University of Puerto Rico
Mayagüez Campus
Mayagüez, Puerto Rico
Department of Electrical and Computer Engineering

Powerline Monitoring Solution for the
Quadrotor Vehicle Platform

Progress Report #5

by

Omar A. Ferrer - Group Leader
Francisco De La Cruz
Gabriel Joel Perez

For: Professor Manuel Jiménez
Course: ICOM 5217, Section 096
Date: November 1st, 2010

Table of contents:

Abstract:

Power line maintenance is a costly endeavor. There is no easy way to constantly monitor them, and more often than not, instead of preventing their failure, any problems are treated after a power line has broken. Currently, to prevent failures, a helicopter is flown over power lines with an infrared scanner to monitor the temperature of power lines. If a temperature is found to be over a certain limit, the line must be treated to prevent future failure. This however is a costly process, which also requires a good amount of man power. To prevent these, a small scale quadcopter aerial vehicle will be employed. While following preset GPS waypoints, verifying with the sonar that it is over the intended cable, and measuring with an infrared thermometer we can monitor the powerlines with minimal human interaction, and when a certain temperature is passed, a picture will be taken, along with a GPS position being logged. This will allow for better and more efficient monitoring of powerlines for a lower cost.

Introduction:

As technology progresses, so does humanity's dependence on electricity. More electricity means more power plants, and more power plants mean more power lines. There are many ways these can be harmed, but unfortunately its not easy to monitor them constantly. A power line failure may cause power outages, and can be costly to recover from these power outages. Not only is it costly to the end client that receives the electricity, but it can be costly to the electric company to fix in a timely fashion a suddenly failed electrical line.

Because of this, some monitoring must be done, to try to treat power lines that may eventually fail preemptively. This monitoring however, is also costly and man power dependent. We intend to provide an alternative solution that depends little on human interaction, is easily scalable, and more affordable.

For our project, we will build an automated vehicle to replace the current procedure. Currently, a helicopter is piloted while a person measures the temperature of the power lines with an infrared thermometer to verify that their temperature is within a certain range. This proofs costly and dependent of man power. Our vehicle would replace this current solution by having some preset GPS waypoints to follow. To make sure that the vehicle is over the intended power line, it uses a sonar, and while following the power line path, it measures with an infrared thermometer the temperature of the power line. If it detects that the temperature is over a certain range of temperature, it takes a picture of the power line, and it logs it's GPS position to be later reviewed. The vehicle will also provide manual controls, to allow for the control of the vehicle, in case the preset program is insufficient or malfunctional. This would provide enough information to the electric company, to inform them when they need to take preemptive action. This unmanned monitoring would provide the data they would have acquired through the use of more expensive resources.

To keep the scope lower, we intend to simplify the project. We will use a controlled environment with little obstruction, where we specify a very limited range that simulates a power lines. The vehicle is set to follow an object that

stretches the intended path through GPS way points assisted by sonar. As it follows the path it would measure the temperature of the object that it is monitoring, and if it detects a temperature higher than a certain range, it should take a picture and log the GPS. This temperature will be simulated by increasing the temperature within a limited area of the path. This information will later be retrievable through serial, allowing for the analysis of it. With this prototype, the project will represent a step forward in allowing for safe unmanned monitoring of power lines to prevent failure preemptively.

Background Theory

Electrical transmission lines:

Electrical transmission lines are vulnerable to a wide range of vulnerabilities. One of this faults is the production of excessive heat in the line which can overheat components. This is of particular concern in in high voltage/ high current systems. For example in substations or switch yards, where contacts and or clamps can overheat until the material melts and causes the welding of the contacts together or even wore the complete disconnection of the power line. [1]

Thermographic systems available from several commercial entities can be used to detect these problems at a very early stage. [2][3] Many transmission authorities use helicopters to survey hundreds of miles of power line with these thermographic systems. These surveys can help increase the reliability of power lines and thus prevent costly damages to the lines.

[1] Davies, A. (1997). *Handbook of condition monitoring: techniques and methodology*. Springer.

[2] *High Voltage Distribution Lines & Substation, Electrical Infrared Surveys*. (n.d.). . Retrieved December 17, 2010, from http://www.electriscan.com/highvoltagedistribution.html

[3] High Voltage - ICS - Infrared Testing, IR Testing, Infrared Thermography Testing, Infrared Consulting Services. (n.d.). . Retrieved December 17, 2010, from http://www.irtest.com/infrared-high-voltage-survey-inspection-testing.html

XBee and Zigbee:

Xbee is a family of radio modules that implement the Zigbee IEEE 802.15.4 communications protocol. It is designed for low power, low speed, short to medium range and secure communications.[4][5] Zigbee has serveral features that distinguish it from other communication protocols such as Bluetooth and Wi-Fi. [6]For example it is way simpler than both Bluetooth and Wi-Fi. One of the advantages of this simplicity is that it gives it very fast connection times. But even though it is very simple it also very flexible. It supports several networking topologies. One can choose from Ad-hoc, peer to peer, star or mesh. Perhaps its most important feature is its low power consumption. This low power consumption comes from the fact that the majority of network devices in a ZIgbee network are able to remain inactive for long periods of time.

