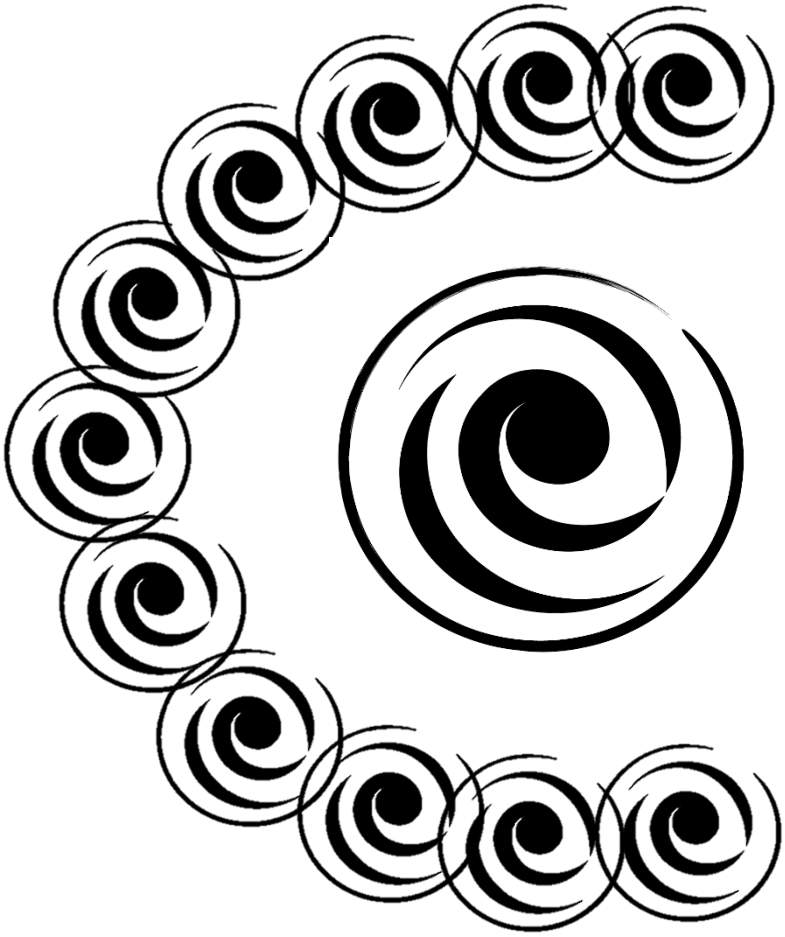


CYCLONE

Ashwin Ahuja, Benjamin Yass, Quentin Guéroult, William Eustace, Hugo Aaronson, Daniel Halstead, James Crompton, Nicholas Palmer, Philip Fernandes

CANSATS IN EUROPE

FINAL INTERIM REPORT

Table of Contents

0 - Team Members 5

1 - Brief Overview 6

1.1 - Proposal 6

1.2 - Progress Synopsis 6

2 - Mission Criteria 7

2.1 - Primary Mission 7

2.2 - Secondary Mission 7

3 - Outreach 8

3.1 - Local Community 8

3.2 - School 8

3.3 - The Wider Public 9

4 - Funding 10

5 - Mechanics 12

5.1 - Aim 12

5.2 - Parachute 12

5.3 - General Design Creation 12

5.4 - Refinement of General Design 16

5.5 - Manufacture, Testing and Further Refinement 18

5.6 - Design Alteration 20

5.7 - Materials 22

5.8 - Arm Unfolding 22

5.9 - Specific Details of Mechanism for Opening Arms 23

5.10 - Latest Design 25

5.11 - Propellers 27

5.12 - Testing of the Mechanics 27

5.13 - Further progress 28

6 - Electronics 29

6.1 - Flight System 29

6.1.1 - Battery – Turnigy Nano-Tech 3s (11.1V) 850mAh 29

6.1.2 - Motors – Turnigy Outrunner v2 30

6.1.3 - Control Board - OpenPilot CC3D Atom 30

6.1.4 - ESCs – Turnigy Nano Tech 20A 30

6.1.5 - TX / RX system – Orange Nano 31

6.2 - Sensor System 31

6.2.1 - Components 31
6.2.2 - PCBs 32
6.3 - Camera System 35
6.4 - Communications 36
6.5 – Initial Single-System Testing 37
6.5.1 - Sensor System 37
6.5.2 - Camera System 38
6.5.3 - Flight System 38
6.5.4 - Battery Management 39
7 - Software 40
7.1 – Algorithms 40
7.1.1 – Designing the Algorithm 40
7.1.2 - Assessing and improving our algorithm 42
7.2 - Website 44
7.3 - Can Code 45
7.4 - Base Station 46
8 - Integration of Electronics and Software within the Mechanics 48
8.1 - Testing 49
8.2 – Further Testing and Improvement of design 49
8.3 – Flight Tests 50
8.4 – Folding Propellers 54
9 – Launch Procedures 57
9.1 – Drone Laws and Safety 57
9.1.1 - Laws 57
9.1.2 – Learning Points 58
9.2 – Data Analysis 61
10 - Ground Support 63
11 - Risk Mitigation 64
12 - Gantt Chart 66
13 – Appendices 67
13.1 – Mechanical Design 67
13.1.1 - Arm – Type 1 67
13.1.2 - Arm - Type 2 68
13.1.3 - Control Board Holder 69
13.1.4 - Top Layer 1 70

13.1.5 - Layer 2 71
13.1.6 - Layer 3 72
13.1.6 - Bottom Layer 73
13.1.7 - Post 74
13.1.8 - Front Post 76
13.1.9 - Stack Assembly 77
13.2 – Electronics Design 78
13.3 – Code 80
13.3.1 – Can Code 80
13.3.2 – Base Station 120
13.3.3 – Website 142

0 - Team Members

Ashwin Ahuja – Ashwin is one of the co-team leaders and also leading the efforts of the electronics and software parts of the team. He is also managing the team's finances, and shouldering much of the team's organisation, ensuring the various parts of the team are working effectively.

Benjamin Yass – Ben is the other co-team leader and leading the efforts of the mechanical design team. He is also shouldering much of the team's organisation, ensuring the various parts of the team are working effectively.

Quentin Guéroult – Quentin is the head of the Outreach team, managing the efforts to inform the greater public about our project. He is also the lead publicist, designing the logos and website.

Hugo Aaronson – Hugo is a member of the Software and Electronics steam, specifically looking at the Data Analysis.

William Eustace – William is also a part of the Software and Electronics team, specifically leading the writing of the software, as a highly experienced coder in a number of languages (including C++ and C#). Additionally, he carries with him the experience of being the team leader of Team Impulse, and CanSat through this.

Daniel Halstead – Daniel is the Head of Flight Management, attempting to survey the specifications we have produced and determining whether the quadcopter will be flyable. He is very experienced in quadcopter design, having flown them for a number of years, and was particularly involved in the preliminary design of the product.

James Crompton – James is a member of the Mechanical Design team, in charge of the launch procedure, ensuring that the arms deploy as expected, and that the quadcopter begins its flight faultlessly.

Philip Fernandes – Phil is also a member of the Mechanical Design team, but is currently leading research into and creating the algorithms for finding Agricultural Viability.

Nicholas Palmer – Nick is also a member of the Mechanical Design team, with particular responsibilities for researching the most effective manufacturing choices and materials.

Figure 0.1: Team photo – from left to right: James Crompton, Nicholas Palmer, Ashwin Ahuja, Benjamin Yass, Daniel Halstead, Philip Fernandes, William Eustace, Monty Evans, Quentin Guéroult

1 - Brief Overview

1.1 - Proposal

The CanSat must fulfil the primary mission of measuring air temperature and barometric pressure and transmit this data over a radio link at a minimum rate of 1Hz, using RF transmission. It will also measure several other variables, including relative humidity, GPS location and acceleration. These data will also be transmitted to the base station. The main part of the secondary mission - to produce a quadcopter that is able to deploy after launch from the specified dimensions of the CanSat - will use these measurements, as well as a gyroscope. The quadcopter will be used to investigate unknown landscapes. For this, both the array of sensors, and a live camera link will be used. The quadcopter will be designed to autonomously move to a set of GPS coordinates, but will also be able to be controlled manually. It will possibly be able to return to the launch-site. Finally, we also hope to estimate the relative agricultural viability of the area, by using a predefined algorithm. This could also be used on other planets to find how viable the area could be for cultivation of crops.

1.2 - Progress Synopsis

Since the Second Interim Report, a vast amount of progress has been made, moving from a working, flying first prototype shortly after Christmas (exactly meeting our objective) and identifying and remedying the problems we encountered. For example, the propellers were too small, resulting in a lack of lift. To address this, we changed the design to include folding propellers, which allow us to meet the regulations while supplying more than enough lift. In particular, the mechanics has gone through momentous progress, shown by the vast number of small changes that have been made. Meanwhile, the electronics had already largely been finalised, while the completion of the large amounts of software required mean that despite some setbacks, the team is very nearly ready for launch. In the background, the outreach program has continued, with a couple of talks to various internal societies as well as to a local preparatory school.

2 - Mission Criteria

2.1 - Primary Mission

1. The CanSat should transmit air temperature and barometric pressure to the ground at least once per second
2. The CanSat should comply with all of the CanSat guidelines, notably:
    a. It should weigh 370g
    b. It should have a maximum diameter of 66mm
    c. It should have a maximum height of 115mm, bar antennae
3. The CanSat should log all data both on the ground and in the Can.

2.2 - Secondary Mission

1. The CanSat should be able to fly comfortably, being relatively stable in the air and low winds
2. The CanSat should be capable of at least 5 minutes of flight.
3. The CanSat should be able to transmit further data, including relative humidity, thus allowing a computer to calculate the agricultural viability of the area.
4. The CanSat should be able to navigate to a set of co-ordinates autonomously.
5. The CanSat should also be able to controlled manually, if necessary.
6. The CanSat should be able to live stream video to the ground.

3 - Outreach

Cyclone's Outreach has continued to develop in the planned outreach towards the local community, within the school and towards the wider public. All three strategies are underpinned by the team's website (http://teamcycl.one) that was developed from scratch by the software team alongside a recognisable and simple domain name being secured. The website contains a description about CanSat and Cyclone's entry, an overview of the team, a public folder with documents as well as videos and a blog that regularly discusses the progress of Cyclone. The website acts as the main platform through which people can learn about the team and follow its progress. It also contains links to all other platforms through which people can find out about the project. Additionally, all source code, designs and plans of the different departments of Cyclone are made available to the general public and community through GitHub by simply searching for Cyclone CanSat. A simple, easy-to-remember logo was also designed by the team, which unites all the team's efforts on all platforms.

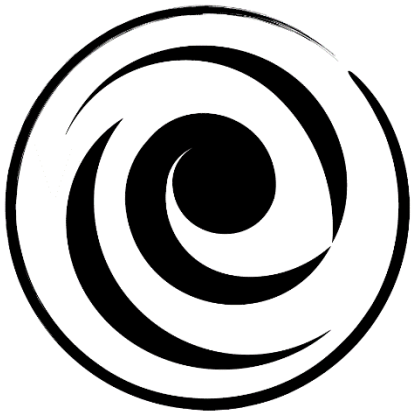


Figure 3.0.1: Cyclone Universal Logo

3.1 - Local Community

Early in January, the long-planned talk to the nearby preparatory school Colet Court occurred. Cyclone gave the assembly about the CanSat competition to inspire future engineers. A demonstrations of the quadcopter spreading out its wings whilst being held securely was a particular highlight along with the demonstration of Impulse's Rover last year. Cyclone's leaders and William Eustace (as the team-leader of Team Impulse) gave a talk on the 27th September to the Surrey Explorer's Club (http://www.surreyexplorers.org.uk/), a group of gifted primary school children where the CanSat competition was discussed amidst topics such as rockets, outer space and the uses of satellites as well as how do they function. Cyclone's entry and Team Impulse's European victory were also discussed. Finally, an interesting competition was organised which allowed the children to produce a paper aeroplane which would be released from a specially adapted paper-aeroplane-launching-remote-controlled- helicopter. The aeroplane with the longest flight time would win and this proved a fun activity for all as tactics were also discussed. Team Cyclone are looking for further opportunities such as the Surrey Explorer's Club to further publicise the CanSat competition. We hope to manage to secure even more schools to talk to in the future, and thus further our promotion of Engineering and the CanSat Competition.

3.2 - School

Cyclone have already made popular talks at societies such as SPS Space outlining the CanSat competition and Cyclone's entry. Cyclone were also at the Societies Fair and the school's open day where we openly discussed

CanSat and our project this year to both pupils and parents. Most recently in February, talks were given to the Physics and Engineering societies, mainly to the younger years within the school. These talks would be often more technical with team members talking in detail about both the technical design and the software and electronics. The engineering concepts behind the design and the whole process that led to several ideas having to be altered were also discussed. During the most recent talks, footage of test flights has also been used, often enhancing the audience's understanding of our entry for the CanSat competition. Additionally, promotion of CanSat has been carried out informally during a number of other societies, notably Robotics Society and HalleySoc. An article on CanSat was published in the school's magazine Black and White that was distributed for free and that can be accessed electronically. The team hoped that this would further increase publicity for CanSat and our entry as the school magazine is widely read by teachers and pupils alike. The PDF version of the article is now also available on Team Cyclone's website to make it available to all and not just pupils of St Paul's School. Meanwhile, the St Paul's school website and magazine reference CanSat in a number of places, talking about Impulse's famous European win last year as well as our continuing mission this year. Plans to have a cake sale in the school for fundraising had somewhat stalled, with certain teachers having reservations about the idea. Team Cyclone have contacted and are planning to talk with the Surmaster (Deputy Head) in order to hopefully carry out the cake sale. More importantly, due to Cyclone's Outreach, a new member, specialised in Software and Electronics, Hugo Aaronson has joined the team. Hugo, as an experienced programmer, will help greatly in this section of the team.

3.3 - The Wider Public

Coupled with the website, the team has decided to be present on multiple platforms to further increase awareness of CanSat and Cyclone. Cyclone has a Facebook account (http://on.fb.me/1jTDXtu) as well as a Twitter account where a briefer, but more up-to-date account of the team's progress is available (@SPSCyclone). Android (http://bit.ly/20cqWM7) and Windows Phone Apps (http://bit.ly/1MW0Cfv) have also been made by the Software Team and can be downloaded. Again, these apps are to further publicise CanSat. Several videos outlining the progress of the different software, electronic and mechanic teams have been uploaded. The team believes that these videos are extremely important when showing our progress. They are a crucial visual aid for the public when following the team in showing exactly what different members of Cyclone are designing or making, and we hope these continue up until the launch date in March. This together with different blog posts from every member of the team would inform anyone precisely what we are doing.

4 - Funding

Firstly, Cyclone have approached a number of sources for sponsorship, and have been very successful to date. Newbury Electronics, the owners of PCBTrain have agreed to sponsor the team to the tune of £150, with free PCBs. Additionally, they have agreed to offer their expertise in checking the PCBs that we have sent. To date, only around £50 of the £150 has been used, with the first round of PCBs. Thus, two more revisions of PCBs would easily be possible, in order to make more improvements if necessary, or to fix any inevitable issues that may arise. Additionally, we have been sponsored by HobbyKing, a large online Remote Controlled parts producer and seller who are providing us with a limited amount of free parts. To date, this amount has been around £500, but there is a possibility of more parts if necessary. This amount has allowed us to easily get the best, rather than cheapest components, maximizing the chance of success of the project. Additionally, the chance of needing more products is also low, given that a number of spares of every necessary part has been obtained.

We are also planning a cake sale at school which as well as being a successful fundraising mechanism, will also act as good publicity. Last year, CanSat (under the guises of Team Colossus and Team Impulse) organised a cake sale, raising just under £300, showing the potential success of such an event. This year, we hope with similar preparation and organization, a similar sum could be generated.

Additionally, three members of the team (Ben, Daniel, William) are Arkwright Scholars, which contributes £200 per person annually to the school engineering department. This money could be used for CanSat if necessary. In fact, our school is also willing to sponsor the project (especially since we are this year's only team) to the tune of a few hundred pounds if necessary, with money coming from the Engineering budget. However, we hope to keep these costs to the minimum necessary, by continuing to seek corporate sponsorship and through a successful fundraising event in school. Since the previous progress report, there have been very few new purchases apart from slightly larger propellers which could be used to make folding propellers (see the mechanics section) and a better Lithium Polymer Charger, since the other charger did not seem to be able to charge the battery effectively.

Section	Expected Cost / Value	Costs to date
Outreach	£60	£60
Hardware	£26	£26
Electronics Components	£500	£500
PCB Manufacturing	£0	£0
TOTAL	£586	£586

Figure 4.1: Basic breakdown of costing – for more detailed breakdown, see Figure 4.3

However, despite the costs being over the acceptable limits of the CanSat cost per unit, it must be noted that this includes a number of spares for each component, in fact, we expect the cost to reproduce our CanSat (including the value of our sponsorship) to be much lower:

Item	Cost
Electronics Components	£290
PCB Manufacture	£50
Hardware	£30
TOTAL	£400

Figure 4.2: Breakdown of cost of a single Can.

Type	Name	Quantity	Cost
Battery	Turnigy Nano-Tech 850mAh LiPo	6	£ 6.56
Board	Hobbyking i8 Control Board	1	£ 10.98
Motors	Turnigy Multistar Outrunner V2 Motors	8	£ 7.46
Escs	Turnigy Multistar 20A Slim ESCs	8	£ 8.88
Servos	Turnigy Analog Nano Servos	10	£ 2.55
FPV TX	Hobbyking FPV Transmitter	3	£ 13.35
Battery	Turnigy 2200mAh battery (for base station)	2	£ 6.39
FPV Receiver	SkyZone FPV receiver	2	£ 12.77
Antenna	Polarized SMA antenna	2	£ 3.19
SMA Wire	SMA wire	5	£ 1.06
OSD	Hobbyking OSD	2	£ 9.28
FPV Camera	Mini FPV Camera	3	£ 18.84
Bags	Lipol Bags	6	£ 1.34
Radio TX	Orange Radio Transmitter	1	£ 41.59
Radio RX	Orange Radio Receiver	3	£ 6.97
LiPo Charger	Hobbyking LiPo charger	2	£ 7.98
FPV RX	Quanum Complete FPV Bundle Set	1	£ 46.01
Propellers	Gemfan Multi-Rotor Prop Set 50mm	15	£ 0.80
Control Board	OpenPilot CC3D	2	£ 11.18
Control Board	OpenPilot CC3D Atom	3	£ 14.74
Wires	Servo Wires	3	£ 1.00
Wires	XT60 Wires	5	£ 3.00
Connectors	3.5mm Connectors	5	£ 1.20
Propellers	**5" x 4.5" Propellers**	**5**	**£ 1.00**
LiPo Charger	**iMax B6**	**2**	**£ 25.00**
MAIN SENSOR BOARD			
Male Headers	Break Away Headers - Machine Pin	2	£ 2.95
Resistor	Panasonic 75kOhm Resistor	100	£ 0.01
Capacitor	Murata 100muF capacitor	10	£ 0.98
Resistor	Bourns 10k SMD 0805 Resistor	50	£ 0.01
Resistor	Bourns 10k SMD 0805 Resistor	50	£ 0.01
Motor Driver	Texas Instruments DRV 8833	10	£ 1.78
Capacitor	TDK 2.2muF	100	£ 0.05
Capacitor	Kemet 0.01muF	10	£ 0.18
MCU	Teensy 3.2	4	£ 13.02
Humidity Sensor	HYT-271	5	£ 22.09
Pressure Sensor	MS5637	10	£ 1.69
IMU	SparkFun LSMDS1 Breakout	1	£ 16.28
GPS:			
GPS Module	GP-2106	2	£ 32.59
GPS Breakout	Sparkfun GPS Evaluation Board (GP-2106)	2	£ 6.49
GPS Module Connector	Interface Cable GP-2106	5	£ 0.98
RF	Hope RFM98W	5	£ 5.99
Micro SD Breakout	Sparkfun OpenLog	1	£ 16.28
POWER DISTRIBUTION BOARD			
5V Voltage Regulator		10	£ 2.00
PCB Building Costs		3	£ 50.00
Can Building Costs	3D Printing	4	£ 5.00
	Grub Screws	2	£ 3.00
Outreach Costs	Website	12	£ 5.00
SUB-TOTAL	**£ 1,284.89**		
HobbyKing Sponsorship	£ 606.78		
PCBTrain Sponsorship	£ 150.00		
Cake Sale	£ 300.00		
TOTAL	£ 228.11		

Figure 4.3: Breakdown of costs (made using Microsoft Excel) – new purchases are highlighted

5 - Mechanics

5.1 - Aim

The goal of the mechanical design time is to design a successful quadcopter which can fit inside the specifications of 66m diameter, 115mm height, etc. at launch.

For most challenges involved in accomplishing the task above, any improvement in any aspect of the design will be beneficial to the overall performance of the Can, unless this is done to such an extent that we begin to suffer from lack of space within the Can. When creating the design, we found that this issue became most acute when trying to maximise the stability of the quadcopter, as this resulted in using up large amounts of space which is needed for other components.

5.2 - Parachute

In the previous progress reviews, it was decided that a parachute, aside from being unnecessary, would impede the proper function of the CanSat. The parachute strings and fabric could easily become tangled in the arms, preventing correct deployment, or, worse still, in one or more propellers. In both scenarios, the CanSat would fall in an uncontrolled manner, because neither the parachute nor the propellers could properly deploy. In the latter scenario, there would be the additional risks of motor burnout or ESC (speed controller) damage through overloading, and the can being propelled into unpredictable and uneven flight as it attempts to correct for the inevitable deviation in attitude from the prescribed value by changing power values for propellers which do not all respond. The UK CanSat authorities were approached during May 2015, and permission was given to fly the CanSat without the parachute, provided evidence could be provided of the quadcopter's correct function. In the UK competition, the parachute cutaway problems would be exaggerated: the comparatively small drop height of a few hundred metres would give very little time for the parachute to deploy, be cut away (if it is cut away before it has deployed it is likely to tangle in the rotors) and for the CanSat to enter the powered flight phase; however, if the team were to get through to the European CanSat competition, it is understood that the drop height is significantly more and that compliance with the parachute regulations may be stringently required. For this reason, it was felt prudent to test the feasibility of a parachute cutaway system. A small servo motor was acquired. This would pin down a small loop at the end of the parachute cable until the software received a release command over radio. This would then turn the servo's horn through 90 degrees and allow the loop to slip off the end of this horn. This firmware has been written already, permitting the plan to be put into action in a short period of time should it become necessary.

5.3 - General Design Creation

The trade-off between space and stability was the deciding factor for one of our high impact decisions on the overall setup of the quadcopter – the orientation that it would fly. The outcome of this decision was to go for the vertical design, as although the horizontal one would be more stable, it would be impossible to build the can within the specifications given to fly in a horizontal orientation.

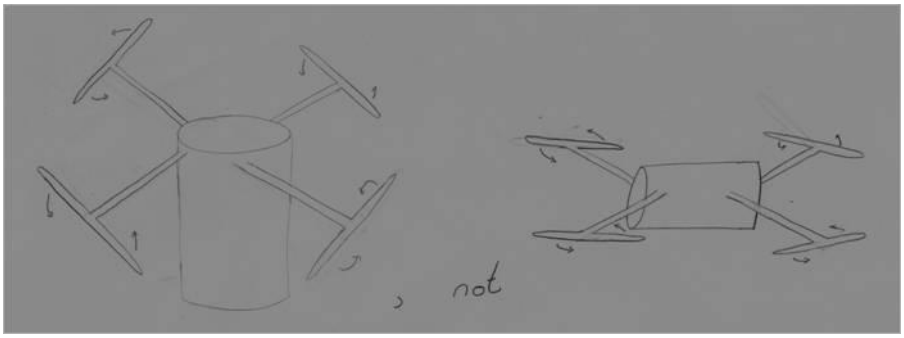


Figure 5.3.1 - Can Orientation Decision

Throughout the design of the Can many decisions were strongly influenced by this space / stability trade-off. For example, the exact length of the arms:

This initial vertically orientated design (Figure 5.3.2), where arm length was maximised to increase arm stability (as the greater the distance between diagonally opposite rotors when unfolded, the greater the quadcopter's stability), had to be altered to ensure sufficient space was left inside the Can for other components (Figure 5.3.3). Due to this change the diagonal distance decreased by about 35mm (see Figures 5.3.2 and 5.3.3).[1]

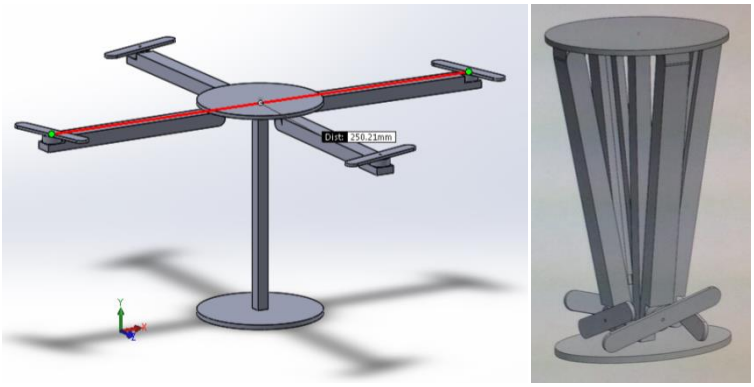


Figure 5.3.2

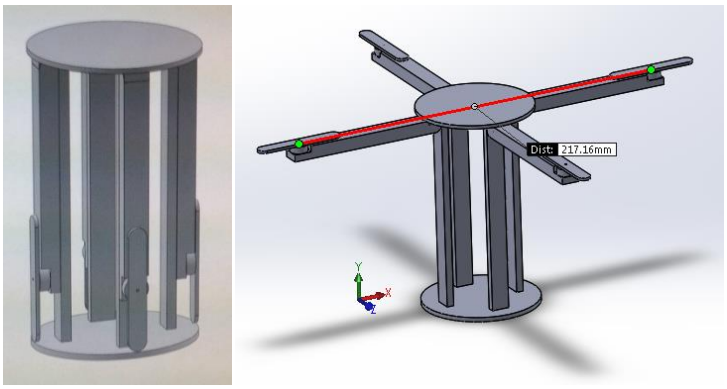


Figure 5.3.3

Other factors of the quadcopter's design were also strongly influenced by the minimal space available.

