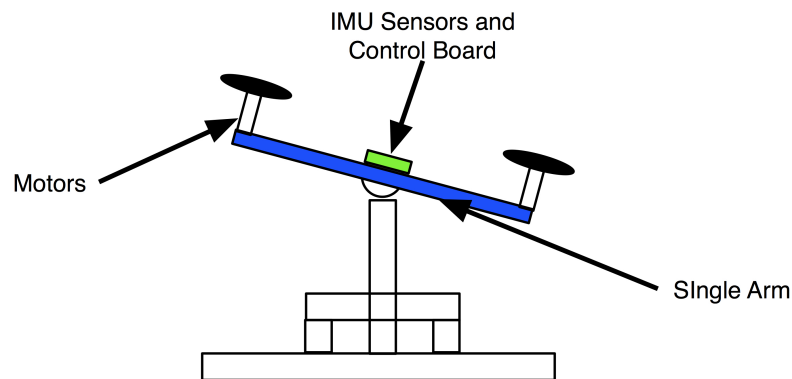


Figure 58: Single axis Testing setup (Conceptual Drawing. Not to scale)

The design will then be subjected to a rotational force on either end of arm and will be allowed to self-stabilize. The rotational force will simply be motion generated by tapping on either end of the arm. This method of testing will be done with each axis of the helicopter to verify pitch and roll.

Testing of the complete design is necessary to verify the stabilization algorithm and the circuitry that is designed prior to the final demonstration. There will be a series of two complete design tests that will be conducted to allow for corrections in the design. Each test will be conducted in the same manner and will be done outside in an open field. The test will consist of: initializing the system, bringing the system to a hovering level to 5 feet and performing maneuvers to verify the stabilization.

11.7 Communications Range:

This test will show how far the quad-copter can go while still being under the control of the User. The test will include both an optimal line of sight location and

one with many obstructions to figure out the accurate amount of range the copter has to fly around the user. For the optimal location the group will use the school parking lot where there is little to no interference because of trees or buildings. For the location with obstructions the group will go to the middle of campus of the University of Central Florida where there are many buildings and trees that will interfere with out signal. The copter will have a feature that when the copter goes out of range of the iPhone it will begin to hover. From the point where it begins to hover the group will then record how far the copter is from the user in feet. This test will be done multiple times to ensure that the group have an accurate range for the quad-copter.

11.8 Software:

To test that the software is working properly on the system the group will think of the system in two parts: the microcontroller code and iPhone application. Both parts have their own functions that they need to accomplish for the project to be working.

The iPhone application will be considered working if it can control the quad-copter, display to the user the video feed from the copter and display the information that the copter is sending to the iPhone. The iPhone application should allow the user to make the copter rise and fall, land with a touch a button on the iPhone and also take off, move the copter in any direction with a touch based interface. The user should also be able to control the speed of the copter in the direction it goes as well. To test that the copter can land with a press of a button on the iPhone the group will fly the copter around for a few minutes and then at predetermined times the group will stop moving the copter and press the land button. If it lands in a timely manner and does it repeatedly when the button is pressed, the landing button will be considered working.

The take off button will be used when the copter is resting on the ground and will make the quad-copter rise to a certain elevation and begin hovering. To test the video feedback is working in a timely and responsive manner the group will have the copter in a crowded air space were the user can control the copter and try to navigate through the air space by just looking at the video screen on the iPhone that is displaying on the iPhone user interface.

11.9 User Interface:

The user interface test will see how well the multiple users can effectively and easily use control the quad-copter. The group will have a sampling group of multiple people that will try and control the copter from the iPhone. The users will then have to respond to a survey about how the copter flew and how easily it was to control the copter from the iPhone. From the surveys the group will then fine tune the user interface and the necessary controls to make the copter easier

control for the general public.

Name:
Age:
Gender:

Description/Identification of Survey Item	P o o r	Scale			E x c e l l e n t
		Good			
How would you rate the control sensitivity of the rise and falls control?	1	2	3	4	5
How would you rate the control sensitivity of the left and right controls?	1	2	3	4	5
How would you rate the display?	1	2	3	4	5
How would you rate the overall user interface?	1	2	3	4	5

Figure 59: Rating Card for the user interface

11.10 Cold Start Test

The cold start test will evaluate the amount of time it takes for the copter and the iPhone to be able to function. To get this time amount, the quad-copter will remain off for an extended amount of time. After the copter has been turned off for a few hours the group will then power up the copter and see and record how long it takes for the copter to become functional with the iPhone user interface. This test will be done several times to get an accurate account for the start up time.

