EEL 4924 Electrical Engineering Design
(Senior Design)

Final Report

22 April 2013

Project Name: **Tail-Gator**

Team Name: **Eye in the Sky**

Team Members:

Name: **Anthony Incardona** Name: **Fredrik Womack**

Table of Contents

List of Tables and Figures

Project Abstract:

Our project is to design a remote control air plane that will be able to send back real time telemetry as well as take in flight pictures. The purpose of this plane will be to design an aerial surveillance vehicle that is capable of getting a bird's eye view of the area as well as sending back relevant information to the base station such as GPS location. The live telemetry will allow the operator to fly the plane without having an eye on it at all times and be able to know the status of the conditions that the plane is currently in when taking a photo.

The two control boards for the plane are designed, one to handle actuation of the plane and another to send back the live telemetry and control the camera. The plane will be designed with the intention of having the capability to make the RC plane become autonomous. This is outside the scope of this project but we want to design Tail-Gator with autonomous flight in mind to be added at a later date. An atxmega32A4U was used be added to interface with the radio controls from the ground station to control the servos that steered the elevator and rudder and another Xmega was used to send back the wireless telemetry and save the pictures taken during flight.

Introduction:

The Tail-Gator will be an aerial surveillance vehicle that will be able to fly via control from the ground station and take pictures on command. The pictures taken will be sent back to the base station using an XBEE radio. While a picture is being sent through the XBEEs in order to get the picture back quickly all live telemetry will be halted. This could be solved by adding another transmitter but we designed the plane with only one transmitter to simplify the first iteration of the design. During flight live telemetry will also be sent to the ground station so that the health of the plane can be viewed as well as its position when a picture is taken.

Currently the military use a similar surveillance plane to scout out an unknown territory. Ours, first of all, won't be made of Kevlar (considering it's not being shot at) and will be equipped to deliver live telemetry of the status of the plane in space. This plane will allow the user to get a bird's eye view of an unknown area so that an area can be surveyed more thoroughly. This has many applications that could be used for both recreational use and use in a situation when someone needs to see over or scout ahead in an unsafe area.

Project Features:

Tail-Gator will have a nice suite of features to allow the user to properly asses that status of the plane as well as the surrounding area in real time. Tail-Gator will provide all the information in a logical manner to eliminate the use of user processing the given information.

1. Aerial Photos:
    - When prompted by the user on the ground station Tail-Gator will take an aerial photo and send the picture back to the base station for immediate viewing. In case packet lose or the signal is distorted causing damage to the viewed picture it will also be stored on an on board micro SD card to be retrieved after the plane lands. When a photo is taken all the data of the plane will be saved as well so that the photo can be associated with the orientation of the plane.

2. Live Telemetry:
   - Telemetry will be sent back to the ground station during flight so that the status of the plane can be assessed during flight. Telemetry that will be sent back to the ground station includes GPS, roll, pitch, yaw, heading, altitude, and battery life.

3. Control of Motor and Servos:
   - The plane will be flown using a single motor. Since Tail-Gator is a plane the motor will only be drove in one direction but will need to have good control over the speed that it's rotating at. This will be controlled using an Electronic Speed Controller (ESC). Two servos will be utilized to control the rudder and elevator to steer the plane during flight.

4. Motor Controller
   - Designed on the actuation board is a three phase brushless motor controller that will allow the user to control the motor via a radio control. The controller is designed in the sensorless configuration and uses BEMF readings on the coils to know the rotor position and current being drawn by the motor to know the speed at which the motor is currently spinning in order to change to the next commutation phase.

5. LiPo Battery Recharger
   - A single cell LiPo battery recharger circuit is designed to be able to charge a single cell Lithium Polymer battery back to its original voltage. This is used to recharge the batteries used for flight and could be altered to recharge a multiple cell battery if a balancer is added.

6. Power Management/Maintaining Center of Gravity
   - The motor will use 275W of power with a no load current of 1.6A. Using an 11.1V 3000mAh LiPo battery the plane will have an estimated flight time of 15 minutes while being able to handle the extra weight with a proper center of gravity.

Technology Selection:

1. *Camera:*

   - The camera needed to have a good resolution so that the pictures taken would give useful data to the user but also needed the files to not be too big that it would slow up the entire processor when trying to send all the information. The solution was 4D Systems microCAM Serial JPEG Camera Module. This camera is programmable to take pictures at low resolution (160x120) or high resolution (640x480) allowing for a wide range of pictures to be taken depending on the user's preference. This camera communicates with the microcontroller by using a TTL interface. The pictures will be in color and in JPEG format which will allow for easy transmision and saving to the micro SD card.