[1] All designs are made using SolidWorks 2015 – made by Dassault Systèmes – http://solidworks.com

E.g. The Can's overall structures that hold it together had to be cleverly redesigned to maximise space for other components (From Figure 5.3.4 → Figure 5.3.5 (Figure 5.3.6 is the same as Figure 5.3.5 except with the arms unfolded to show the wall)):

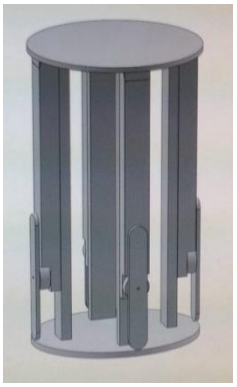


Figure 5.3.4

Figure 5.3.5

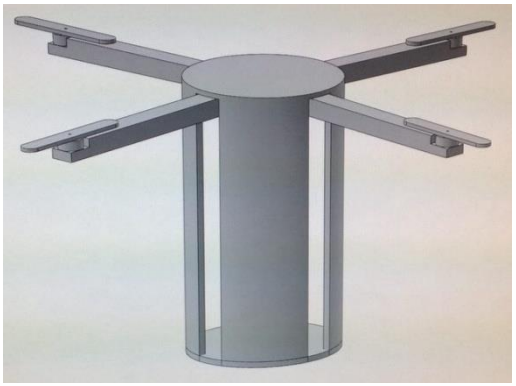


Figure 5.3.6

Another decision we made with reference to the overall design of the Can was deciding how to stabilise the arms when unfolded – hence we experimented with various solutions.

Figures 5.3.7 shows a possible solution we came up with to increase arm stability

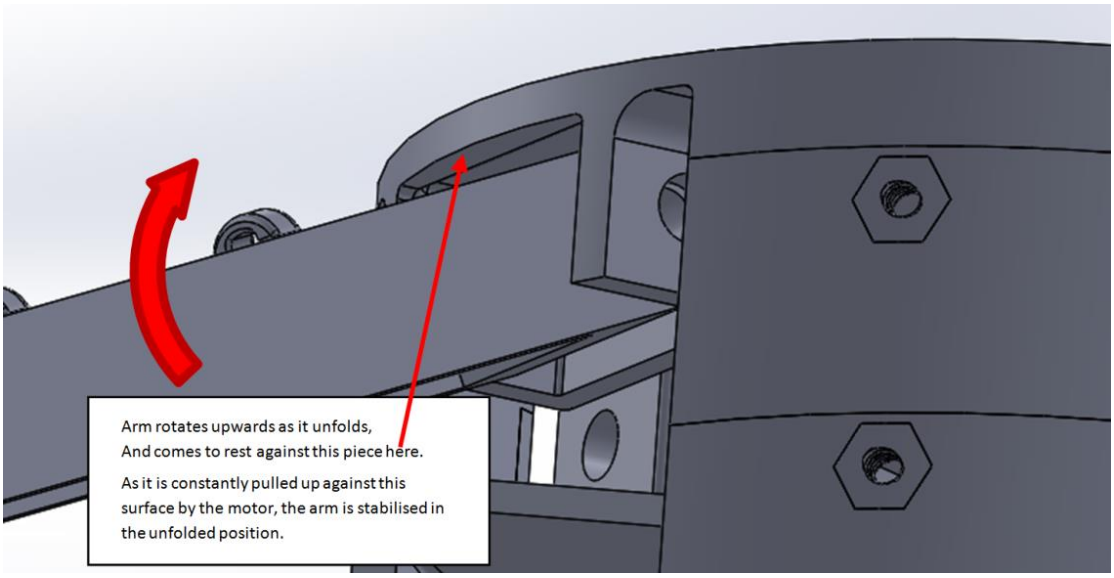


Figure 5.3.8 shows another possible solution we came up with to increase arm stability

We made a decision to use the solution shown in Figure 5.3.8 as we thought it would work sufficiently well, yet would be simpler and so more reliable than the supports shown in Figure 5.3.7.

The final major part of designing the general Can structure was working out layer spacing. This we did by modelling each component in the Can and working out the best way to fit them all in it. From this we then created layers at specific heights within the Can to facilitate easy mounting and organisation of components during assembly and integration of electronic components within the Can. The plan for this can be seen in Figure 5.3.9, and the consequent layer creation can be seen in Figure 5.3.10.

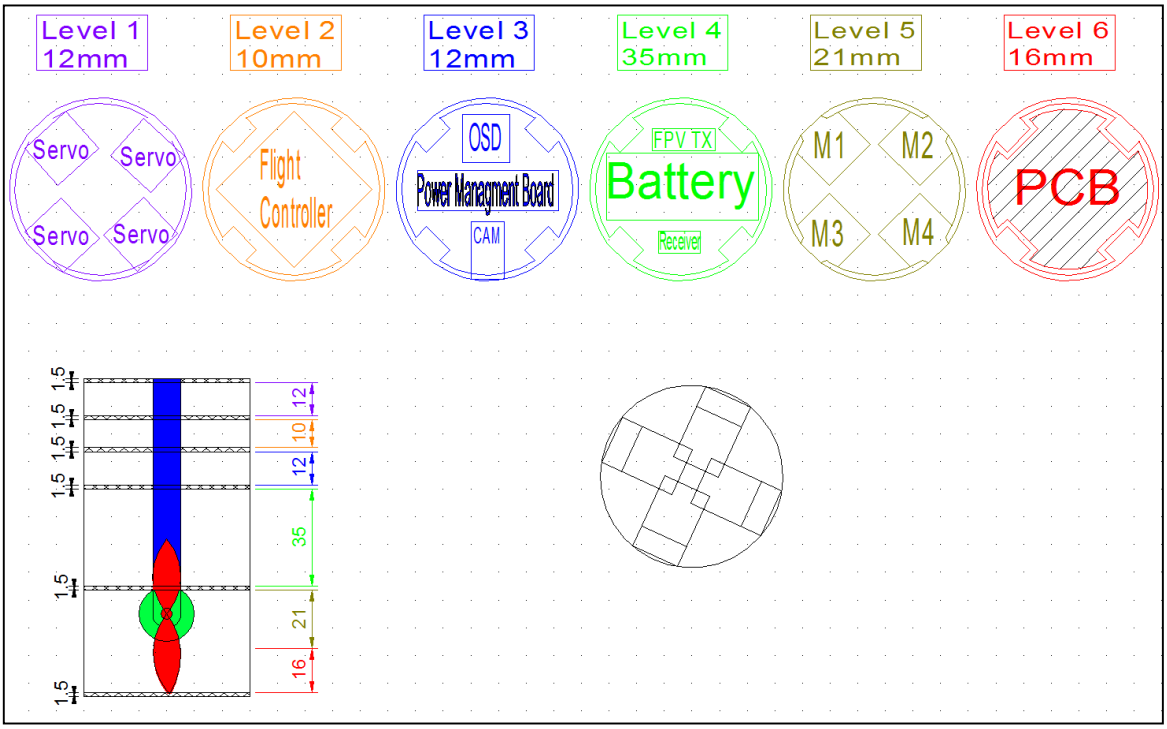


Figure 5.3.9 [2]

[2] 2D design made using 2D Design V2 - http://www.techsoft.co.uk/products/software/2D_Design_V2.asp

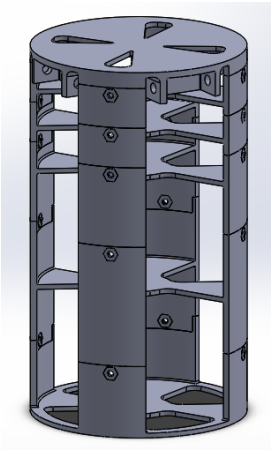


Figure 5.3.10

Before beginning the detailed refinement of the Can we decided to fillet (round) every edge that would be under stress (i.e. almost all of them) – shown in Figure 5.3.11. This strengthens the Can as it means any force acting to snap a corner is spread across an arc of material instead of being concentrated at a single line along an edge.

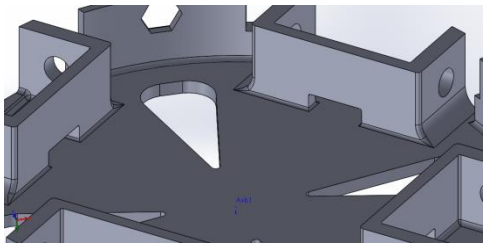


Figure 5.3.11

5.4 - Refinement of General Design

Now that the general design of the Can was complete, we begun work on the designing of the specific component parts of the Can.

This began with designing a way of being able to dismantle and reassemble the stack easily. We went for a modular system which utilised teeth like fittings to connect the layers securely together (as shown in Figure 5.4.1). Through the teeth can be seen holes which are for pins, so that we can secure the layers together.

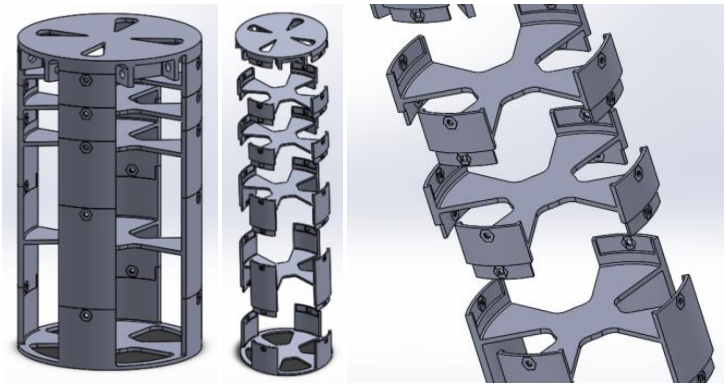


Figure 5.4.1

All pieces of this modular system are very similar apart from the base piece which simply does not have connections for a layer below it, and the top piece (Figure 5.4.2) which does not have connections for a piece above it, but does have hinges for the arms built into it, as well as the arm stabiliser as seen in Figure 5.3.8.

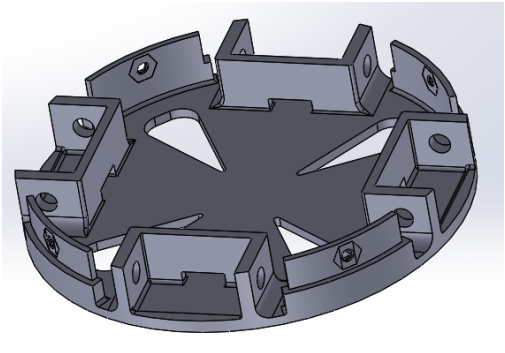


Figure 5.4.2

Furthermore, we designed the arms in such a way that they would be strong, hold the motors securely and also contain cavities in their underside for the ESCs so that they would be efficiently stored away (see Figure 5.4.3). Note that the bobbles on top of the arms are rings built into them for the wire (which will be used to unfold the arms) to run through. This will allow the wire to run along the top of the arm without it becoming a danger to the rotors.

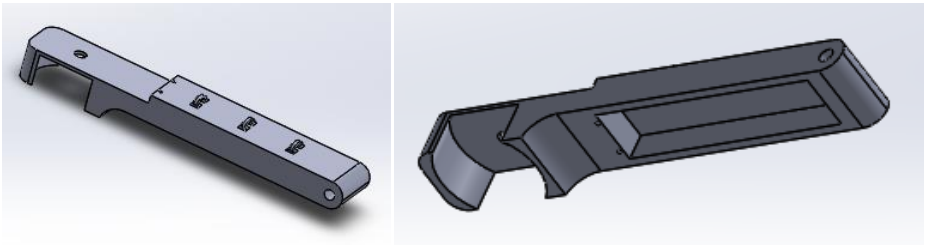


Figure 5.4.3

When constructed in SolidWorks the model looked as follows:

Figure 5.4.4

5.5 - Manufacture, Testing and Further Refinement

To test our CAD we 3D printed this latest design. While normally this would have been fairly easy, as our school has a 3D printer, it was unfortunately broken at the point we wanted to 3D print. Therefore, we decided to outsource the printing and contacted a local hobbyist group who printed the pieces at a low cost for us.

When these parts arrived we set about testing them and discovered some issues:

- The motor housing we had designed appeared to barely hold the motor;
- The layers did not slot together smoothly.

Hence we redesigned the motor housings to provide greater purchase on the motors, using secure screw fittings:

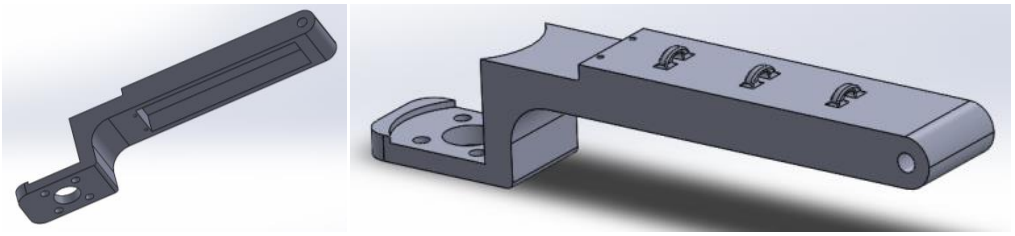


Figure 5.5.1

By this point the school's 3D printer had been fixed so we 3D printed an arm to see if the problem had been solved and we believe it has – the motor feels secure, however we will not know for sure until we carry out tests with the motor running at full speed (which we plan to do in the next couple of weeks). In addition, when we do this testing we also will be looking to test the hinge integrity.

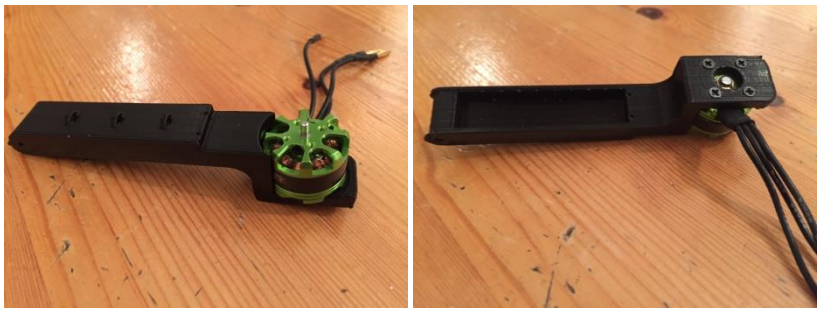


Figure 5.5.2 – motor screwed securely into arm

Furthermore, while the layers had fitted together perfectly in SolidWorks, due to the inaccuracies of the 3D printer we are using, they did not fit together once printed. The issue arose at the end of the thin tabs which slot into a matching section in the layer above (Figure 5.5.3). The problem was that when the 3D printer printed this tab, when the printer head rounded the corner, it deposited a blob of material on that corner as it could not make such a tight turn. The tight turn was due to the thinness of the tab. Hence, on every layer of material that the printer built up, additional material was deposited on this corner – and every other similar corner on the piece. Hence the width of this tab was increased meaning that it no longer slotted into its matching section on the next layer. Hence, to solve this, we thinned the tab and the matching section which it slots into by 0.4mm (see Figure 5.5.4 (before) and 5.5.5 (after)). Thus even when the pieces became wider due to the inaccuracies of the 3D printer the pieces still slotted together.

Figure 5.5.3

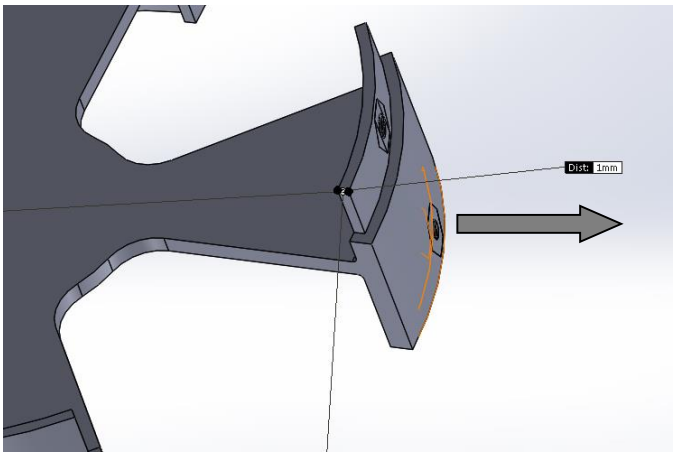


Figure 5.5.4

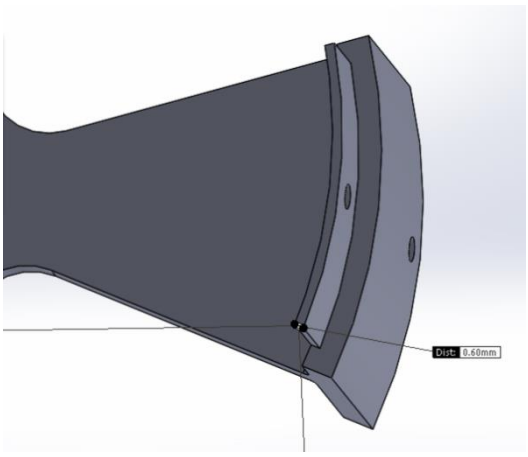


Figure 5.5.5

Hence, the layers can easily fit together, as shown with the top layer in Figure 5.5.6.

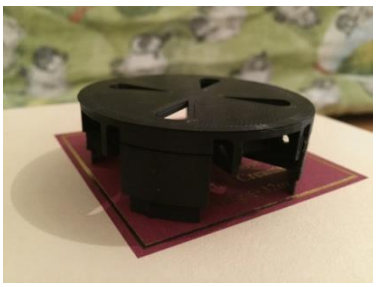
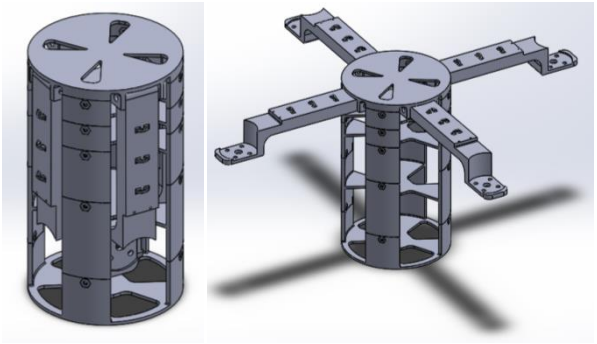


Figure 5.5.6

When constructed in SolidWorks including those changes the model now looks as follows:



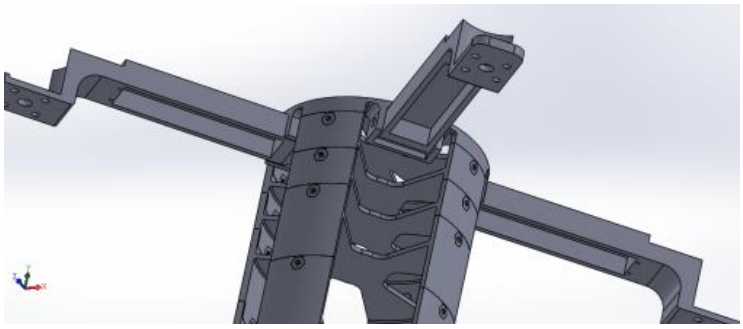


Figure 5.5.7

5.6 - Design Alteration

Although this fully functional design fulfilled all our requirements, we believed we could still improve it. This was because the tabs which connected the layers together were very fragile and the small knocks that the pieces were receiving in the day to day handling of them had already begun to show – one of the pieces had a broken tab and two others were clearly weakened. Hence, we decided to see if we could redesign this aspect of the Can, without affecting any other part of it.

The new design we came up with did just this. It consists of four vertical posts (the back and front of one of the posts can be seen in Figure 5.6.1) which replace the vertical extensions that protruded from the plates in the earlier design. These posts, situated at the circumference of the Can, fix onto plates at different heights, holding the plates securely in place at these heights. The plate spacing is the same as in the previous design, for the same reason – when we modelled each component in the Can this spacing optimised their packing and allowed all the components to be mounted securely and organised coherently. Each plate (Figure 5.6.2) has four teeth in place of their four protrusions which previously were used to link the layers together. Each tooth fits into a matching slot in one of the posts, and these teeth are fixed to the posts with M2 nuts and bolts which act as pins.

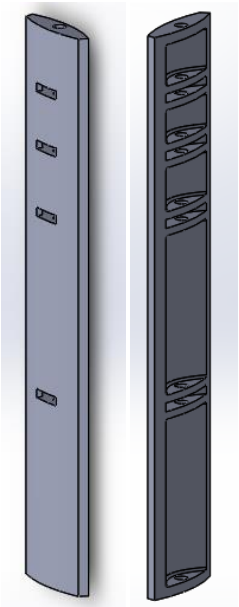


Figure 5.6.1

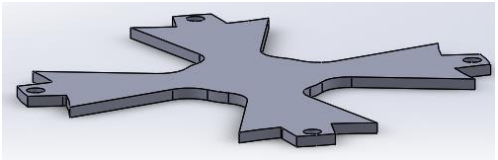


Figure 5.6.2

This new design has many advantages over the old one, and only a single disadvantage. The disadvantage is that on each layer there is less room than before. This is due to the fact that the horizontal cross-section of the posts is greater than that of the previous interlayer attachments. However, this reduction in space is so minimal (see Figure 5.6.3) due to the efficient design of the new system that this does not affect whether components fit into their allocated space on each layer, as previously planned (see Figure 5.3.9).

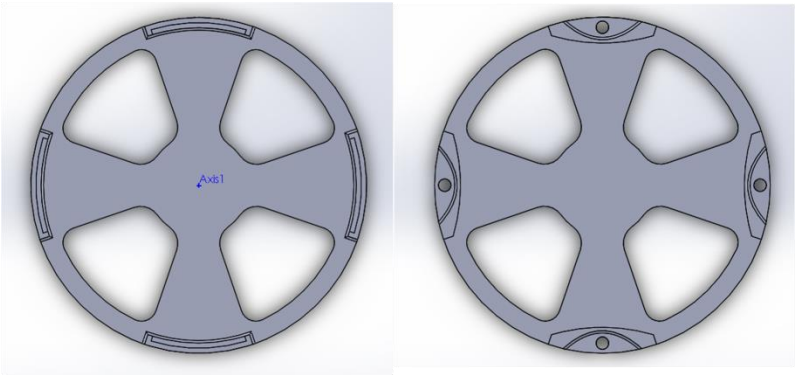


Figure 5.6.3 – Before vs. After

The benefits of the new design include:

- The interlayer connections are less fragile and stronger with the direct result that the Can is now too.
- The pieces are more 3D printer friendly. This is due to the extensive reduction in the number of thin protruding structures extending from the pieces – there is now nothing less than 1mm thick whereas before some sections were 0.5mm thick.
- An unforeseen benefit is that now there are only 4 designs of pieces – the top piece, bottom piece, plate and post; whereas before there were six – each of the six layers was slightly different.
- The new design still allows for easy access to components on each layer when the Can has been assembled. This is because all the fixings holding each post in place are mechanical, and the removal of one post allows access to all the layers.

The new design when assembled can be seen below:

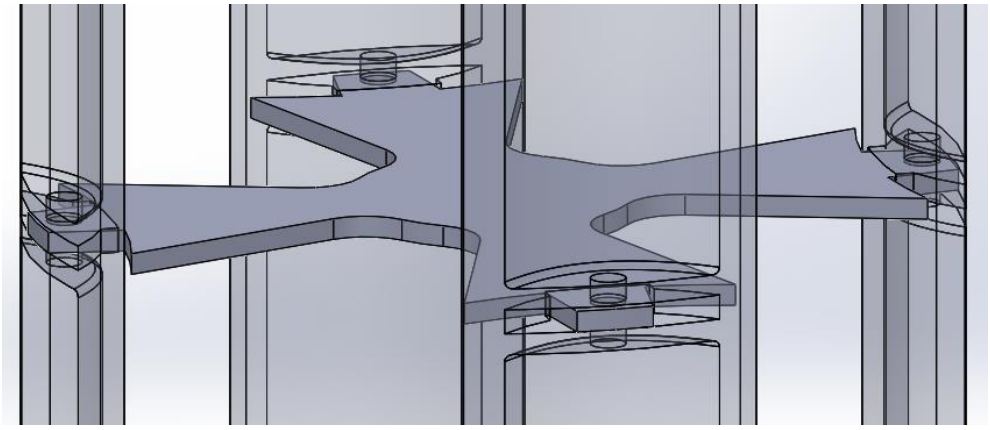


Figure 5.6.4

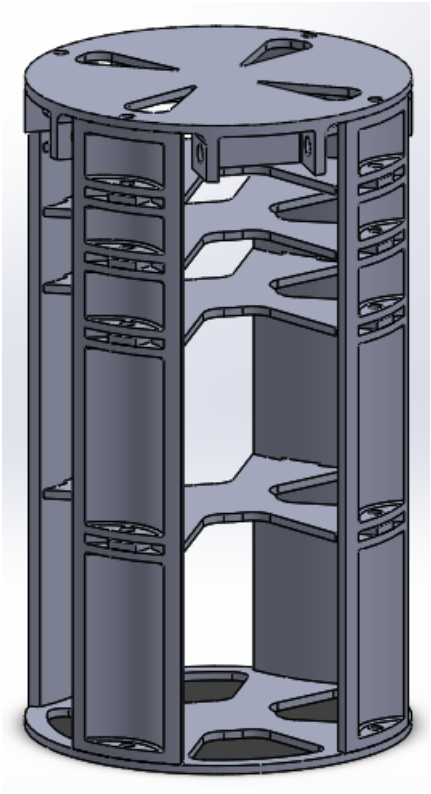


Figure 5.6.5

5.7 - Materials

For our testing so far we have been using some PLA and some ABS as these are cheap materials which work well in the school's 3D printer. For the testing of whether pieces are the correct size to fit components within them, these materials are suitable. However, we are also conducting research into stronger materials, in case ABS and PLA fail in our upcoming testing – notably the use of SLS (3D printing) to allow for the production of exactly the parts designed but using a stronger (metal containing) filament.