11.11 Video Stream Delay

This test will determine the time difference between when the copter records the video and when the user sees the video on the iPhone. To perform this test the group will synchronize 2 clocks and place one of the clocks in front of the camera on the quad-copter and the other clock in front of the user looking at the iPhone user interface. To measure the amount of delay for the system you subtract the time on the clock on the video display from the clock that is right next to the user with the iPhone. This will give you the video delay for the system. This test will be done while the copter is stationary on the ground, while hovering and also while traveling in a vector away from the iPhone. This test will be performed many

times to achieve an average time for the video delay for the three different scenarios.

12.0 Budget

Table 17 is a list of components that the quad-copter will need. The list also includes supplies needed for designing, building and configuring the copter throughout the design and building stage. Also included is the amount for extra back up parts that will be needed in case of other parts breaking and finding the right configuration for the project needs. The total cost projection for the prototype of the project is \$531.37.

Components	Estimated Quantity	Estimated Price per Unit	Estimated Cost	Actual Quantity	Actual Price	Total Actual Price
Camera	1	20	20	1	39.43	39.43
GPS device	1	35	35	0	0	0
magnetometer	1	5	5	0	0	0
IMU	1	75	75	1	74.95	74.95
4400 mAh battery	1	30	30	1	26.79	26.79
610 mAh battery	1	10	10	1	3.92	3.92
Battery charger	0	0	0	1	23.99	23.99
Battery Sensor	0	0	0	1	2.10	2.10
microcontroller	1	40	40	1	0	0
Header Board	0	0	0	1	40.95	40.95
Break away Headers	0	0	0	1	2.50	2.50
IPhone	1	0	0	1	0	0
Motors	4	15	60	4	7.99	31.96
power sensor	1	5	5	1	6	6
body	1	40	40	1	12.99	12.99
PVC	1	1	1	1	1.04	1.04
Blades	4	12	48	3	9.07	27.21
Electronic speed controllers	1	30	30	4	6.50	26
Wireless Transceiver	1	15	15		49	49
Apple computer	1	0	0	1	0	0
IPhone SDK	1	99	99		99	99
Shipping	0	0	0	1	63.96	63.96
Total Projected cost			513	Total Cost		\$531.37

Table 17: Budget

Figure 60 shows how the cost for each item differed between the estimated values and the actual prices that the group paid for. The group estimated the project to cost \$513 dollars and the price that the group have come up with is \$531.37. Judging from the difference between the numbers, the group believe that the group had budgeted the project wisely and used the money as effectively as possible without any waste to unnecessary items.

