EE2EDP - Avionics Module

Lancer Programme

Ben Cartwright
Siôn Abraham
Matthew Marriott
Thomas Lewis

Aston University

The Institution of Engineering and Technology

Contents

Table of Contents

Team Members Role and Responsibilities

Ben Cartwright – Team Leader, Software and general technical issues.

Matthew Marriott – Project Management, Financial management, Antenna build, assembly technician.

Thomas Lewis – I^2 C code management, sensor research, mathematics and error calculations for tuning network.

Sion Abraham – PCB Design, Project Management, module testing.

Introduction

The goal of this project was to design, develop, build and fly a High Altitude Balloon craft capable of flying up to at least 30km into the stratosphere and returning both high resolution photography and scientific data. This was done as part of Aston Astronomy and Astronautics Society and as a project of the EE2EDP module in 2015-16. While the AstroSoc has been tasked with handling the production of airframe hardware and sourcing the equipment we needed to layout the foundation for the electronics. The 4-man team that authored this report have produced the avionics and communication hardware as part of our EE2EDP module.

The duration of this report will detail the steps and development of the GPS module unit, which is pinnacle for the success of the flight. This GPS module project aims to meet all the criteria required to pass the EE2EDP module to the master's degree program level.

Project Management

The project has been closely monitored and managed by all members of the team. The schedule for the project was centred on a detailed Gantt chart. This chart provided individual, broken down tasks that are needed to be completed over the project, along with scheduled times to meet.

Communication is paramount to the success of a team project; therefore our team have been in constant communication via group messaging on social media platforms along with utilising a resource called Slack [1], which is a messaging environment for teams. In addition to messaging we can share files and create feeds that the general public can access see to view our progress.

At every EE2EDP scheduled lab sessions we set time aside for a 10-15 minute meeting where action minutes were taken and progress and issues were broken down and evaluated, followed by new tasks being allocated.

Gantt Chart Screenshot Figure 1.1
Action Minutes Screenshot Figure 1.2
(For figures see reference figures)

Specification

It was vital that the specification was set out in advance in order for components to be selected that will meet our requirements.

Power

Power will be supplied by a pack of several Energizer L91 batteries. [5] These batteries are non-rechargeable, however they use a specific lithium chemistry that allows them to operate between -40°c and 60°c. Whereas other batteries, for example LiPo or NiCad, will take substantial hits to their efficiency at extreme temperatures, L91s maintain a consistent voltage output, and are therefore suitable for the largely un-insulated airframe that we're using for both L1 and L2. These batteries produce a nominal voltage of 1.5V and can produce upwards of 2000mAh. A set of these combined with a voltage regulator will therefore be suitable for our purposes.
We have chosen a Vcc of 3.3V, which allows us to easily power the CPU and sensors/modules. Additionally we will be able to use an LD1117 power supply, which significantly reduces complexity on the PCB over building a power supply out of discrete components.

Battery life

The battery life of the system needs to be at least 6-8 hours at minimum. This allows for a 2-3 hour flight and additional time while the craft is on the ground for successful recovery. If possible, the battery life of the avionics system should be extended wherever possible.

GPS

The GPS unit is one of the most crucial systems on the craft. It provides complete positional telemetry, in longitude, latitude and altitude. This does require some initial configuration and so research will need to be done to ensure the system is set up correctly.
The GPS module we will be using is the uBLOX MAX-M8Q – with different models available depending on our target Vcc. [2]
The GPS module will need to be initialized and set to Aerial Dynamic Mode before it can be used in a HAB launch – if this mode isn't successfully enabled then the GPS will be limited to its artificial ceiling of around 12km.

Radio

The radio module we will be using to transmit telemetry/pictures to our ground station will be the NTX2B module, made by Radiometrix. [3] This module has been chosen as it has the highest power legally available for amateur radio in the UK (10mW). Additionally, the module is suitable for use on a HAB vehicle as it contains

a TCXO (Temperature Compensated Crystal Oscillator). This means that as the surrounding temperature changes, the transmission frequency remains stable and does not fluctuate, meaning that we don't have to constantly retune the ground station. Although the datasheet [4] states an operational range of 500m, in reality, with clear line of sight (clouds do not affect operation) a range of around 500km is achievable. This means this module is perfect for use on a HAB vehicle.
The communication protocol the craft will use will be RTTY. This is very similar to UART, but will require some research in regards to proper implementation. The transmitter will be configured to transmit at a data rate of 50 baud, ensuring no data loss during transmission.
The transmitter is configured to transmit at a frequency of 433MHz (Europe) and 458MHz (UK). Research will need to be done into the exact frequency we will need to use. The transmission frequency is able to be tuned somewhat via a control pin on the transmitter.

Data rates

The NTX2B transmitter will transmit data at a rate of 50 baud.
The GPS will communicate with the ATmega328 via the USART port on the microcontroller. This will be at 9600 baud for both transmission and reception.
The datastring is approximately 256 characters long at most, corresponding to 2816 bits ((8+3) * 256). At 50 baud, this can take up to 56 seconds to transmit. In general the datastring will be much shorter, up to around 128 characters. This means most datastrings will take around 28 seconds to transmit, which will be easily enough time to measure the next set of data from the sensors.

Form factor

The avionics computer that we produce must be confined to a certain form factor in order to fit into the equipment bay of the SkyRay airframe. Additionally, it must be kept as compact as possible, so we have more space available for batteries, backup trackers, cameras or further science experiments.

Craft – Tx antenna equipment

An antenna will be required to be positioned on the outside of the craft, ideally positioned to aid transmission to the ground. This will require an electronic socket to be attached to the PCB in order to facilitate this. Research needs to be done on the types and sizes of antennae that are suitable for our transmission frequency and into the type of wire connection these antennae use.

GND – Rx antenna equipment

The signal being transmitted from the Lancer needs to be received and interpreted on the ground. The RTL-SDR Dongle was selected as it is one of the cheapest on the market for what was needed. It has a large frequency range (24 – 1766 MHz) and is well respected by HAB enthusiasts.
In order to receive the signal up to 30km, a larger antenna was required. A 434MHz YAGI Antenna was purchased, matching the frequency of the transmitted signal. It is

directional, meaning it is stronger when facing the Lancer. This was proved during testing.

System Block Diagram

The above diagram defines the modules within the software and hardware of the avionics system.

The software will need to initialize the GPS and radio transmission modules, as well as setup I^2C protocols in order to talk to the sensor bank. From there it will initiate a loop to read the GPS data, append it into a string, and then transmit that string using RTTY. Though I^2C implementation is not required for the EE2EDP module it will be integrated in the future so everything has been designed with this in mind.

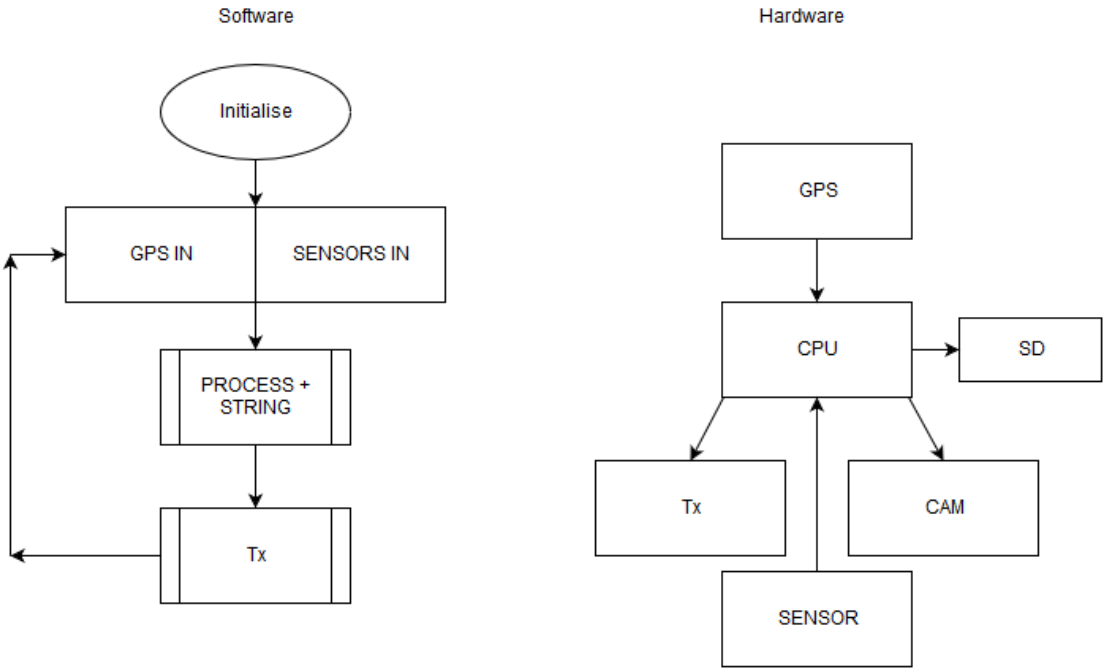


Figure 1

Component Choice

The aim of the Lancer 1 mission is to collect stratospheric climate data, by measuring factors in the environment and transmitting the information to ground. In order to manage the sensors I^2C, also known as Two Wire Interface (TWI), will be used. The information from these sensors, along with the GPS information, is then formatted into a single string by the microcontroller and transmitted.

The environment of the Earth's atmosphere is far different from the ground, and changes with altitude. Pressure decreases with altitude and asymptotically approaches a vacuum. Temperature, however changes depending on what layer of the atmosphere the lancer is in. The two main heat sources are the heat from the Earth and heat from the sun with the amount of air between the lancer and the heat sources affecting its temperature.

As the lancer rises, the pressure inside the balloon will differ from the barometric pressure. Once a large different is met the balloon will exploded and the lancer will head back to ground. Using the graph below, it becomes clear that the pressure is very close to a vacuum from 30km and above. This means that we have to make sure our sensors can survive and operate correctly within the environment up to 30km.