Zigbee uses digital radios to enable communications between devices.

Zigbee operates in two main modes: non-beacon and beacon mode. “Beacon mode is a fully coordinated mode in that all the devices know when to coordinate with one another.” “Non-beacon mode, on the other hand, is less coordinated, as any device can communicate with the coordinator at will. However, this operation can cause different devices within the network to interfere with one another, and the coordinator must always be awake to listen for signals, thus requiring more power.” [7]

[4] *Overview.* (n.d.). . Retrieved December 17, 2010, from http://www.zigbee.org/Standards/Overview.aspx

[5] *ZigBee* - Wikipedia, the free encyclopedia. (n.d.). . Retrieved December 17, 2010, from http://en.wikipedia.org/wiki/ZigBee

[6] *ZigBee versus other wireless networking standards*. (n.d.). . Retrieved December 17, 2010, from http://www.stg.com/wireless/ZigBee_comp.html

[7] *How ZigBee Works.* (n.d.). . Retrieved December 17, 2010, from http://homepage.uab.edu/cdiamond/How%20Zigbee%20Works.htm

Sonar:

Sonar is an abbreviation for “Sound Navigation, and Ranging”. It's initial work was developed with military application. It was used to detect submarines underwater. However, as time has passed, the resources needed have become increasingly inexpensive, and today sonar can be used for multiple other tasks.

The main components in a sonar are quite simple. At it's most basic, a

sonar sensor is mainly composed of a transmitter and a receiver. Each sonar device is designed to transmit only to a certain angular range, called it's “Beam Width”. The transmitter transmit a high frequency (usually to such high frequencies that it can't be heard by human ears) sound wave in a specific direction covering the range it was designed to cover. If there is an obstacle detected within that beam width, the sound bounces back to the sonar senor, and it's receiver receives the sound similar to how a microphone receives sound. This is called the echo[8][9]. The sensor then is in charge of measuring the time it took from the time it sent a pulse wave, to the time it received back a signal. With this information, and keeping in mind the speed of sound in the medium that the sonar is being used on (be it water or air).

Some sonar sensors allow to receive more than one echo back, allowing the sensor to detect more objects within it's range. It's important to note, that though the sonar by itself is enough to know the distance at which an object is detected, it isn't enough to know at which angle it is found. To know this, a more complicated array of sonars must be used. However, you will know that the object is within the sensor's beam width.

[8]Society Of Robots (n.d.) *Sensors – Robot Sonar* Retrieved Dec 14, 2010 , from http://www.societyofrobots.com/sensors_sonar.shtml

[9]LEI (n.d.) *How it Works* Retrieved Dec 14, 2010 from http://www.lei-extras.com/tips/sonartut/howitworks.asp

Infrared Thermometer:

All objects that have a temperature higher than absolute zero emit infrared light. This is called infrared radiation[5]. This light wave is found between visible light and radio waves, and because of this it is not visible to the naked human eye. All objects (except some theoretical ones) reflect this infrared light. According to Stephan Boltzmann Law, the hotter an object becomes, the more infrared energy it emits[10]. Using this principle, one can measure the infrared light emitted by an object to know it's temperature.

An infrared thermometer is a sensor which measures a temperature of a target according to the infrared energy emitted by the material. Each sensor is designed to measure the temperature of an object within a certain degree range, deemed the field of view. This is achieved through the use of a system that consists principally of a lens, optics, detector, pre-amp, ADC and a micro controller[11]. The lens takes in the infrared light which is detected by the optics and the detector. This is taken in as an analog signal which gets amplified ,and then passed to the ADC. The micro controller then receives this information from the ADC and is ready to use it as necessary.

To get accurate readings, one must make sure that the field of view of the sensor is small enough that the field of view covers mostly the object to measure. If not, the measurement may not accurately represent the temperature of the desired object. This can be mitigated by using a sensor with a small field of view.