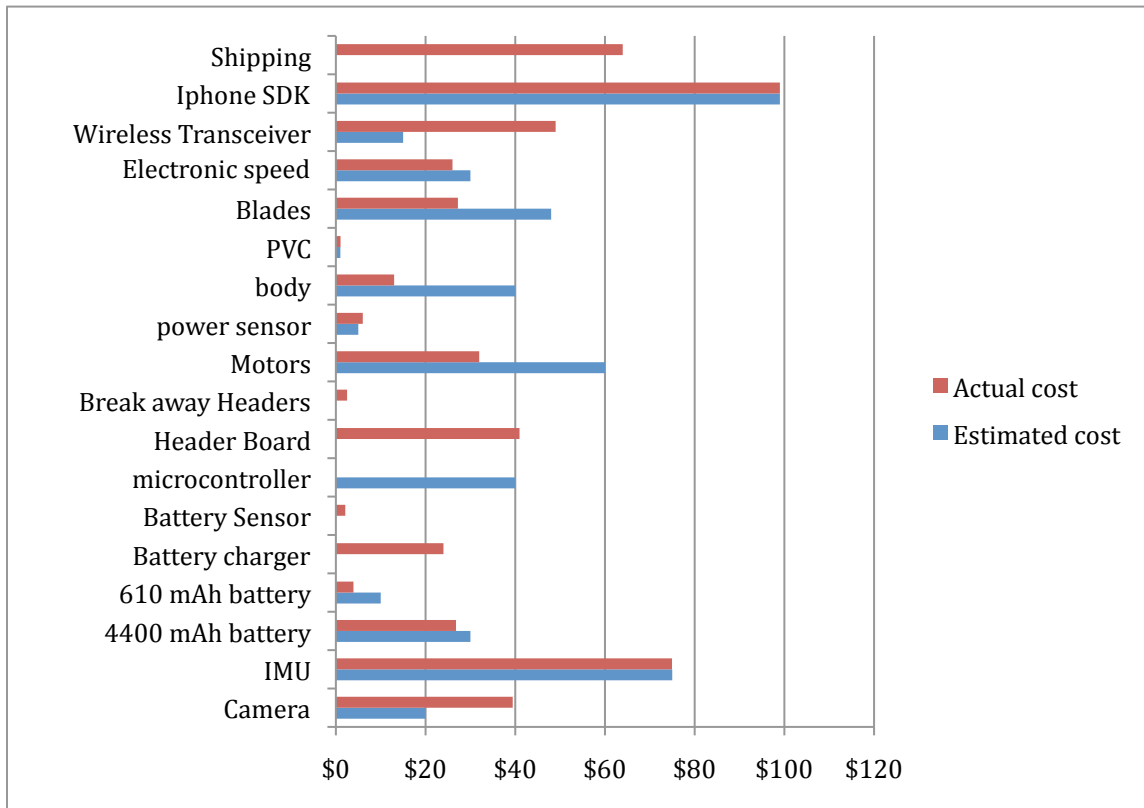


Figure 60: Individual Item Difference

Appendix A: Works Cited

- [1a] "Specific Strength." *Wikipedia, the Free Encyclopedia*. July-Aug. 2010. Web. July-Aug. 2010. <http://en.wikipedia.org/wiki/Specific_strength>.
- [2a] Julio, Jose. "ArduIMU Quad-copter Part II (mini)." *DIY Drones*. 24 Feb. 2010. Web. July-Aug. 2010. <<http://www.diydrones.com/profiles/blogs/arduimu-quad-copter-part-ii>>.
- [3a] Hepperle, Martin. "How a Propeller Works." *Willkommen / Welcome*. 8 Sept. 2003. Web. July-Aug. 2010. <<http://www.mh-aerotoools.de/airfoils/propuls4.htm>>.
- [4a] McKinney, Martyn. "Prop Rpm vs Thrust Chart - RC Groups." *RCGroups.com: The ABCs of Radio Control - Aircraft, Boats, and Cars!* 21 May 2005. Web. July-Aug. 2010. <<http://www.rcgroups.com/forums/showthread.php?t=232300>>.
- [5a] Joe. "Propeller Basics." *AmpAviators | Unbiased Information on All Aspects of Radio Controlled Electric Planes*. Amp Aviators. Web. July-Aug. 2010. <<http://www.ampaviators.com/2007/05/15/propeller-basics/>>.
- [6a] Ays, Feliks. "AeroQuad." *Your Friendly Neighborhood Quad-copter Platform* (2010). Print.
- [7a] "A Tutorial on Electrical Motors." *Welcome to STMicroelectronics*. ST. Web. July-Aug. 2010. <<http://us.st.com/stonline/products/support/motor/tutorial/tutorial.htm>>.
- [8a] "Brushless vs Brushed Motors." *High Quality, Precision Solutions for Motion Control Applications*. Dynetic Systems. Web. June-July 2010. <[http://www.dynetic.com/brushless vs brushed.htm](http://www.dynetic.com/brushless%20vs%20brushed.htm)>.
- [9a] Kirsch, Matt. "Outrunner v. Inrunner Motor." *RC Universe Features Rc Cars, Rc Airplanes, Rc Helicopters, Rc Electric Planes, Rc Boats, Radio Control Jets, Rc Discussion Forums, Rc Classifieds and Auctions*. 15 Mar. 2006. Web. July-Aug. 2010. <http://www.rcuniverse.com/forum/m_4033256/tm.htm>.
- [10a] "All Outrunner Motors." *HobbyKing Online R/C Hobby Store*. Web. July-Aug. 2010. <http://www.hobbycity.com/hobbycity/store/uh_listCategoriesAndProducts.asp?catname=All+Outrunner+Motors&idCategory=63&ParentCat=59>.