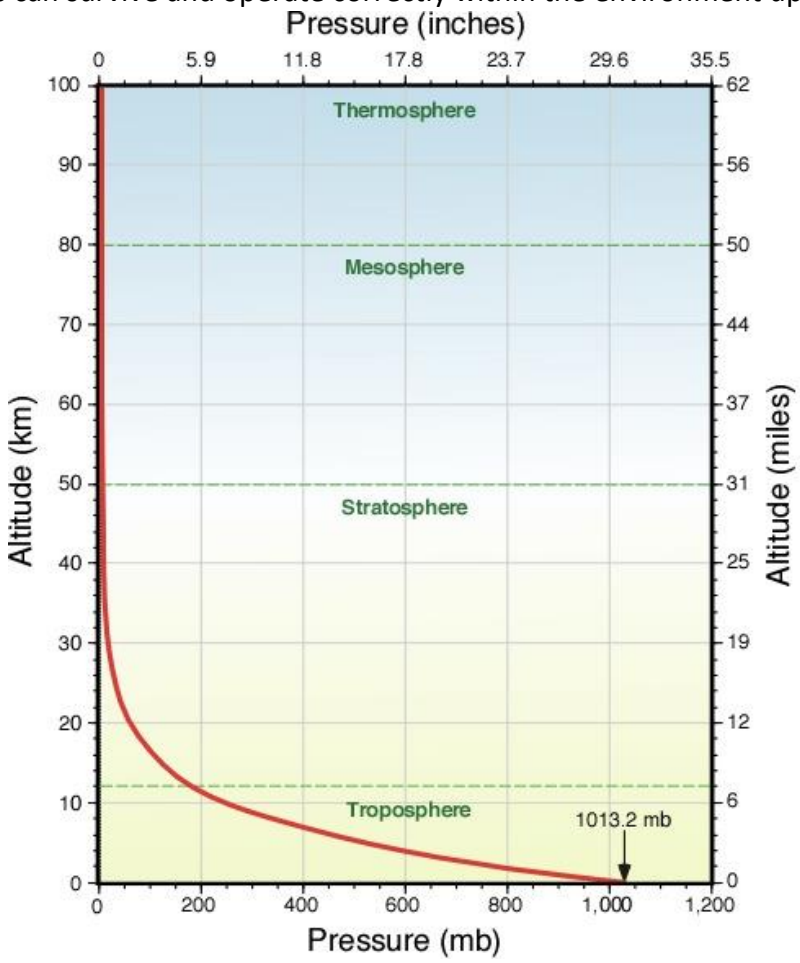


Figure 2: Pressure change over altitude

A major factor when looking at operating range for components is temperature. Using the graph below we can see how the temperature fluctuates as the lancer travels though different layers of the atmosphere.

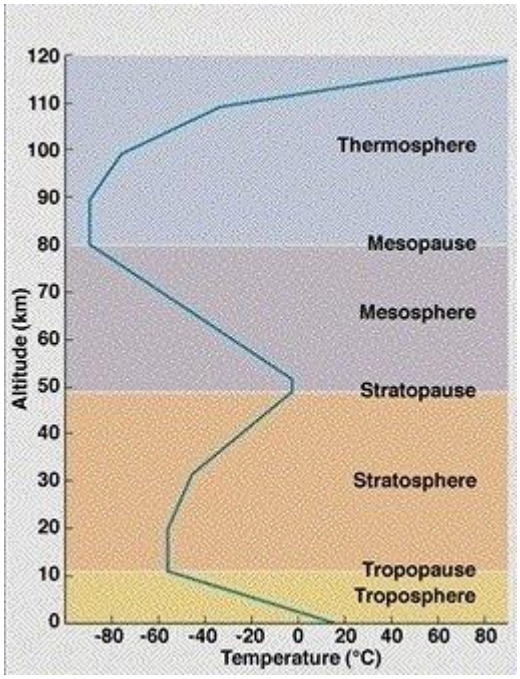


Figure 3: Temperature change over altitude

The minimum temperature that the lancer will have to meet is around – 55°C, the maximum being the temperature at ground. However, many of the components will be sealed inside of the lancer's body, shielding it from the colder temperatures. Due to the lower pressure it is far harder for the heat to radiate from the board as there is less of a medium (air) to radiate into, so components will be above -40°C for the majority of the mission. The lack of wind chill also keeps the temperature from dropping. The sensor needs to also be able to run off 3.3V and have a high resolution.

The GPS Module we are using is an Breakout Board consisting of a u-Blox Max M8-Q GPS Chip and a SARANTEL/MARUWA SL-1252R antenna. It is a professional grade GPS unit utilizing the M8 chip and can utilize two concurrent GPS signals at any one time thus making it highly accurate. U-Blox Max M8-Q and the SARANTEL antenna both have an operating range of –40° C to 85°C. It is produced and assembled by Hab Supplies.

Atmospheric Sensor

Although not part of the EE2EDP module, it was important that a suitable sensors were chosen in order to allow easy integration in the future. This would lead to the PCB board and software being designed around the aims of the next generation lancer.

The keys areas when monitoring the environment is temperature, humidity and pressure. Early on it was agreed that I^2C would be the best method to use as multiple sensors could be added without having the change the PCB design, making it very versatile.

The BME280 was chosen as it is 3 sensors in one breakout board and can operate at temperatures as low as -40°C, and only takes up 2 pins on the microcontroller. Basic elements of the C code was drawn up for testing and using Bray's terminal we can see that we are able to communicate with the sensor, though the data is not being correctly interpreted. However, we gathered all the information we needed for PCB board design and how the code can be integrated with the GPS code.

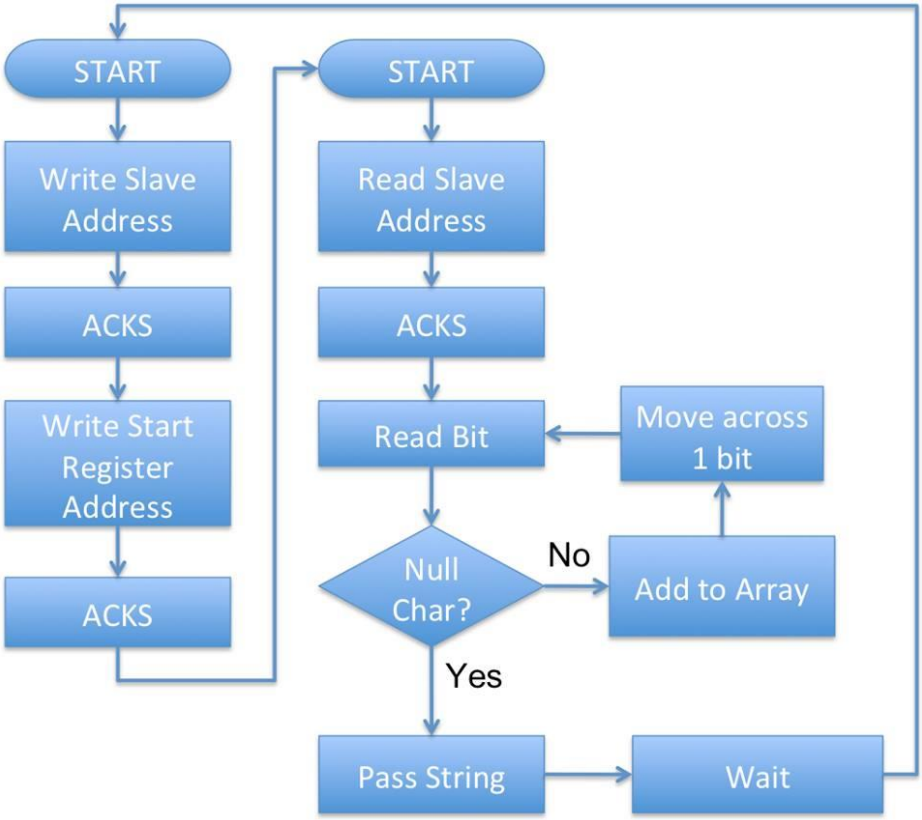


Figure 4: Flow Diagram for I2C

Board Specifications

Weight Battery Connector Pitch Default baud rate Power Usage (from 5V Line)
10g 1216 Lithium 3V Cell 2.54mm 9600bps Acquire 25mA / Tracking 21mA / Cyclic PSM Mode 9mA
Board Absolute Maximums
5V Pin Max Voltage 30V Operating Temperature -45°C to +85°C

Pin Configurations

SCL	I^2C Clock Line. Not connected by default.
SDA	I^2C Data Line. Not connected by default.
TX	Serial TX Line 5V logic level. Connect to microcontroller RX line.
RX	Serial RX Line 5V logic level. Connect to microcontroller TX line.
EN	Enable pin. Connect to 5V to enable the module
TP	Time Pulse Output. When Locked the GPS will output a 1hz pulse to this pin.
5V	5V Supply Pin
GND	GND

GPS Setup, testing and data recording
To setup the GPS module we will follow guidance steps on Ava High Altitude Balloon Project main website, where they share their experiences with using the module that we have, thus hopefully making it easier for us to setup our module and avoid as many errors or potential issues as possible.
http://ava.upuaut.net/?p=738

GPS Data Interpretation
Use software provided by u-Blox:
https://www.u-blox.com/en/product-search?keywords=u-center&utm_source=en%2Fevaluation-software%2Fu-center.html

Software Development

From the start of this project, the most challenging aspect was always going to be the software. Our microcontroller has to handle a lot of data gathering and manipulation before making use of several different time critical outputs. As a general rule, we built up each module individually before integrating them together before final testing. This allowed us to easily identify error cases and problem areas such as timing errors – without modularising the software would be impossible to debug.

The first module we looked at was the GPS. A key challenge for our board was that it had to consistently identify and isolate the exact information we need from the stream of data provided by the GPS module, the uBlox MAX M8Q. This module uses NMEA, a standard format for conveying GNSS information. These NMEA messages are always precluded with a $, followed by 5 letters denoting the message type, followed by comma delimited data and a checksum. First, the code listens for the $ so it knows it's at the start of the message. It then checks the 5 character sentence identifier to look for the GNGGA sentence – which contains most of the position data we need.