[10]John Merchant, (n.d.) *Infrared Temperature Measurement Theory and Application* Retrieved Dec 14, 2010 from

https://www.omega.com/techref/iredtempmeasur.html

[11]Everest Interscience Inc., (n.d.) *Theory of Infrared Thermometery* Retrieved Dec 14, 2010 from www.ictinternational.com.au/brochures/everest/irt-theory.pdf

ESC:

The aircraft has 4 ESCs, one for each motor. The ESCs serve two main purposes. First and foremost the ESC or **E**lectronic **S**peed **C**ontroller serve as the main interface between the KDA20-22L brushless motors and the APM microcontroller board. The ESC takes a PWM waveform input from the microcontroller indicating the desired power level of the motor. This power level is determined by the received PWM's duty cycle, the duty cycle is modulated by making the pulse width longer or shorter, hence the name of the scheme[1]. The ESC's microntroller analyses the signal and according to it's internal configuration and the feedback received from the brushless motor generates an appropriate signals for the motor's FETs the motor up to the desired speed[2].

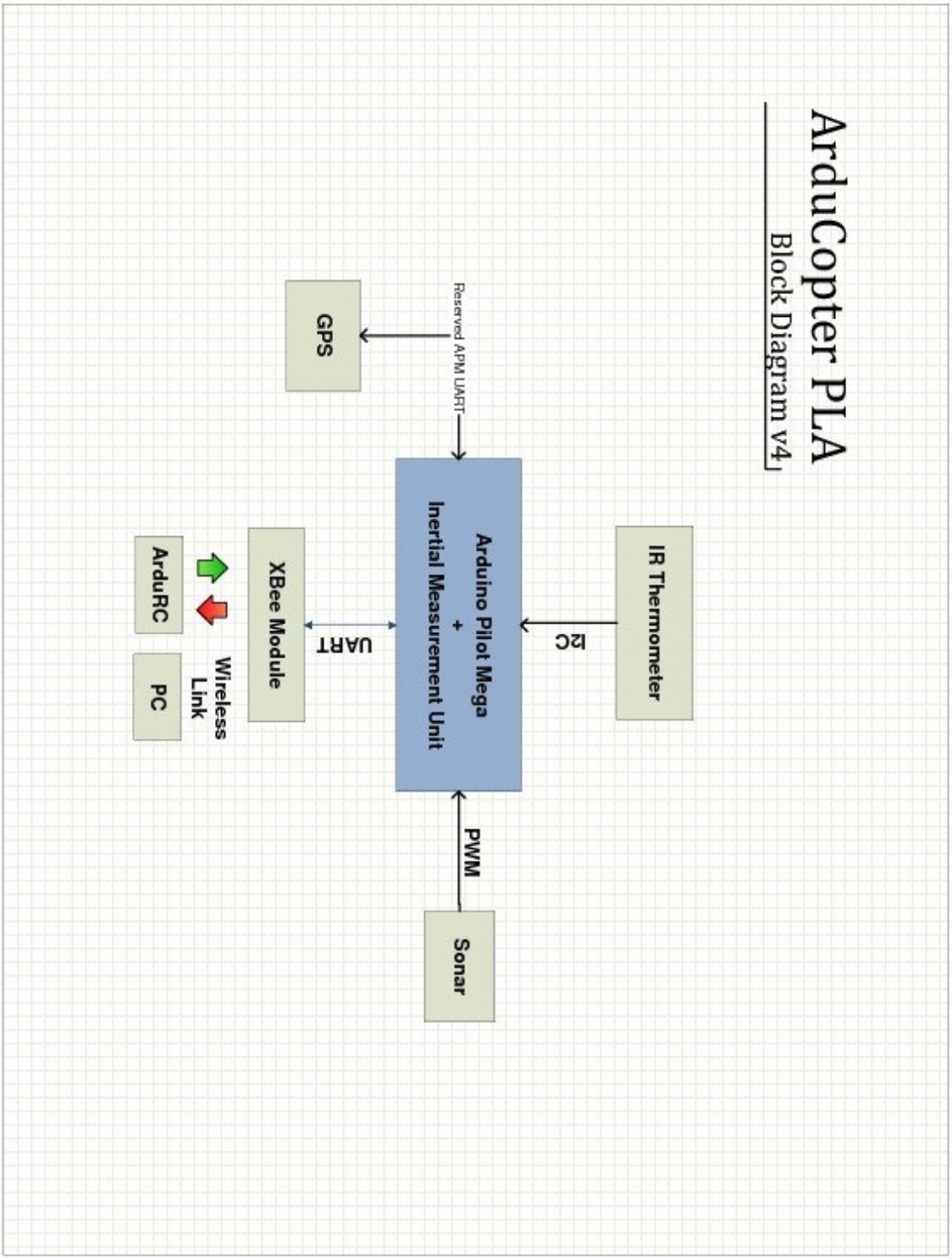
Additionally, the ESCs provide a BEC or Battery Elimination Circuit. This internal circuitry in the ESC provides the user with a 5V line at a maximum current draw of 3A. The Arducopter PLA uses the BEC line from one of the ESCs to provide power for the APM and sensor boards.