- [11a] Buchmann, Isidor. "What's the Best Battery?" *Welcome to Battery University*. Cadex, Nov. 2006. Web. July-Aug. 2010.
<<http://www.batteryuniversity.com/partone-3.htm>>.
- [12a] "Battery Eliminator Circuit - RC Helicopter Wiki." *Main Page - RC Helicopter Wiki*. 26 Nov. 2009. Web. July-Aug. 2010.
<http://www.rcheliwiki.com/Battery_eliminator_circuit>.
- [13a] "Electronic Speed Control." *Wikipedia, the Free Encyclopedia*. 13 June 2010. Web. July-Aug. 2010.
<http://en.wikipedia.org/wiki/Electronic_speed_control>.
- [1] Premerlani, William, and Paul Bizard. *Directional Cosine Matrix IMU: Theory*. Tech. DIY Drone, 17 May 2009. Web. 15 June 2010.
<<http://diydrones.com/profiles/blogs/dcm-imu-theory-first-draft?id=705844%3ABlogPost%3A77893&page=6#comments>>.
- [2] Bak, Thomas. *Lecture Notes Modeling of Mechanical Systems*. Tech. Aalborg University, 19 Feb. 2002. Web. 15 June 2010.
<<http://www.control.auc.dk/~jan/undervisning/MechanicsI/mechbook.pdf>>.
- [3] Welch, Greg, and Gary Bishop. *An Introduction to the Kalman Filter*. Tech. Department of Computer Science, University of North Carolina at Chapel Hill, 24 July 2006. Web. 20 June 2010.
<http://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf>.
- [4] Stengel, Robert. "Dynamic Systems." Lecture. MAE345 Lecture Notes. *MAE345 Lecture Notes*. Princeton University. Web. 20 June 2010.
<<http://www.princeton.edu/~stengel/MAE345Lecture6.pdf>>.
- [5] "A Guide to Using IMU (Accelerometer and Gyroscope) in Embedded Applications." Web log post. *Starlino*. 29 Dec. 2009. Web. 17 June 2010.
<http://www.starlino.com/imu_guide.html>.
- [6] "Direction Cosine Matrix." *PlanetPhysics*. 25 Aug. 2005. Web. 15 June 2010.
<<http://planetphysics.org/encyclopedia/DirectionCosineMatrix.html>>.
- [7] Euston, Mark, Paul Coote, Robert Mahony, Jonghyuk Kim, and Tarek Hamel. *A Complementary Filter for Attitude Estimation of a Fixed-Wing UAV*. Tech. The Australian National University. Web. 25 June 2010.
<http://engnet.anu.edu.au/DEpeople/Jonghyuk.Kim/pdf/2008_Euston_iros_v1.04.pdf>.
- [8] "Magnetometer." *Wikipedia, the Free Encyclopedia*. Web. 01 July. 2010.
<<http://en.wikipedia.org/wiki/Magnetometer>>.