Fig. 1: *JPEG Camera*

2. *GPS:*

Fig. 2: *Venus GPS*

- The GPS is a Venus638PFLx which will is able to output NMEA-0183 with an adjustable baud rate of up to 115200 but for our purposes a baud rate of 38400 was used. This module has a fast update rate of up to 20Hz so that Tail-Gator will have accurate GPS coordinates throughout the entire flight. It allows for SPI flash memory to be added if desired to log the entire flight path so that it could be viewed once the plane had landed using Google maps to retrace its steps. Also the on board flash memory allows the Venus638FLPx to be programed so that it only outputs the lines of NMEA data that is required for Tail-Gator so that the XBEE won't get bogged down by transmitting useless data for our application.

3. *Three Axis Gyroscope/Accelerometer:*

- To implement a proper Inertial Measurement Unit (IMU) both a triple axis gyroscope and accelerometer is needed. This allows for roll, pitch, and yaw of the airplane to be calculated. A gyro only measures change in rotation along an axis and doesn't reset to 0 immediately when there is no more movement thus causing noise. For these reasons a triple axis accelerometer is needed so that proper orientation can be determined. But accelerometers also have their limitations in that they are susceptible to mechanical vibrations. In order to incorporate both of these into our design a MPU-6050 will be used that includes both accelerometer and gyroscope. This allows both sensors to share the same axis and be positioned so that they are both in the middle of the board for the most accurate readings to be given.

4. *Triple Axis Magnetometer:*

- The magnetometer is used to give accurate heading for North, South, East, and West. When used with the triple axis gyro and accelerometer a proper IMU unit will be completed. Even though the GPS gives directions of heading it only gives a simplified picture such as only North where a magnetometer will be able to provide more accurate data in degree units as well as provide faster data rates.

5. *Barometric Pressure:*

- A barometric pressure sensor is used so that the altitude of the plane can be recorded and sent back to the base station. The BMP085 is a high precision barometric pressure sensor that will allow the user to know the height of the plane. This pressure sensor has a resolution that goes down to 0.03hPa in ultra-high resolution mode which correlates to .25m of altitude. With this sensitivity reliable altitude data can be logged and the flight path will be made clearer but at the cost of decreased battery life making the sensor draw more current.

6. *XBEE Transceivers:*
    - The XBee Transceiver is used to transmit all the sensor data gathered on the plane to the base station. The WRL-08742 is an XBee Pro 60mW Wire Antennae – Series 1 that operates at 2.4GHz. This particular transceiver is similar to the other XBee transceivers except that it has an increased output which is better for long range. This transceiver has reliable serial communication between a microcontroller and computer anywhere closer than 1 mile with line of sight using a wire antenna.

Fig. 3: *XBEE Transceiver*

7. *Microcontroller:*
    - The ATxmega32A4U is an Atmel microcontroller that will be used to complete these tasks. It is the single most important component of the entire system, because it controls and reads the data as well as maintains operation of the plane. The microcontroller has to read the data from the IMU, magnetometer, barometric pressure sensor, and GPS. This 8 bit microcontroller has all the functions we need in a 44 pin package. With enough UART (5), PWM (15 channels), I2C, and a SPI to be utilized for control and sensing, this microcontroller will be able to handle the task. The max clock speed of 32 MHz is used and is more than fast enough to interface with all the components while maintaining reliable flight. Also the on board DAC allows for expansion options in the future.

8. *Servos:*
    - Two HS-82MG Hitec micro servos will be used to control the rudder and elevator. These servos fit inside the servo bay and have enough torque and speed to be able properly to steer the plane during flight. Communication to the servos will be via pulse width modulation and by controlling the duty cycle of the servos controls the angle that they are currently at.

9. *RC Transmitter/Receiver:*
    - To control get signals to the plane for remote control a 5-channel Hi-tec receiver and transmitter will be used. The three channels used for control of the plane are throttle speed, elevator control, and rudder control. The other two channels are intended for future expansion of the system so that the entire product can be optimized for the intended use of the user.

10. *Micro SD Card:*
    - The on board micro SD card slot is used to be able to save the picture that is being read from the microcontroller. The micro SD uses the SPI peripheral in the microcontroller acting as a slave

Block Diagram:

Below is a block diagram that demonstrates how our components connect to each other to obtain our objectives outlined in project features. Two processors will be used in our design. One controller will maintain all the sensor values, live telemetry, as well as take pictures. The other processor controls the motor and servos while reading the incoming PWM signal from the receiver. Interfaces utilized by our processor are I2C, USART, PWM, ADC, and SPI.

When the battery has lost charge the microcontroller will put the system in power saving mode. This will prevent the motor from running at full throttle as well as notify the user that the plane needs to land to have the battery replaced. For the analog component of this design we are going to design our own ESC for throttle control. This will allow us to customize how the motor works specifically for our applications.