[1]Electronic speed control - Wikipedia, the free encyclopedia. (n.d.). . Retrieved December 17, 2010, from http://en.wikipedia.org/wiki/Electronic_speed_control

[2] SPEEDY-BL. (n.d.). . Retrieved December 17, 2010, from http://www.aerodesign.de/peter/2001/LRK350/SPEEDY-BL_eng.html

Hardware Analysis:

Block Diagram:



General Functionality:

GPS:

The onboard MediaTek MT3329 provides global positioning information that is used by the navigation software to track preset, on board, waypoints. The GPS provided localization data will also be used to tag points of interest like such as high temperature conditions in and around electrical lines. The GPS communicates via a serial TLL interface via a binary protocol. The interfacing aspects of this sensor are accounted for by the included software by the Arducopter team.

IR thermometer:

The Melexis ML90614ESF-ACF provides us with infrared temperature sensing abilities. The thermometer is placed pointing vertically down on the bottom side of the aircraft. This arrangement allows for convenient sensing of temperature data of objects below. When a temperature above a certain threshold is detected the software records the current GPS coordinates and optionally triggers another device such as an optional Camera. Communication with the microprocessor is done via an I2C bus. This bus is where the magnetometer and barometer are connected as well.

Sonar:

The downfacing XL-MaxSonar-EZ0 (MB1200) sonar serves as a navigation aid for the aircraft. Its main purpose is work as an object sensing device. The

data sent by the sonar is analyzed by the on board software and assessments are made to check if the is below the aircraft or not. Notice that the sonar is not used as an altitude hold sensor. The sonar communicates with the microprocessor via PWM signal.

Power Analysis:

	IOL	IOH	IIL	IIH	Isupply
Thermometer (U4)	7mA	7mA	30uA	30uA	25mA
Sonar (U1)	8.5mA	-3.0mA	0.1uA	1uA	100mA
Micro (U0)	20mA	20mA	1uA	1uA	200mA
XBee	20mA	1mA	-2mA	50uA	210mA
GPS	20mA	1mA	-2mA	50uA	53mA

Driver Analysis:

The GPS and Thermometer are connected to ports who's total pin current must not exceed 200mA. Hence:

Low:

200mA > -2mA + -2mA ; **OK**

High:

200mA > 50uA+50uA ; **OK**

The sonar is connected to a port with a total sourced current limit of 200mA. As the sonar is connected to a different port, the analysis is straight forward:

Low:

200mA > 0.1uA ; **OK**

High:

200mA > 1uA ; **OK**

Microprocesor Driver data:

Since our connections are point to point our weakest driver becomes our microprocessor:

Lowest driver low: 40mA (micro)
Lowest driver high: 40mA(micro)

Low state analysis:
40mA > 30uA + 0.1uA + 1uA – 2mA -2mA; no problem
High state analysis:
40mA > 30uA + 1uA + 1uA + 50uA + 50uA ; no problem

$\sum Isupply$ = 25mA + 100mA + 200mA + 210mA + 53mA = 588mA

588mA * 1.2 = $705.6 \approx 750$

750mA at 5V < 3 A provided by the ESC Battery Eliminating Circuits.

Timing Analysis:

Sonar: The micro has to wait 175ms after power on to use the sonar for measurements. The data gets updated every 99ms. The sonar will be connected through PWM, and it must measure how long the pin is held to high. To prevent this call from slowing down the whole system, it will be called only occasionally, not on every loop iteration.

GPS: The micro has to wait 35 seconds after power up from cold start to get proper measurements. It has a USART connection, so the the controller handles the timing responsibilities.

Thermometer: Requires a 20ms wait after boot before acquiring measurements.

The communication will be handled via an I2C bus. The controller will handle the timing. The connection must be setup and acknowledged within 1.5us, and hold said signal for another 1.5us.

Xbee**:** Connected via a dedicated USART on the micro. Timing concerns are handled by the USART controller.

Memory Map:

The onboard **A**rduino**P**ilot**M**ega (APM) board has two microcontrollers, an ATmega1280 and an ATmega328p. The ATmega1280 serves as the main operating microcontroller running all navigation and control code while the ATmega328p serves as a PPM encoder. As the code developed for this project runs exclusively on the ATmega1280 will focus on it and its memory map.