Once we detect the correct sentence, the software copies that sentence into a buffer called gpsLatestReading for parsing. There is a flag included in the code such that the loop that reads in new sentences can't copy a sentence into gpsLatestReading while parsing, to avoid errors. Because each piece of data is comma delimited, we can use a for loop with several variables to parse out the individual data. We used three variables, used to denote the segment of information we're in (longitude, latitude, etc) the character index of that segment, and the character index of the strings we used to hold the parsed data. The software then cycles through the entire gpsLatestReading until all of the data has been parsed. Before each parse cycle, the data arrays are overwritten with '\0' to avoid any errors being introduced.
In order to read in the GPS data we had to set up a UART port with 8 data bits, 2 stop bits and no parity bits – reading at 9600 baud.

The next section of the software dealt with transmitting the data we've collected in a specifically formatted string. Our stringbuilder simply made use of a series of snprintf() functions to compile the information, calculate and append the CRC-16 checksum, then combine the two together to be transmitted in a buffer called txDatastring. This stringbuilding function was called at the end of every string transmission to allow a constant stream of data.

In order to transmit our information, we need to have a method of transmitting 50 bits/second, or 50 baud. Unfortunately we're already using the UART port on the ATmega328 for the GPS and at any rate, it doesn't support bitrates anywhere near 50 baud. Therefore, we used an overflow timer and interrupt routine to raise a flag when the next bit is ready to transmit (every 20ms). This then triggers a routine to

cycle through the txDatastring bit by bit and change the logic on our transmission pin accordingly.

We initially had some issues with the timing of the transmission due to the way we'd written the interrupts, however refactoring them to only assert a flag seemed to fix most of the timing issues we were having, as the software wouldn't get stuck in an ISR for too long.

It was important that we go out of our way to prevent any errors from occurring during the flight. If a data error happens after we release the payload, it could compromise the system's ability to correctly transmit its position and therefore result in a lost flight. Measures such as overwriting each data array with null characters and including a framework for implementation for a watchdog timer will help with this. The watchdog timer will be setup to monitor the transmitted output and match it against a known format. If the transmitted data is consistently mismatched against the template format, the microcontroller will be commanded to restart.

Transmitter Tuning Network

A resistor network needed to be set up which would tune our signal for the transmitter. The output from this network needed to be one of two voltages, which would be seen as high and low. It is know that for our situation 2000Hz/V corresponds to a 425Hz shift.

A voltage shift, the gap between the high and low signal, could then be calculated. Through a method using equivalent circuits, equations can be derived in order to calculate what values of resistors are needed.
However, due to the numerous possibilities of combinations, and our desire to have it as finely tuned as possible, a script was written in Python 2.7 to test all combinations and see if they met the set conditions.

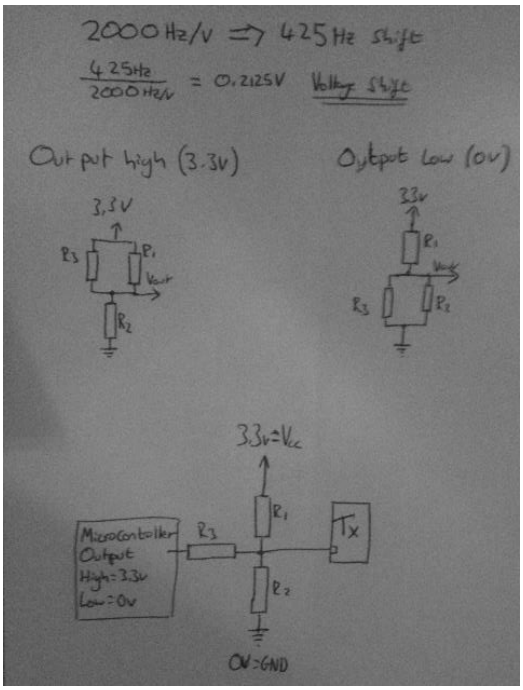


Figure 5: Equivalent circuits

The voltage shift was calculated as 0.2125V. The lowest diagram shows how the network will be implemented, while using the two equivalent circuits, two values of Vout can be calculated using a potential divider equation.

```
#   Calculating optimum tuning resistors
#   Written by Thomas Lewis, lewistr

Vcc = 3.3;
res1= [1000, 1200, 1300, 1500, 1600, 1800, 2000, 2200, 2400, 2700, 3000, 3300, 3600, 3900, 4300, 4700, 5100, 5600, 6200, 68
res2 = res1;
res3 = res1;
VS = float(format((0.2125),'.5f')) #VS is the target voltage shift
VSA = float(format((0.001),'.5f')) #The Voltage Shift Allowance is the +-

################

def results():
    print "########"
    print "Res1 = ", r1
    print "Res2 = ", r2
    print "Res3 = ", r3
    Rh = r1pr3 + r2
    Rl = r2pr3 + r1
    Ifluc = abs((Vcc/Rh)-(Vcc/Rl))
    print "Vdiff: ", Vdiff
    print "Ifluc: ", Ifluc


#The results function prints all the needed information if the conditions are met
################


#Loop through all possible resistor combinations

for r1 in res1:
    for r2 in res2:
        for r3 in res3:
            r1pr3 = (r1*r3)/(r1+r3) #r1 parallel r3
            r2pr3 = (r2*r3)/(r2+r3) #r2 parallel r3

            Vh = (Vcc*r2)/(r2+r1pr3)    #value of Vout for high state
            Vl = (Vcc*r2pr3)/(r1+r2pr3) #value of Vout for low state
            Vdiff = abs(Vh-Vl)          #Difference between Vh and Vl i.e the Voltage Shift

            if r1 == r2: #if r1 and r2 are the same value. This means that the total resistance of the network is constant,
                if (VS - VSA) <= Vdiff <= (VS + VSA): #is our voltage shift close to our target?
                    results()
```

Figure 6: Tuning Network Script

All the constants are defined between lines 4 – 9, which can be referenced later. The main aim of the script was to run through every combination for this 3-resistor

network and calculate the outputs for both the High Output and Low Output equivalent circuits (Figure 6). Resistors tested were 1KΩ – 91kΩ in the e24 series.

The difference between the two outputs in called the Vdiff, and was compared to the VS, our set Voltage Shift. VSA, the voltage shift allowance which was a value saying how close the calculated Vdiff could be to the VS for it to be printed as an answer.

It was decided to keep r1 and r2 the same value as it made component selection simple and kept current constant, something, which would very unlikely affect, our system but when trying to be this precise it is better to have fewer variables with which to be concerned.

Results and Error Analysis

The script gave us the following output.

```
########
Res1 =  2200
Res2 =  2200
Res3 =  16000
Vdiff:  0.212336719884
Ifluc:  0.0
########
Res1 =  3300
Res2 =  3300
Res3 =  24000
Vdiff:  0.212336719884
Ifluc:  0.0
```

Figure 7: Tuning Network Results

The needed voltage shift was previously calculated as being 0.2125V with the result of 0.2123V (to 2 d.p) being acceptable for what was needed and 3k3Ω and 24kΩ being chosen.

It is important to know the level of error that is being dealt with when designing and building circuits that require precision. With the help of a Python script the error for the tuning resistor network was calculated.

```
#Error Calculations

r1 = 3300 # r1 = r2
r3 = 24000
tol = 0.05 # 5%

dr1 = r1 * tol
dr3 = r3 * tol

#print dr1, dr3

##########
# Min and Max for parallel combination

r1max = r1 + dr1
r1min = r1 - dr1
r3max = r3 + dr3
r3min = r3 - dr3

r1pr3 = (r1 * r3)/(r1 + r3)
r1pr3max = (r1max * r3max)/(r1max + r3max)
r1pr3min = (r1min * r3min)/(r1min + r3min)

#print r1pr3max, r1pr3min, r1pr3

Diff = r1pr3max - r1pr3min

#print Diff

print "R1 = ", r1, "\tR2 = ", r1, "\tR3 = ", r3
print "Error: ", ((Diff/2)/r1pr3)*100
```

Figure 8

The values for the resistors and the tolerance are first declared. The ðR1 and ðR3 is then calculated using our 5% tolerance. The maximum and minimum possible values for R1 and R3 are found and then used to find the largest and smallest values possible for them in parallel together.

```
R1 =  3300      R2 =  3300      R3 =  24000
Error:  5.00017046036
```

Figure 9: Error Calculation Results

The result can be taken as 5%. As all resistors have 5% tolerance, and assuming the 3.3V supply has little to no error, the High and Low voltage signals will have an error of 5%. This is very small and it was agreed that this would make no significant difference.

We can assume the supple voltage to have very little error due to the testing of the LD1117, shown later in the report.

Prototype 1 Specification and Review

The purpose of the initial prototype is to allow testing of the power supply, GPS module, transmitter and receiver before progressing onto PCB design and further development.

Using our Proteus simulation and calculations we produced a theoretically viable design. We then ordered all of the relevant components (see: bill of materials) and progressed onto building the design on breadboard.