5.8 - Arm Unfolding

Whilst originally we intended to use to raise each arm on a separate servo attached to the top plate, as seen in Figure 5.3.9, we quickly decided that this would not give a robust enough connection between the arms and the top plate. This is because the servos would need to be small. Therefore, the axel onto which the arm would

be attached would be about 3mm in diameter and 4mm deep. This is too small to give a link that would endure the launch and then support the can's weight during flight.

Therefore we decided to redesign this system. During this process we realised that we would be able to attach the arms are more securely to the Can's body, and we would be able to free up more space within the Can if the arms were controlled by a motor which would wind up wires attached to and which ran along the loops on, the top of each arm. This principle we have tested with some of the 3D printed parts, with huge success (see Figure 5.8.1) – however for this test we were manually pulling on the wires to simulate what would, in the actual Can, be done using a motor to shorten the lengths of wire passing around the arms (as discussed in conjunction with Figure 5.4.3), thus lifting the arm structure up.

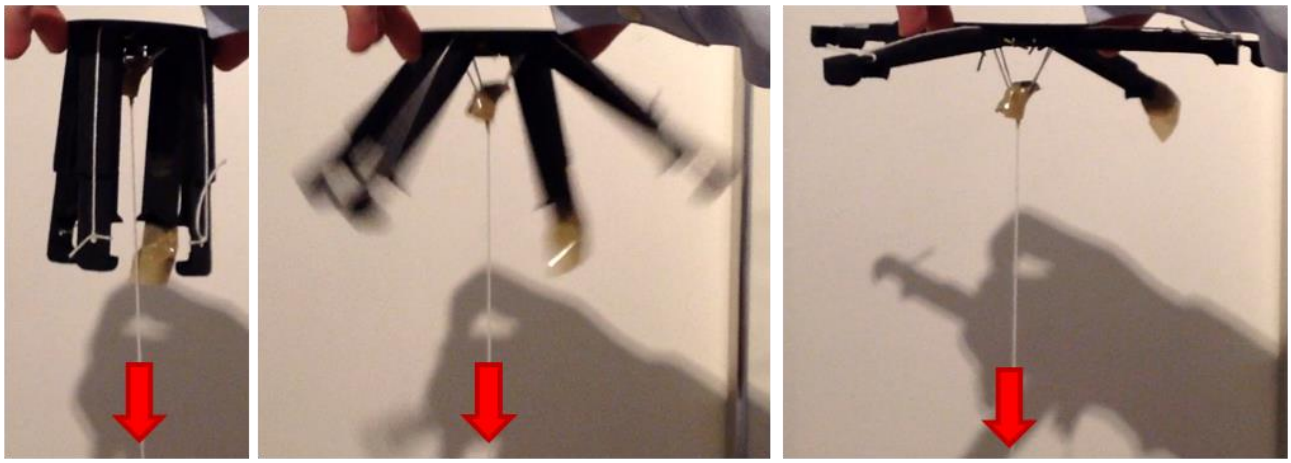


Figure 5.8.1

5.9 - Specific Details of Mechanism for Opening Arms

To raise the arms into their flight position we have decided to use a continuous piece of thin brass wire attached from one arm through the top plate to the opposite arm. In order to fully raise the arm piece, the wire must be shortened by 10mm to 16mm (measured from the 1st test assembly of the can). To do this we will wind the wire around a motorised spool.

Initially the idea was to wind up the wire on a spool attached to a dc motor. The smallest suitable motor was Como Drills Brushed DC Motor, which is 15mm long. The left picture in Figure 5.9.1 shows how this would not fit well into the design vertically, whereas if placed in horizontally one of the pieces of continuous wire would need to come on and off the spool in line with the axis of rotation, which would make it likely to jump and come off.

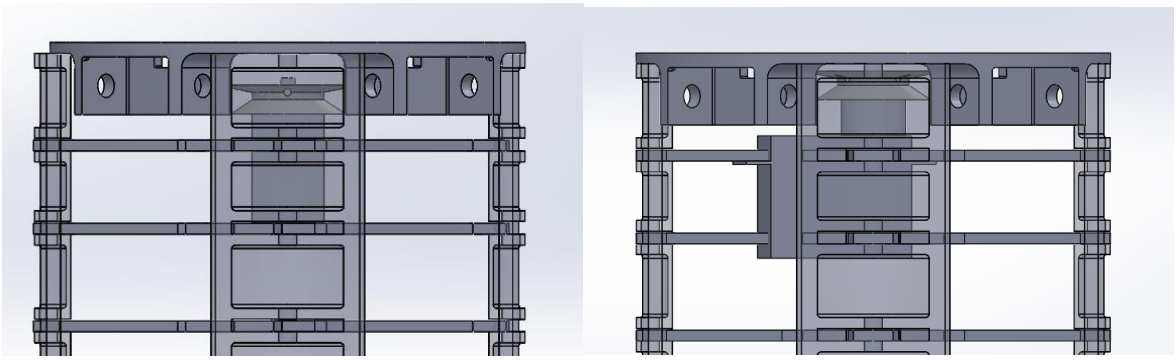


Figure 5.9.1

The right picture shows a similar arrangement for the Turnigy Nano Servo. This is much deeper and would pass through 2 layers, taking up part of 3 cavities, making this arrangement unworkable.

Therefore the best arrangement would be a horizontal servo. To get around the issue of one of the wires jumping two servos and spools were needed- one for each wire. This enabled us to utilise the ability of servos to lock with minimal impact on volume.

In the initial design we created structures (of the exact size of the servos) in the underside of the top plate to mount the servos. For this we designed pillar-like constructions for the servos to attach to and a raised platform for them to sit on so as to leave space underneath the servos for the wire to pass from one side of the top plate to the other. Figure 5.9.2 shows the design which we prototyped in the 3d printer.

Figure 5.9.2

The issues with this design was that there was no tolerance to insert the servos without breaking off two of the pillars first. Another problem was that the holes were too small to allow the antennae through the top plate.

The spools are also shown in Figure 5.9.2. They have a minimum diameter of 8mm giving a circumference of 25mm which will easily provide enough distance to raise the arms within the limit of 360° rotation for these servos. There are also 12mm diameter plates on either side of the smaller diameter part to ensure the wire does not jump off the spool.

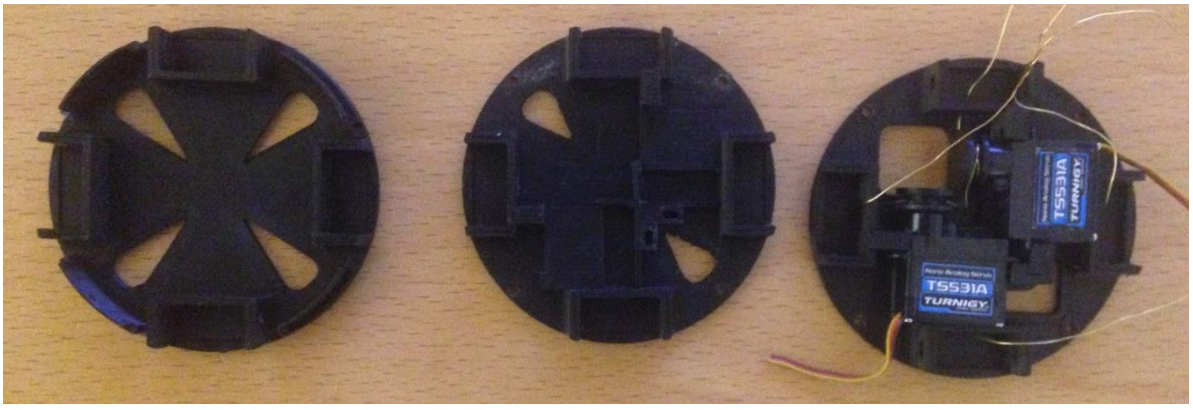


Figure 5.9.3

Figure 5.9.3 shows the evolution of the design of the top plate over the development period. The current design (Figure 5.9.4) has much larger holes to allow antennae through, while still giving a strong base for the servos to attach to. The pillars have been moved to allow a looser fit for the servo and the pillars that snapped

last time have been joined together to make them stronger. The wire attaches to the spools through a small hole through their centre along a diameter. The spools then attach to the servos through friction fit joints, which when their design has been finalised, will be secured with adhesives.

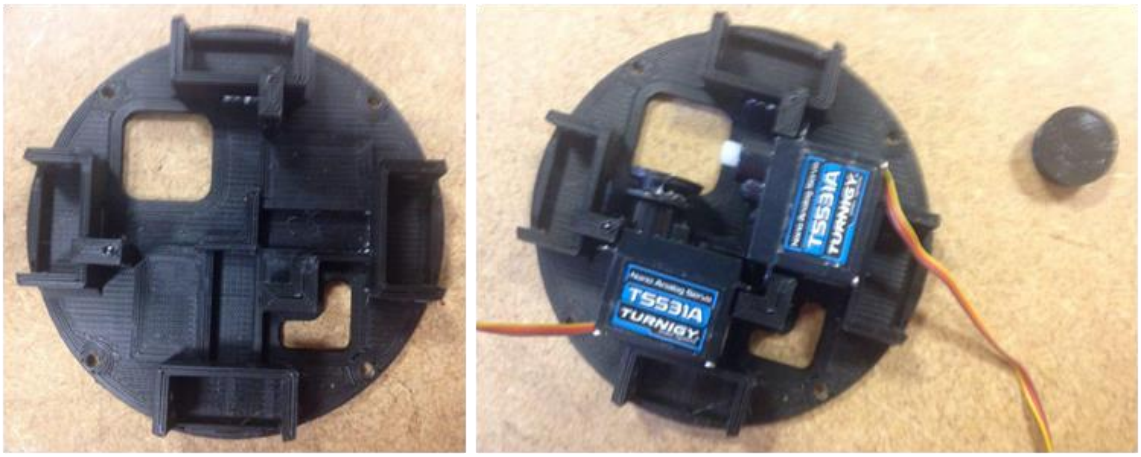


Figure 5.9.4

5.10 - Latest Design

Here (Figures 5.10.1 and 5.10.2) are our pictures of our latest design. It has been 3D printed on the school's 3D printer in PLA. In Figure 5.10.2 the servos are not in place to hold the arms open so the wire is manually being held taut.

Figure 5.10.1

Figure 5.10.2

5.11 - Propellers

Most of the information about propellers is decided by the existing mechanics of the product. It is clear that we want to maximise the size of the propellers, in order to ensure that the maximum air can be pushed downwards, thus increasing the lift of the can, maximising the efficiency of the system. However, there is inevitably a compromise between the length of the arms (which when increased, increases stability of the craft) and the length of the propellers, since the maximum of the arms + half the propeller diameter must equal 115mm or less, the maximum height we can use, as per the ESA regulations. To start off with, through carrying out research into various specifications, we determined that the minimum propeller size was 50mm, hence we started work with that. However, there is the possibility, that, if testing reveals that the lift is not great enough, we will reduce the length of the arms and increase the size of the propellers. The next decision was how to obtain propellers, with there being two main options, the first buying them, the second making them. If we were to make specific propellers for our purposes, it would be clear that we would need to make use of Injection Moulding as the only suitable mechanism for producing the precise shapes required for the propellers. However, the process is painful, requiring the creation of a mould, and very expensive. Unlike 3D printing, we do not have the facilities to carry out Injection Moulding at school, and outsourcing the production of the propellers to those who have the correct equipment would be very expensive, given the low volume required. Injection Moulding only becomes viable when the volumes increase, since the starting costs (that of the mould and equipment) which must be overcome, are very high, hence would not be possible for us. Additionally, we would not even have just one type of propeller, in fact requiring both CW and CCW propellers, since by the specifications of a X-type quad, two motors spin clockwise and two spin counter-clockwise (hence CW and CCW), effectively doubling the already high costs. Therefore, we decided to purchase ready-made propellers of 50mm size. Additionally, buying propellers means we could easily buy other sizes later if they are required. In fact, these would also be free, since we were sponsored by HobbyKing, the company from which we chose to obtain the propellers.

5.12 - Testing of the Mechanics

So far there has been rigorous testing of all individual components (i.e.: the 3D printed parts):

- We have tested and refined each piece such that now all of them fit snugly together.
- Each piece has been stress tested to ensure it can take reasonable stresses – this we did by bending, flexing and compressing each piece by hand until they began to show signs of snapping. Based on how hard it was to reach this point we decided whether the pieces would be strong enough. The only piece which we decided needed improving was the part of the arm which the motor attached to (Figure 5.12.1). This piece we have since increased the thickness of. Luckily increasing the thickness of this piece did not intrude on the space allocated to any other component.

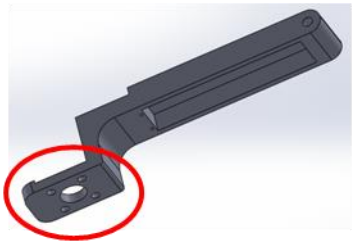


Figure 5.12.1

- We have also tested how well the housings for specific components hold the intended components. The description of the refinement of the arm design details how we encountered difficulties and resolved them for the motor housing, and the description of the specific details of the mechanism for opening arms details how we did the same for the mounting of the servos for the deployment of the arms.

In addition to this we have begun testing how well the different components perform when as part of a subsystem:

- We have fixed all the pieces together and performed stress tests as before, encountering no problems. We hope to improve these tests by fixing the actual motors in place and running them – further details of this can be seen in the integration of electronics and software within the mechanics section.
- We have also, as seen in Figure 5.8.1, tested the arms opening by the tightening of wire; we hope to test this system using the actual servos soon.

All further testing is non-inter-mechanical-component, and so is detailed in the integration of electronics and software within the mechanics section.

5.13 - Further progress

In addition to all further testing of the mechanics, all further development of the mechanical design is detailed in the integration of the electronics and software within the mechanics section.

The reason for this formatting of the report is to allow us to clearly show how the testing has dictated the further changes to the design – as this is the only reason for changes to be made to the design (and all further testing is non-inter-mechanical-component, and so is detailed in the section mentioned above).

6 - Electronics

The software and electronics were largely split into three parts, which would not be interconnected except for sharing the same power source, as we wanted to ensure a fail-safe that if one system failed, the others would continue to work. Additionally, to provide more flexibility, the system that would be used in order to open the arms, with the two servos, will be connected to the sensor system. Additionally, this means we could trigger the system to work with changes in altitude if we wanted (though this may unreliable, and so too risky).

6.1 - Flight System

For the quadcopter system, the first thing we decided was the exact parts we would choose. The actual types of parts we required rather decided themselves, since we did not have the space to add any unnecessary parts. Thus only vital components were chosen. The first main decision that we made was that we would buy as many parts as possible for this system given the high complexity required to keep a quadcopter in flight, and the lack of space and time for parts that we might make ourselves. Additionally, parts such as ESCs and the Control Board, which we could theoretically replicate were very good, reliable and cheap, and since we were sponsored by HobbyKing, became free. Below, is the list of the exact parts we chose and why we decided that they were the right choices. Since the first progress report, there was one major change, the Control Board, which was required, simply due to the fact that the old control board was ineffective, not providing enough flexibility. Additionally, this was backlogged with no prospect of it returning to stock, thus if we broke the one board we had inherited from a team member, we would have no control board. Additionally, by looking outside of this board, we in fact managed to find an equally easy to use, but far more effective board.

6.1.1 - Battery – Turnigy Nano-Tech 3s (11.1V) 850mAh[3]

The power supply system is vital to the entire project, since any failure in the system could prevent the operation of the entire CanSat. We need to thus ensure that all components of all systems receive a safe voltage. Additionally, the battery needs to be very small, given the lack of space that we have in the can. However, the smaller the battery the lower the capacity of the battery. In order to maximise the capacity per unit area, the use of a LiPo battery is the best choice. Though it is a very powerful battery, there are a few issues related to the safety of the battery, since LiPos are liable to explosions if they are overcharged, discharged or indeed short-circuited. However, certain batteries, such as the battery that we have chosen to use has specific protection built into the system to ensure that there is no excess in voltage or indeed current, which could be damaging. Though this could have been achieved using a Zener Diode and a collection of transistors, or indeed specific Integrated Circuits, it would have taken up some space and have probably been less effective than the system integrated in the battery.

Given that the motors have to lift 370g of the can, the majority of motors we could use appeared to require the use of a 3s (11.1V) battery, so we chose to use this battery, since it was the one which best fit into the existing mechanics of the can. Additionally, using our existing expertise and extensive online research we have found that this should be able to provide around 5 minutes of flight time, our aim, while also powering the sensor system. However, also through testing, we have found that the camera system takes a lot of power, so we hope to include a high-current logic-level MOSFET to control the camera remotely, so that we could keep the camera off when the can is idle.

In order to distribute all the power according to the wiring chart (see Figure 6.4.1), we have designed a power distribution PCB, (see Figure 6.1.1) in order to allow power to go to the correct places. Additionally, on board we have a 5V voltage regulator, which will provide the correct voltage for the motor which opens the arms of

[3] http://bit.ly/1GAFMWi

the quadcopter. We have chosen to rely on the voltage regulator of our microcontroller (the Teensy 3.2) to supply the 3.3V required for the rest of the sensor system.

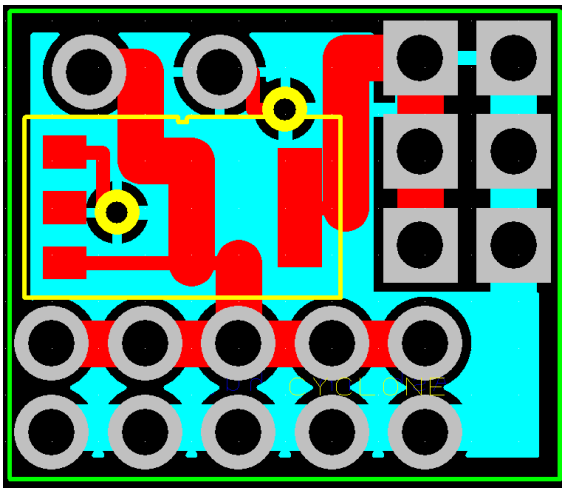


Figure 6.1.1: Power Distribution PCB (designed using DesignSpark PCB[4])

6.1.2 - Motors – Turnigy Outrunner v2[5]

For the motor choices, we were forced to consider two important aspects, the size of the motors and their power. In order to provide a little leeway, we wanted to ensure that the motors we chose had at least a thrust of 400g, thus it could lift 400g. This meant that the four motors together had a thrust of over 1.6kg, more than enough to comfortably lift the 370g of the CanSat and accelerate it upwards. In order to calculate the thrust, we used online calculators. We soon found out that we would not struggle with thrust, with the majority of motors meeting our specification, but would struggle with size. The Turnigy Outrunner v2 motors were selected because they were the smallest motors (15.5g) we could find but also one of the most powerful, with a KV specification of 1900RPM/V. Though normally very expensive, the sponsorship by HobbyKing (from whom we purchased the motors) meant that they were the best choice.

6.1.3 - Control Board - OpenPilot CC3D Atom[6]

This specific control board was chosen due to its simplicity to use, as well as it having many people online who have used and support the CC3D, with tutorials on how to get started with the board. Additionally, it is tiny, with a size of 16x16mm, thus reducing the amount of space required for the board. Finally, it had sufficient specifications, including auto-stabilisation using a barometer to reduce the amount of coding required. Additionally, it was easy to program, with a very nice, cross-platform GUI called Ground Control Software, where the precise settings of the board, as well as calibration could be easily changed. As opposed to the previous choice, the Hobbyking i86, this is much smaller, as well as being better supported. Additionally, according to most, it is more effective, as, as opposed to the i86, it has the options on how to change the calibration settings of the ESCs and RX system.

6.1.4 - ESCs – Turnigy Nano Tech 20A[7]

Given our motor choice, we knew we needed a 20A ESC. By working together with the Mechanics team, we chose this ESC, as it was long and thin, thus would fit in the arm as required, most easily. Additionally, the ESC

[4] http://bit.ly/1AWvlGh
[5] http://bit.ly/1PUlKbW
[6] http://bit.ly/1PUlKbW
[7] http://bit.ly/1LFUOc8

was highly recommended with most users online having been able to use it with few issues. However, as we progressed through the project, we in fact had a number of issues with the ESCs. In fact, the first ESC we connected to power was faulty and burst into flames! Later, we also learned that the ESC did not have a BEC on board, thus meaning it could not be used to power the control board, thus explaining a number of issues with the control board not powering up. This was not in fact a large issue, since we already had a 5V output from the power distribution board that we could use if necessary, yet, this made a significant impact on the Wiring Diagram.

6.1.5 - TX / RX system – Orange Nano8

This TX / RX system was chosen because it was very compact, thus most easily fitting in the little space we had in the Can. It was also very highly rated by many other users, who had found it easy to set up. Additionally, Orange, the manufacturer is a high end manufacturer, and normally produce reliable products.

6.2 - Sensor System

6.2.1 - Components

The first choice we made was to use an Arduino microcontroller, an obvious one, given the team's familiarity with it, and the vast availability of parts and examples. We chose to use a microcontroller board rather than a microcontroller, given the significant amount of supporting circuitry required for regular operation and further circuitry required for reprogramming. We have chosen to use the '**Teensy 3.2**[9]' a very small board (as the name suggests), which includes all the equipment required to reprogram the Can. Additionally, it is very powerful, containing an ARM Cortex M4, far superior to the Atmel chips on other Arduino boards. It also contains a 3.3V voltage regulator, rated for 500mA which we could rely on. For many of the components, we chose to work off many of the choices made by Team Impulse, of whom a couple of members (including their Team Leader (and Head of Software and Electronics) – William Eustace) we had inherited. Thus we immediately chose to use the **MS5637**[10] and **HYT271**[11], the pressure sensor and relative humidity sensor that they had used, since they had been very effective. Though we had considered using the BME280, a Bosch sensor which included temperature, pressure and humidity sensing in one chip, we felt that it was too inaccurate, and too small to feasibly be soldered by hand. Additionally, we chose to harness both the MS5637 and HYT271's abilities to sense temperature to find a more accurate temperature of the surroundings by averaging their results. We also chose to use the **Hope RFM98W**[12], since by using Spread Spectrum Technology, the module is able to more accurately send all the information, over a longer distance. In fact, even without the use of a Yagi, the sensor has been found to be able to send data with little error over 3km, a larger distance than we would ever encounter over CanSat. Additionally, the RFM98W can be used to perform cyclic redundancy checks, ensuring the amount of data received is equal to the amount expected, which would allow us to ensure that errors in receipt of data can be ignored. Though originally we also planned to build in the DRV8833PWR to control the motor to open the arms, this is no longer required, since we are now moving towards two servos. However, this does mean that a second revision of PCBs would be required, since the old PCB is no longer correct.

However, from here the similarities with Team Impulse's electronics end, as we chose different parts. Firstly, we have chosen to use the **GP-2106**[13] GPS module as it has been very effective in testing, and has a very small footprint, much smaller than the GPS module used by either Team Colossus or Team Impulse (last year's

[8] http://bit.ly/1M4X4Ln
[9] https://www.pjrc.com/teensy/
[10] http://www.meas-spec.com/product/pressure/MS5637-02BA03.aspx
[11] http://www.hygrochip.com/index.php?id=3854&L=1
[12] http://www.hoperf.com/rf/lora/RFM98W.htm
[13] https://www.sparkfun.com/products/10890

teams). Additionally, we have chosen to use the Evaluation Breakout[14] produced by Sparkfun, to help use the module, which has proprietary connectors, and makes use of 1.8V logic. Finally, we are going to use the **Sparkfun 9DOF Breakout**[15], as an IMU, containing a 3-axis accelerometer, 3-axis gyro and 3-axis magnetometer. This is because it is very small, very accurate, and is well supported by Sparkfun, with them having produced a very good Arduino library for the device.

6.2.2 - PCBs

The main sensor system PCB (Revision 1) is shown below:

It was designed in RS / Allied DesignSpark PCB 6[16]. All PCB files are available on the GitHub repository[17], under the Electronics Design section.