ATMega1280 Memory Map:

The ATmega1280 has three different types of memory:

Flash: 128KB

SRAM: 8KB

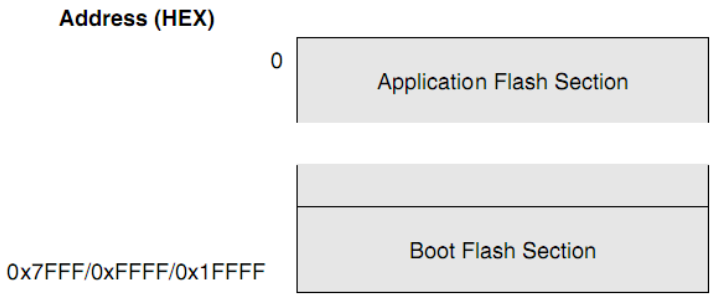
EEPROM: 4KB

This project made no use of external Flash or SRAM (XMEM) memory. The ATMega1280 splits it's program memory in two sections. Starting at address 0x00000 we should find the application memory. This is where our program will reside. Since our microcontroller has 128K of addressable flash the memory extends from **0x00000** to **0x1FFFF** as can be seen in the figure below [1].

Notice that there is a section reserved for “Boot Flash” this is where the Arduino bootloader is stored. The Arduino Boot Loader comes pre-flashed on to our APM board and hence takes up about 2K of flash[2]. The remaining 126K are free for use.

The latest revision of our code consumes **59994** bytes of the “Application Flash Section”. Rounding up we find that we still have about **66K** of flash memory for future expansion.

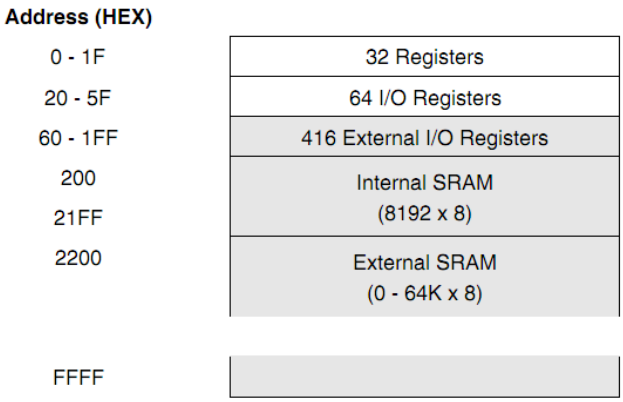
Program Flash Memory Map:



SRAM Memory Map:

Below we can observe a diagram of the microcontroller's SRAM [1]. This is where our register file is located. This houses registers R0-R31, the I/O Registers and space for internal, runtime storage such as stacks.

The On-board Atmel's DataFlash



(AT45DB161B):

Telemetry logging and GPS “Point of interest” in written onto an internal **16megabit** [3] dataflash serial memory.

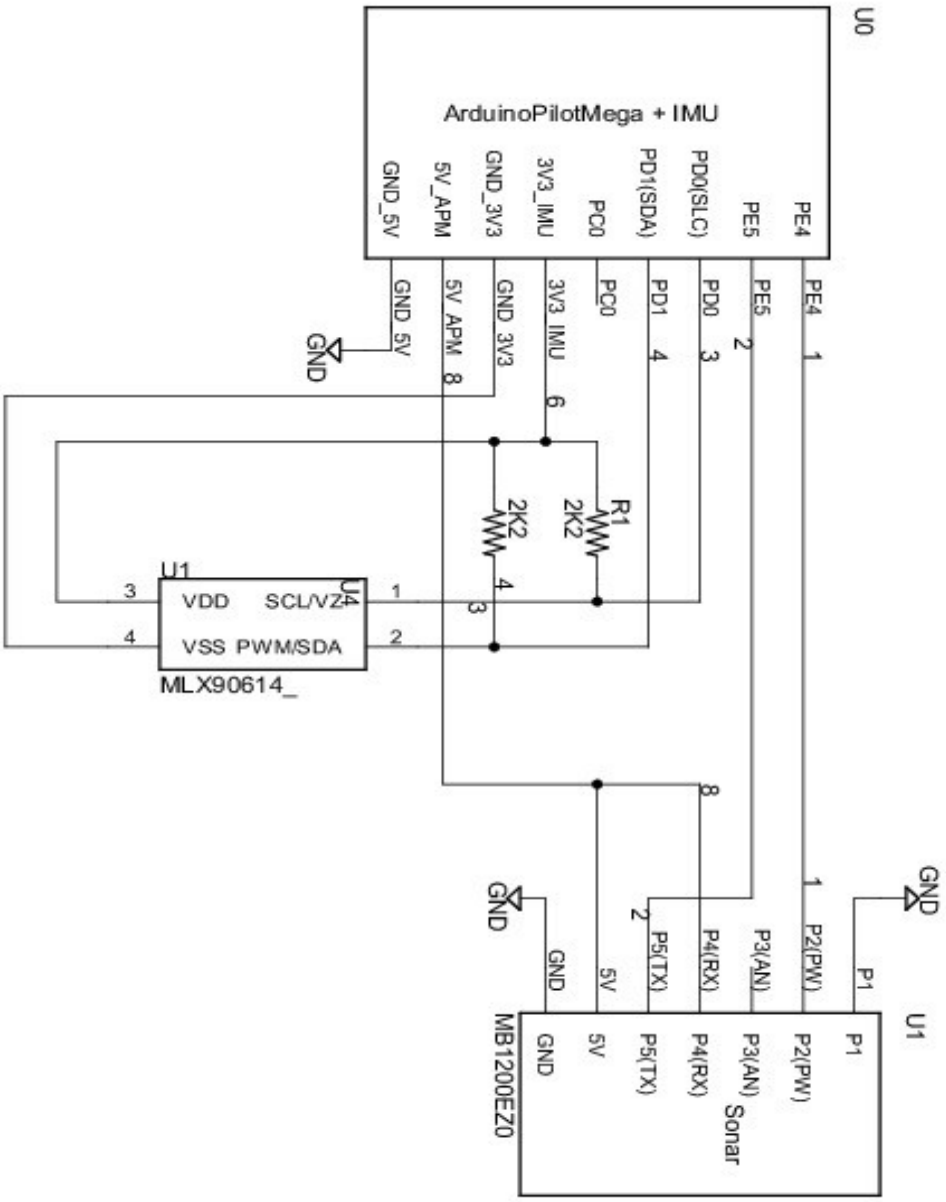
References:

[1] ATmel. (2010, September 0). 8-bit Microcontroller with 128K Bytes In-System Programmable Flash. Retrieved from www.atmel.com/dyn/resources/prod_documents/doc2549.PDF

[2] arduino.cc. (n.d.). The Arduino Boot-Cloner. *Arduino playground - Boot-Cloner*. Retrieved December 17, 2010, from http://www.arduino.cc/playground/BootCloner/BootCloner

[3] ATmel. (2010, May). AT45DB161B 16-megabit 2.7-volt Only Serial DataFlash. Retrieved from http://www.atmel.com/dyn/resources/prod_documents/doc3500.pdf

Schematic:


U0
ArduinoPilotMega + IMU
PE4
PE5
PD0(SLC)
PD1(SDA)
PC0
3V3_IMU
GND_3V3
5V_APM
GND_5V
GND
2K2
R1
2K2
U1
U4
VDD
SCL/VZ
VSS
PWM/SDA
MLX90614_
P1
P2(PW)
P3(AN)
P4(RX)
P5(TX)
5V
GND
MB1200EZ0
Sonar

Design Considerations:

Since the Arducopter PLA main platform is an aerial vehicle it is crucial that we hold security and safety of the operators, vehicle, citizens at high priority and preserve the integrity of the monitored lines. For this reason the several considerations have been made, most on the software side, that allow us to achieve a this high level of redundancy. A dedicated Atmega328p (aside from the main Atmega1280) handles Radio Receiver PPM encoding. This additional microntroller, although increasing the complexity of the system, allows for separation of tasks and allow the navigation software to run less encumbered.

Additional, software-side mechanisms are included in the software provided by the Arducopter team. Among the safety considerations the following two stand out:

- Motor arming and disarming safety mechanisms via special Radio Transmitter stick patterns.
- Accidental Sudden Throttle protection: Automatically detects surges in throttle settings avoiding sudden motor jolts and or loss of control.

Beyond this, we have decided to add an on-board LiPo battery monitor to the aircraft. This battery monitor displays preset led colors and buzzer sounds indicating approximate battery charge left. This gives the operator visual and auditory guides allowing him or her to asses remaining battery life without landing the aircraft.

Hardware Termination Table

Task	Status	Notes
Sonar connectivity	Complete	
IR Thermometer connectivity	Complete	Changed Thermometer
JPEG trigger and camera connectivity	Incomplete	We have connection to the camera, but the JPEG trigger doesn't work.
Xbee connectivity	Complete	
GPS connectivity	Complete	
Frame, ESCs and Motors connectivity	Incomplete	Though its connected, we are still tuning and verifying that the connections are correct, because we haven't been able to achieve stable flight.

Software Analysis:

Software flow charts and plan:

Software Termination Table:

Task	Status	Notes
Sonar monitoring	Complete	Verifies that the vehicle is within a certain range.
Sonar reaction	Partial Implementation	Preliminary code done to control vehicle according to sonar monitoring, but still must debug further
Thermometer monitoring	Complete	Monitors current temperature in F.
Thermometer single shot	Complete	When a certain temperature is passed, it sets a specific pin to high (will be used together with JPEG trigger)
Picture taking	Incomplete	Because of issues previously mentioned, we haven't been able to take pictures yet.
Xbee commands and data passing	Complete	All the communication software is done, and slight protocol established.
Stable flight	Incomplete	Issues with the base architecture have proven difficult.
GPS waypoints	Partial implementation	We have information from the GPS already, and preliminary code to move according to GPS data, but hasn't been fully prepared.