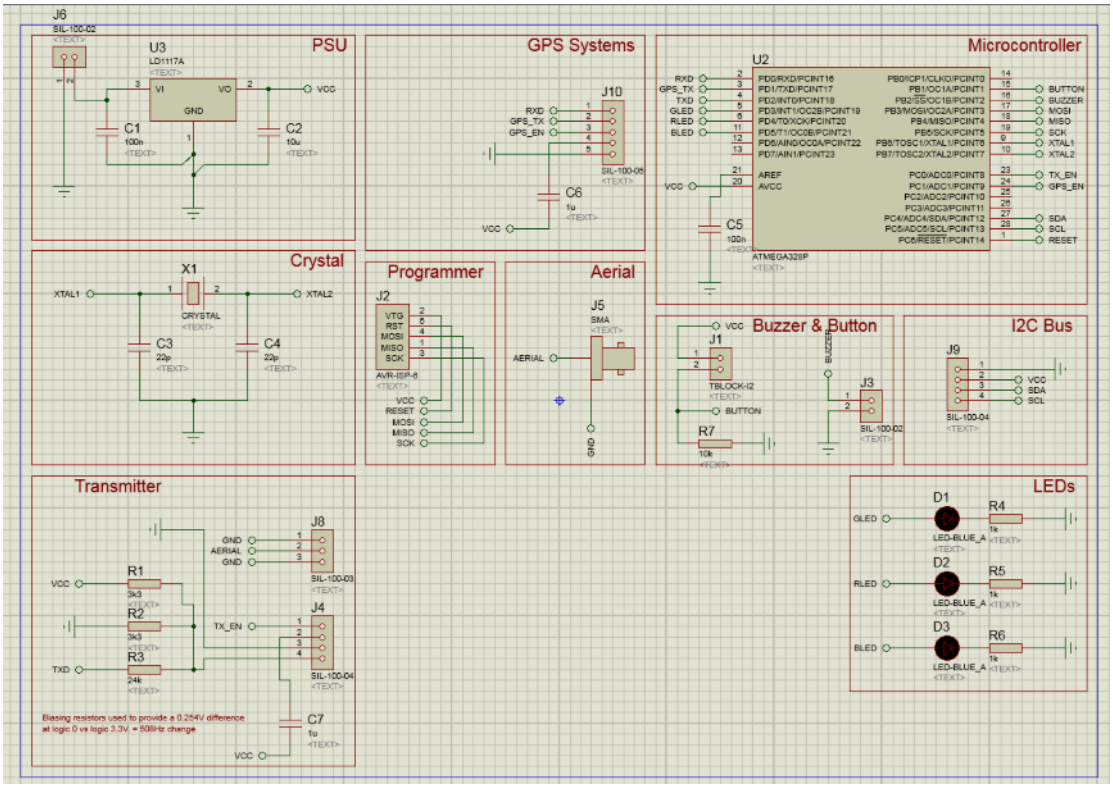


Figure 10

Above is our finalised circuit design with theoretical calculations for component values.

Below is a photo of our bread-boarded circuit, with a 6 pin programming connector for the chip. Also note the transmitter and the receiver that is located in the PC's USB slot.

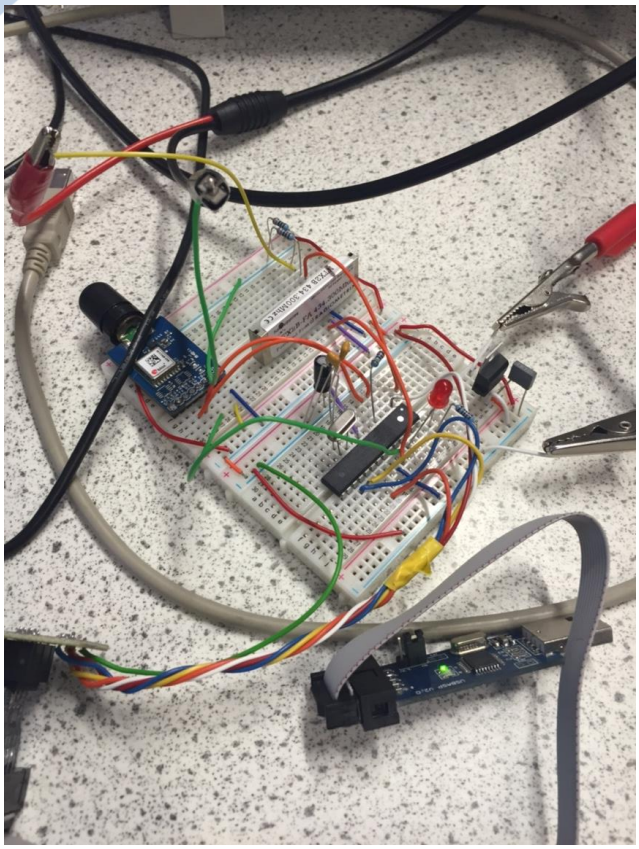


Figure 12

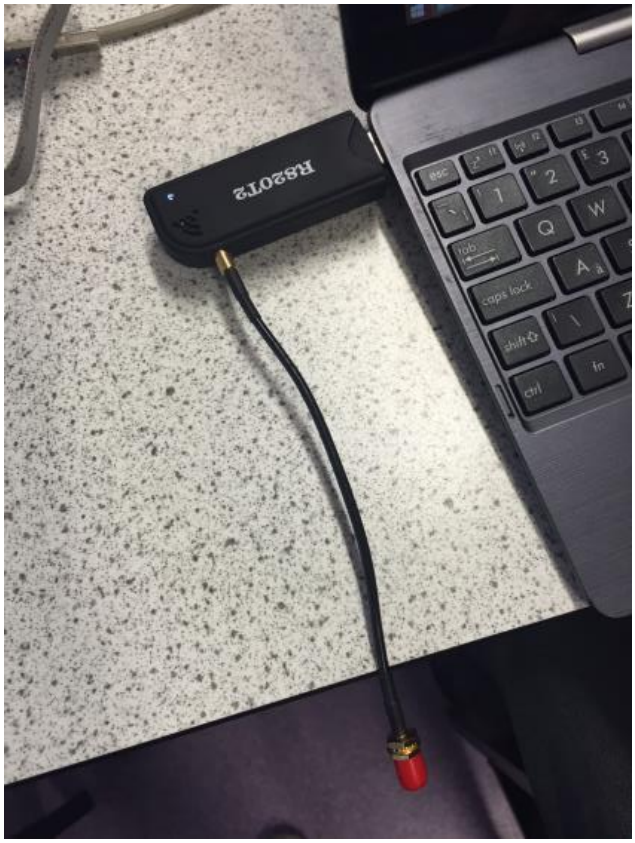


Figure 11

Above is the final design for P1. You can see the GPS Antenna (top left), 6 pin programmer input (bottom right) and the long strip in the centre of the board is the transmitter. Also on the board are an ATMEGA 328, 8MHZ Oscillator and 3.3v voltage regulator.

Figure 11 shows the receiver that was used for short-range communication to the design; we used the more substantial LPRS YAGI-434A antenna with a larger range with the completed PCB. The software we have use to interpret the data received is SDR software defining the radio transmission (allows us to communicate with receiver), use VB cable to send signal to DL-FLDigi from the receiver, which interprets the data in a way which we can understand.

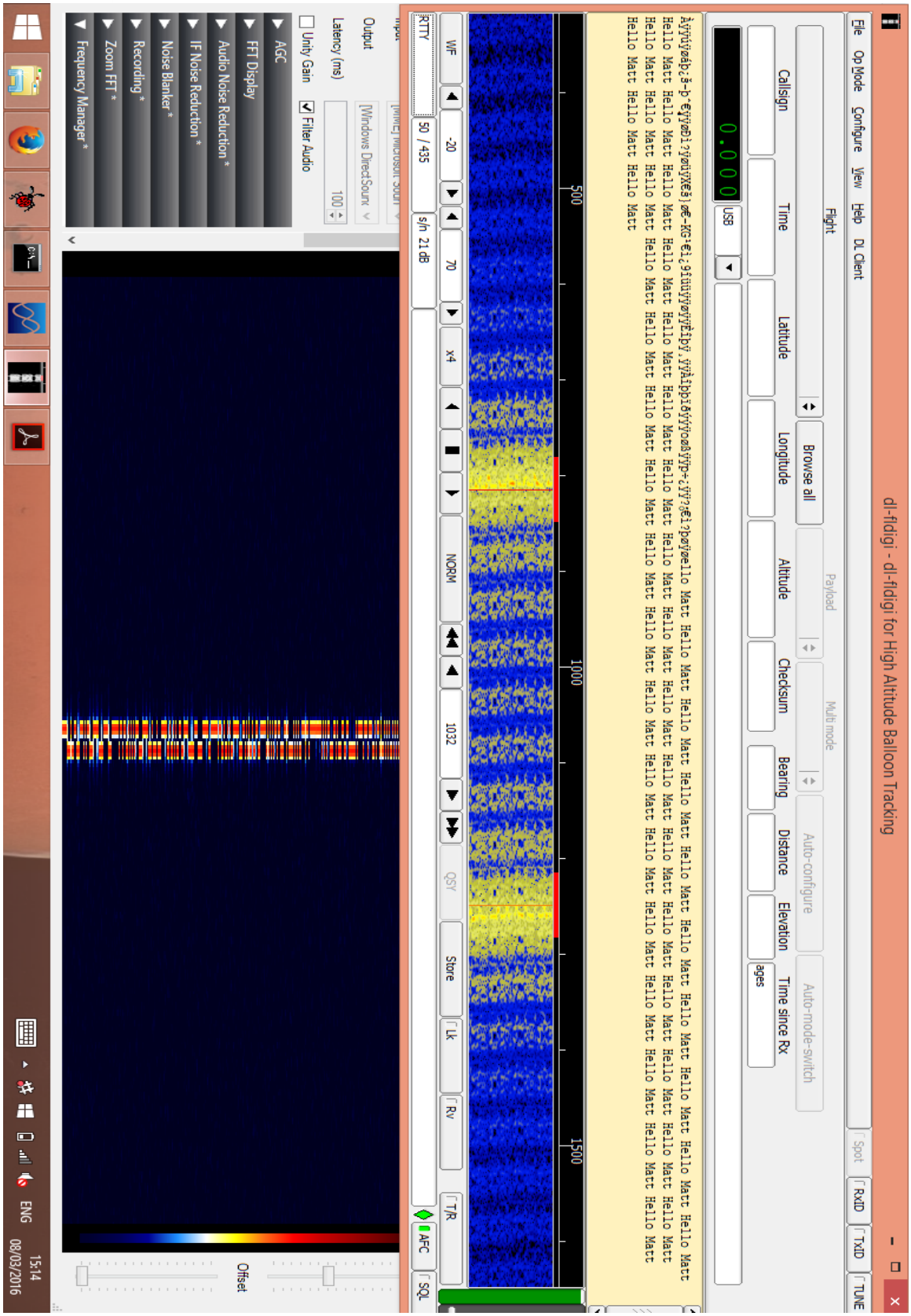


Figure 13

The image above demonstrates that we can transmit a simple string of "Hello Matt" wirelessly.