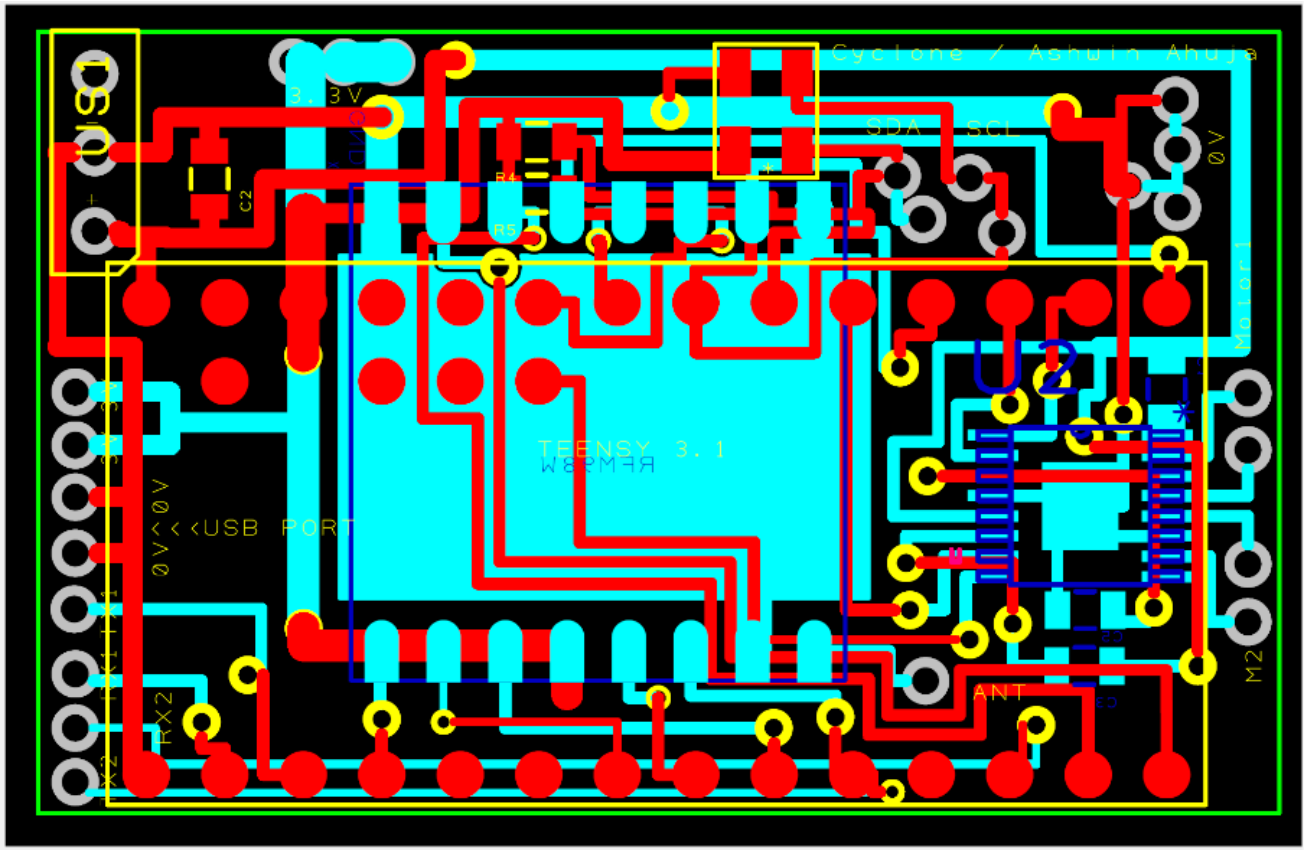


Figure 6.2.2.1a: The design for the main sensor PCB

The board shows how the Teensy 3.2 is mounted on the top layer of the board, using Surface Mount Soldering. This was done so that we may conserve space, and could be achieved by ensuring that some insulation is placed between the board and the Teensy, and then soldering the pads of the Teensy directly to the pads on the board. Also on the top layer is the MS5637 – a 4 pin QFN package, as well as resistors required for the I2C line. On the bottom side, there is the Hope RFM98W, a breakout which will be surface mounted to the PCB. There is a ground plane under this, as recommended by the manufacturers, in order to reduce noise. Also, under the Teensy, (labelled U2) there is the Motor Driver (DRV8833-PWR), using the HTSSOP-16 package, and associated resistors and capacitors. It is clear however, that this board does not have the capability to manage

[14] https://www.sparkfun.com/products/10995
[15] https://www.sparkfun.com/products/10736
[16] http://www.rs-online.com/designspark/electronics/eng/page/designspark-pcb-home-page
[17] http://www.github.com/CycloneCanSat

the opening of the arms, since it was produced when the plan for arms opening was using a single motor. Hence, in the second revision, we plan to build in this ability as well as fixing a number of other irritations that have been found when the board was soldered. Finally, dotted around the board, there are a number of connectors for the many breakouts and sensors which must be on flying wires. Given this PCB is at the bottom of the Can, the Humidity Sensor will be just below, so that it may have exposure to the air. Additionally, the IMU breakout and GPS evaluation board will be on the same layer, attached with short wires. The GP-2106 module, however has to be at the very top of the Can, to ensure that it can get a fix, and find the Can's position.

PCB Assembly and Review

The manufactured PCB was as below:

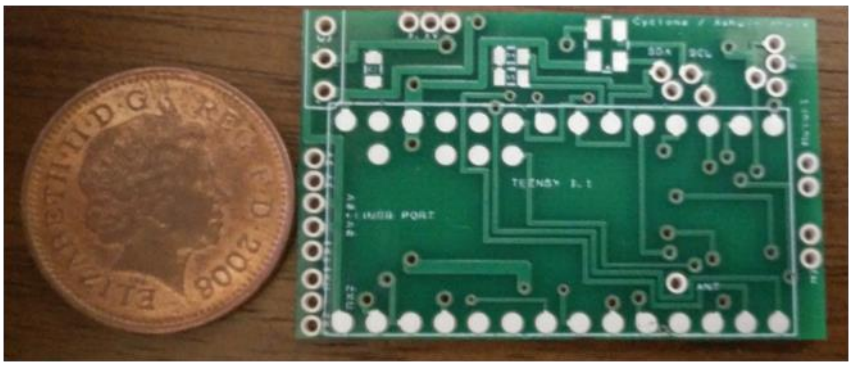


Figure 6.2.2.2 – Front view of manufactured PCB

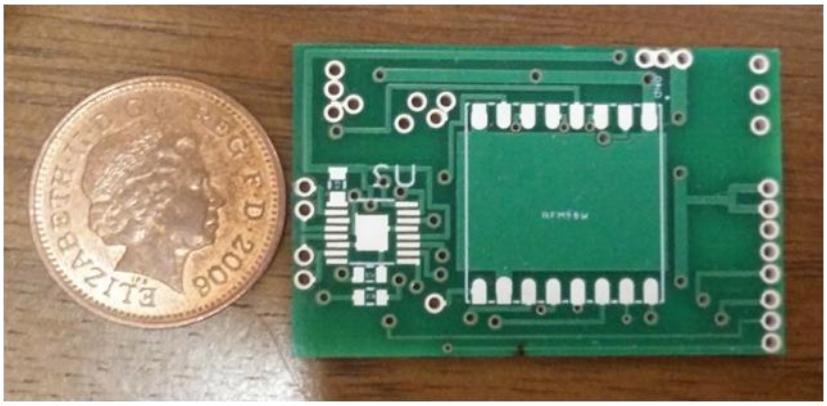


Figure 6.2.2.3 – Rear view of manufactured PCB

When all the parts had been soldered to the board (with the exception of the Hope RFM98W which had not arrived at the time of writing), the board, looked like this:

Figure 6.2.2.4 – Front view of assembled PCB

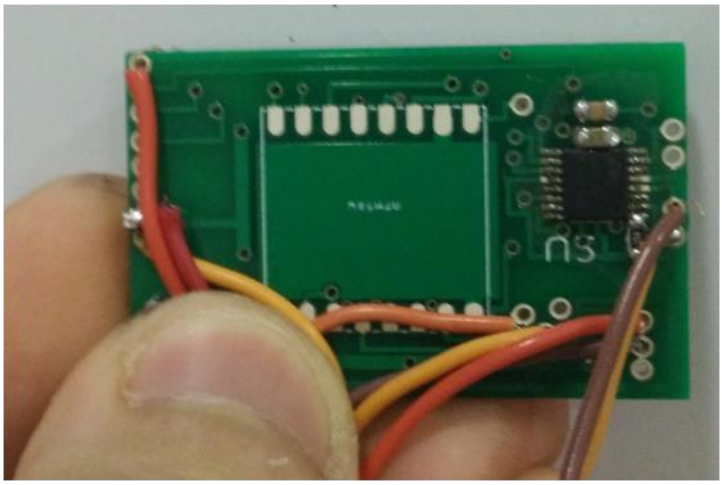


Figure 6.2.2.5 – Rear view of assembled PCB

From preliminary testing, all the components on the PCB work, including the ports which are made for Off-Board connectors, which have tested using easily removable Female Header Wires, connecting to Male Header Pins on the breakouts, to ensure no damage is done to the expensive breakout boards. This suggests that all the electronics design is entirely correct. However, there are a couple of minor issues. Firstly, the mechanism of soldering the Teensy to the PCB, by soldering through the holes in the Teensy straight to the PCB underneath despite working, was quite painful, requiring a lot of care. However, given the amount of space gained through this mechanism, it is likely that we will stick with this as a method of connecting the two. Additionally, a number of pins, placed directly underneath the Micro-USB connection of the Teensy were very hard to solder, and are always very close to touching the connector, sometimes shorting it out. Thus, in the next revision of the board, as shown below, these were fixed, as well as removing the unnecessary connections that the first revision provided, for example, the provision of a Motor Driver (DRV-8833) – a preliminary idea for the opening of the arms was no longer needed.

The updated PCBs have been ordered and are to be delivered in the next couple of days, hence should be ready for launch, however, the old PCBs are still functional and could be used in case there are any problems with the new revision.

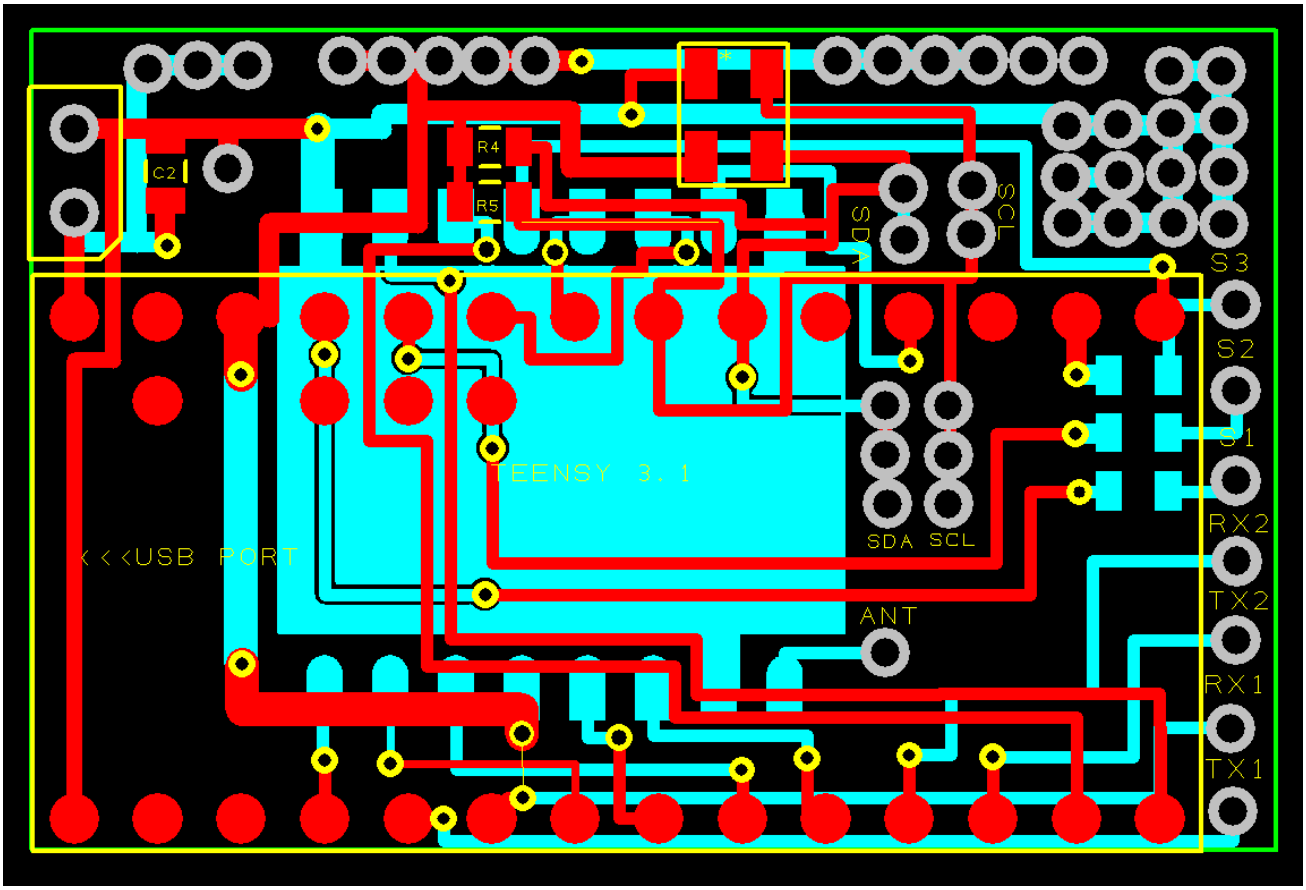


Figure 6.2.2.1b – Revision Two PCB Design (made using DesignSpark PCB)

6.3 - Camera System

For the camera system, we have chosen to use a pre-made FPV system, with the **Mini FPV Camera**[18] combined with the **HobbyKing FPV Transmitter**[19]. Though we had considered using an Intel Edison and Wi-Fi transmission with a NTSC webcam, it transpired that it would take up more space, as well as being a lower quality. Thus, we had the choice of 5.8GHz or 2.4GHz FPV. Though there were some problems regarding legality of using certain powers of 5.8GHz transmission, through more research, it is clear that one can use devices with a transmit power of under 25mW without the necessity of an Amateur Radio License[20]. Though this severely limits the quality of image we can use, it is still superior to the quality one receives when 2.4GHz is used. Additionally, there are similar issues regarding the maximum transmission power of 2.4GHz. Despite originally planning to make use of an OSD we have found that it takes up too much space for little gains, since the voltage of the batteries can easily be felt by monitoring the effects of the thrust applied by the transmission system – since this will reduce over time.

[18] http://bit.ly/1PYEpTe
[19] http://bit.ly/1RhYGBA
[20] Ofcom IR2030/27/3 – Available here (page 63): http://stakeholders.ofcom.org.uk/binaries/spectrum/spectrum-policy-area/spectrum-management/research-guidelines-tech-info/interface-requirements/IR_2030.pdf

6.4 - Communications

Given the large amounts of data required, a number of different data transmission methods are being used to stream data to and from the Can. As most RC systems work off 2.4GHz, we are following this trend, as this allows to exploit the large number of small TX and RX units specifically built for RC helicopters and indeed quadcopters. Additionally, the use of this transmission method links up well with the **OpenPilot CC3D Control Board**[21], without much work, thus allowing us to simplify the process required to get the flight system working manually, thus allowing more time to produce autonomous movement.

For the camera system, as discussed above, we have chosen to make use of the 5.8 GHz transmission systems, though there are some problems with the low penetrating power of this high frequency. However, given the low distance that the CanSat will travel, we have concluded that this is not a large problem, especially given that we will always have line-of-sight with the Can.

Finally, for the sensor system, we are going with the tried-and-tested 434 MHz RF transmission frequency, given the lower path loss and thus large distances we could easily get using this protocol. This ensures a high reliability of data transfer, and though this means we could transfer less data, this is not a large problem, since the data, probably 32 bits in length, would only need to be transferred once a second. Given the specifics of the spectrum provided to us, we can also begin to determine the specifics we can use. We have been provided with a specific frequency of 434.07 MHz. Given the teams around us have 433.98MHz and 434.25MHz[22], a safe bandwidth would be around 100kHz, thus the nearest setting, 62.5kHz will be used. Making use of the LoRa Modem Calculator Tool made by SemTech[23], this will allow a data rate of 3348 bps, assuming the lowest possible Spreading Factor (6), with a link budget of 139 dB and a receiver sensitivity of -122 DBm.

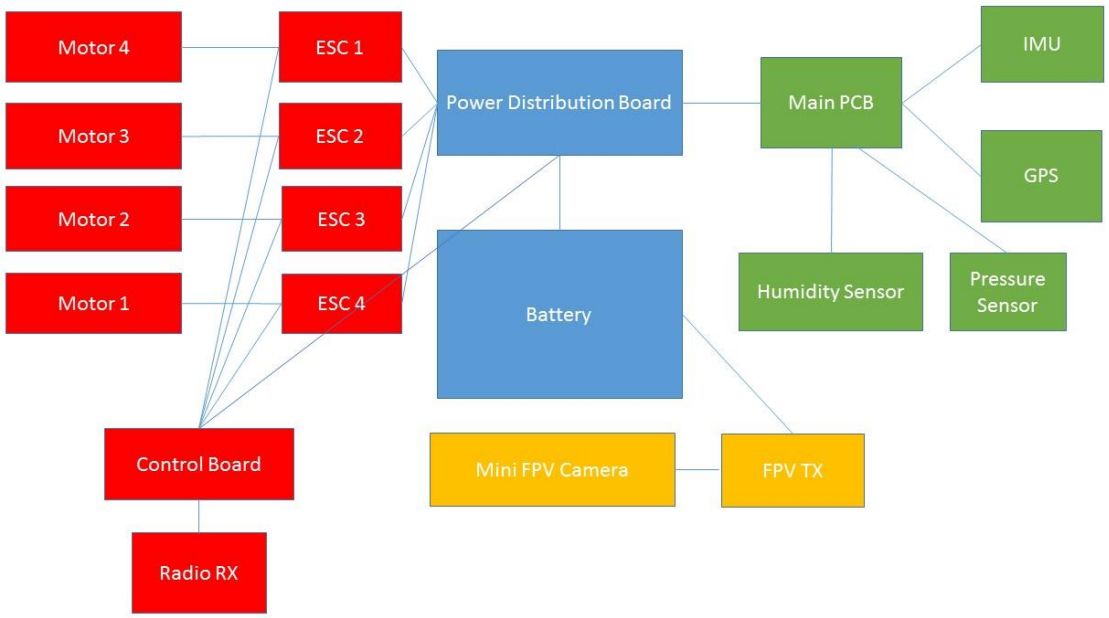


Figure 6.4.1: Diagram of system wiring (made using Microsoft Powerpoint[24])

[22] According to the spreadsheet of allocated frequencies provided during the 2015 CanSat Teacher's Workshop
[23] http://www.semtech.com/apps/filedown/down.php?file=SX1272LoRaCalculatorSetup1%271.zip
[24] https://products.office.com/en-gb/powerpoint

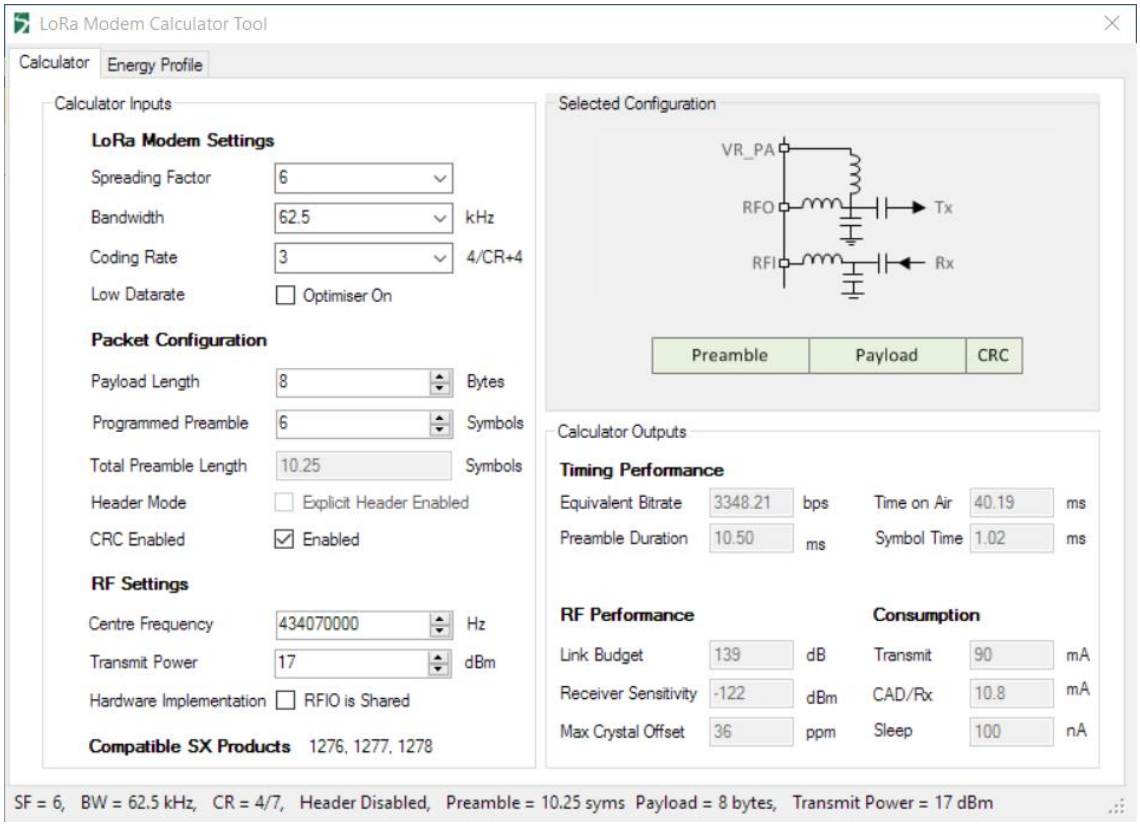


Figure 6.4.2: Calculations for Hope RFM98W, made using the SemTech LoRa Modem Calculator Tool

6.5 – Initial Single-System Testing

Prior to the combination of the Electronics and Mechanics it was import to test each system separately, hence finding any problems with the electronics in and of itself, hence allowing us to resolve them before combining it with the electronics. Hence, we followed the following steps. Once this testing was completed, the integration of electronics and mechanics occurred, which is documented in 8, including the changes which occurred because of the full-system testing.

6.5.1 - Sensor System

In order to test the sensor system prior to the PCBs' arrival, working alongside the software elements of the team, we tested the individual components, producing code individually for each one, before attempting to combine them. This tactic was however, ineffective for the pressure and temperature sensor, where the SMD package meant that it would impossible to test before receiving the PCBs, thus the previous working uses of the product as well as the datasheet was very closely investigated, in order to ensure the wiring (though in essence very simple) was not incorrect.

Once the PCB had arrived, we attempted to replicate the same process, using the PCB, and this was relatively successful, though revealed the problem that some of the pins were not connected to the Teensy upon the board first being soldered, due to the rather tricky mechanism of connecting the two, though the mechanism was clearly worth the pain. However, once all the components were soldered, we were able to test them one by one before moving to a complete system test, which clearly showed that the design of the PCB was perfect with no apparent problems, apart from the location of a couple of the pads meaning that the USB port was slightly obstructed.

6.5.2 - Camera System

The camera system was initially tested easily, with the setup being well documented. Though the quality of the images produced was not very high resolution, it was at a very fast frame rate, which ensured there was little to no lag, a very important and necessary part if the quadcopter was needed to be piloted manually using the camera. Additionally, we found the field of vision rather low, with it being around 90 degrees. Hence, we found a 120-degree wide angle lens, which would make it much easier to fly using the camera if necessary. Finally, we are storing the video using a recorder which outputs the video into an SD Card, to allow the video to be kept for posterity and in case the link gets cut.

Figure 6.5.2.1 – Camera system sample frame

6.5.3 - Flight System

The first part of the flight testing consisted of using a frame equivalent to our CanSat and mounting all the parts on it, in order to test it would be able to fly. However, due to the lack of our power distribution board, we were also forced to use an external BEC and a bulky bought Power Distribution Board, thus increasing the size of the setup. However, this (as shown in the image) was able to be assembled, and then flown (with a mass of 370g), easily being capable of lifting its weight with less than 30% throttle required, however with larger propellers than the ones which we plan to use. Hence the next step was to use the correct propellers, attempt to do the same, before moving on and testing the system inside the actual can, with the other systems, which we hope to accomplish before the end of the year. From here, any problems could be fixed, with any required redesigns to the electronics or mechanics.

Figure 5.5.3.1 – Frame for Flight System testing

6.5.4 - Battery Management

A central concern around the electronics system is the battery size and whether the system will fly for a sufficient period of time. This was attempted to be found during testing. The power was left on for the motors for in excess of 15 minutes, depleting the entire battery in this time. Given the power requirements of the sensor system and camera systems are minimal in comparison to the flight system, this is largely promising. In comparison to this, the camera system remained on for in excess of five hours in testing, in fact starting to heat up the FPV Transmitter before anywhere near running out of power. It is important to note, while the motors will begin to turn off when the voltage drops below around 11V, the sensor system would continue to work until around 7V (the limit of the voltage regulator), while the camera system would work until around 6V! Hence, if this were to be flown to an unknown area, the can has the benefit of continuing to transmit positional data to help locate it if required, even if the battery is too low for the quadcopter to fly anymore.