- [9] "PID Controller." *Wikipedia, the Free Encyclopedia*. Web. 01 July. 2010.
<http://en.wikipedia.org/wiki/PID_controller>.
- [10] Hua, Wensheng, Dan B. DeBra, Corwin T. Hardham, and Brian T. Lantz.
Polyphase FIR Complementary Filters for Control Systems. Tech.
Stanford University. Web. 28 June 2010.
<<http://www.ligo.caltech.edu/docs/P/P040020-00.pdf>>.
- [11] *AeroQuad Forums - The Front Page*. Web. 08 June 2010
<<http://www.aeroquad.com/>>.
- [12] "ATMega328 Data Sheet" Technical Literature. *Atmel Corporation*. Web.
27 June 2010.
<http://www.atmel.com/dyn/resources/prod_documents/8271S.pdf>.
- [13] "dsPIC30F4011 Data Sheet" Technical Literature. *Microchip Corporation*.
Web. 27 June 2010.
<<http://ww1.microchip.com/downloads/en/DeviceDoc/70135F.pdf>>.
- [14] "STM32F103x8/B Data Sheet" Technical Literature. *STMicroelectronics*.
Web. 28 June 2010.
<<http://www.st.com/stonline/products/literature/ds/13587.pdf>>.
- [15] "STM32F103x8/B Reference Manual" Technical Literature.
STMicroelectronics. Web. 29 June 2010.
<<http://www.st.com/stonline/products/literature/rm/13902.pdf>>.
- [16] "STM3210B-EVAL User Manual" Technical Literature. *STMicroelectronics*.
Web. 08 July 2010.
<<http://www.st.com/stonline/products/literature/um/13472.pdf>>.
- [17] "STM32-H103 User Manual" Technical Literature. *Olimex LTD*. Web.
08 July 2010.
<<http://www.olimex.com/dev/pdf/ARM/ST/STM32-H103.pdf>>.
- [18] "STM32-P103 User Manual" Technical Literature. *Olimex LTD*. Web.
08 July 2010.
<<http://www.olimex.com/dev/pdf/ARM/ST/STM32-P103.pdf>>.
- [19] "USB ARM JTAG DEBUGGER WITH FREE GCC AND DEBUGGER
SUPPORT." *OLIMEX Ltd. - Electronic Design and PCB Sub-contract
Assembly OEM/ODM Service*. Web. 30 July 2010.
<<http://www.olimex.com/dev/arm-usb-tiny.html>>.
- [21] "Hardware Design Getting Started" Technical Literature. *STMicroelectronics*.
Web. 10 July 2010.

<http://www.st.com/stonline/products/literature/an/13675.pdf>>.

[22] "Oscillator Design Guide" Technical Literature. *STMicroelectronics*. Web. 10 July 2010.

<http://www.st.com/stonline/products/literature/an/15287.pdf>>.

[23] "Standard Peripheral Library" Support Files. *STMicroelectronics*. Web. 08 July 2010.

http://www.st.com/stonline/products/support/micro/files/stm32f10x_stdperiph_lib.zip>.

"iPhone SDK Articles." iPhone SDK Articles. 25th July 2010.

<http://www.iphonesdkarticles.com/>>

iOS Reference Library. 25th July 2010.

<http://developer.apple.com/iPhone/library/navigation/index.html>>

Appendix B: Figure Permissions

Image usage Inbox | X

Angel Rodriguez Hello, my name is Angel Rodriguez and I 7:43 PM (12 hours ago)

OLIMEX Ltd Support to me [show details](#) 2:06 AM (6 hours ago) Reply

Hi
you can use the images, as far you write that these images are Copyright by Olimex Ltd
and reference our web site
Best regards
Tsvetan
- Hide quoted text -

----- Original Message -----
From: [Angel Rodriguez](#)
To: info@olimex.com
Sent: Friday, July 16, 2010 2:43 AM
Subject: Image usage

Hello, my name is Angel Rodriguez and I am currently using your STM32-H103
development board and USB-TINY JTAG to do development for my senior
design project. I am currently writing the documentation for my project and would
like to use some of the product images you have on your website in my document.
May I have permission to use the images on your website?

From: Carol BROWN
Subject: RE: Permission to use figure
Date: July 29, 2010 7:43:38 AM EDT
To: Jared Rought

Hi Jared,

Do you have a copy of the figure already? As long as you attribute the figure to STMicroelectronics, you can use it.

Best regards,
Carol

-----Original Message-----
From: Jared Rought [<mailto:jrought@me.com>]
Sent: Wednesday, July 28, 2010 6:53 PM
To: carol.brown@st.com
Subject: Permission to use figure

Hello,

I am currently an EE major at the University of Central Florida. Our senior design group is developing a quad-rotor helicopter and will be utilizing your single axis gyro, LY530AL. I would like to ask your permission to use the figure depicting the rotational axis of the gyroscope.

Thanks,
Jared Rought

Subject: Permission to Use Figure
From: Jared Rought
Date: July 28, 2010 6:49:15 PM EDT
To: pr@invensense.com

Hello,

I am currently an EE major at the University of Central Florida. Our senior design group is developing a quad-rotor helicopter and will be utilizing your dual axis gyro, IDG500. I would like to ask your permission to use the figure depicting the rotational axis of the gyroscope.