Reviewing Prototype 1 we can clearly see that we can transmit data over a short distance. Our calculations for component sizes were correct and produce a strong enough signal to transmit data.

The next steps will involve refining the design and produce a PCB design that we will mount the design on for the flight.

PCB Design

Circuit and PCB Design

The initial design of the board started with a ISIS 8 circuit design, that incorporated all the possible features that were intended to be used. Due to the limitations of the program many T-Blocks were used to simulate the hardware when transitioning over to ARES. There was initial difficulty in getting the right package for the ATMEGA328-PU, however this was quickly resolved by downloading and installing the correct shape package, that was conveniently the same size as an Arduino's socket. The individual components, indicated by the red outlines in Figure 14. Are connected via "default" pins, which are then allocated the target's name, simulating a wire.

8MHz Crystal

Due to the sensitive nature of GPS calculations, it was imperative that the timing was much more accurate than the internal clock could provide. As such it was decided that an external clock would be needed, which was provided with a HC49 8MHz Crystal. To keep the voltage constant on both sides of the crystal, 2 capacitors were used, with the values specified in the data sheet. The IDE was then used to set the IC to read from the external crystal by setting the internal fuse bits.

Programmer

It was decided for ease of access that a programmer should be installed on the board to allow for hotfixes and patches to be applied during the testing phase. We also had quite a bit of legroom inside out payload to allow for this. The alternative would have been to take the IC out of its housing to then programme, which would have lead to the risk of bending the pins irreversibly.

I2C Bus

Though not an intended part for this module, we added an I2C bus onto the board so that we could implement a sensor breakout later on. This would allow us to take far more data on the initial flight, and contribute to the “Act in Space” Hackathon which is mentioned in another section of this report.

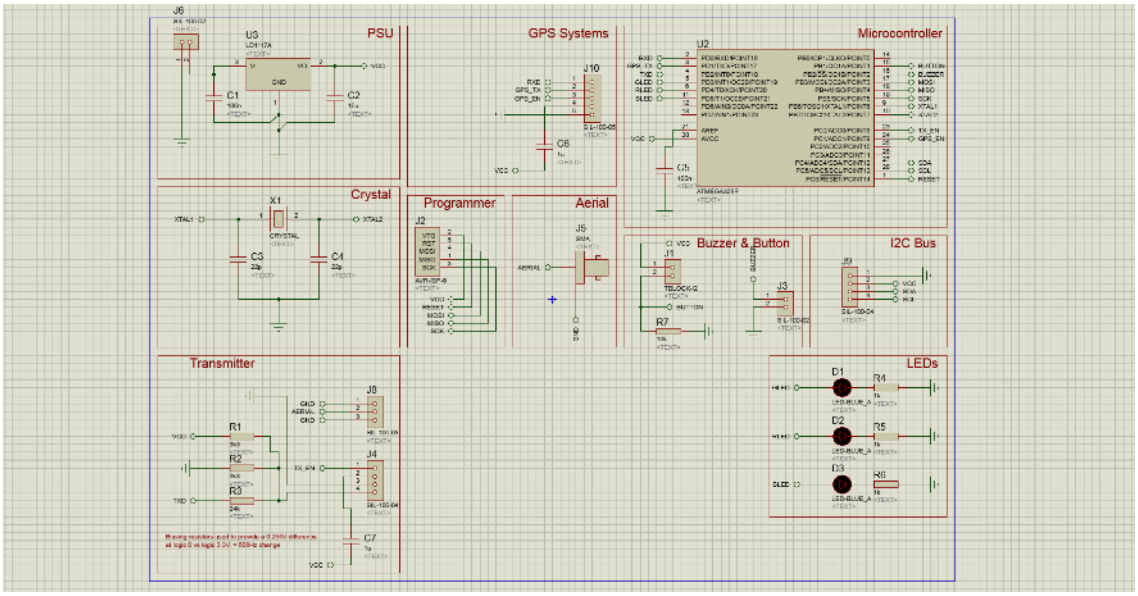


Figure 15: ISIS 8 Circuit Design

ARES PCB

Below in Figure 15 shows the final design and track display of the final PCB board. Due to the complexity of the design, and the specific size requirements of the board, it made it almost impossible to keep all the tracks on a single layer, without compromising on these constraints. The decision was therefore made to make a 2 layered board, with the red tracks indicated top copper, and blue the bottom copper. Our SMA pin had to be mounted either on the leftmost, or rightmost sides of the PCB due to the polystyrene wall that the board would be mounted next to.

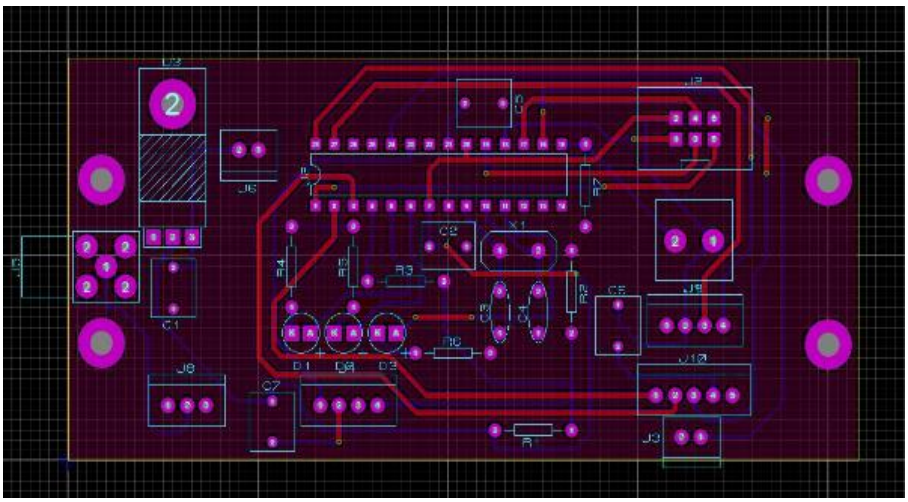


Figure 16: ARES 8 Circuit Design

Our board was etched and created by Ragworm, on a very quick turn around to give us enough time to test and validate the board before the demonstration date. Unfortunately, it appeared that the SMA pin package that was native to PROTEUS was in imperial units, not metric and therefore we had to use a drill to make our SMA fit. The mounting hole with the label '2' in figure 16 is to be used for a heat sink for our regulator, as it was found that as the atmosphere reduces in pressure, there is less surface area for the heatsink to dissipate heat.

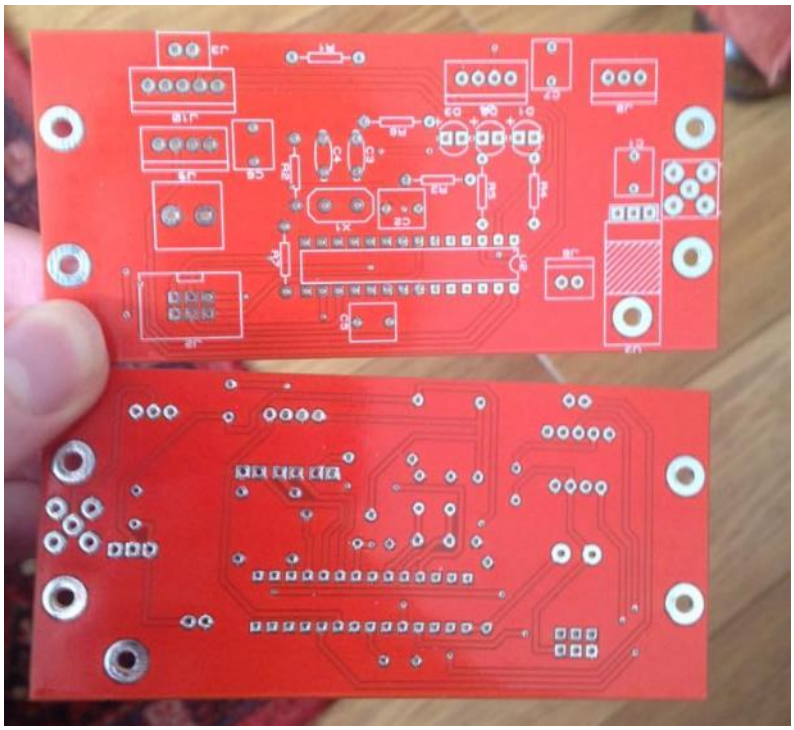


Figure 17: Final PCB Etching

Halo Antenna Design

The payload antenna will be based on an already proven design by the UK High Altitude Society (UKHAS).

It is a ¼ wave ground plane antenna, which is useful as it has an omnidirectional radiation pattern meaning it will emit equal radio power in the direction of the all the receivers, thus producing a better signal.

It is constructed from a single core coaxial cable (referenced below). We used the UKHAS's build instructions and modified the cable to meet out specifications.

A detailed description can be found on the UKHAS website [6].

Below is an image of our finished antenna.

It will be fixed to the fuselage of the weather balloon and connected to the transmitter module.