User Guide

Commercial RC Control

Arming the motors

To arm the motors move the thrust stick (the left joystick) to the lower right position and hold it there for 3-5 seconds and then release.

Disarming the motors

To disarm the motors move the thrust stick (the left joystick) to the lower left position and hold it there for 3-5 seconds and then release.

Vehicle Control

The left stick controls the vehicle thrust. When you hold the control upright if you put the lower the stick to its lowest position the thrust will be at its minimum. If you rise it to the top the thrust will be at its maximum. The right stick controls vehicle pitch, yaw and roll. Moving the stick vertically upwards increases roll and moving it horizontally to the right increases pitch. The opposite direction decrese these values.

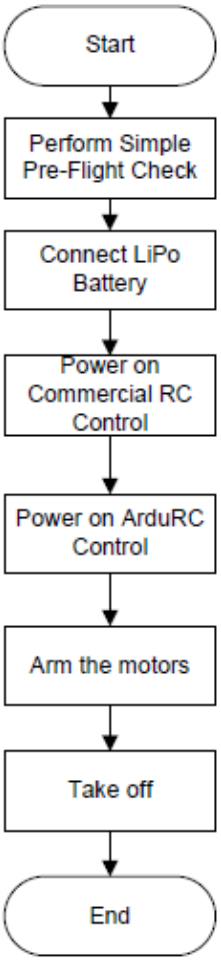
Switching to ArduRC control

In order to switch control of the vehicle from the commercial RC control to the ArduRC control move the switch displayed on the picture to the 0 position (the position closest to the back of the control). Only perform this change while the vehicle is safely placed on the ground.

Switching to Full UAV (unmanned autmoatic vehicle) mode

In order to switch to Full UAV mode for automatic power line inspection move the switch displayed on the picture to 0 position (the lower position if the control is upright). Switching to Full UAV mode should be done with much care and requires proper prior configuration.

Vehicle Takeoff



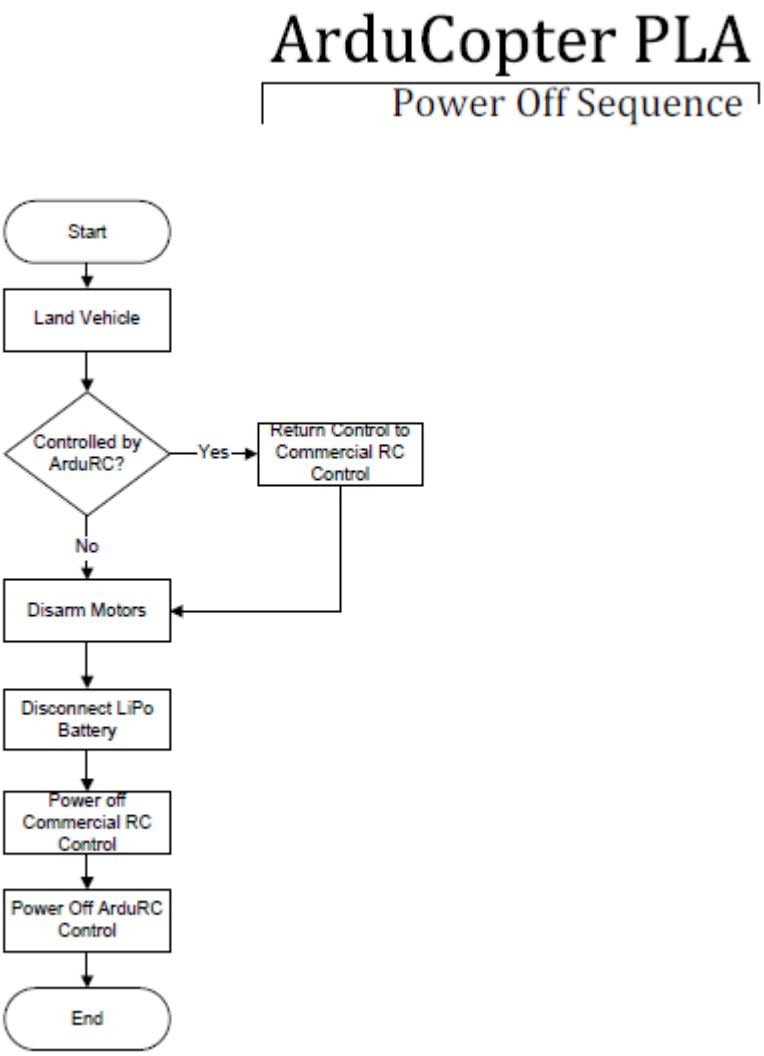
Before taking off make a simple pre-flight check to make sure the vehicle is ready for flight:

- Check if the battery is fully charged.
- Check if all the propellers are clean and without damages.
- Check if the propeller screws are tight.
- Check for damage in the arms.
- Check that all the sensors and electronics are properly placed, in good shape, well connected and tightly adjusted to the vehicle.