Thanks,

Jared Rought

Subject: Permission to use figure
From: Jared Rought <roughtenator@gmail.com>
Date: July 28, 2010 6:55:19 PM EDT
To: maria.tagliaferro@analog.com

Hello,

I am currently an EE major at the University of Central Florida. Our senior design group is developing a quad-rotor helicopter and will be utilizing your triple axis accelerometer, ADXL335. I would like to ask your permission to use the figure depicting the sensory axis of the accelerometer.

Thanks,

Jared Rought

Subject: Permission to use figure
From: Jared Rought
Date: August 1, 2010 10:34:49 PM EDT
To: general@electronics123.com

I am currently an EE student at the University of Central Florida. I will be utilizing the C3038 camera module in my senior design project and was wondering if I would be able to use the picture that you have on your site in my documentation.

Regards,

Jared Rought

From: general@electronics123.com
Subject: Re: Permission to use figure
Date: August 2, 2010 12:46:25 PM EDT
To: Jared Rought

Jared,

Yes you can go ahead and use the picture.

Thanks

On Sun, Aug 1, 2010 at 10:34 PM, Jared Rought <jrought@mac.com> wrote:

I am currently an EE student at the University of Central Florida. I will be utilizing the C3038 camera module in my senior design project and was wondering if I would be able to use the picture that you have on your site in my documentation.

Regards,

Jared Rought

--

Johan

Electronics123.com

Toll Free: 1-888-549-3749

Tel: (330) 482-9944

Fax: (330) 266-7307

Address: 102 East Park Ave, PO Box 21, Columbiana, OH 44408-1353, USA.

Permission to use table

Inbox | X

★ ● **Daniel Goodhew** to ar [show details](#) 4:26 AM (7 hours ago) [↩ Reply](#) ▼

Hello,

I am a student in senior design at the University of Central Florida, and I would like to request permission to use your table of common characteristics of rechargeable batteries in my project documentation. I will be sure to give credit to Cadex for the table in the documentation.

Thank you,
Daniel Goodhew

[↩ Reply](#) [➡ Forward](#)

★ **Brandon Crick** to r [show details](#) 11:28 AM (0 minutes ago) [↩ Reply](#) ▼

Hi Daniel,

That would be fine, thanks and good luck!

Brandon

- Show quoted text -

- Show quoted text -

This email has been scanned by the MessageLabs Email Security System.
For more information please visit <http://www.messagelabs.com/email>

From: Mitch Dale [mailto:mitch@rovingnetworks.com]
Sent: Tuesday, August 03, 2010 11:52 AM
To: 'John Sullivan'; info@rovingnetworks.com
Subject: RE: Permission to use figures, tables and information

John,

Of course you can. I have attached the word documents to make it easier for you.

In the course of your project if you find error or areas of the documentation that needs improvement I would be thankful for any suggestions or improvements you come up with.

Best of luck on your project. What are you building?