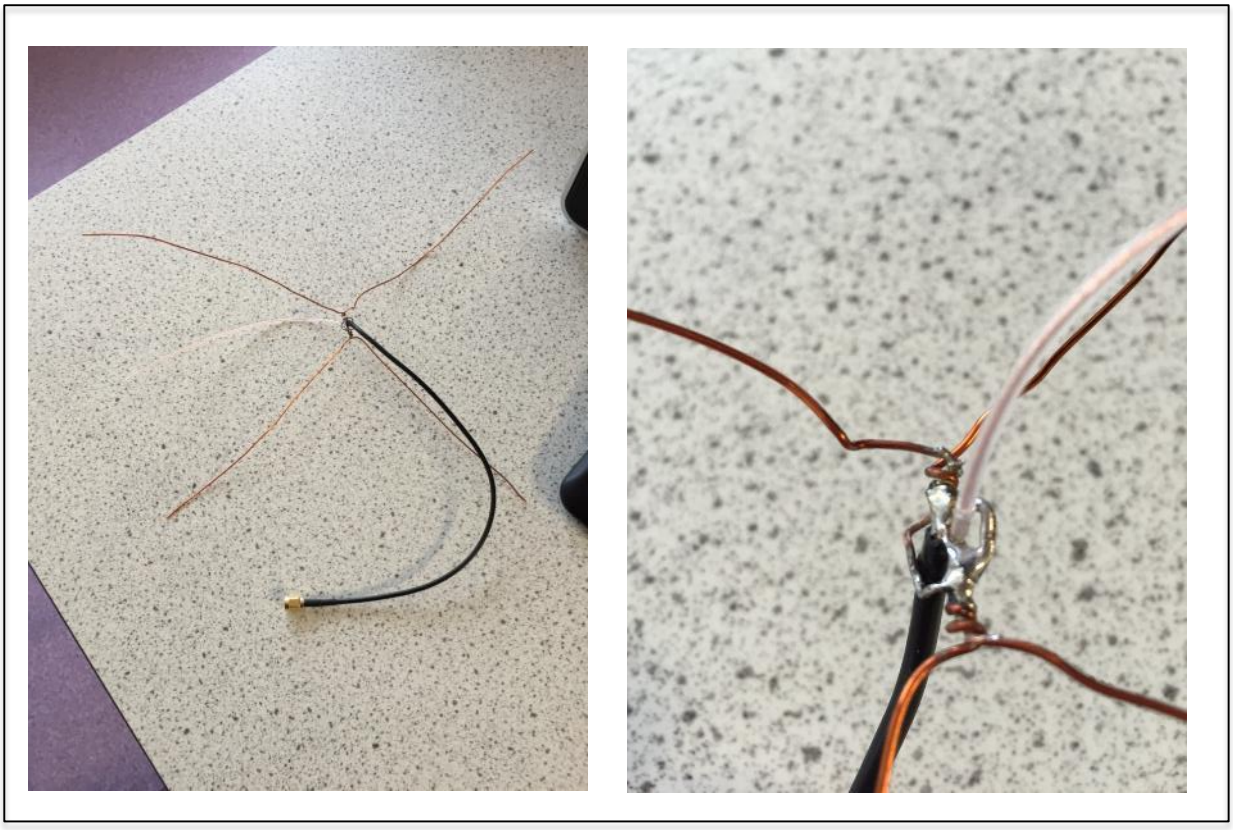


Figure 18

Testing

Before a launch all aspects of hardware and software need to be tested and prove that the requirements have been met. The major risks are components not surviving the harsh environment of the atmosphere and loss of signal.

Voltage Regulator Testing

As the data sheet for the LD1117 did not state a minimum operating range, the component was tested within the range of 0V-12V.

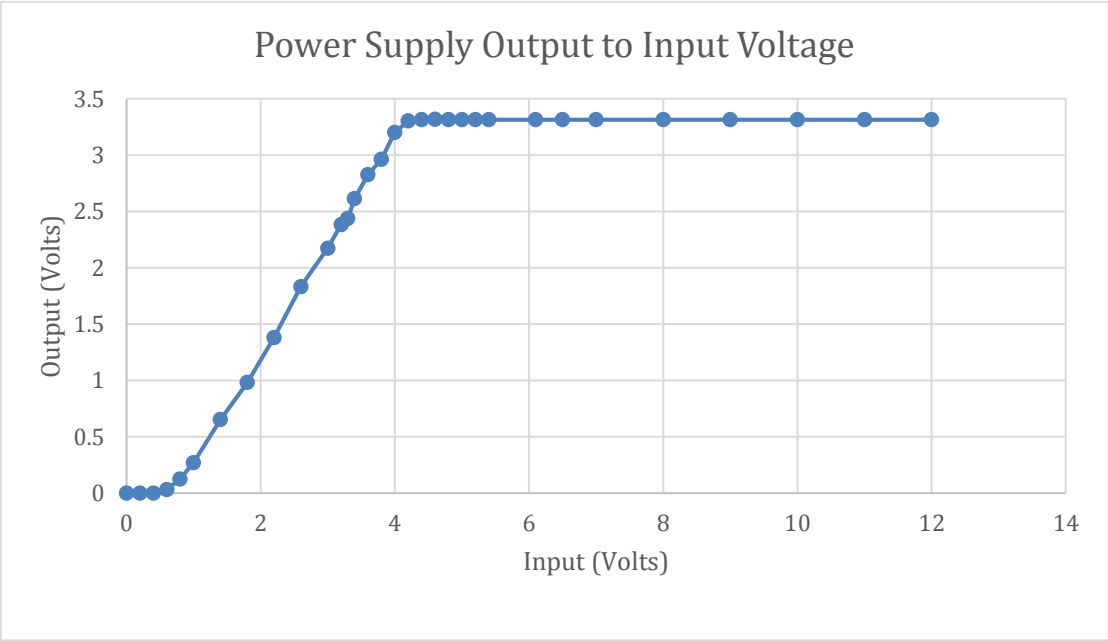


Figure 19: Data gathered from testing the PSU (LD1117) on breadboard

It was determined that if the input voltage to the regulator drops anywhere below 4V, we'd see a drop on the output. That being said, this was done without any current draw and it has not been observed what temperature may do to the regulator.

GPS Testing

Initial Testing

Once we had the data-string transmission finalised and working we took the system out into Birmingham to conduct a simultaneous test of the GPS and radio systems. We expected the signal to drop as soon as the antenna left line of sight, however we discovered that we could still receive signal at the main doors of Aston University while the board was out at Moor Street Station and the Bull. We suspect that the radio signal may have been bouncing between buildings. We then decoded the GPS

location data from the NMEA format (DDm.mmmmm) into standard decimal, which we then put into Google maps to display an approximate test path.

Long Range Testing

In order to do another test of the radio in line of sight, we sent the payload out into the country in order to put as much distance between the two as possible before the flight. The signal was decoded perfectly cleanly, with huge amounts of strength despite approximately a mile between the two.

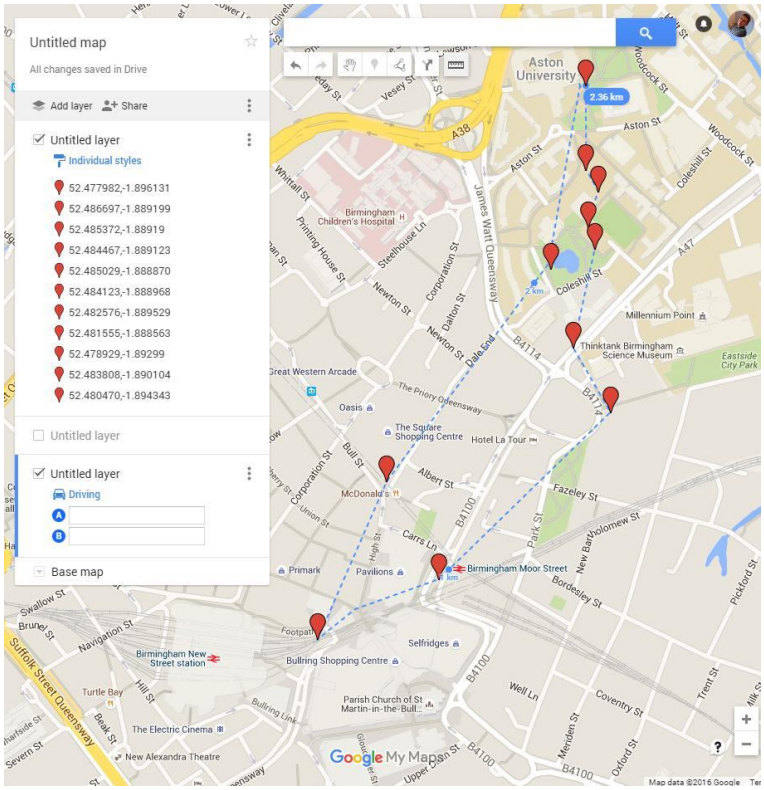


Figure 20

Temperature Testing

Before we fly it's vital that we try and qualify our board in as close to the environmental conditions that it would experience in flight as possible. Originally we approached the Chemistry and LHS departments, knowing that they make use of freezers for storing samples, so see if they had any freezers that could be adjusted to -40degC. Unfortunately the only ones available were fixed at -80, which would have killed our electronics.

However, we then had the idea to use a home freezer, as they go down to roughly -22. This would simulate the expected temperature in the insulated and sheltered payload bay. The first test was used with a standard 9V battery to see what would cause the transmission signal to cut out. We found that the board performed perfectly however there was a point where the battery died due to the temperature dropping too low.

We positioned several thermocouples on the voltage regulator, PCB and elsewhere in the freezer. Interestingly we found that the regulator was slightly above the PCB temperature until it got to really low temperatures. This difference would be amplified at these low temperatures at altitude, as there would be much less air to aid in heat radiation.

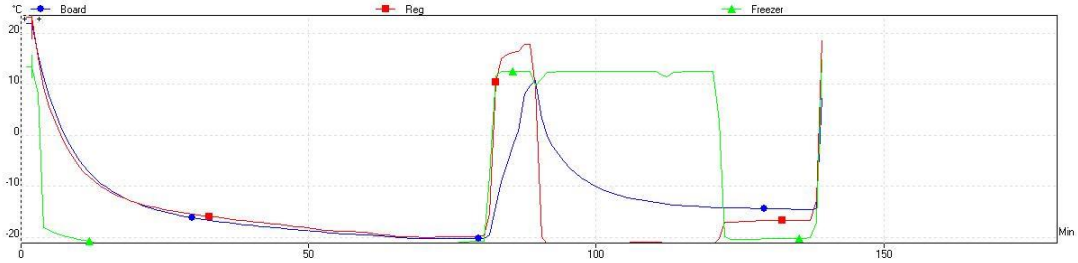


Figure 21

We then did a second test with the L91 lithium ion batteries, designed to supply voltage down to at least -40. This time the board performed perfectly throughout the duration of a three hour test.