Do not attempt to flight unless you have checked all of these items and fixed any problems you find. Take off on a level surface. A level grass field is recommend since it provides for cushioning in case of a crash.

Vehicle Landing

Land on a level surface. Land very slowly to avoid damage to the vehicle and to the landing gear. Then follow the power off sequence.



ESC cofiguration:

To configure the ESC we need to use the beep codes the ESC outputs. To

configure them, they must each be connected individually to the battery, so make sure all other ESCs are disconnected from the battery. The input signal from the ESC must be connected to the “throttle” signal from the RF receiver, and the ESC should be connected to the motor. After the connections are done, put the throttle on the controller to zero. Then, connect the battery to the ESC. Once connected, the ESC will begin giving out it's beep codes.

The first beep code is a beep, silence, and a beep. To configure the ESC to be used with a LiPo battery, one must push throttle to the top between these beeps. Once done this is done, the ESC will give out a longer lasting beep, indicating that option has been set. Disconnect the ESC to make further changes. In the controller, turn throttle to zer and connect the ESC to the battery again to go into the beep code menu once more. To then turn off the “break” feature, wait until the beeps are three consecutive beeps in a row. These will repeat twice. Between the repetition, push the throttle to the top, to set the “break” feature off. The ESC will now give a long beep sound, indicating the option is set. Disconnect the ESC from the battery, turn the throttle to zero. Now repeat this complete process for each ESC to configure them all.

Battery connection, disconnection and charging:

The project depends heavily on a LiPo for operation. To charge the battery, you need an appropriate LiPo battery charger. For our project we used the “Turnigy Accucel-8 150W 7A Balancer/Charger”. To charge it, set the charger to LiPo charge, then set it to 3S (we are using a 3 cell LiPo battery), set the current

to 3.6A, and the voltage to 11.1V. Then connect the charger to a power supply that is outputting 16V, and that has current protection set to 3.6A. As the battery charges, you'll see the current being provided by the power supply to decrease over time. Eventually, when the current is around .2 A, it means that the battery is fully charged and ready to use.

To connect the battery, make sure that all the ESCs and the Arduino is connected before hand. If you are connecting the Arduino through USB, make sure to disconnect the voltage being provided from the ESC to the Arduino, but do not disconnect the ground between the ESC and the Arduino. Also, make sure the USB is disconnected before connecting the battery. When ready, connect the battery to the Power Distribution board (making sure the you are connecting it positive with positive, and negative with negative). After this is connected, if you wish to connect the USB, now would be the time. Now the vehicle should be fully powered and ready to use.

Conclusion:

Building embedded systems is a complicated task. It's complexity doesn't necessarily lie in understanding the concepts behind the project being built, but rather on the fact that theory and practice are two wildly different things. In practice you'll notice a staggering amount of variable of how things can go wrong. Amongst these is trusting that a device or a platform that you depend on

will not necessarily work as advertised. Because of this, even if using a fully built device or platform, it must be taken into consideration the time it will actually take to make that platform work, if it ever will. We clearly learned this with the “Arducopter” platform that we are using in our project. We didn't take into account how incomplete the platform actually was, and how much work it is to fully integrate it, and because of this, our whole project stalled. The same thing happened with the JPEG trigger, which did not work with the camera we bought, even though the seller clearly advertised it as so.

Embedded systems have a lot of possible variables where things can go wrong. Though we may plan for some of these by getting extra parts, its not always easy to find which part needs to be replaced. This is even more obvious when dealing with devices that have to interact with the environment through physical movement. Our project, depending on flight, is especially sensitive to this. We haven't been able to have steady flight because our sensors may be malfunctioning, or because they are incorrectly calibrated. Because of this, we believe that this project may have been a bit inappropriate as an introduction to embedded systems. We had to understand a fully built project first, so that we could add to it. This turned to be far more problematic than we predicted.

Overall, we were able to add to the system the functionality of object detection under the copter through sonar, and thermometer measurements. However, the complexity of the flight system took far more time than we had planned. Nevertheless, we feel that we have a complete system that built on top of the already before built system significantly.

References:

(Each section's references are self contained).

Appendix:

(The appendix information has been included in the DVD)