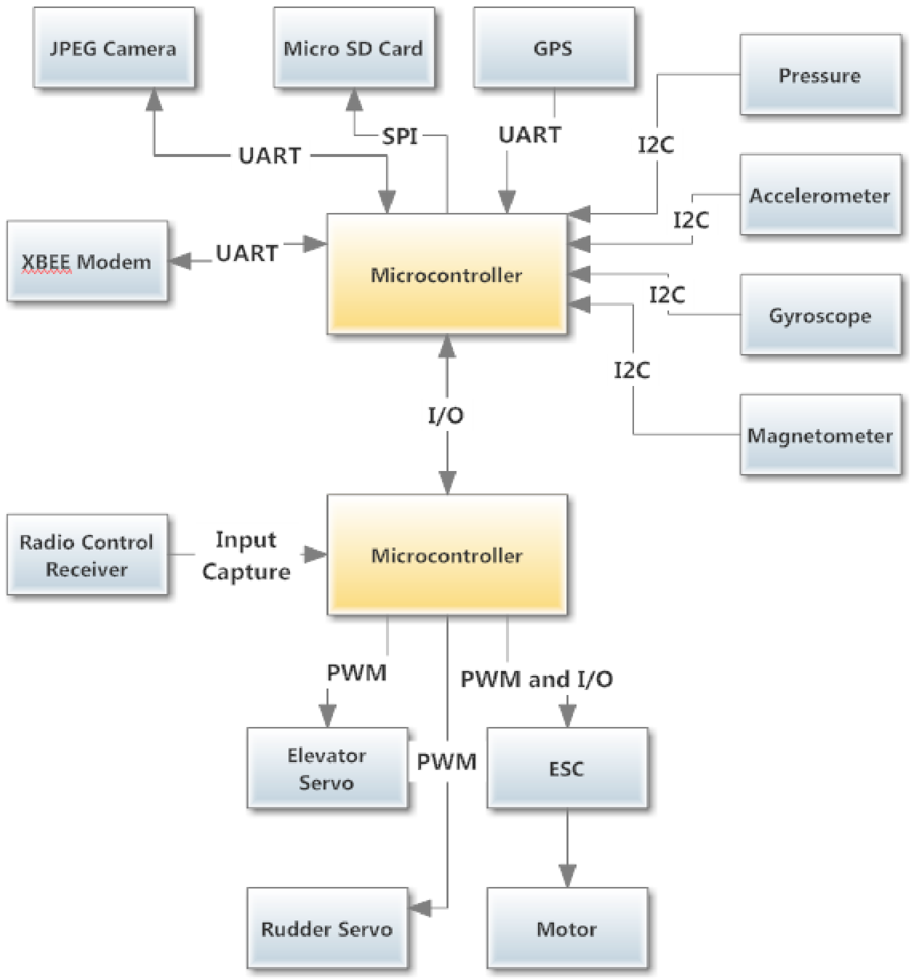


Fig. 4: *Functional Block Diagram*

BLDC Motor Driver Overview:

The motor used for radio controlled air planes is a three phase brushless DC motor. To reduce the cost of these motors most brushless motors are used in the sensorless configuration meaning that there are no hall effect sensors on the motor to sense the rotor position therefore in order to drive the motor through the different commutation phases another method needs to be used. They way this is achieved is by utilizing the back EMF voltage that is read via an ADC pin on the microcontroller of the drive phase that is currently not being driven.

Even though there are some cost advantages of using a sensorless BLDC motor there are some disadvantages as well. In order to generate a large enough BEMF to be read by the microcontroller the motor must be moving this means that to begin the motor must be ramped up in the open loop configuration until enough BEMF is on the coils to be read by the ADC's of the microcontroller then one can enter into closed loop control. BEMF can only be read with the motor speed is within certain operation range which may limit the BLDC motor that you currently have. Also commutation rates faster than the ideal rate will result in losing the lock on the zero crossing point of the BEMF and will result in discontinuous motor response.