Current Draws from Sensor system components (used to help determine battery life)

Component	**Current Draw (mA)**	**Conditions**
Teensy 3.2	*38*	*Flashing an LED*
GP-2106	*65*	*Reporting GPS Data at 1Hz*
HYT-271	*4*	*Reporting Humidity and Temperature Data at 1Hz*
MS5637	*2*	*Reporting Pressure and Temperature Data at 1Hz*
LSM9DS1	*10*	*Reporting all possible Data at 1Hz*

Figure 6.5.4.1 – Current Draw of Sensor System

7 - Software

From the fore, the team knew that there was a number of different strands of software that were needed in order to control the can well, allowing it to become an autonomous drone. We decided, alongside the Electronics team, to keep the flying systems (through the main control board), the sensor systems and camera system separate, thus allowing us to ensure that in case of failure of one system, the others would continue to work. We also divided the team into a number of different strands, with some working on the website, alongside the outreach team and others with the electronics team. We also aided in the outreach, putting all of our code on GitHub (http://github.com/cyclonecansat) under the MIT license, thus allowing anyone else to use our code if they wanted to. Additionally, all the code is included in the Appendices. We are also attempting to produce documentation to better explain how everything works, for this to be even easier. In many ways, this process also works the other way, as we rely off some work others have completed, especially in the sensor system, learning from their examples. The flying system of the Can required little to no software, given that we were using an advanced control board which includes auto stabilisation. Additionally, we decided that the autonomous aspect of movement would be achieved by altering the signals sent to the flying system, using a computer to calculate the signals to send to the RX of the Can. However, we were still required to write a very complex base station software to allow us to send the correct commands to the Can, so it could monitor the place of the Can at all times and issue commands to move it to a user-defined location. Since the first progress report much of the sensor library has been written / adapted from other sources, producing a system which has been used to individually test specific components. From here, the aim is to put the sensor system together, forming one complete program which will control all sensors as required.

7.1 – Algorithms

7.1.1 – Designing the Algorithm

To design an equation for estimating agricultural viability of a particular site, we thought about how the different inputs (temperature, humidity and pressure) would affect the growth of crops. We came up with several mathematical equations to illustrate these relationships, plotting several graphs where the y-axis is a measure of agricultural viability with a maximum value of 1 and the x-axis is the input.

Temperature would control a plant's enzymatic activity, meaning that temperatures near the optimal temperature for a given enzyme would favour plant growth. To show this representation graphically, the graphs of $y = \frac{-2}{3x}$ and $y = 1 - \frac{x^2}{3}$ were combined to produce the graph shown here. The lines cross at x=-1 where the two gradients are the same, ensuring the curve is continuous. When x is less than -1, $y = \frac{-2}{3x}$ is used, showing the increasing enzymatic activity as temperature increases and hence the increasing agricultural viability. For values of x greater than -1, $y = 1 - \frac{x^2}{3}$ is used, with the steep drop representing the effect of enzyme denaturing at high temperatures. For appropriate use by the can, the measured value for the temperature would be altered such that the optimal temperature (e.g. 27°C for rice) returns a value of 1 for agricultural

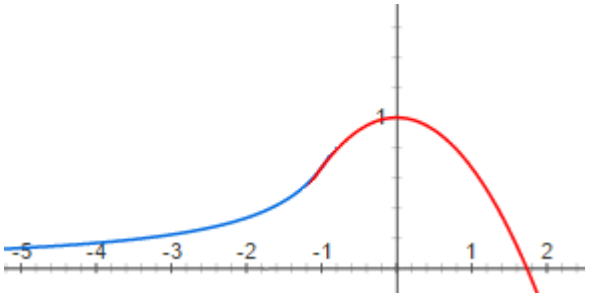


Figure 7.1.1.1: Graph used to find Agricultural Viability from temperature

viability and normalised such that a range of suitable temperatures still return relatively high values for agricultural viability. The graphs could also then be stretched to allow a sensible range of temperatures to yield high viability. For rice we will replace x with $\frac{T-27}{9}$ which creates a maximum temperature of T=42.5.

We decided that humidity could be a proxy for soil water content, with higher humidity resulting in more water available for crops and so a higher agricultural viability. Therefore, we decided that $y = -x^2 + 2x$ could be used. The humidity measured by the can would be divided by 100 so that 100% humidity returns a value of 1 for agricultural viability.

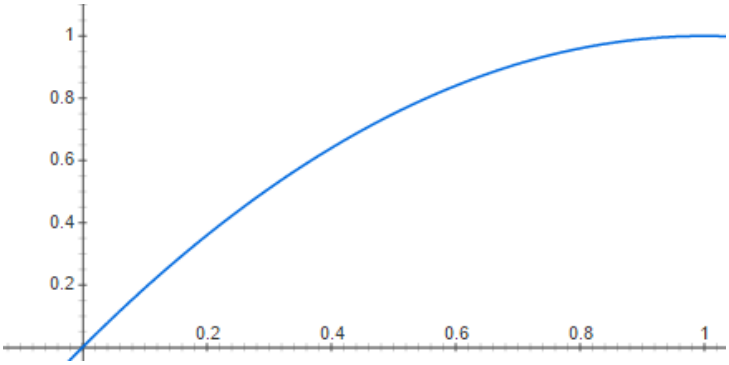


Figure 7.1.1.2: Graph to find Agricultural Viability using humidity

We assumed that for Earth-originating crops, a pressure of 1atm would be ideal for a plant, with a vacuum being more detrimental than higher pressures. Taking this into account, the graph of $\frac{-x^2+3x-1}{x}$ was chosen as it displayed the features we wanted. As it peaks when x=1, this graph would not require any further calibrating provided the pressure is in atmospheres. The three values for agricultural viability multiply together to give an overall value between 0 and 1.

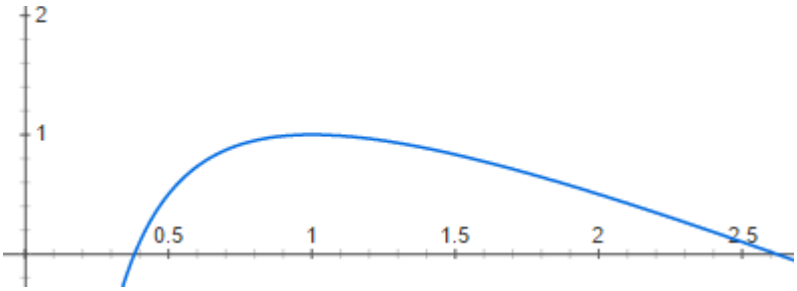


Figure 7.1.1.3: Graph to find Agricultural Viability using Barometric Pressure

Summary:

$$Agricultural\ Viability = V_T \times V_H \times V_P$$

where;

	Calculation	Condition	Notes
V_T = Viability of Temperature	$V_T = \frac{-2}{3(\frac{T-27}{9})}$	For $T < 18$	Where T = temperature in degrees Celsius.
	$V_T = 1 - \frac{-(\frac{T-27}{9})^2}{3}$	For $18 \leq T \leq 42.5$	
V_H = Viability of Humidity		For $0 \leq H \leq 100$	Where H is relative humidity in %

$$V_H = -\frac{H}{100}^2 + 2\frac{H}{100}$$

V_P= Viability of Pressure $V_P = \frac{-P^2 + 3P - 1}{P}$ For $0.4 \leq P \leq 2.6$ Where P = pressure in atmospheres

Figure 7.1.1.4 – Table of equations for calculating Vt, Vh and Vp

7.1.2 - Assessing and improving our algorithm

	Nebraska	Minnesota	Illinois	Nevada	Texas	Virginia	Alaska	Copenhagen	Timbuktu
Temperature/C	10	5.8	11.4	11.4	21.1	14.2	5.6	11	29.1
Pressure/atm	1.002714	1.002714	0.9967925	1.006662	1.002714	0.9987664	1.00074	0.990781	1.004688
Humidity/%	45	55	50	20	55	60	90	85	5
V(temperature)	0.545454546	0.394736842	0.625	0.625	1.000041152	0.882352941	0.38961039	0.6	1.27
V(pressure)	0.999992654	0.999992654	0.999989679	0.999955912	0.999992654	0.999998476	0.999999453	0.999914219	0.999978125
V(humidity)	0.6975	0.7975	0.75	0.36	0.7975	0.84	0.99	0.9775	0.0975
Agricultural Viability	0.380451751	0.314800319	0.468745162	0.22499008	0.79752696	0.741175341	0.385714075	0.58644969	0.123822291
Actual Agricultural Output	90	50	75	10	90	50	10		
High									
Higher Middle									
Lower Middle									
Low									

Figure 7.1.2.1 – Table comparing predicted agricultural viability versus actual agricultural output

To test out our algorithm for agricultural viability, we input existing data for temperature, humidity and pressure for a range of locations. Because of the abundance of available data and the diversity in climate across the region, we decided to use data from across the USA. The algorithm was calibrated for growing wheat, the most widespread crop in the area, by setting the optimal condition for temperature to 21°C. In addition to the data from the USA, we also tested the algorithm for two more locations that we expected to be at opposite ends of the spectrum for growing wheat: Copenhagen, Denmark, one of the world's largest producer of wheat per hectare, and Timbuktu, Mali, a city in the Sahara Desert.

To measure how well our algorithm worked, we compared the algorithm outputs with data for the percentage of land that is farmland in each region, assuming that areas which have a higher natural agricultural viability would contain a higher percentage of farmland.

Overall the algorithm outputs match up relatively well to our expected values. The greatest dissimilitude between an output and the real world situation was the value for Nebraska, with the algorithm returning a relatively low value for agricultural viability despite it being one of the most agriculturally productive regions that we looked at. We believe this mismatch is largely down to other factors that our algorithm does not take into account, such as soil quality and precipitation, areas in which Nebraska performs well.

There are two main ways that we seek to improve the algorithm. Firstly, when calculating the viability of the temperature, our algorithm returned values that were greater than 1 for Texas and Timbuktu. This should be impossible as 1 should be the maximum possible value, indicating that the temperature is at the optimal temperature. This is an issue we will look to fix. Also, while researching the agricultural output of certain

regions, we noticed that often areas that produce a lot of a certain crop have conditions that are not necessarily close to the optimal conditions. This is most evident in the data for Nebraska and Copenhagen where the average temperature is around 10°C below the optimal temperature, yet the crop production is still very high. As it stands this difference causes the output value for agricultural viability to be significantly affected. To address this, we will improve the equation for the viability of temperature to allow a wider range of inputs and still return a high value for overall agricultural viability, more accurately reflecting real life.

Fixing the equation for calculating the viability of the temperature proved to be merely fixing a single sign error. We then wanted to improve the algorithm by making it less sensitive to small changes in temperature, as we found that most crops have a wide range of suitable temperatures for growing. Our first algorithm did not account for this range of acceptable values so we redesigned the equation for calculating the viability of the temperature. By transforming the graph of the equation used for the first algorithm, stretching it horizontally to return high values for a wide range of temperatures, we decided that the graph of:

$$V_T = 1 - \frac{(\mathrm{T} - 27)^2}{972}$$

would work better. Replacing our first algorithm with this new one yielded the following results when tested against the same data as before:

	Nebraska	Minnesota	Illinois	Nevada	Texas	Virginia	Alaska	Copenhagen	Timbuktu
Temperature/C	10	5.8	11.4	11.4	21.1	14.2	5.6	11	29.1
Pressure/atm	1.002714	1.002714	0.9967925	1.006662	1.002714	0.9987664	1.00074	0.990781	1.004688
Humidity/%	45	55	50	20	55	60	90	85	5
V(temperature)	0.7026749	0.53761317	0.74962963	0.74962963	0.96418724	0.83144033	0.52884774	0.73662551	0.99546296
V(pressure)	0.99999265	0.99999265	0.99998968	0.99995591	0.99999265	0.99999848	0.99999945	0.99991422	0.99997813
V(humidity)	0.6975	0.7975	0.75	0.36	0.7975	0.84	0.99	0.9775	0.0975
Agricultural Viability	0.49011214	0.42874335	0.56221642	0.26985477	0.76893368	0.69840881	0.52355897	0.71998967	0.09705552
Actual Agricultural Output	90	50	75	10	90	50	10		
High									
Higher Middle									
Lower Middle									
Low									

The main noticeable improvement is that higher values for agricultural viability are returned for some of the locations with temperatures that are relatively far from the optimal temperature (which in this case was taken to be 27°C). This was the desired effect so it would appear that the changes to the equations were successful. To check whether the algorithm really had improved we checked the Spearman's rank correlation coefficient for the results of both algorithms. We compared how our range of locations ranked when put in order of our agricultural viability calculations with their ranking when put in order of their actual agricultural output. This would give us an indicator of how well our models reflect reality. The Spearman's rank correlation coefficient (rS) for each algorithm would return a number between -1 (perfect disagreement) and 1 (perfect agreement). The results are as follows:

Old Algorithm – rS = 0.671

New Algorithm – rS = 0.746

The correlation coefficient has increased, indicating that our new algorithm now models real life more accurately.

Hence, the complete algorithm is:

	Calculation	Condition	Notes
V_T = Viability of Temperature	$V_T = \frac{-2}{3(\frac{T-27}{9})}$	For $T < 18$	Where T = temperature in degrees Celsius.
	$V_T = 1 - \frac{(\mathrm{T}-27)^2}{972}$	For $18 \leq T \leq 42.5$	
V_H = Viability of Humidity	$V_H = -\frac{H}{100}^2 + 2\frac{H}{100}$	For $0 \leq H \leq 100$	Where H is relative humidity in %
V_P= Viability of Pressure	$V_P = \frac{-P^2 + 3P - 1}{P}$	For $0.4 \leq P \leq 2.6$	Where P = pressure in atmospheres
V_A= Complete Agricultural Viability	$\boldsymbol{V_A = V_T \times V_H \times V_P}$		

7.2 - Website

The Software Team has worked hard to design (and continue to develop) a responsive website, now available at: http://teamcycl.one which is both very attractive aesthetically, as well as containing a lot of information, allowing visitors to easily find out more about the project and indeed us. The website even has an embedded blog where each team has so far written an article, including the Software and Electronics team. We hope to continue doing this and even make them more frequent as the launch draws nearer. This website has been extensively promoted using our social networks, and so far has encountered over 3500 page views[25], thus allowing this many people to learn more about our project. The website proved an interesting challenge for

[25] Statistics found using Google Analytics

us, as we attempted to use all aspects of web development to our advantage, using JavaScript to allow the website to be responsive, CSS for stylistic aspects, and HTML(5) for the core programming. Additionally, a mobile website was designed, as this responds to the idea that more and more of the visits we would receive would be from mobile devices. The mobile website was designed to be as simple as possible, while still very usable. This has proved very successful, as the average time spent on our website (approximately 2 minutes on other devices) is closer to 4 for mobile[26]. Finally, we decided to produce mobile apps for Android and Windows Phone, in order to reach a greater audience even more easily. To do this, we employed the abilities of the Software Teams, producing apps in Swift, Java and C# respectively. To date, Android and Windows Phone apps have been launched to their respective App Stores, and been successful, with the Android app having more than 400 downloads[27].

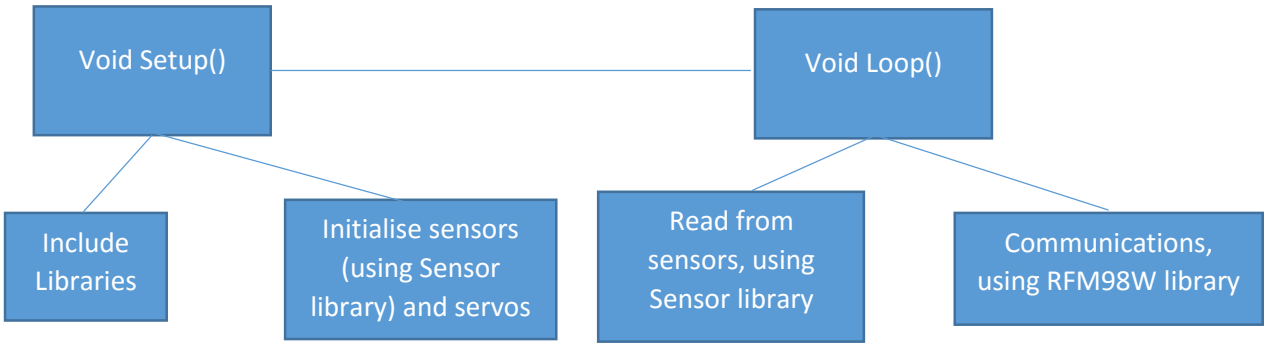
7.3 - Can Code

Since the last progress report, the Can Code has been completed, alongside the final completion of the libraries that govern the communications between the various sensor and the microcontroller. Ultimately, most of the components chosen use the relatively simple I2C protocol, apart from the GPS sensor which will make use of TTL Serial, another well documented protocol. Unfortunately, many of the components do not have recognised libraries, so we are producing our own libraries. We produced two main libraries, one for the sensors, and one for communications. The Sensor library manages the reading of data from all sensors which will in fact make use of other libraries for each sensor. While the MS5637 and HYT271 libraries are being made in-house, the library for the IMU has already been produced by Sparkfun, so we are making use of this. For the GPS sensor, we are going to use the TinyGPS++[28] (an open source GPS library) to parse the NMEA statements that the breakout produces. This library is very well acknowledged and the team has lots of experience having used it in the past (though with other GPS units). Hence the list of libraries we have used is as follows:

1. Sensor Library
2. RFM98W Library
3. LSM9DS1 Library
4. TinyGPS++

Additionally, a basic plan, illustrated by the flowchart below, for the main loop (since we are using Arduino, a language which has a cyclic program structure), where the data is collected and then transmitted, has been created.

With regards to the main microcontroller program, the following diagram shows the main structure of it, with it calling the various libraries at regular intervals.



[26] Statistics found using Google Analytics
[27] Statistics found through analytics on the Google Play Developers' Console
[28] http://arduiniana.org/libraries/tinygps/

Figure 7.3.1: Design for Can code (made using Microsoft Powerpoint[29])

We found the vast majority of sensors largely unchallenging to read from given the documentation provided. In fact, the only system that posed any challenge is the communications using the RFM98W, given the complex protocols that it makes use of. This has been hugely assisted by William's personal experience using the module, since he has used it on a number of occasions for a number of different projects. However, despite managing to read from the LSM9DS1 relatively easily, we have found a problem with the sensor, notably that the inbuilt sensor calibration, which does not appear to show the correct heading, a very important part of the system, since the autonomous motion relies upon this ability. Hence, we are currently experimenting with other physical manners of calibrating the magnetometer, in order to prevent this from occurring. Additionally, though concerned that the proximity of the motors to the magnetometer in the can (directly above it) could have a large impact on the results, by testing the sensor nearby motors, (especially given that when the motors begin to spin, the arms will open and the motors will no longer be in proximity to the LSM9DS1), we found little impact to the results. Additionally, we hope that the software would be able to compensate for any inaccuracies in the magnetometer by making use of the GPS position, and thus extrapolating the heading of the can.

7.4 - Base Station

During the last progress report, we defined our exact requirements for the base station, and over the last couple of months, the Software team has been hard at work, to the point where as of now the software is complete and working through testing. The specification we defined was thus:

1. The software should parse and store sensor data from the Can (in a '.CSV' – Comma Separated Values file. The data transmitted will be exclusively in bytes, as required by the RFM98W, thus there would likely be a requirement of adapting the exact input in order to arrive at the correct data received from the can.
2. The software should display the temperature, pressure, relative humidity and IMU data in real time from the can.
3. The software should be able to calculate the altitude, also making use of a barometer on the ground station, hence calculating relative height, and displaying this to ensure the person flying the quadcopter has sufficient data to fly comfortably.
4. The software should be able to display information about the exact information about the quality of the link to the Can, providing the signal strength, and information about the received packages.
5. The live camera feed should be displayed on a separate monitor, ensuring that the systems are separated, even in the base station. This will make use of a separate battery supply, with FPV RX and antenna, as has been tested.
6. The software should display the current Can location, preferably on a map (making use of Google Maps[30] or OpenStreetMaps[31] to help display the location of the can.
7. The software should use the incoming data to calculate altitude (using barometric pressure) and the agricultural viability (using the predefined algorithm)
8. The software should be able to autonomously monitor the position of the quadcopter and move it to a user-defined location, using (6), taking input from the user.

[29] https://products.office.com/en-gb/powerpoint
[30] http://maps.google.com
[31] http://www.openstreetmap.org/

Since then, the entire ground software has been written, meeting the entire specification, with an attractive Graphical User Interface, written in XAML, and the backend written in C#. In fact, as well as the FPV image being shown on a separate monitor, it will also show in a corner of the Ground Software, by using an EasyCap AV Input device to allow the computer to take it as an input. Additionally, instead of using Google Maps or OpenStreetMaps, the system is making use of the Bing Maps API for C# which produces a live adjusting system, which in fact allows the user to zoom in and pan around the map. The software also has a section for allowing the user to output information to the can, such as setting the QFE (the ground pressure) either using an amount specified by the user, or using the input from the pressure sensor on the ground station PCB. Meanwhile, the system also can communicate with the Transmitter system, controlling the throttle, pitch, yaw and roll sent to the Can, hence allowing it to move in a specific direction, through monitoring the GPS inputs from the Can. [All code is available in the Appendices]. Finally, I added in the ability to produce live graphs, which proved more of a challenge than expected, especially compared to other programming languages such as Python, where the MatPlotLib library was used. For this, the best possible function that I could find was the rather underdeveloped 3rd Party Charts function, produced by beto-rodriguez (a user on GitHub) however, this was rewritten and improved for this. Most importantly, the ability to change charts was added to allow a user to make a live chart. Additionally, an innovative method was added, in which the computer attempts to create more smooth curve of best fit (instead of just linking points with a line) by looking at the last few points (at least three) and attempting to guess at a quadratic function which would connect as many of them as possible.

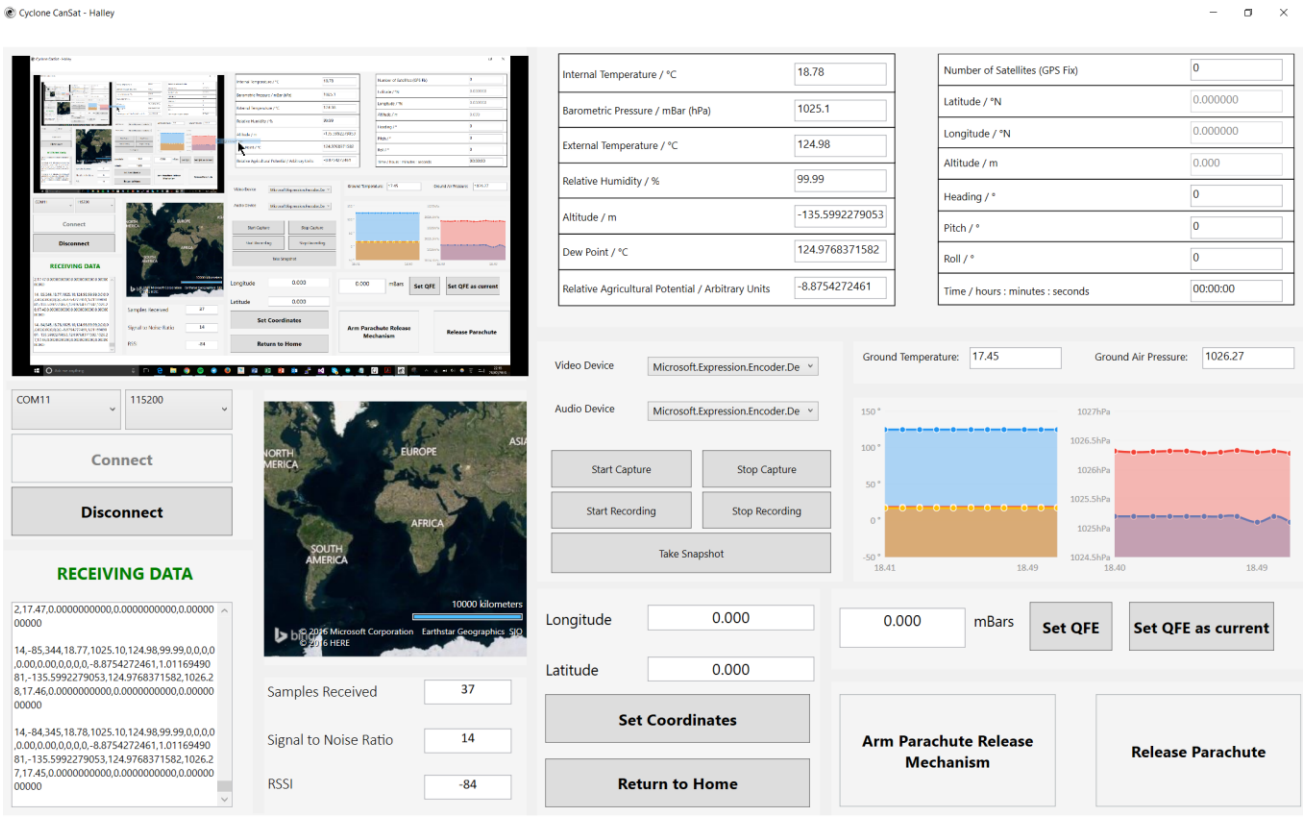


Figure 7.4.1 – Base Station Software

8 - Integration of Electronics and Software within the Mechanics

This aspect of the Can design and manufacture, as expected, had to be very closely coordinated between the electronics and mechanics teams. These designs of these subsystems are dependent upon each other as the limitations of the mechanics define the limitations of the electronic complexity, but the specific electronics design defines the actual mechanics design. Hence, constant and efficient communication was necessary – something we believe we achieved.

Initially, the outline of the maximum possible space within the Can (the simplest shell and arm setup) was calculated by the mechanical design team. This was passed onto the electronics team who worked out what our potential complexity could be, and we tried to achieve this.

With these objectives set, the electronics team set about designing their system and when they had created an initial overview of the components that they required, they supplied this to the mechanical design team who created, from this, a plan of the layer arrangement (Figure 8) – a way of fitting in all the components within the Can in a way that facilitated efficient packing and allowed all the components to be mounted securely and organised coherently.