Final Build

Once the circuit board and ¼ wave antenna had been build, we were ready to look into integration with the payload and begin the first full scale testing. We discovered that the PCB fit really snugly, although the size would need to be adjusted for a more consistent fit in the future.

Due to the need to develop the system into a glider – enabling flyback capability – we needed to both make sure the batteries were balanced inside the fuselage but also out of the way of the main payload bay, in order to fly external experiments and sensor payloads. Therefore we rigged up a harness by wiring two 2xAA battery packs in series, and mounted them either side of the electronics bay, allowing the power cable to tangle neatly onto the PCB.

Next, we mounted the antenna by threading the coaxial tail through the nose of the fuselage and plugging it into the SMA connector. Finally, we cut holes in the sensor bays on the airframe and threaded our sensors in place.

It quickly became apparent that we had left the antenna coaxial and the sensor wiring much too long, so finally we measured and cut the wiring down to a more efficient length. Although we installed regular header pins to the wires, in the future we would like to use a more secure connector format.

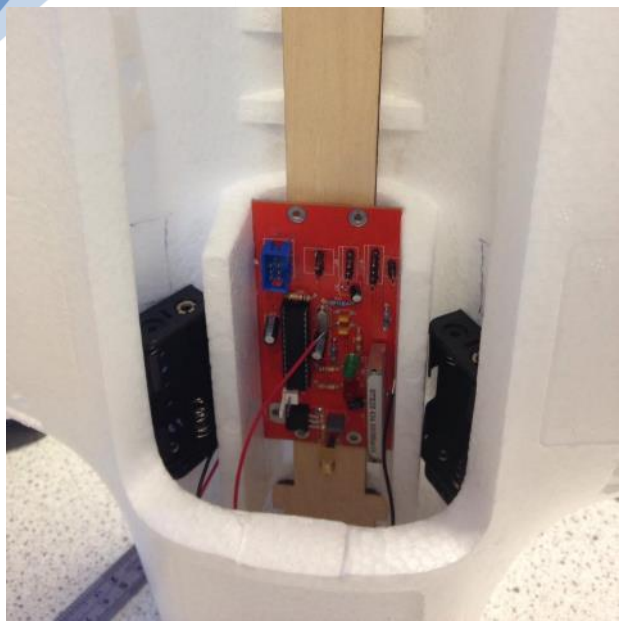


Figure 23

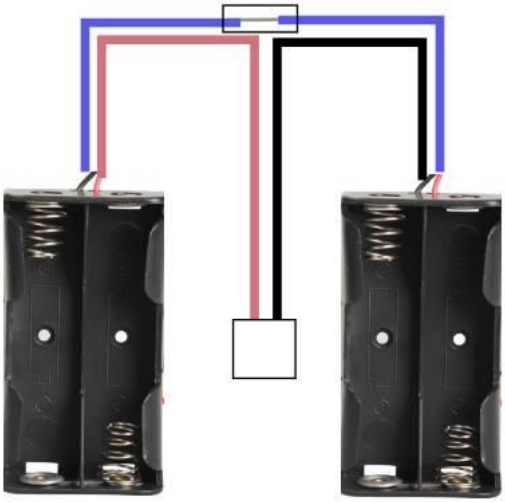


Figure 22

Finance

The finance of the project has been closely monitored and continually updated. Along with the Gantt chart and action minutes it is a vital part of the project and in industry it would paramount.

The finances were broken down into two sections. The first being costs to the project via our standard university suppliers. For each purchase a product code, web link, quantity, individual and total price was quoted. The second category was costs from non-university suppliers where we purchased the components ourselves. Again the same information was present for each purchase.

Using the details stated we have had clear control on total spending for the whole project. It also allows us to know where we sourced our components from so in the future we can quickly source more. Although costly to produce one, if other high altitude groups adopted our design we could severely bring the build price down for bulk production.

See Appendix 1.3

Uses and Future Development

The Lancer II is being designed to fulfil the use of a reliable experiment and data collection platform in the upper atmosphere. By switching to a glider based final descent stage, the Lancer II will be able to accurately pinpoint and land in an area of the user's choice. These landing zones can be positioned along the predicted flight path of the HAB and will therefore drastically reduce the numbers of HAB flights that were lost and unrecovered. It will also help to eliminate the possibility of the HAB landing in an unsafe area, for instance in a built up area or over a lake.

The Lancer programme also aims to provide an interface for external experiments, such as those set up by high school or university students. We will build a weight margin into the launch system (balloon and parachute) enabling us to launch a variety of experiments and return them safely to the ground.

In addition to general HAB flights, the Lancer II will be perfect for use as a replacement for atmospheric probe rockets. In order for orbital launch vehicles to safely launch, they need to have very precise atmospheric readings in order to prevent anomalies such as a lightning strike in cloud cover or extremely high wind shear. Previously HAB flights were not used for this capability due to an inability to determine where the payload would land; obviously the launch site itself needs to be kept clear at all costs. This led to disposable rockets being used to determine atmospheric data. However, these rockets are expensive and do not provide a full spectrum of atmospheric data during the ascent.

The Lancer II will solve both of these problems by allowing data collection to run for the entire flight and then enabling a controlled landing. This will ensure that the launch controllers have very accurate and detailed data from the environs immediately around the Launch Vehicle without having to worry about the airspace immediately over the rocket being violated.

There are numerous avenues that we will be exploring further. The main features we hope to integrate for revision 2 are:

- Ability to co-ordinate multiple sets of sensors at once
- Ability to control an autopilot module
- Ability to process navigational data in-flight
- Ability to record all flight data on local SD card

We are also looking into converting to a surface mount format in order to save on weight and, more critically, space on the PCB. This will allow us to fit more electronics into the small space we've allocated for the Lancer's flight computer and therefore make more space available for scientific payloads.

Project Review

During the process of developing and building the revision 1 avionics board for Lancer I, we came across several things to update or improve during the building of revision 2.

The first major change would be to transition from PTH (plated through-hole) technology to surface mount, allowing us to make use of smaller and lighter components in the same space. While the board will become harder to populate by hand and will require professional etching, the benefits make it worthwhile.
We will be reducing the width of the board by approximately 2mm in order for it to fit properly at the bottom of the electronics bay in the airframe.

One large mistake we noticed was the positioning of the C7 capacitor, which in revision 1 was positioned between the pins for the radio transmitter, which has a group of 3 and a group of 4 pins spread apart. Having only caught this error in the populating of the PCB, we were forced to adapt the mounting of the capacitor in order to make it fit. The capacitor would be repositioned for revision 2.
Another issue we had was the size of the pin holes for the SMA connector. While everything else fitted perfectly, the SMA connector needed to have its pins bored out a fraction of a millimetre to make them fit.

After fixing these problems, the main priority will be to reduce the size and weight of the board to make room for additional features. These include better header pins for external sensors, an interface for an SD card datalogger, as well as a power source and data collection pins for external experiments. In particular we are keen to incorporate a CO2 sensor as part of a project with a member of the UKHAS (UK High Altitude Society) community.

References

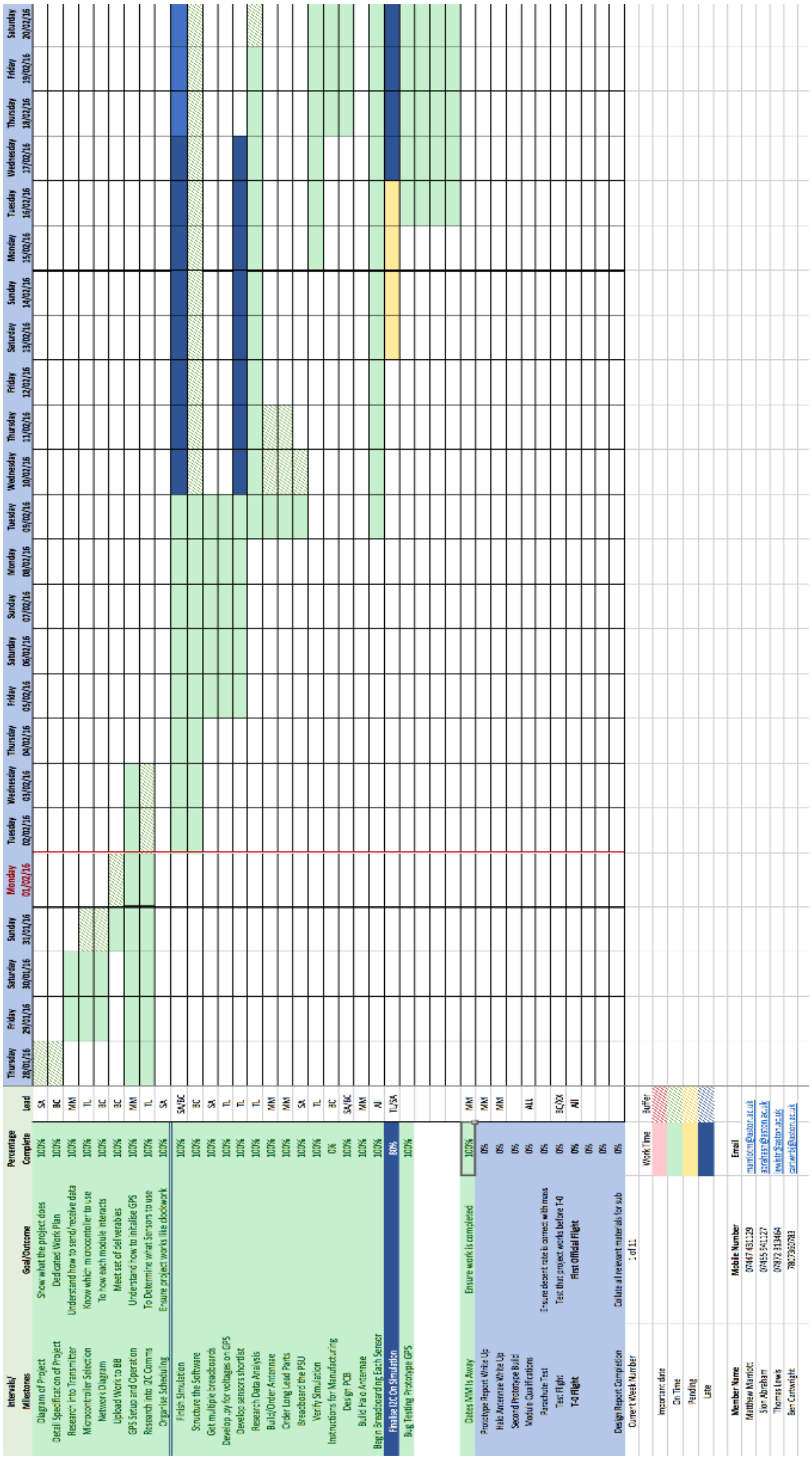
[1] - Ben Cartwright. 2016. Aston AstroSoc Slack Page. [ONLINE] Available at: https://astrosoc.slack.com/messages/eng-comms/details/. [Accessed 19 April 2016].
[2]. GPS module – accessed 26/01/2016 - http://ava.upuaut.net/store/index.php?route=product/product&path=59_60&product_id=68
[3] Radio module – accessed 27/01/2016 - http://www.radiometrix.com/content/ntx2b
[4] Radio module data sheet – accessed 27/01/2016 - http://www.radiometrix.com/files/additional/ntx2bnrx2b_0.pdf
[5] Battery specification PDF – accessed 26/01/2016 - http://data.energizer.com/PDFs/l91.pdf
[6]-Payload Antenna – accessed 20/02/16 https://ukhas.org.uk/guides:payload_antenna