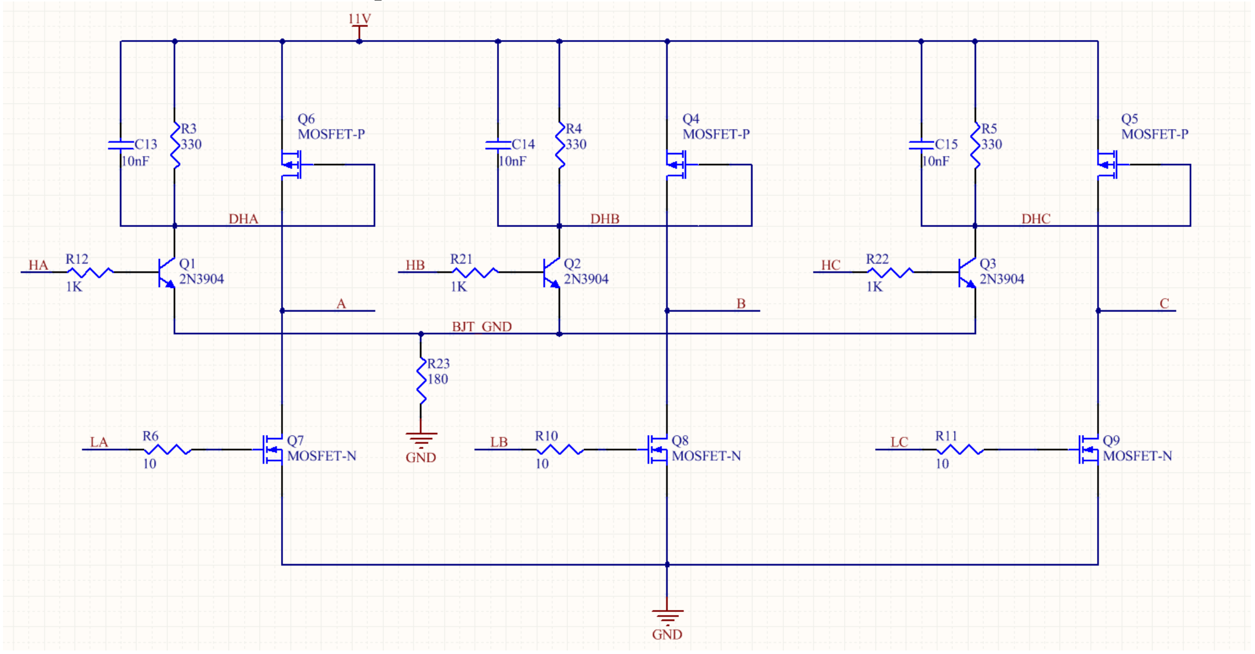


Fig. 5: *BLDC Motor Driver*

Now that the disadvantages are known a proper motor controller can be designed. For the analog component of the motor controller the schematic is shown in Fig. 5 above. A BLDC motor needs to be driven by 3 signals each 60 degrees out of phase with each other. Therefore there are three similar circuit each one controlling one phase labeled A, B, and C. The high side of the motor driver uses a PFET. By driving the PFET with a BJT it allows for a PWM signal to be sent from the microcontroller to the BJT turning on and off the BJT and thus biasing the PFET. Only the high side PFETs needs the BJT to bias them because the NFETs on the low side can be biases via I/O from the microcontroller. For example one commutation phase would be to drive the high side of A with a PWM signal while holding the low side of B low by turning on the 3.3V output then reading the BEMF from phase C and waiting until the reading is ½ the motor voltage. This point is called the zero crossing point. A timer is set at the beginning of each commutation phase and counts till the zero crossing point. Once the zero crossing point is read the timer is set again but this time waits for the same amount of time it took to get to the zero crossing point then switches commutation phases. Table 1 shows the commutation phases for driving a BEMF motor.

Drive Phase	High Side on	Low Side on	Read BEMF
1	AH	CL	B
2	BH	CL	A
3	BH	AL	C
4	CH	AL	B
5	CH	BL	A
6	AH	BL	C

Table 1: *Drive Phase Commutation Table*

The BEMF is a voltage signal that spans from 0V-the voltage that the motor is being driven by. In our case that is 11.1V. In order to read such a large signal in the incoming signal must be ran through a voltage divider so that the voltage level is acceptable for our ADC pins. A filter capacitor is added to the divider to filter out any of the high frequency noise that could enter our BEMF signals so that zero crossing detection can be read at the right time. In order to protect our microprocessor from harmful voltage and currents locking diodes on the ADC pin lines are added. This means that if ever an incoming voltage gets larger than 3.3V the pin gets shorted to the 3.3V power rail so it can never go above that voltage and likewise is the voltage goes below 0V the pin gets shorted to ground. Throughout the flight the motor voltage will change from about 12.5V to 10V. In order to properly get the correct zero crossing point the analog reference pin must be tied to the voltage of the motor. But as the motor voltage changes so will the scaling so an internal reference on the atxmega32A4U must be read in with our new voltage reference that proper scaling can be achieved and our system is independent of loss of battery power.

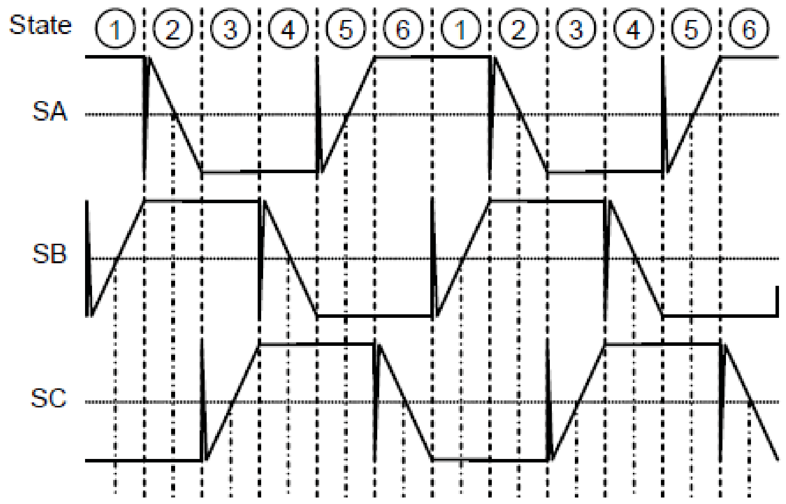


Fig 6: *Drive Phase Commutation graph*

Main Sensor Program Overview:

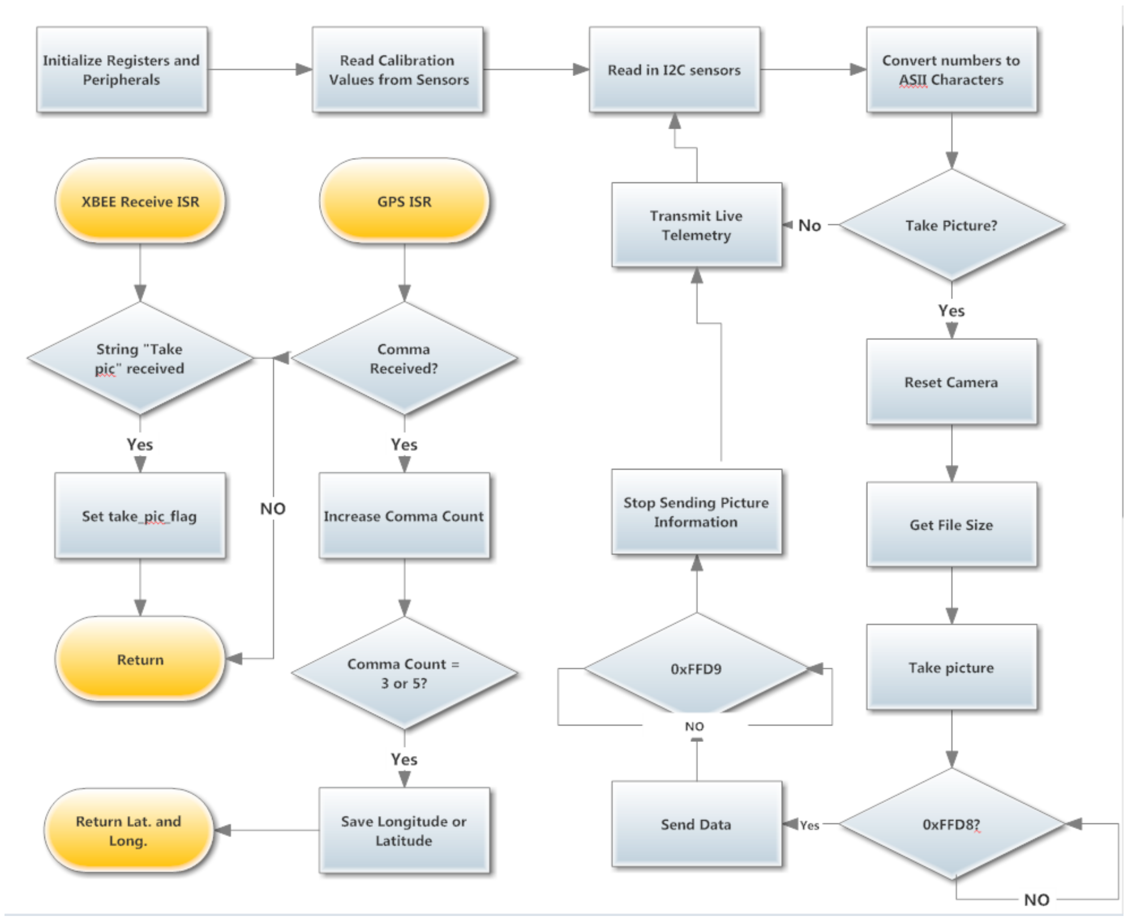


Fig. 7: *Main Controller Flow Chart*

The flowchart above shows how the main sensor program checks values and sends back live telemetry. Upon start up all the ports are initialized to the correct peripherals. Making sure the correct port it connected to the right sensor and then the registers for each peripheral are set to the corresponding configuration. This involves setting the baud rate for the I2C, USART, and SPI, then setting the clock speed to 32MHz.

Once all the ports are initialized then the I2C sensors need to be read to get the offset coefficients and storing the values in memory so that the correct sensor values can be read in. The sensors that involve offset coefficients are the pressure, accelerometer, and gyroscope. Once all the sensor coefficients are saved in memory the main loop is entered where the sensor values are read in through the I2C and USART then converted to characters to be outputted to the XBEE to be transmitted back to the base station. The data is transmitted in the same string with each data value separated by commas. This allows for the GUI at the base station to know when a new value is being read in by using the comma as a delimiter and the new line character lets the GUI know that a new set of data values are being read in to update the corresponding GUI display boxes.