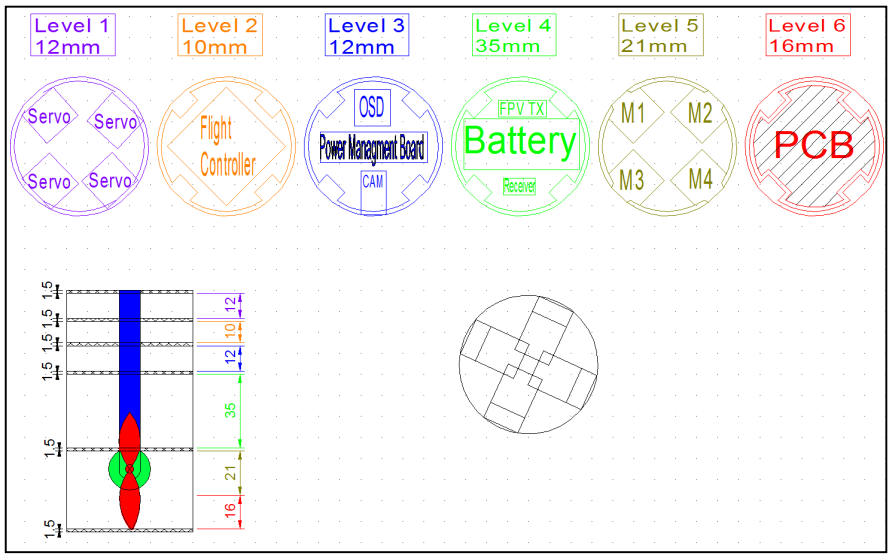


Figure 8.0.1 – Structure of Integration

With this plan having been finalised, and with the confirmation from both teams that this was viable, both teams set about creating their respect subsystem. To ensure a functional CanSat at the end of the project, regular discussions have existed and will to continue to take place between the teams. This ensures that any changes that take place either have no effect on the others system or that any change that needs to be made can be.

In addition to the space allocated to all components, as in the plan above, some components also required specialised mounting systems – for example the motors and ESCs. This was organised by the electronics team and mechanics team discussing the requirements of each component and whether or not they needed this special treatment. If they did the exact requirements were also discussed and decided on. Then the mechanics team created mounting platforms and structures which would be able to cope with the stresses that the component would place on them; and the viability of this was confirmed by the electronics team.

8.1 - Testing

Constant testing has been taking place and will continue to do so. This consists of activities such as placing the components within the Can to ensure that the space allocated to them is sufficient, wiring up sets of components which form certain subsystems to check whether the space left between each component is sufficient to hold the wiring that we require, and also – when each part of the electronics has been completed it is run inside the body of the Can to ensure that both it can do so and also to ensure that the body of the Can is able to securely hold the components and also to itself be able to survive the stresses that each subsystem creates.

Looking into the future, we plan to run many further tests on each subsystem, especially within the Can's body – something we may have been complacent about doing to the extent we should have. Furthermore, we plan to do extensive testing of everything together in the Can as soon as this is possible, making sure that we have time to correct any issues that may arise.

8.2 – Further Testing and Improvement of design

Following on from the previous extensive intra-system testing, we have now carried out (and will continue to carry out until launch) extensive inter-system testing. For this we have aimed to make the tests as realistic as possible to best inform our design for the actual launch. Detailed below is one example which shows the benefits of testing with as complete a set up as possible. In addition, the consequent and subsequent design changes that followed the test are also detailed.

In this test we investigated the functionality of the servo system designed to open the arms of the Can during its descent. For this we fully fitted the upper sections of the Can with their respective components, and also fitted out each arm with its own components – ESCs and motors. Then we sent the command to the servos to open the arms, however nothing happened. (Unlike in the tests when we were just testing with the bare frame of the Can and the servo system raised the arms perfectly.) In order to see if there was a clear area where the arms were sticking, we manually raised each arm to approximately a 45^{O} angle. While from here none of the arms opened, none of them slipped back into the folded up position they had naturally fallen to before – without the servo system in place. From this we concluded that the servos controlling the arms, while not powerful enough to raise the arms, were powerful enough to keep the arms in a given position. We found that the reason for this was that the wires going through the top plate from the servo spool to the arms found a lot of resistance to being tensioned – a process necessary to raise the arms. (The cause of the resistance was due to the large number of other components in this area.) Furthermore, the capacity of the system to lift the arms was already being stretched due the moment the motors were exerting on the system – as they are heavy and situated far from the pivot (the hinge). In order to try and get around this issue we tried to increase the torque the servos were exerting on the wire. We first considered doing this by redesigning the spools, however this idea was discarded as the spools were designed just to fit in the available space and increasing the size would eat into the space for the control board. In addition to this we considered using more powerful servos; unfortunately, the only more powerful servos we could find within our price range were much larger in size and so would not fit inside the confines of the Can. Thus we decided that we shouldn't use the servos as they used space and weight and only held the arms in place, which didn't justify their presence. In place of this servo system to open the arms, we considered putting springs to push the arms out immediately after release from the balloon. This would avoid the propellers clashing into each other and the pillars when first spinning up. However, during testing of this idea, we found that the arms sit naturally in a position where the propellers don't clash – due to the natural spring like effect of the wires packed into the upper section of the Can, which means that neither springs nor any other arm opening system was required. This is hugely beneficial as it leaves an increased amount of space for other components within this upper section of the Can.

A subsequent change we made to the Can following the decision to omit the servos was that we removed one of the layers from the upper section of the Can. We were able to do this as previously we were going to mount components on it, however, now the top plate was free and no longer had servos mounted to it, we could mount components to the underside of it instead. This further freed up space and allowed the wiring to be simpler and neater in this area. The new design without this layer can be seen below.

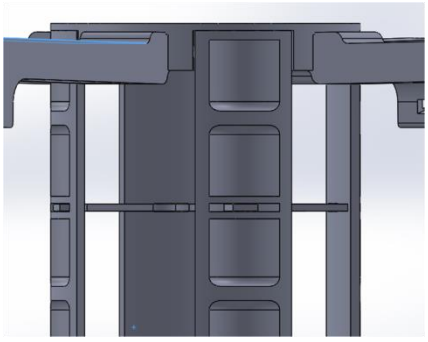


Figure 8.2.1 – Zoom in on top of Can

8.3 – Flight Tests

We were now at a stage where we were able to fly the Can for the first time. As expected we did not fly the enter Can in its complete state at the first test flight, but instead phased in the different sub-systems that made up the Can over many test flights.

For the first flight – and for the use of subsequent flights as well – we created a launch platform to allow us to fly the quadcopter off the ground. This consisted of a thick heavy block of wood with a small raised platform in the centre. This meant we could place the Can on the platform, and retreat to a safe distance before firing up the propellers. In addition, the propellers, when rotating with the arms in the closed position, have points in their cycle when the ends of they are well below the level of the base of the Can; so the raised platform of the launch platform allows the propellers to spin up without clashing with the ground.

The first test flight was short, although very pleasing for a first flight: the Can took off fairly vertically, but after reaching a height of around 10m, it began to veer off to one side rapidly and out of control, and in order to prevent the Can flying into a building the power had to cut and the Can fell from around 15m. The reason for this problem was that the control board – which contains the sensors to tell the Can what angle it is and so allows for the correction of tilting of the Can when in flight – had become dislodged due to the vibrations of the motors. Hence, as the control board had become unstuck from where it had been stuck down to a plate, and slipped to one side of this plate, it now sat at an angle relative to where it had started off. Thus the Can thought it was permanently tilted one way, and so was constantly trying to correct itself – the reason for the veering uncontrollably to one side. The simple solution to this was to fix the control board more securely to the plate on which it had previously been stuck. To do this we drilled mounting holes in the control board case and in the plate on which it sat and bolted the two together.

With the control board now securely mounted we tried another test flight. The Can took off, hovered well, and could even swoop and dive. To see a video of this second test see here [*http://teamcycl.one/flighttest2.mp4*].

Figure 8.3.1 – Snapshot from video of Second Flight Test

However, after a strong gust of wind caught the Can, it crashed. The main damage to the Can was that the hinges on the top plate had broken – which called for a redesign of the hinge system to a much more robust setup. The new design can be seen below.

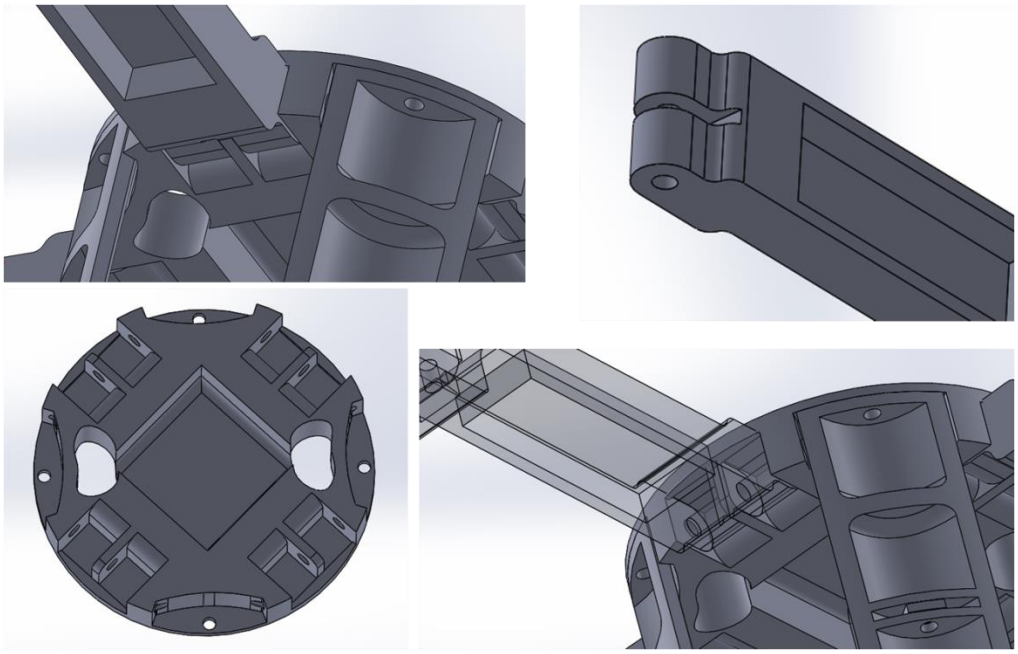


Figure 8.3.2

Once again we flew the Can, this time with the much stronger hinge mechanism. The test went amazingly, and only after six or seven brilliant flights (each with their own harsh landings), a particularly bad crash from a height of 20m caused complete destruction to the Can: all but one of the plates shattered (not including the top plate in this category of 'plates'); the hinge was sound although the top plate had hair line cracks running across its upper surface; one of the posts was slightly cracked, and one arm had snapped. The main reason for this particularly complete destruction was that the Can, which had been flying around fine, was accidently flown over an area of tarmac – whereas previously it had been flying over grass. In the pilot's effort to get the Can away from the tarmac – as we knew it would not be good for the Can to land on tarmac, the pilot cut the power to the Can as soon as it returned near some grass. While normally this would have been fine and the Can would have crashed onto grass not particularly coming to harm, the pilot tried to pull out of the fall, when they realised that they might be able to fly the Can back towards where we had launched it from. This meant, as the Can had turned upside down during its free fall period, that now the Can was pulling itself towards the ground on which it then smashed with a huge force – destroying it. This however is not of great concern, as there will be a much larger area on the actual launch day, and our pilot will have had much greater practice controlling the Can. Furthermore, we do not need to worry about the issue of the Can turning over mid fall as if the Can turns over during its free fall period when dropped from the helikite, it will have time to turn back the right way up and so fly well – which did not happen in the test as there was not time for the propellers to start spinning and so right the Can before it hit the ground.

Nevertheless, we still tried to strengthen every part of the Can: we increased the thickness of every layer from 1.5mm to 2mm (including the top layer); added additional support ribs to the posts; and thickened the arms. See below for these designs.

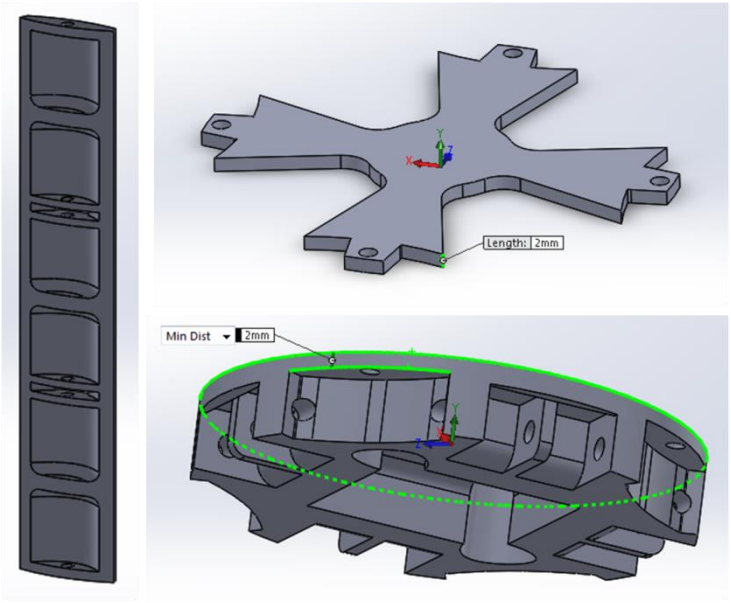


Figure 8.3.3

With these parts reprinted we tried another test flight. For this test flight we added the FPV (camera) system in and also the sensor board system – although were still flying with fixed props. It was when fitting all this inside the Can that we encountered some serious space issues in the top section. We managed to solve this by, instead of having the control board attached to the top face of the second layer, turning it over and attaching it to the underside of the top piece. The result was that we could hugely decrease the amount of wiring and so create enough space for all the components. The design of this new piece can be seen below.

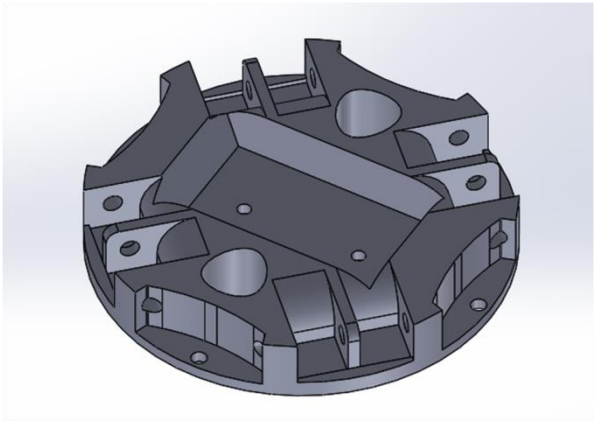


Figure 8.3.3

In addition to this problem we also encountered another problem: that the arms were not folding into space properly. This was partly due to the LiPo not being the exact size stated on its specifications and also that the sensor system at the bottom would not stay in its allocated space – as there was nothing to ensure that it did. To get around this we created extrusions in the arms that formed around the battery and also created an extra plate which sat between the motors (when the arms were in the folded up position) and the sensor system. When reprinted and fixed into place these alterations allowed the arms to slot nicely into place. The design for these altered parts can be seen below.

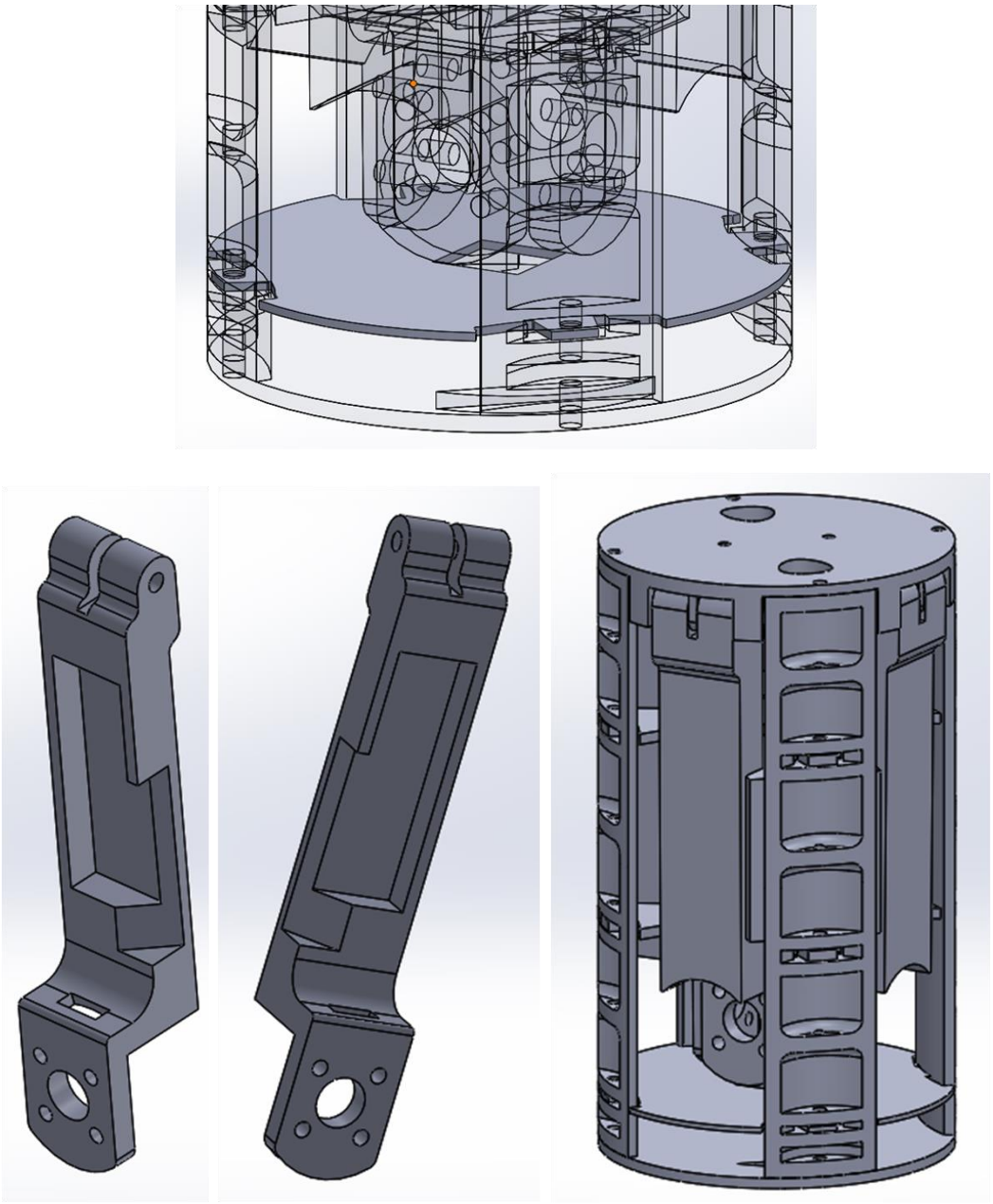


Figure 8.3.4

While these changes were being made, work was being done on the creation of folding propellers (see folding propeller section).

When these propellers had been made we mounted them to the Can and after some very preliminary testing, they seem to be working – we hope to test them in a full flight soon.

8.4 – Folding Propellers

It was necessary to create our own propellers as after much research we concluded that there were none that we could simply purchase. This was because the only folding props which folded parallel to the plane of rotation of the propeller (rather than perpendicular to it – which were of no use to us) came in a minimum of 15 inch diameter. As we needed 5 inch propellers, we clearly could not get away with just cutting down existing ones.

To make the folding props we bought in standard 5 inch props and cut them in half, in the direction shown below:

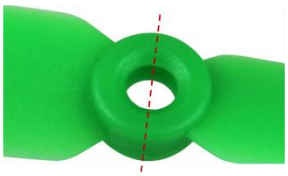


Figure 8.4.1 – Annotated image of conventional 5x3 propellers

Then we drilled holes in the propeller that line up with those mounting holes already present on top of the motors:

Figure 8.4.2 – Annotated image of Turnigy Outrunner V2 Motors

Next we filed the edges of the propeller's hub to a shape that allowed one half to rotate as described shortly.

Finally, one half of the propeller was mounted with both holes through its hub, and the other half only mounted with one of the holes through its hub.

This was all done to allow the propeller to work as follows:

- one half does not move;
- the other half can rotate in the opposite direction to the direction it is pushed on by the airflow caused by the spinning of the motors.

Thus the prop can be folded into a closed position, yet when the motor spins; it is forced open and held open by the airflow caused by the spinning of the motor.

While this does not sound very complex, the details above are a much simplified version of the process. Furthermore, the precision involved when manufacturing and perfecting these tiny mechanisms such that they work reliably and precisely under the extreme conditions they encounter when spinning at incredibly high speeds truly gives credit the craftsmen in the team who have manufactured these pieces of high precision engineering.

One of the folding propellers can be seen below in the folded and unfolded positions.

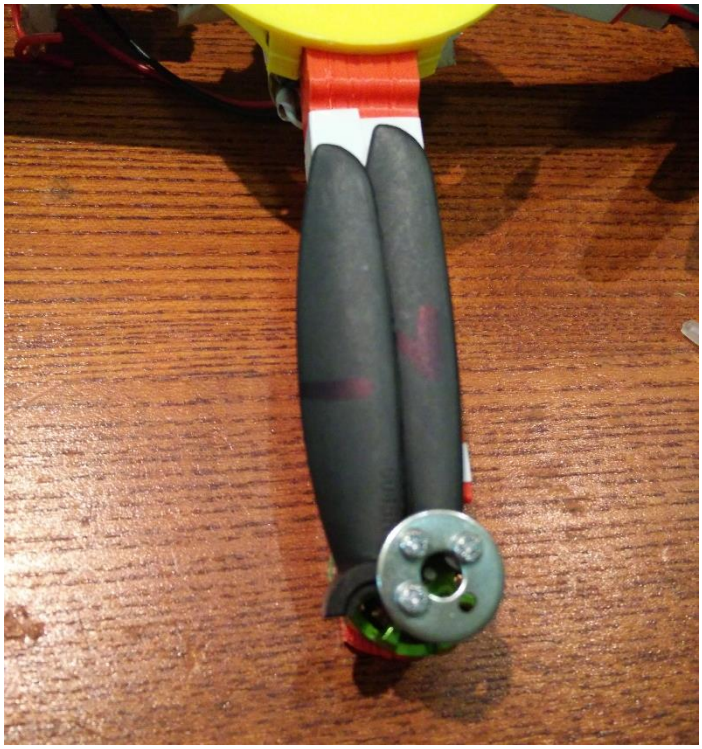
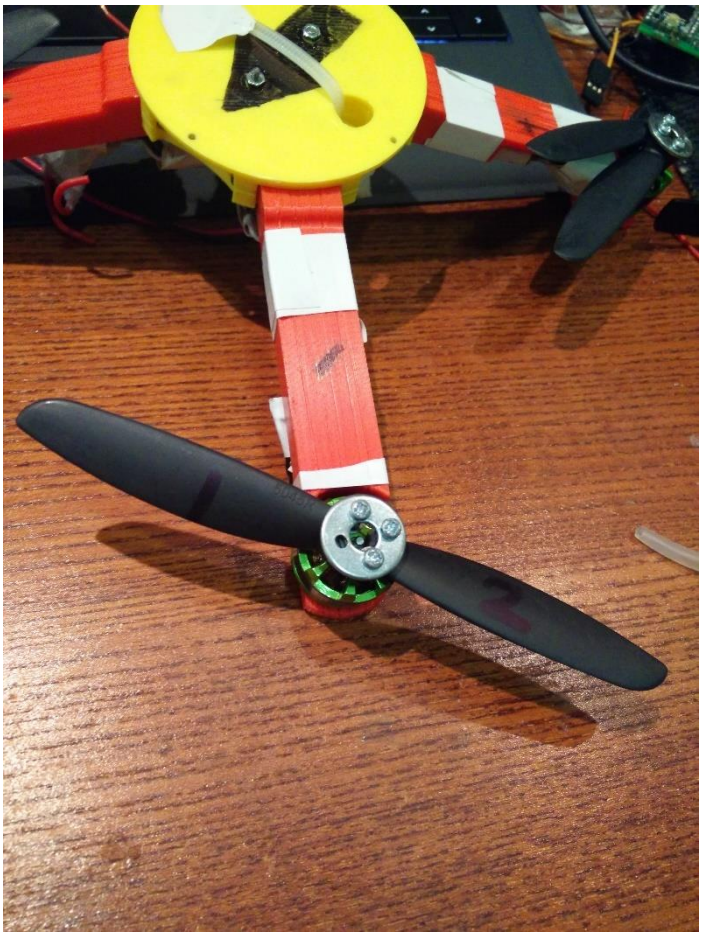
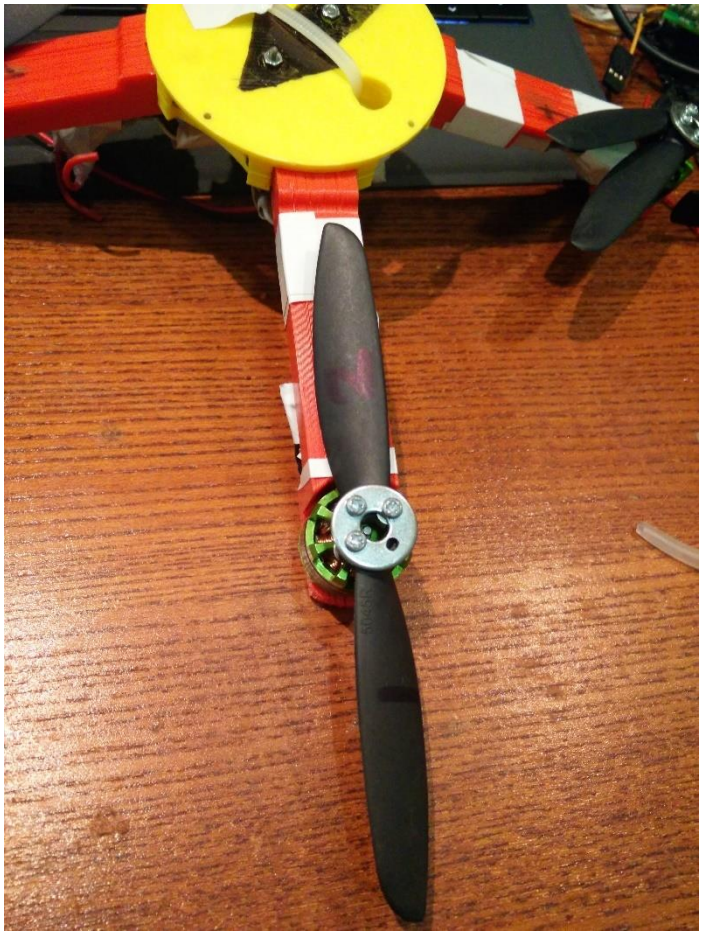


Figure 8.4.3 – Folding propeller shown in closed position





Figures 8.4.4 and 8.4.5 – Folding propeller shown in open position

9 – Launch Procedures

9.1 – Drone Laws and Safety

Safety is of paramount importance when flying any sort of quadcopter, due to the number of possible hazards. Improper operation can cause serious injury and property damage. Due to this, there are certain UK laws surrounding the operation of quadcopters.