Reference Figures

Figure 1.1-

Date	26-Jan	Tasks	Responsible	Due	Notes
Attendance	SA	Detail Specification of Project	BC	27/01/16	Start up of the project. Our roles were discussed and it was decided we had to write up a spec to allow for the rest of the project to continue. Whilst the spec was being written up, the rest of the team was to research into there induvidually allocated tasks. It was decided that report work would be continuous
	MM	Research into I2C Comms	TL	30/01/16	
	BC	Microcontroller Selection	SA	30/01/16	
	TL	Network Diagram	BC	30/01/16	
		Diagram of Project	SA	30/01/16	
		GPS Setup and Operation	MM	28/01/16	
		Research into Transmitter	MM	30/01/16	
		Organise Scheduling	SA	26/01/16	
Date	02-Feb	Tasks	Responsible	Due	Notes
Attendance	SA	Research into I2C Comms	TL	03/02/16	Microcontroller selected ; ATMEGA328PU GPS chosen UBLOX MAX-M8Q 5V Ben is currently abstracting the network, with each induvidual segment needed SA and BC working on Sim for the next week TL and MM finalising our knowledge on sensor communication (I^2C) Agreed that MM would be treasurer, keeping Excel of costs
	MM	GPS Setup and Operation	MM	03/02/16	
	BC	Finish Simulation	SA/BC	09/02/16	
	TL	Structure the Software	BC	09/02/16	
Date	05-Feb	Tasks	Responsible	Due	Notes
Attendance	SA	Get multiple breadboards	SA	09/02/16	A quick coffee break, whereby the finer details of the project were discussed. A number of issues were raised and resolved. We need to do I2C calibration, all the GPS items and the final Tx in the simulation
	MM	Develop .py for voltages on GPS	TL	09/02/16	
	BC	Develop sensors shortlist	TL	09/02/16	
	TL	Begin breadboarding each sensor	SA, MM, TL	16/02/16	
		Finalise I2C on simulation	TL SA	16/02/16	
Date	09-Feb	Tasks	Responsible	Due	Notes
Attendance	SA	Finish Simulation	SA/BC	09/02/16	SA working on gantt chart, expanding upon our goals to make clearer. MM asked for everyone to contribute to costs, group agreed. MM also ordering parts today. BC Finishing GPS passer, GPS initialisation and Bit banging. TL at Black and Decker Interview
	MM	Build/Order Antennae	MM	17/02/16	
	BC	Order Long Lead Parts	MM	17/02/16	
Appologies	TL	Build and test the PSU	SA	09/02/16	
Date	16-Feb	Tasks	Responsible	Due	Notes
Attendance	SA	Sensor Shortlist	TL	**16/02/16**	Ordering of the sensors has delayed pretty much all of our tasks. Agreed that we are to work hard on this date to get everyone back on track.
	MM	Finish Simulation	SA	**16/02/16**	
	BC	Order Sensors	MM	**16/02/16**	
	TL	Software	BC	**23/02/16**	
		Update Finances	MM	**16/02/16**	
Date	23-Feb	Tasks	Responsible	Due	Notes
Attendance					Code generation is going well. The PCB design is coming together we are just finalising the simulation, the aim is to have it done by Friday in time for the second set of deliverables. Our breadboarded circuit proves to us that we can receive and decode data wirelessly from the design. MM is going to build the halo antenna which will be attached to the fuselage. We will also look at the predicted weight of the total fuselage with components to enable to determine which parachute and balloon to order.
	SA	Minutes and gannt chart update	MM	**23/02/16**	
	MM	Finish and test string building	BC	**23/02/16**	
	BC	Build quaterwave halo antenna	MM	**23/02/16**	
	TL	PCB Design	SA	**26/02/16**	
		Finish Simulation	SA	**26/02/16**	
		Finish Sensor Writeup	TL	**23/02/16**	
		I2C Implementation	TL	**23/02/16**	

Figure 1.2-



Intervals/ Milestones
Goal/Outcome
Percentage Complete
Lead
Thursday 28/01/16
Friday 29/01/16
Saturday 30/01/16
Sunday 31/01/16
Monday 01/02/16
Tuesday 02/02/16
Wednesday 03/02/16
Thursday 04/02/16
Friday 05/02/16
Saturday 06/02/16
Sunday 07/02/16
Monday 08/02/16
Tuesday 09/02/16
Wednesday 10/02/16
Thursday 11/02/16
Friday 12/02/16
Saturday 13/02/16
Sunday 14/02/16
Monday 15/02/16
Tuesday 16/02/16
Wednesday 17/02/16
Thursday 18/02/16
Friday 19/02/16
Saturday 20/02/16
Diagram of Project | Show what the project does | 100% | SA
Detail Specification of Project | Dedicated Work Plan | 100% | BC
Research into Transmitter | Understand how to send/receive data | 100% | MM
Microcontroller Selection | Know which microcontoller to use | 100% | TL
Network Diagram | To how each module interacts | 100% | BC
Upload Work to BB | Meet set of deliverables | 100% | BC
GPS Setup and Operation | Understand how to initalise GPS | 100% | MM
Research into I2C Comms | To Determine what Sensors to use | 100% | TL
Organise Scheduling | Ensure project works like clockwork | 100% | SA
Finish Simulation | 100% | SA/BC
Structure the Software | 100% | BC
Get multiple breadboards | 100% | SA
Develop .py for voltages on GPS | 100% | TL
Develop sensors shortlist | 100% | TL
Research Data Analysis | 100% | TL
Build/Order Antennae | 100% | MM
Order Long Lead Parts | 100% | MM
Breadboard the PSU | 100% | SA
Verify Simulation | 100% | TL
Instructions for Manufacturing | 0% | BC
Design PCB | 100% | SA/BC
Build Halo Antennae | 100% | MM
Begin Breadboarding Each Sensor | 100% | AI
Finalise I2C On Simulation | 80% | TL/SA
Bug Testing Prototype GPS | 100%
Dates MM Is Away | Ensure work is completed | 100% | MM
Prototype Report Write Up | 0% | MM
Halo Antennae Write Up | 0% | MM
Second Prototype Build | 0%
Module Qualifications | 0% | ALL
Parachute Test | Ensure decent rate is correct with mass | 0%
Test Flight | Test that project works before T-0 | 0% | BC/XX
T-0 Flight | First Official Flight | 0% | All
0%
0%
Design Report Completion | Collate all relevant materials for sub | 0%
Current Week Number | 1 of 11
Work Time | Buffer
Important date
On Time
Pending
Late
Member Name | Mobile Number | Email
Matthew Marriott | 07447 431129 | marriotm@aston.ac.uk
Sion Abraham | 07455 941127 | abrahasn@aston.ac.uk
Thomas Lewis | 07872 313464 | lewistr@aston.ac.uk
Ben Cartwright | 7807360783 | cartwrbj@aston.ac.uk

Figure 1.3-

Expendature Tracking

Purchase No.	Item Name	Item Code/ web link	Supplier	Quantity	Individual Cost	Total Cost	Cost per member (split between 4)
1	GPS Breakout Board	http://ava.upuaut.net/store/index.php?route=product/product&product_id=51	Hab Supplies	1	28.79	28.79	7.1975
2	Radio Module 434MHz	http://ava.upuaut.net/store/index.php?route=product/product&path=71_63&product_id=92	Hab Supplies	1	21.07	21.07	5.2675
3	Postage	Postage from Hab supplies	Hab Supplies	1	4.2	4.2	1.05
13	Atmospheric Sensor Breakout	https://www.coolcomponents.co.uk/atmospheric-sensor-breakout-bme280.html	Cool Components	1	16.56	16.56	4.14
14	Sensor Breakout Postage	Postage from Cool Components	Cool Components	1	2.31	2.31	0.5775
15	PCB	Ragworm	Ragworm	2	20	40	10

	Ordered
	Need to order
	We have

Team Member	Amount Paid	Amount to pay
MM	18.5	9.7325
SA	0	28.2325
BC	13.5	14.7325
TL	18.5	9.7325