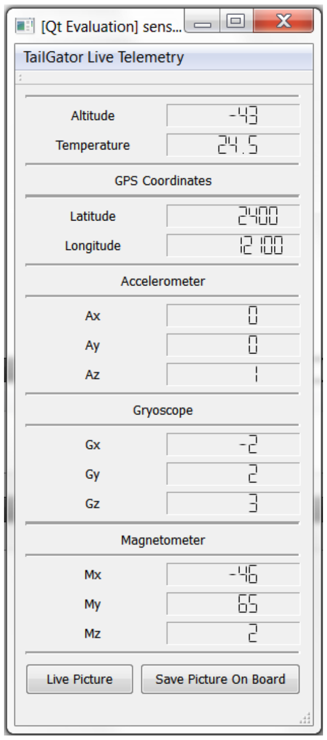
In order to just get the latitude and longitude data that is needed for our application the GPS had to be reprogrammed to work at 38400 baud rate and send back only 1 line of the NMEA message. Then by counting the commas of the incoming string that is being received the latitude and longitude data can be extracted from the string and written to the XBEEs to update the GPS location at 5Hz

When the take picture button is pressed on the GUI it send the string “Take pic” through the XBEE on the bases station and in the ISR for the received data on the XBEE it checks for that string by comparing every character until that series of characters is detected. Once detected the take_pic_flag is set and the main routine starts to send commands to the camera initializing it to take a picture. In the camera returned ISR it is constantly looking for the string (0xFF, 0xD8) which is the beginning of the picture. Once those bytes are read in the picture bytes are sent out through the XBEEs until the characters (0xFF, 0xD9) are read into the buffer which indicates the end of the picture. Once the end of the picture is sent through the XBEE the main program jumps back into the routine for sending back live telemetry and waiting for the take_pic_flag to be set.

GUI Overview:

The GUI was designed using key components from the QT library to utilize the visual aspects of the design with C++ and boost asio library to be able to read from the com port. The GUI utilizes the asio library functions to read from a connected com port until a delimiter character is read in from the com port serial stream for the GUI to update the numerical screen the delimiter used is ‘\n’. The GUI reads the input stream from the com port and converts it from the com port buffer to a string and then separates the variables by knowing that the incoming string is comma separated and allocates the proper variables to the new values and updates the on screen numbers. Multi-threading was used to be able to connect the com port asio thread to the graphical UI thread.

Since the com port was being used to both send and receive data only one operation could be operated at one time therefore when performing an operation the com port needed to know which operation had command of the com port and this was done by locking the program while the com port was either receiving new data or writing out data to be sent through the XBEEs. When a picture is received the incoming data stream is saved in a .jpg format to a specified location on my computer then by having windows picture viewer open the new image can be instantly viewed once the new picture is finished saving to the computer. Below in Fig. 8 is a visual of how the GUI is laid out to display our sensor values


[Qt Evaluation] sens...

TailGator Live Telemetry

Altitude	-43
Temperature	24.5
GPS Coordinates	
Latitude	2400
Longitude	12100
Accelerometer	
Ax	0
Ay	0
Az	1
Gryoscope	
Gx	-2
Gy	2
Gz	3
Magnetometer	
Mx	-46
My	65
Mz	2

Fig. 8: *TailGator GUI*

PCB Overview:

Two PCBs where designed for the electronics on board maintained all the sensors and sent back all the live telemetry while the other was in charge of all the actuation on the system which included the BLDC motor driver and the control of the servos. The sensor board could be powered by an 11.1V battery or if plugged into a USB port on a computer it could be powered by the 5V output. This allowed for the board to be debugged easily when outside the plane but also allowed for it to be powered by the battery during flight. The accelerometer and gyroscope were placed in the middle of the board so that the sensor values read in by the sensors where as precise as possible to characterizing the plane's flight. An FTDI chip was added so that the XBEE could be reprogramed right on the board when changing baud rates for testing and debugging purposes Fig. 9 below shows the layout of the sensor board.