9.1.1 - Laws

The use of various types of unmanned aircraft, popularly known as drones, has increased rapidly in recent years - both for private leisure use, and for commercial 'aerial work'. Unmanned aircraft are generally fitted with cameras, unlike traditional remote controlled model aircraft which have been used by enthusiasts for many years. As such drones are likely to be operated in a way that may pose a greater risk to the general public and other aircraft. Unlike manned or model aircraft there are no established operating guidelines so operators may not be aware of the potential dangers or indeed the responsibility they have towards avoiding collisions. Anyone flying a drone either recreationally or commercially has to take responsibility for doing so safely.

For aircraft of 20 kg or less, these are referred to as a 'small unmanned aircraft', for which the requirements are a little less stringent and are covered within Articles 166 and 167.

Article 166

1. A person shall not cause or permit any article or animal (whether or not attached to a parachute) to be dropped from a small aircraft so as to endanger persons or property.
2. The person in charge of a small unmanned aircraft may only fly the aircraft if reasonably satisfied that the flight can safely be made.
3. The person in charge of a small unmanned aircraft must maintain direct, unaided visual contact with the aircraft sufficient to monitor its flight path in relation to other aircraft, persons, vehicles, vessels and structures for the purpose of avoiding collisions.
4. The person in charge of a small unmanned aircraft which has a mass of more than 7 kg excluding its fuel but including any articles installed in or attached to the aircraft at the commencement of its flight, must not fly such an aircraft:
    a) in Class A, C, D or E airspace unless the permission of the appropriate air traffic control unit has been obtained;
    b) within an aerodrome traffic zone during the notified hours of watch of the air traffic unit (if any) at that aerodrome unless the permission of any such air traffic control unit has been obtained; or
    c) at a height of more than 400 feet above the surface unless it is flying in airspace described in sub-paragraph (a) or (b) above and in accordance with the requirements for that airspace.
5. The person in charge of a small unmanned aircraft must not fly such an aircraft for the purposes of aerial work except in accordance with a permission granted by the CAA.

Article 167

1. The person in charge of a small unmanned surveillance aircraft must not fly the aircraft in any of the circumstances described in paragraph (2) except in accordance with a permission issued by the CAA.
2. The circumstances referred to in paragraph (1) are:
    a) over or within 150 metres of any congested area;

b) over or within 150 metres of an organised open-air assembly of more than 1,000 persons;

c) within 50 metres of any vessel, vehicle or structure which is not under the control of the person in charge of the aircraft; or

d) subject to paragraphs (3) and (4), within 50 metres of any person.

3. Subject to paragraph (4), during take-off or landing, a small unmanned surveillance aircraft must not be flown within 30 metres of any person.
4. Paragraphs (2)(d) and (3) do not apply to the person in charge of the small unmanned surveillance aircraft or a person under the control of the person in charge of the aircraft.
5. In this article 'a small unmanned surveillance aircraft' means a small unmanned aircraft which is equipped to undertake any form of surveillance or data acquisition.

Summary

In summary of the two articles:

- The operation must not endanger anyone or anything.
- The aircraft must be kept within the visual line of sight (normally taken to be within 500 m horizontally and 400 ft. vertically) of its remote pilot (i.e. the 'person in charge' of it). Operations beyond these distances must be approved by the CAA (the basic premise being for the operator to prove that he/she can do this safely).
- Small unmanned aircraft (irrespective of their mass) that are being used for surveillance purposes are subject to tighter restrictions with regard to the minimum distances that you can fly near people or properties that are not under your control. If you wish to fly within these minima, permission is required from the CAA before operations are commenced.
- CAA permission is also required for all flights that are being conducted for aerial work (i.e. in very simple terms, you are getting paid for doing it).
- The 'remote pilot' has the responsibility for satisfying him/herself that the flight can be conducted safely.

9.1.2 – Learning Points

From this information, we devised some key safety points to adhere to when flying. The following are the recommendations drawn from both UK laws, articles 166 and 167, and from drone flying experience.

Pre flight

1. Check the reliability of the Quadcopter Multi-Rotor.
2. The flight battery and transmitter battery must be fully charged.
3. Always turn on the transmitter first and then the Quadcopter Multi-Rotor.
4. Watch for visible damage such as loose screws, broken, unbalanced or damaged propellers, faulty connectors or solder joints, broken pipes, etc.
5. The propeller must be in a good condition and securely mounted. The rotors must spin smoothly. Please ensure that there are no objects are in the rotational plane of the propellers or within a

distance that poses a risk of obstruction. Rotating propeller ends are dangerous to touch - never touch a rotating propeller with fingers or other body parts.

6. Ensure the appropriate to the audible beeps when auto LiPo-detection is enabled. 4 * beep means 4s LiPo
7. Ensure that the selected channel on the remote control is free and that you're within range of the transmitter.
8. Ensure the sensors are calibrated (when the motors are off move yaw and gas stick into upper left corner until it beeps).
9. Different settings can result in fundamentally different flight characteristics and occupation of the Channel functions. Make sure that you are familiar with the setting and its features.
10. After starting the motors, check that all motors are running and rotate evenly. Please fly carefully at a low altitude until you know that everything is working fine.
11. Perform a pre-flight check (see checklist below).
12. Start manually - switch off altitude control and GPS.

During the flight

1. Do not take risks! Never fly towards viewers and avoid flying over spectators.
2. Keep in mind that viewers could get close to the Quadcopter Multi-Rotor, without being aware of potential dangers.
3. Always turn off the Quadcopter Multi-Rotor first and disconnect the flight battery before turning off the transmitter.
4. Only fly in sight. In the case of manual control, you must be able to see the position and attitude.
5. Never rely 100% on functions such as GPS, compass, or altitude control. You must always be able to take manual control of the Quadcopter Multi-Rotor.
6. Reduce the gas or switch off the motors in case of a crash or failure.
7. Do not fly in blocked air space, such as in the vicinity of airports, etc.
8. Pay attention to under-voltage warning -flying on an empty battery can cause damage to the LiPos and crash.
9. If a defect or malfunction has occurred, it must be corrected before the next start.
10. When using GPS, take note that the position of the Quadcopter Multi-Rotor can change suddenly some meters.

After the flight

1. Disconnect the battery and check all over for damage. Make sure that there is no damage to the propellers.
2. See the Handling and safety precautions of LiPos.
3. After a crash the sensors and electronics might be damaged. Before the next start everything must be checked.

Additionally, we have created a checklist below, to ensure that all these recommendations are met, which will be checked off both during all testing and during the launch.

Pre-flight Checklist		
Item to Check	Reason to check	Tick
Observation		
Check surrounding area 2km radius for: Physical intrusions such as hills, lakes, trees Man-made issues, such as people, buildings and vehicles Visibility of over 2km, unobstructed by terrain	"Drones cannot within 50 metres of people, vehicles, buildings or structures" "over congested areas or large gatherings such as concerts and sports events"	
Assign and confirm everyone's role during the operation of the drone, including designating both the pilot and the spotter	Legally the pilot needs to be designated, along with the spotter	
Examine a TCAS map of the area, assessing the airspace for where and when not you can fly	In order to not intrude into restricted airspace	
Set-up		
Remove drone from protective packaging and inspect for damage	A damaged can could result in a crash	
Check battery is fully charged and that cells are balanced	To ensure that the flight does not end prematurely	
Turn on Transmitter, place all buttons in the down position, and make sure throttle is idle	To ensure that the motors do not prematurely spin up	
Place on flat level surface and plug in battery, waiting for the Control board to initialise	The board needs to be flat to set the gyros correctly	
Confirm that the transmitter is bound to the receiver	Interference could lead to loss of control of the can	
Plug in FPV receiver and monitor, and then plug in FPV transmitter	In order to maintain video link	
Individually test all the systems: Test the arms up and down multiple times Spin the motors up to 5% thrust, in order to verify that they are working Whilst the motors are on 5% thrust, test pitch and yaw controls Confirm that the video signal is solid and locked	Component failure could have catastrophic effects	
Flight		
Order everyone, except the pilot and the spotter, to back away from the drone	To prevent collateral damage	
Make sure arms are extended and locked in the outer most position	So they don't collapse mid flight	
Increase throttle aggressively to 50% until the can starts to rise, and then reduce it back to 30%	Known as 'punching it' up, this ensures that the drone does not tip over or catch grass	

Put the drone into a level flight, and then trim it, using the paddle switches beside each stick on the controller	To make it significantly easier
Make sure that the observer is shouting out any obstacles that you could be approaching, and the orientation of the drone	So the pilot can focus on flying
Keep eye on electricity levels in all cells of batteries, once the drone is low on power, land it, leaving at least 3 volts in every cell (9v total voltage across 3 cell)	So the drone does not fall out of the sky/catch fire
Land the drone by hovering around 5-10cm above the ground, and cut the power.	As variable power landings will be difficult on limited power control

9.2 – Data Analysis

Given the possibility of launch delays, there is a likelihood that the team will have very little time to analyse any data. In order to reduce the time taken to analyse all the data that we input, we considered two main manners. One involved the use of Microsoft Excel to plot a series of graphs, and then analysing them, finding the correlation factors and suchlike individually for each graph. Though very simple to use, this method is time inefficient and in fact could produce imperfect results, with a lack of customisability in Excel. The other method is making use of MathWorks MATLAB, a high level language, used to model data and visualise it. Though this is in many ways much more of a challenge, since it is a new programming language for much of the team, it is much more efficient, and produces better results. Additionally, it would require far less effort from an already tired team, at the launch. Hence, we chose to make use of MATLAB, and determined the specification for the code that we would use for Data Processing:

1. Read in a CSV file, individually finding and separating the readings of different sensors.
2. Graph all possible pairs of data (i.e. temperature vs relative humidity, pressure vs temperature, altitude vs time) using a number of different graphs, and calculate relevant data for each graph, including the correlation factors and best fit lines and curves. From here, we are likely to select the most interesting graphs to incorporate into our presentations to the judges.
3. Produce an interesting GUI, making use of MATLAB's built in publishing functions, containing all data and graphs which we could put on our website, to allow all users to easily and more comprehensively read about the results of our project.

Initially, the task appeared relatively simple, through the use of the csvread() function that MATLAB provides and then putting it into a single matrix, after which the data was extracted into individual column vectors to be used for graphing. The plot() function could then be used for all the graphs, while the surf() function for all the three dimensional graphs. Finally, the geoshow() function was used to show the longitude and latitude on a map, with the Google Maps API being used.

However, there were a number of mistakes within this method. Firstly, due to the fact, that some of the parts of the csv files included strings, it would not be able to use the csvread function. Hence we needed to rewrite the importdata() function to allow the system to ignore the data which includes errors. Additionally, there was an understanding that instead of a line graph that the plot() function would create. Hence we would use a scatter() function to produce a scatter graph and then produce a curve of best fit, alongside finding the Spearman's Rank Correlation values, as well as finding the polynomial curve coefficient. In order to find this

coefficient, the system used the equation of the curve of best fit, however, we found that if the values were too consistent then the function would give misleading results.

10 - Ground Support

During launch, we need to ensure that we are able to easily monitor and control all the different systems as required. As with most of the other electronics, we chose to keep them largely separate, in order to ensure that one failure would not ruin the entire system. The only separation from this is the connection from the flight system to sensor system, with the TX controller also connected to the computer, so that it could be connected automatically by the sensor system base station software, which will be running on the computer. This will allow the system to autonomously control the quadcopter, sending signals through the Endurance R/C PCTx, which connects from a computer to the 2.4GHz transmitter. In addition to this, the controller itself will also be ready to be used if necessary, if autonomous travel is not working. In fact, this might be particularly necessary towards the start of launch in order to stabilise the quadcopter, and may also be necessary at the end to land the quadcopter, though we are hopeful that through testing, the landing system could be built into the autonomous software. In order to run all the computer functions, there will be at least one computer running the C# program. Since this program would likely make use of the .NET framework, this would be a Windows computer. In addition to this one computer, there would likely be at least one more computer in case the first does not work, or it runs out of battery (as happened last year during the launch procedure). Entirely separate to the computer, there would also be at least one FPV RX setup, using a 3000mAh 3S LiPo battery connected to a FPV RX (bought as part of a set from HobbyKing), and a small 7" monitor. We are also currently looking into acquiring a set of FPV goggles, in order to further enhance the flying experience of the pilot. This FPV RX will also be connected to a computer which will act as a second monitor. Combining this with free screen capture technology, this would allow us to both easily see the video in a larger screen if necessary and store the video to be viewed later. Finally, the sensor system ground system will be using the same sensor board as the can, since it would have all the required components, as well as always allowing us to compare data in the sky with data from the ground, allowing us to assess changes with altitude. Additionally, the use of a barometer on the ground station would allow me to use a more accurate ground pressure to find the height at which the can is flying. Also, the GPS on board the ground station would enable functions such as 'Return to Home' where the quadcopter would return to where it started, landing at the ground station. In fact, since the can also must have a capability of receiving data (for when to open the arms) the code for the base station electronics can also be very similar, hence reducing the amount of work required by the software team. The only electrical change will be in connecting the RFM98W to a Yagi Directional Antenna rather than a wire antenna. This is simply to ensure that the data transmission will be as successful as possible. In fact, this is not really necessary, with distances of over 3km across London being achieved without the use of the Yagi. However, the Yagi requires little work, since our school already owns one, and in fact would increase our range dramatically. In fact, in a recent High Altitude Balloon flight, the link between base station (with directional Yagi) and RFM98W was maintained at even 110km, a rather ridiculous and unnecessary distance. In fact, for this line of sight was the limiting factor as opposed to the range of the transmission method. Additionally, the choice of using a premade Yagi was made, given that (using experience gained during last year's CanSat missions) the bought one was far more effective than the other Yagi that was built by Team Colossus. In addition, with no other team in the competition from St Paul's this year, the necessity of making one is nil, since the bought one would not be used, if we did not.

11 - Risk Mitigation

Risk	Mitigation
Team Members unable to work due to illness or other reasons	Since the vast majority of the work has been completed this poses much less of a risk. In order to mitigate, there would be the possibility of Skype calls and suchlike to ensure that others had the requisite information. Even if these are not possible, everyone's work is very well documented both in these reports and in internal documents, from which someone else can learn
Delays in construction	Since the vast majority of the can has been constructed, this no longer poses a risk. Additionally, we have multiple spares of every part (both mechanical and electrical) which could be used if the later version is delayed. Additionally, we could entirely revert to previous prototypes which have been proven to work.
Malfunction of tools or equipment	This risk materialised during our prototyping phase. We mitigated the impact by outsourcing the 3D printing of our prototype Can parts to a local company. These parts took longer to be delivered than had been promised and turned out to be of poorer quality than we could have manufactured ourselves. Whilst we were able to use the parts to prototype our design, we have reverted to in-house manufacture (on repaired equipment) for our final parts. However, now, since the vast majority of work has been completed, with the only tools likely to be required are hand tools which are highly unlikely to fail, this risk is highly unlikely.
Receipt of late, poor quality or damaged components	Our primary mitigation against receiving unusable components is to plan ahead and order parts early with float in the schedule, allowing for some re-work or re-ordering if necessary. This requires good team co-operation in order to be able to order long-lead items ahead of schedule. However, now given all the parts have arrived, there are unlikely to be any issues with this.
Problems with software production and testing	The key to successful software development is to have clear and unambiguous requirements. And by working across disciplines, we are able to review and confirm the developer's understanding of these requirements. Our plan includes both software testing and integrated testing with the Can components, as well as an allowance for some software re-work should this be required. Since the vast majority of the software has now been written and tested, there are very few issues related to this.
Malfunction of electronic components	All the components have been reviewed and are working currently, and the risk of failure has been judged to be low. However, we have also acquired spares of every component such that any problems could be resolved.
Overheating of electronics once in the CanSat	Having considered this risk, we are confident that overheating is unlikely to be an issue. The heating effect of our rotor motors will not be an issue as they will be at the ends of the arms and away from the electronics within the Can. Additionally, air flow through the Can will cool the electronics.

Injury during manufacture	Relevant team members are experienced in the use of school DT equipment and will nonetheless review the proposed manufacturing techniques (laser cutting, 3D printing, etc.) with the teacher in charge and agree safety procedures, levels of supervision and conditions for operating the equipment. School safety procedures will apply in the unlikely event of any accident.
Injury during testing and operation e.g. lacerations from spinning rotors	We have consulted with our teacher in charge and have agreed that all initial testing will take place with the Can tethered to a table and behind a Perspex screen. This is due to the inclusion of high speed rotors in our quadcopter design and the risk of parts not being safely attached to the Can. Additionally, during complete testing, we will review the procedures documented in the launch procedures, to reduce the problems with any safety issues.
LiPo explosions in case of malfunctions	The batteries we have chosen to use mitigate this problem by monitoring the usage of the battery and turning the system off, if the battery has too high or too low a voltage, or if there is an over-current.
Destruction of Can during testing or transportation	The first prototype, though of a lower quality than the newest Can, is still partially assembled, and could be used in the case of an emergency. Additionally, we will carry spares of every part in order that we could make another Can if at all necessary.

12 - Gantt Chart

		Oct		Nov					Dec				Jan				Feb					Mar		
	Week Commencing:	19	26	2	9	16	23	30	7	14	21	28	4	11	18	25	1	8	15	22	29	7	14	21
Electronics																								
	Sensor PCB design																							
	Sensor PCB assembly																							
	Quadcopter electronics assembly																							
	Camera system assembly																							
	Full can assembly																							
Software																								
	Write libraries for sensor																							
	Write libraries for RF98W																							
	Write Can code																							
	Data Analysis																							
	Base Station																							
Mechanics																								
	Layer design (using CAD)																							
	Arm design (using CAD)																							
	Layer manufacture (using 3D printing)																							
	Arm manufacture (using 3D printing)																							
	Manufacture of arm opening system																							
	Can assembly																							
Testing																								
	PCB testing																							
	Quadcopter system testing																							
	Camera system testing																							
	Full electronics testing																							
	Test arm opening																							
	Flight test																							
	Launch procedure testing																							
Outreach																								
	Fundraising																							
	Talks to primary school students																							
	Inside school promotion																							
Base Station																								
	Software																							
	Communications system design																							
	Base station electronics design																							
	Base station electronics manufacture																							
Paperwork																								
	Progress Reports																							
	Submission of documentation to sponsors																							

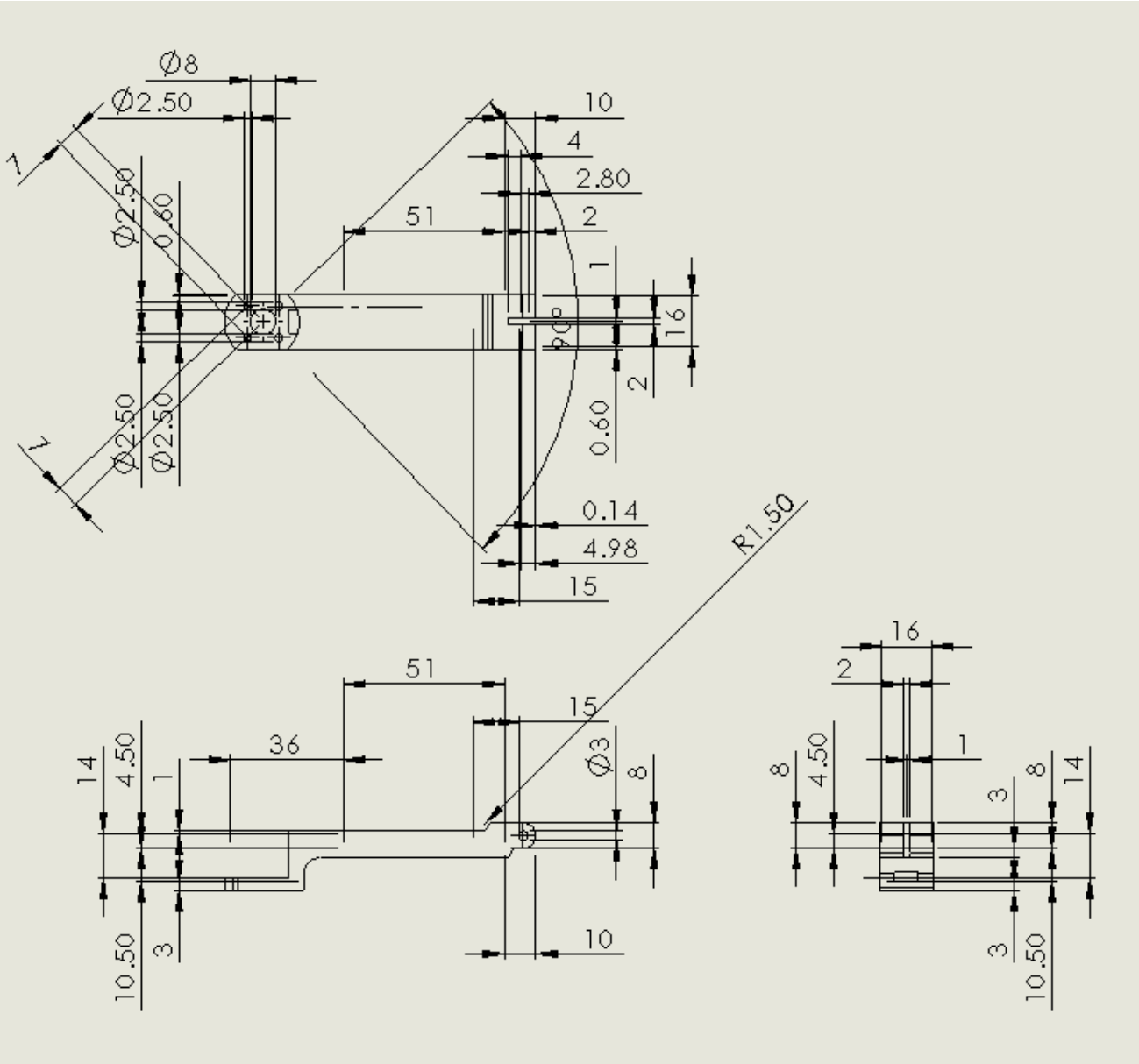
Figure 12: Gantt Chart, outlining future plans (made with Microsoft Excel[32])