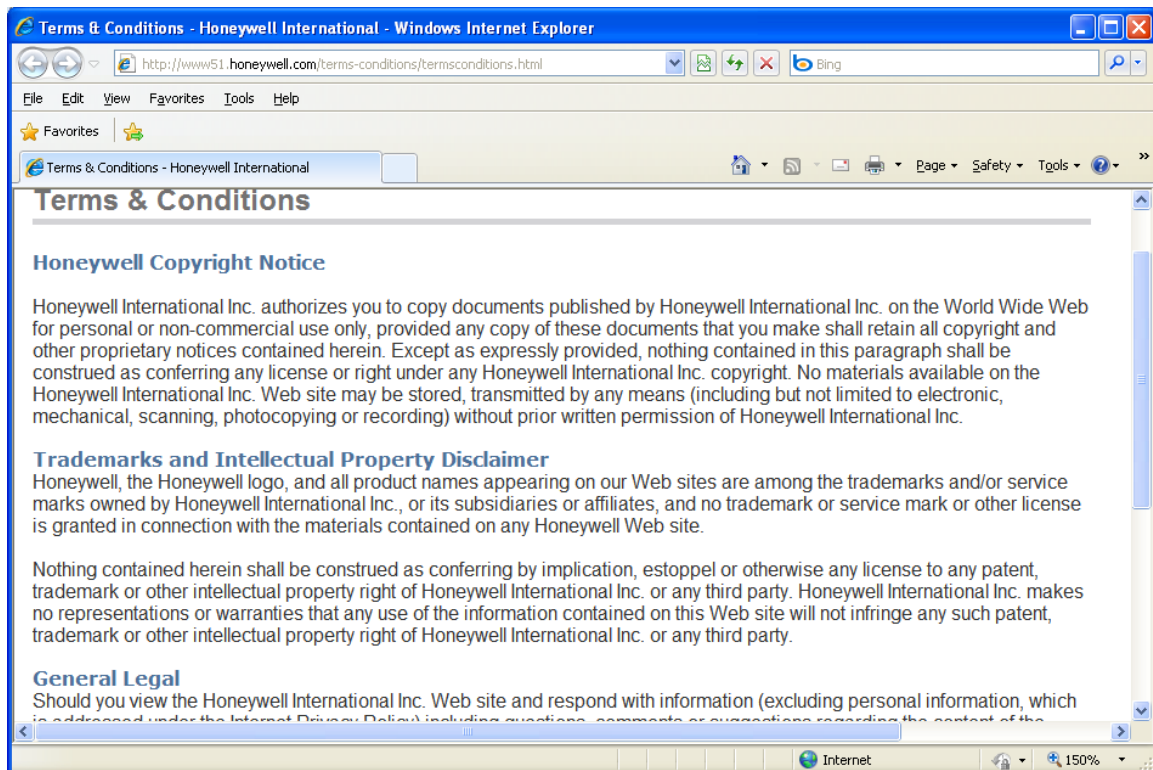
Regards,
Mitch

From: John Sullivan [mailto:johnsullivan@knights.ucf.edu]
Sent: Tuesday, August 03, 2010 7:13 AM
To: info@rovingnetworks.com
Subject: FW: Permission to use figures, tables and information

To whom it may concern,

I am a student at the university of Central Florida and am currently working on a group Senior Design Project which we will be using some of your Wi-Fi modules. For this project we need to write up a document with our research and design. We wish to ask permission to use your figures, tables and information present in your datasheets and manuals in this final report.

Thank you,
John Sullivan
johnsullivan@knights.ucf.edu



From: Beeta Ouraie <bouraie@invensense.com>
Subject: **RE: Permission to Use Figure**
Date: August 2, 2010 7:27:47 PM EDT
To: Jared Rought
▶ 2 Attachments, 7.0 MB [Save](#) [Quick Look](#)

Hi Jared,

Yes – you have permission to use InvenSense released imagery, photos and diagrams, but please use this in small print the bottom of the image:

Copyright © InvenSense, Inc. All rights reserved.

I've added some graphics for you to use for your project, as well.

Please let me know if you have any questions, thank you.

Best,
Beeta Ouraie

Jared,

Yes you can go ahead and use the picture.

Thanks

On Sun, Aug 1, 2010 at 10:34 PM, Jared Rought <jrought@mac.com> wrote:
I am currently an EE student at the University of Central Florida. I will be utilizing the C3038 camera module in my senior design project and was wondering if I would be able to use the picture that you have on your site in my documentation.

Regards,
Jared Rought

--
Johan
Electronics123.com
Toll Free: 1-888-549-3749
Tel: (330) 482-9944
Fax: (330) 266-7307
Address: 102 East Park Ave, PO Box 21, Columbiana, OH 44408-1353, USA.
From: sales.maxbotix@gmail.com [mailto:sales.maxbotix@gmail.com] **On Behalf Of** MaxBotix Inc., Sales
Sent: Tuesday, August 03, 2010 9:06 AM
To: john Sullivan
Cc: scott@maxbotix.com
Subject: Re: Permission to use figures, tables and information

Thanks for the email. Yes, you may use the information provided from our website. Please provide proper credit for any information that you intend to use.

If you have any further questions please let me know.

Best regards,

Carl Myhre
Sales & Shipping
MaxBotix Inc.
Phone: (218) 454-0766
Fax: (218) 454-0768
Email: carl@maxbotix.com
Web: www.maxbotix.com

Optimum Efficiency Graph

© 1996-2005 Martin Hepperle

You may use the data given in this document for your personal use. If you use this document for a publication, you have to cite the source. A publication of a recompilation of the given material is not allowed, if the resulting product is sold for more than the production costs.