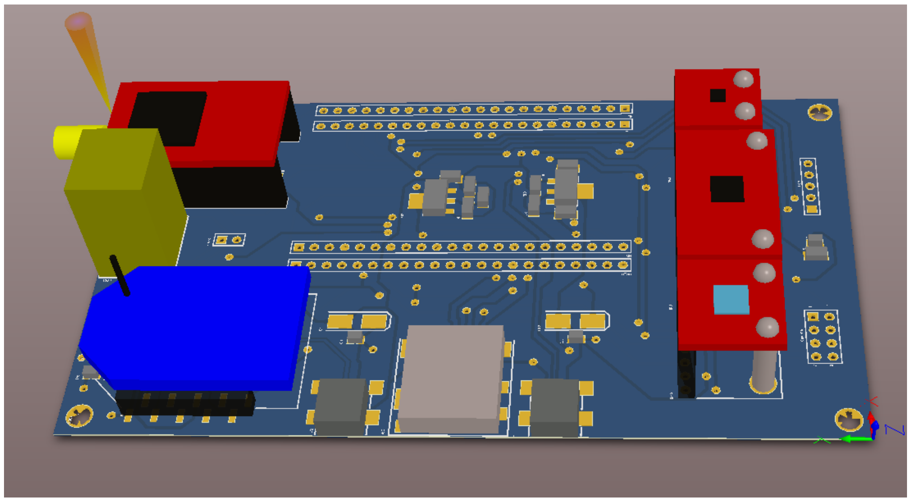


Fig. 9: *Main Sensor Board*

To have a fast response for the controls of our plane a second board was designed specifically for this purpose. It will input capture the incoming PWM signals from the receiver and then output a PWM signal for the servo or elevator based on the read in pulse width. These controls must be smooth and not clunky or else the flying of the plane will not be stable. The actuation board also contained the BLDC motor driver to control the motor. Since the motor driver was one the board a few other components where needed to have proper functionality such as a current sensor to know the speed that the motor was spinning and a low quiescent current 3.3V regulator so that the current consumption from the microprocessor isn't calculated into the speed of the motor. Also a 5V regulator that could source 3A to the servos was chosen so that the servos could not be current starved and be able to apply the torque needed to control the plane. Below in Fig. 10 is the layout for the actuation board.

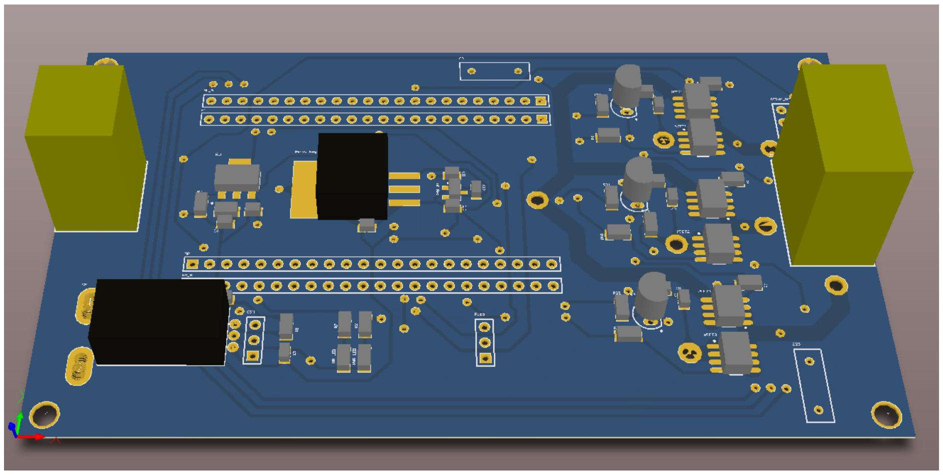


Fig. 10: *Actuation Board*

LiPo Batter Recharger

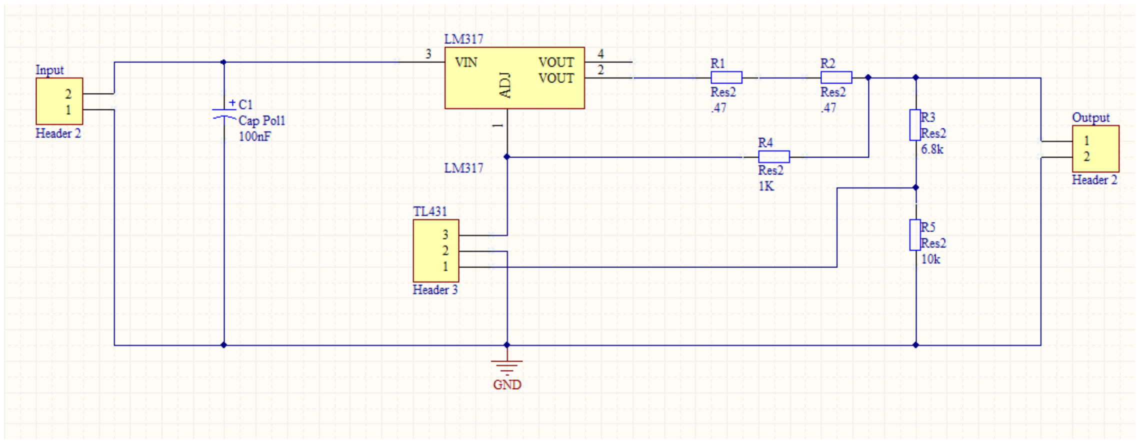


Fig. 11: *Single Cell LiPo Battery Recharger*

This is the schematic for a single cell 3.7 Volt lipo battery charger. Recharging a lipo battery is complex because it cannot be overcharged or the battery will be ruined. The key is to charge the battery close to 4.2 Volts, while not exceeding it and then let the battery discharge a little of its charge, so that the battery never stays precisely at its limit. The input needs to be plugged into a DC source with a range of 9V to 18V and the battery is connected to the output. The LM317 is an adjustable output voltage regulator and the TL431 is an adjustable shunt regulator. The shunt regulator is connected to a voltage divider (R3 and R5) in parallel to the output voltage so that when the output reaches its maximum voltage the regulator forces the output of the TM317 to be 0 Volts. The shunt regulator does this at around 2.5 Volts. When this occurs the battery cannot be overcharged. The reason why R1 and R2 cannot be combined is because of how low the resistance is in both resistors combined with the amount of power they need to supply. The capacitor and R4 were selected to achieve the correct output.

PDR: Tail-Gator

Division of Labor:

	Anthony Incardona	Fred Womack
Preliminary Research	50%	50%
Component Selection	65%	35%
Base Station GUI	100%	0%
Microprocessor Software	70%	30%
Actuation	100%	0%
Live Telemetry	70%	30%
UART Driver	0%	100%
I2C Driver	100%	0%
PCB Design	100%	0%
LiPo Recharger	0%	100%
Camera Operation	40%	60%

Table 2: *Division of Labor*

Gantt Chart:

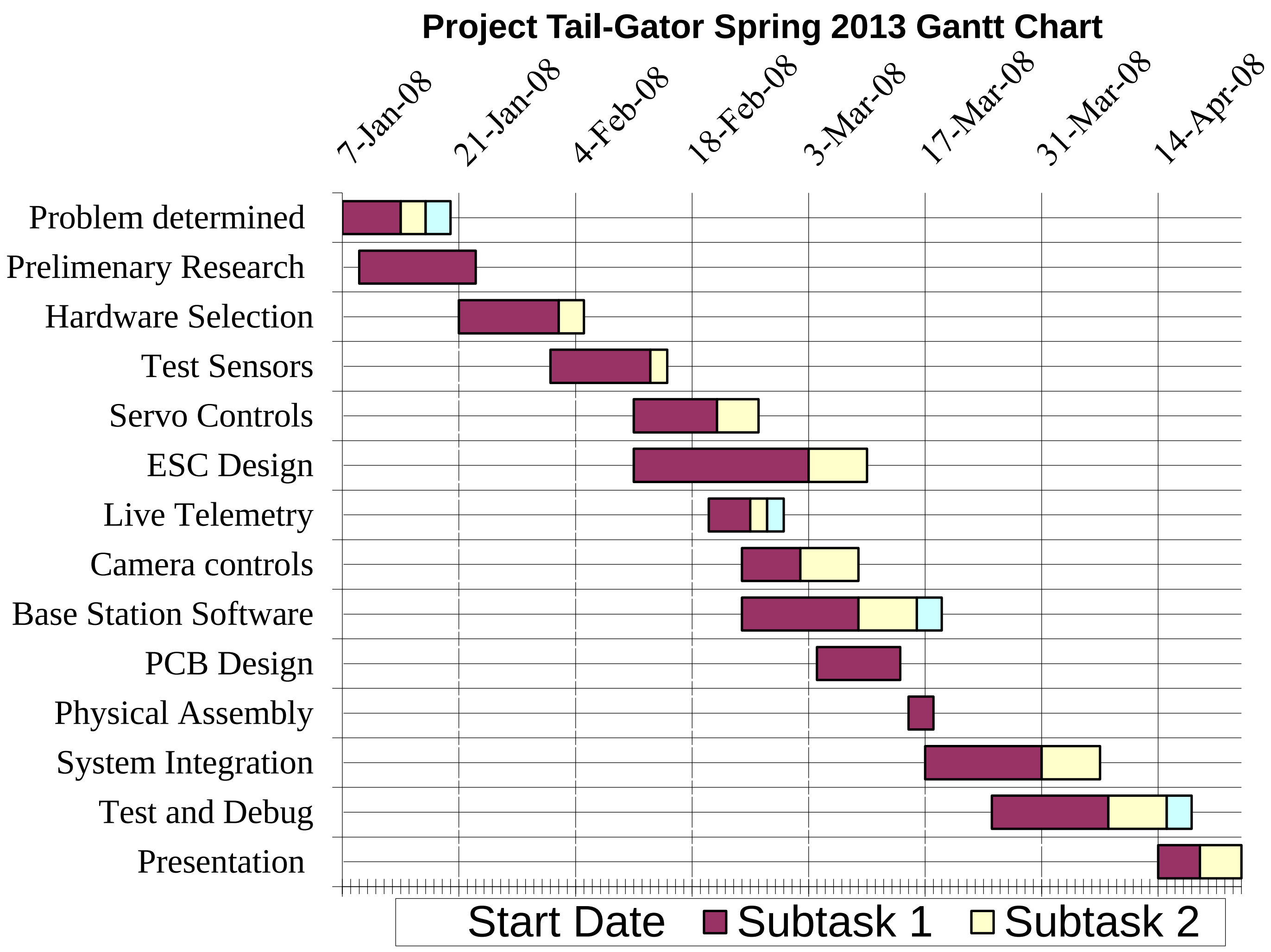


Table 3: *Gantt Chart*