[32] https://products.office.com/en-gb/excel

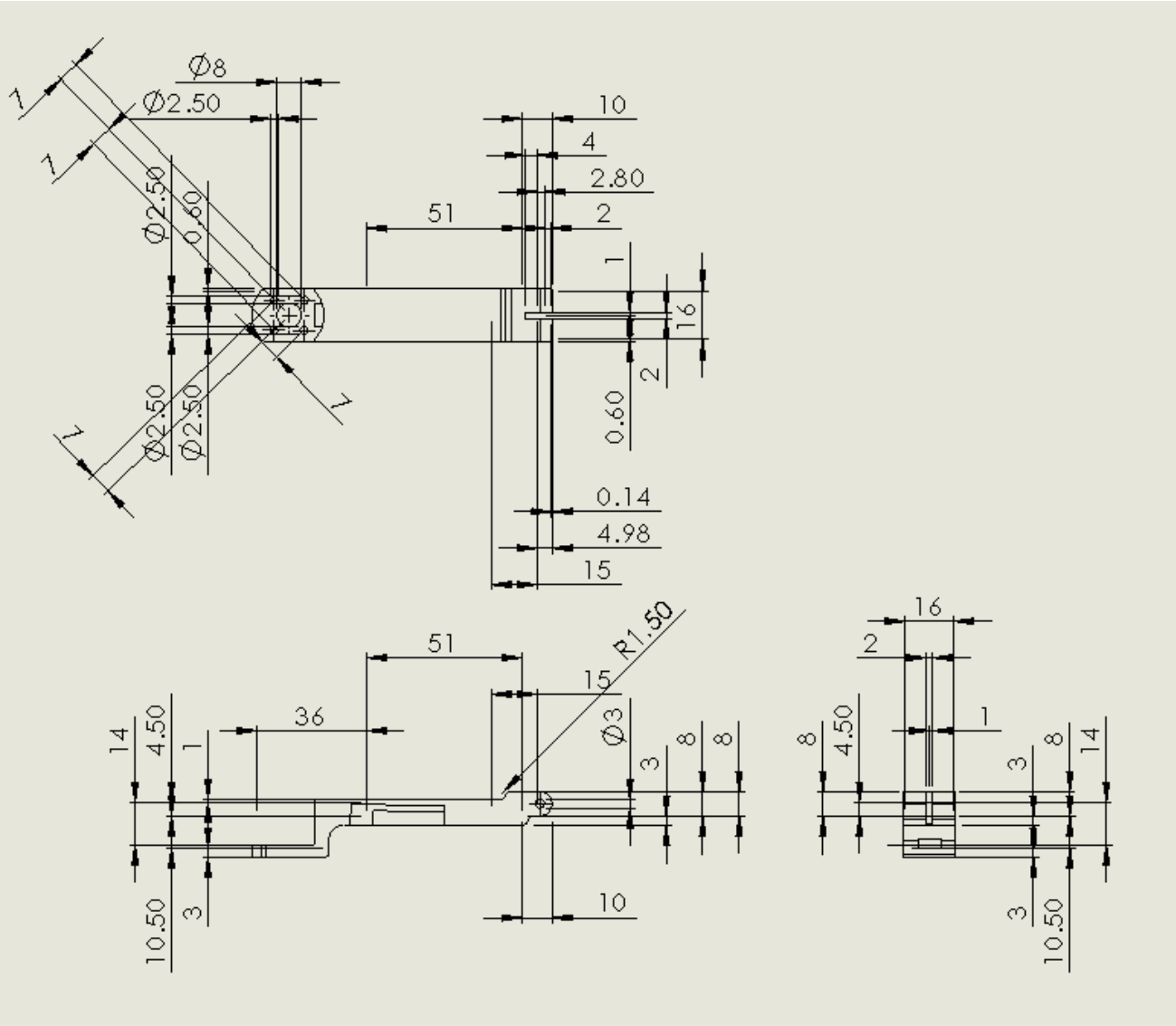
13 – Appendices

13.1 – Mechanical Design

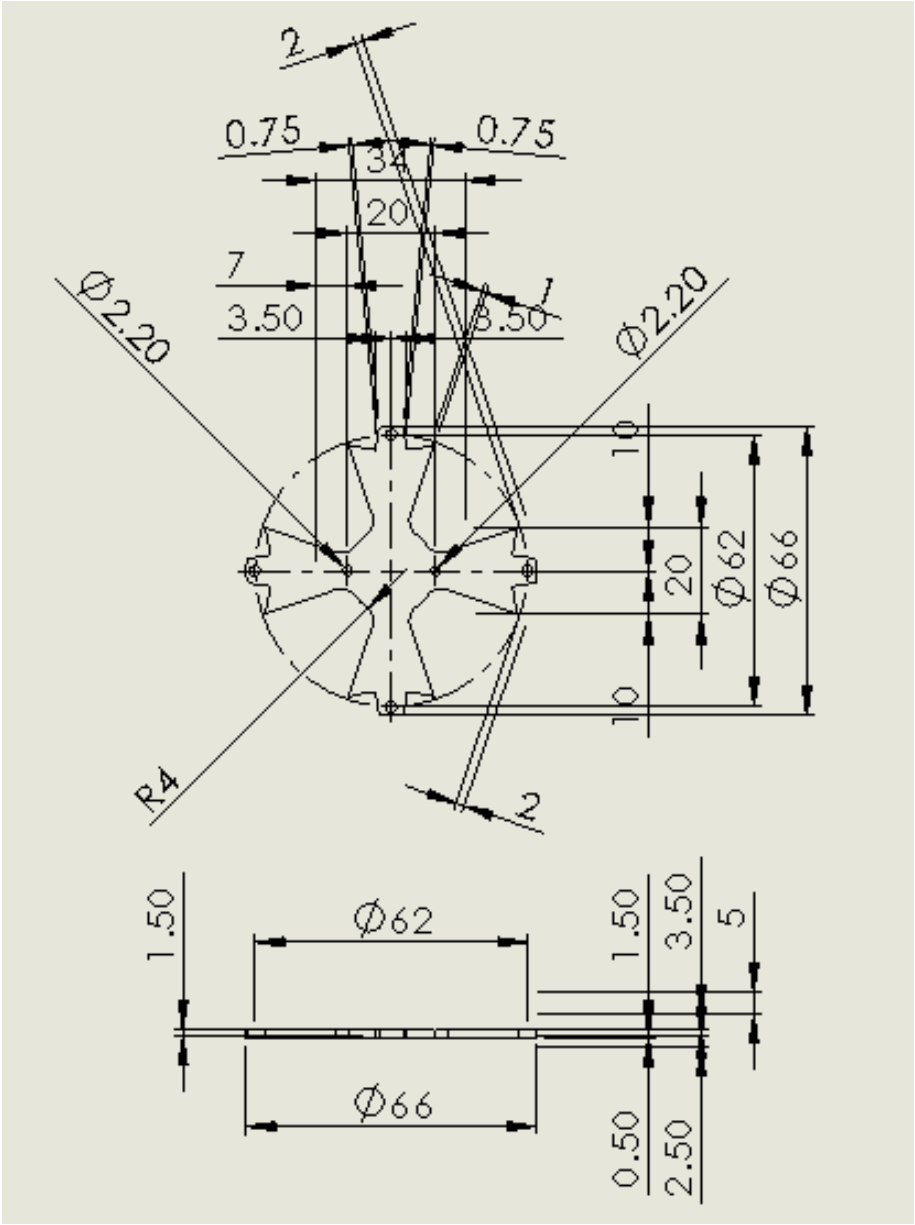
13.1.1 - Arm – Type 1



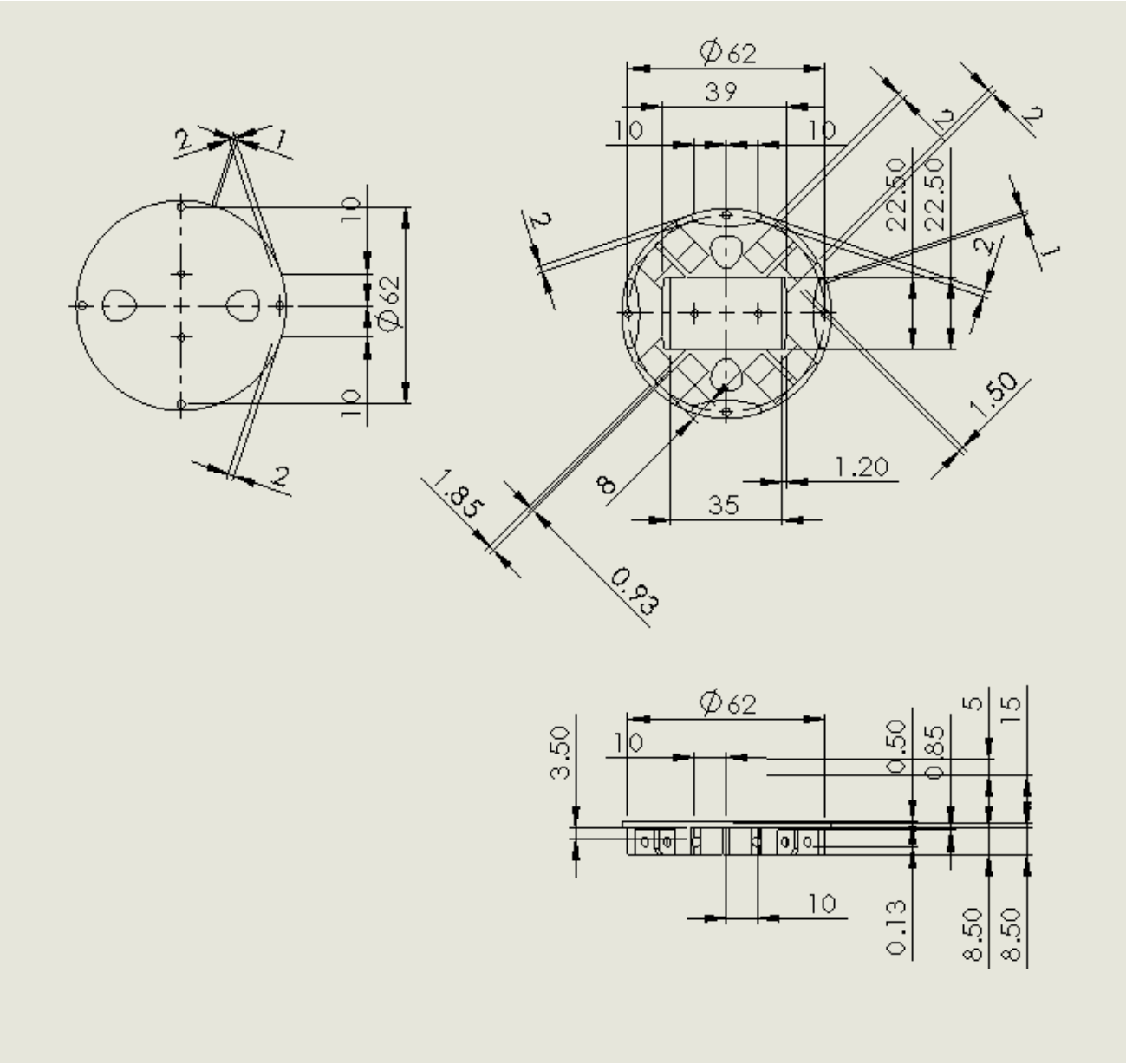
13.1.2 - Arm - Type 2



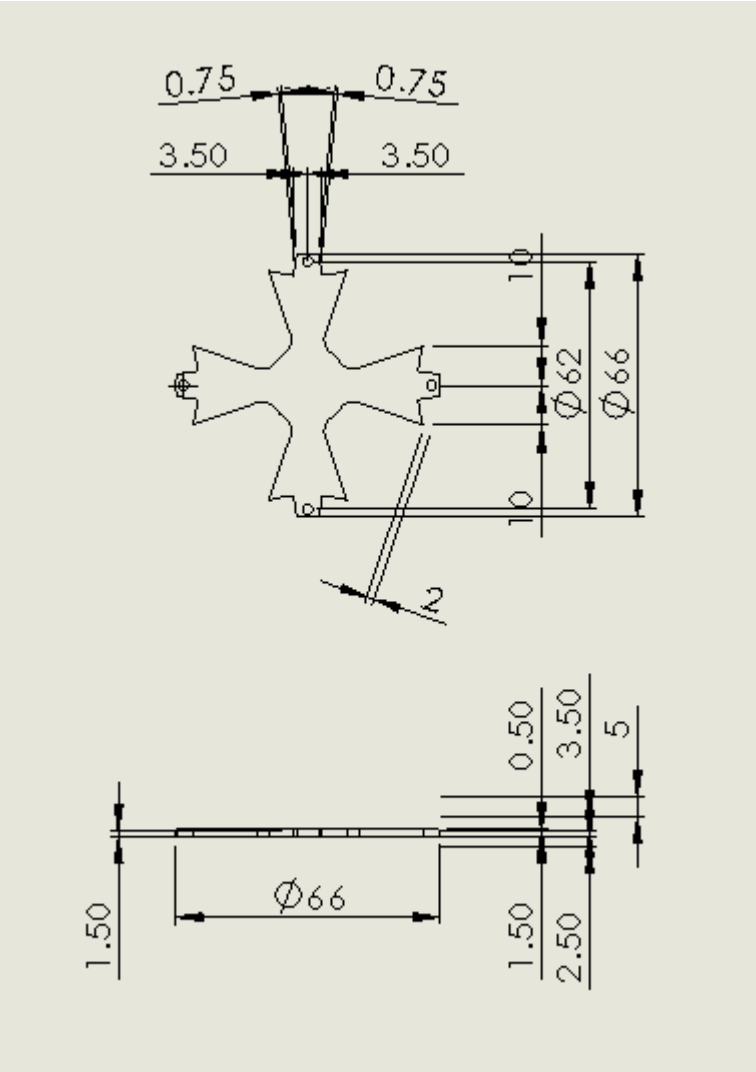
13.1.3 - Control Board Holder



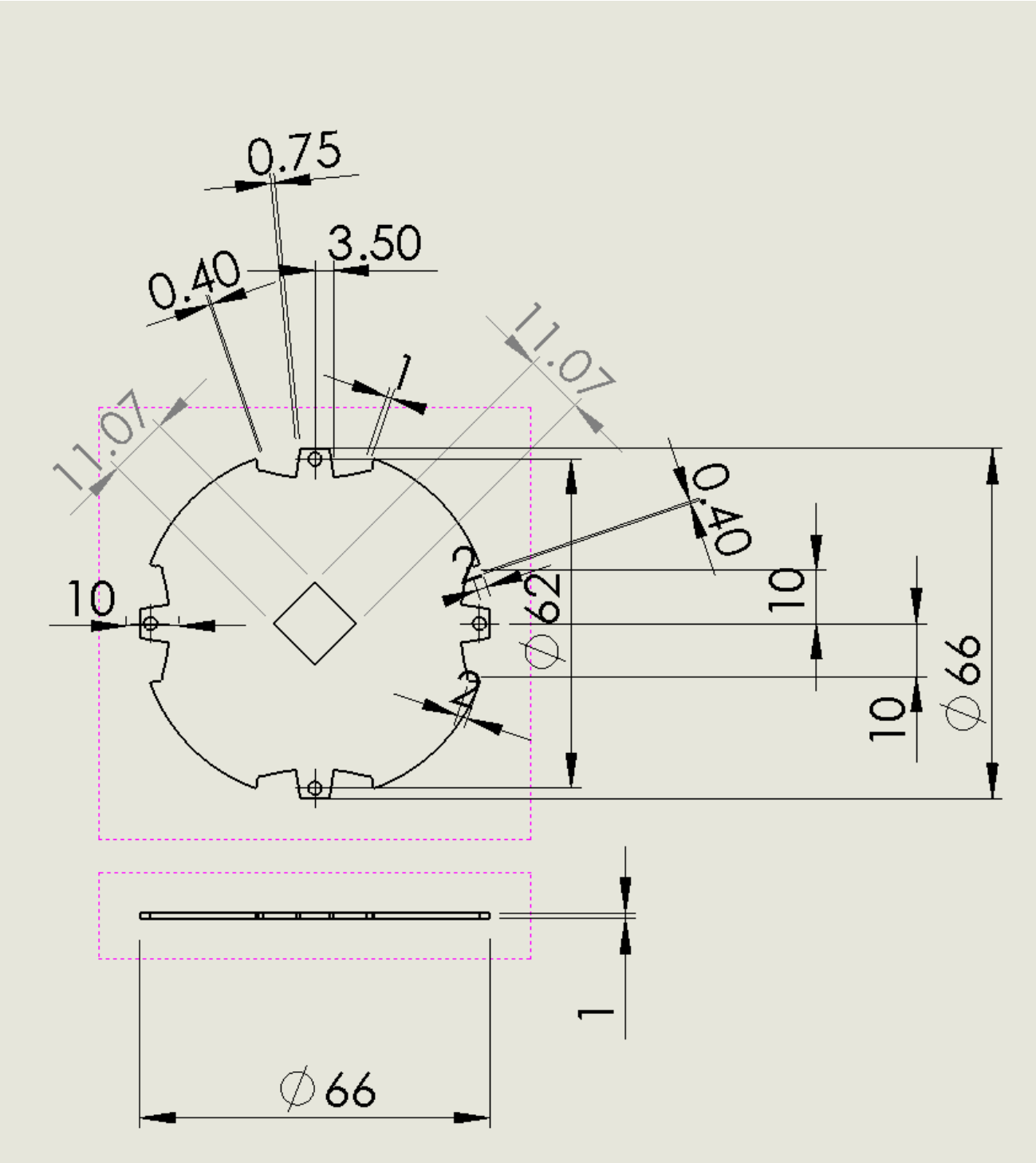
13.1.4 - Top Layer 1



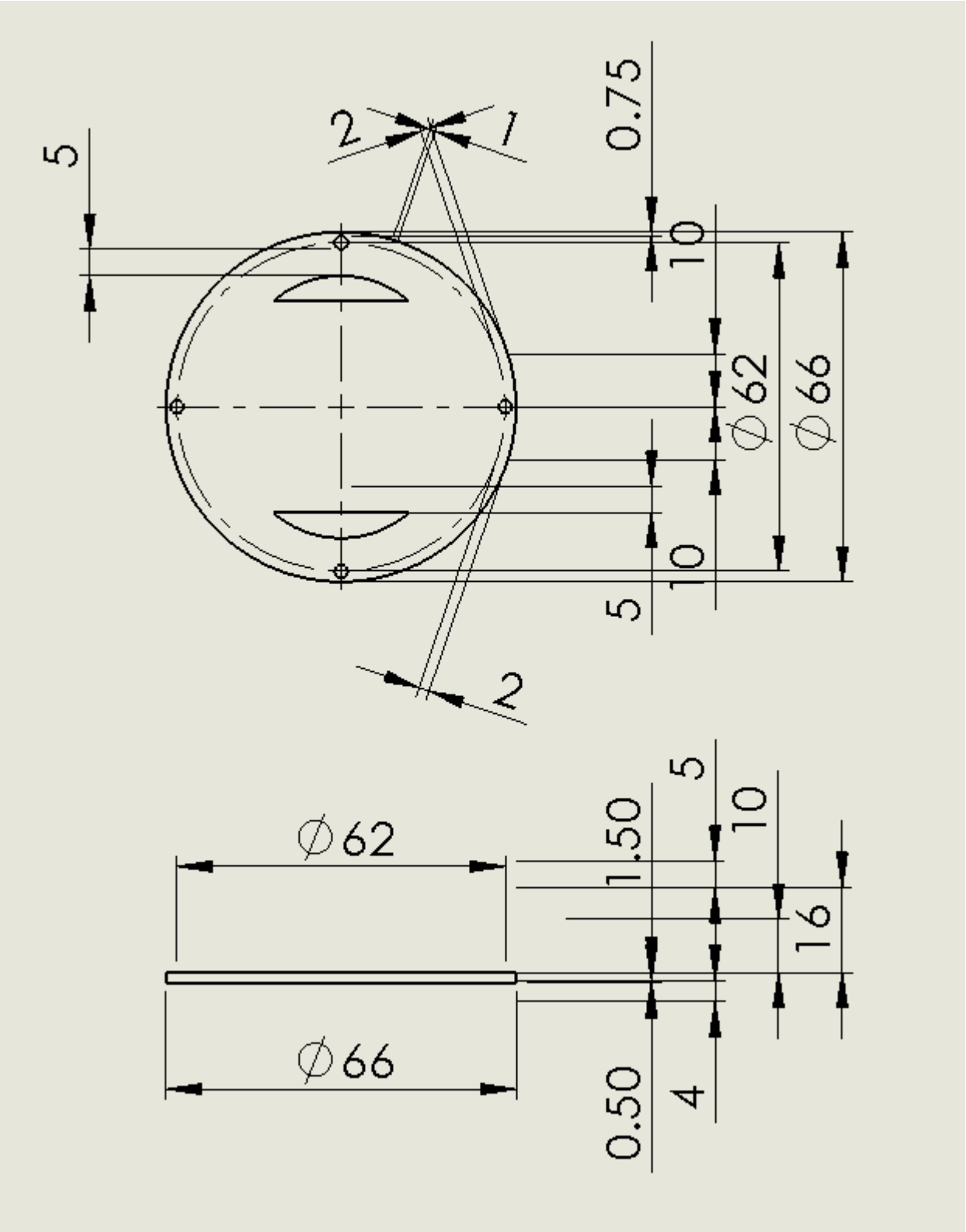
13.1.5 - Layer 2



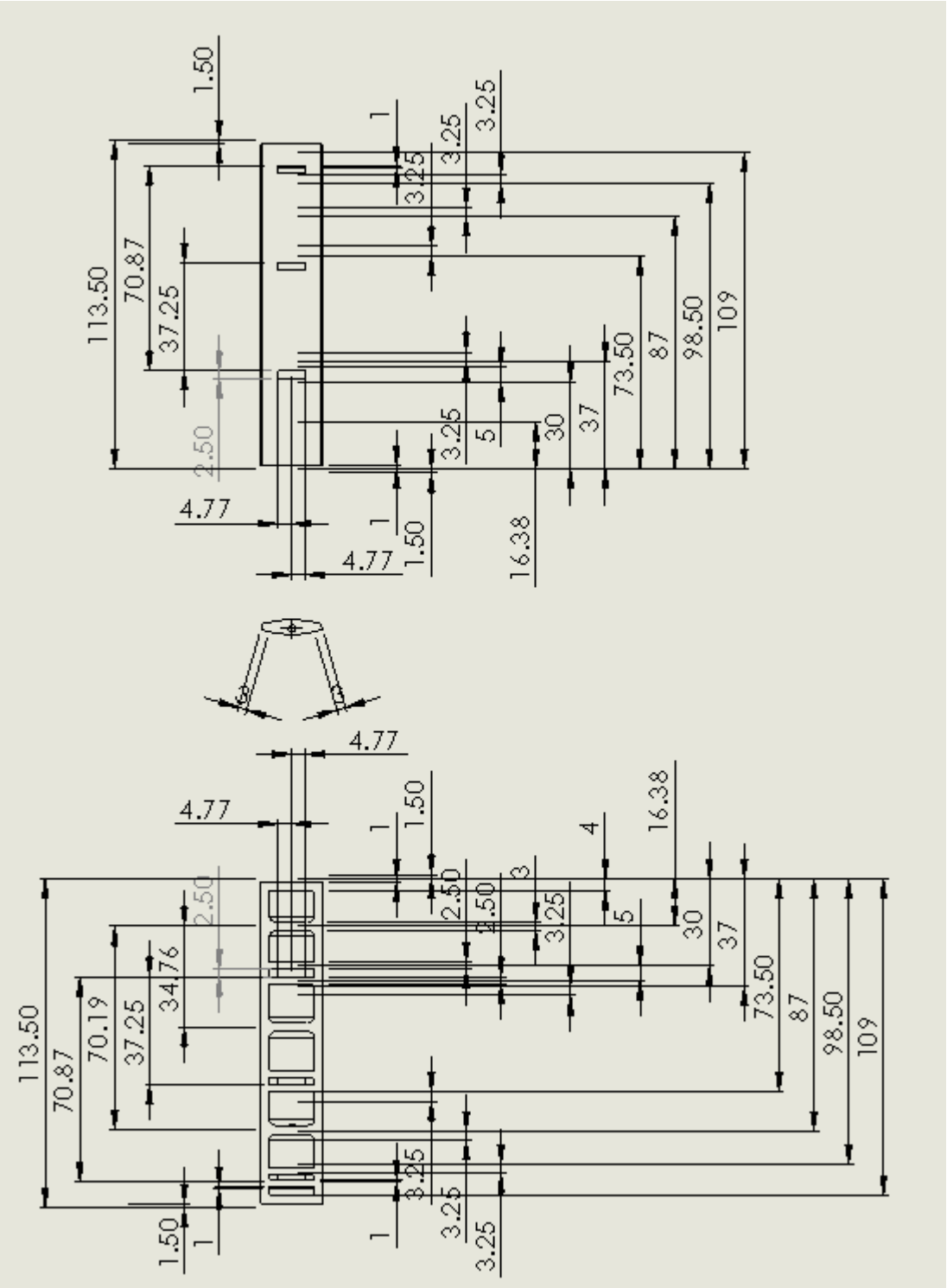
13.1.6 - Layer 3

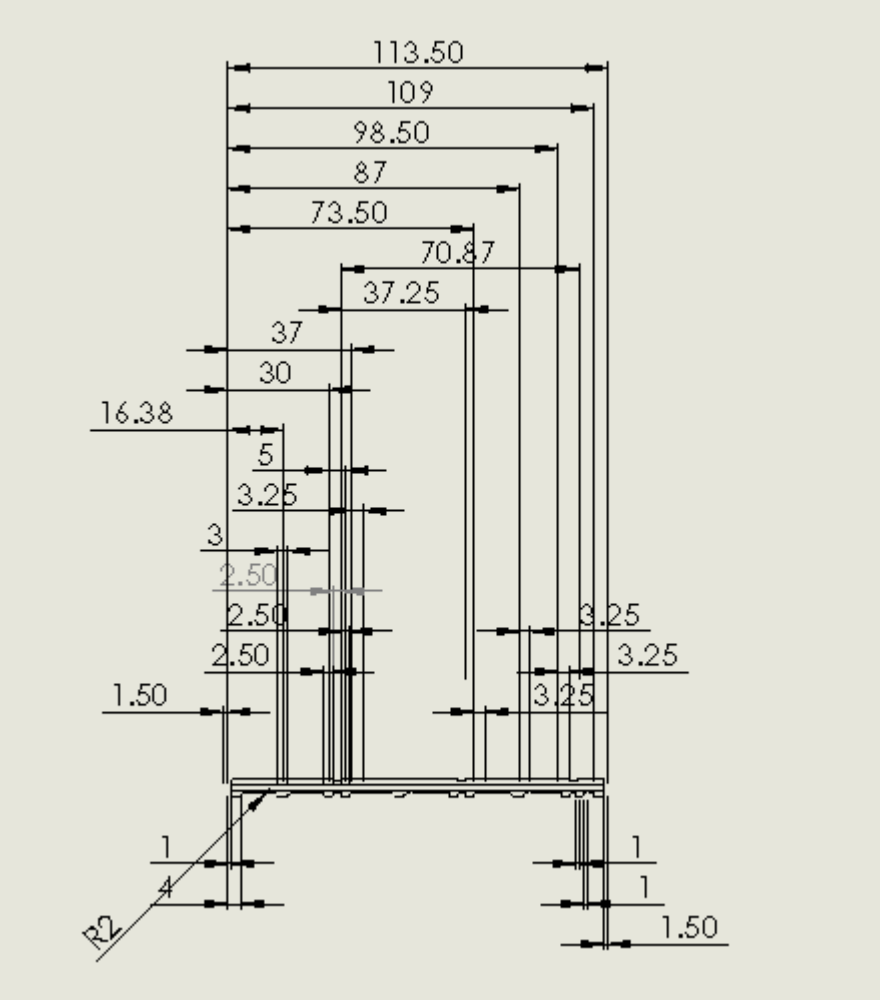


13.1.6 - Bottom Layer

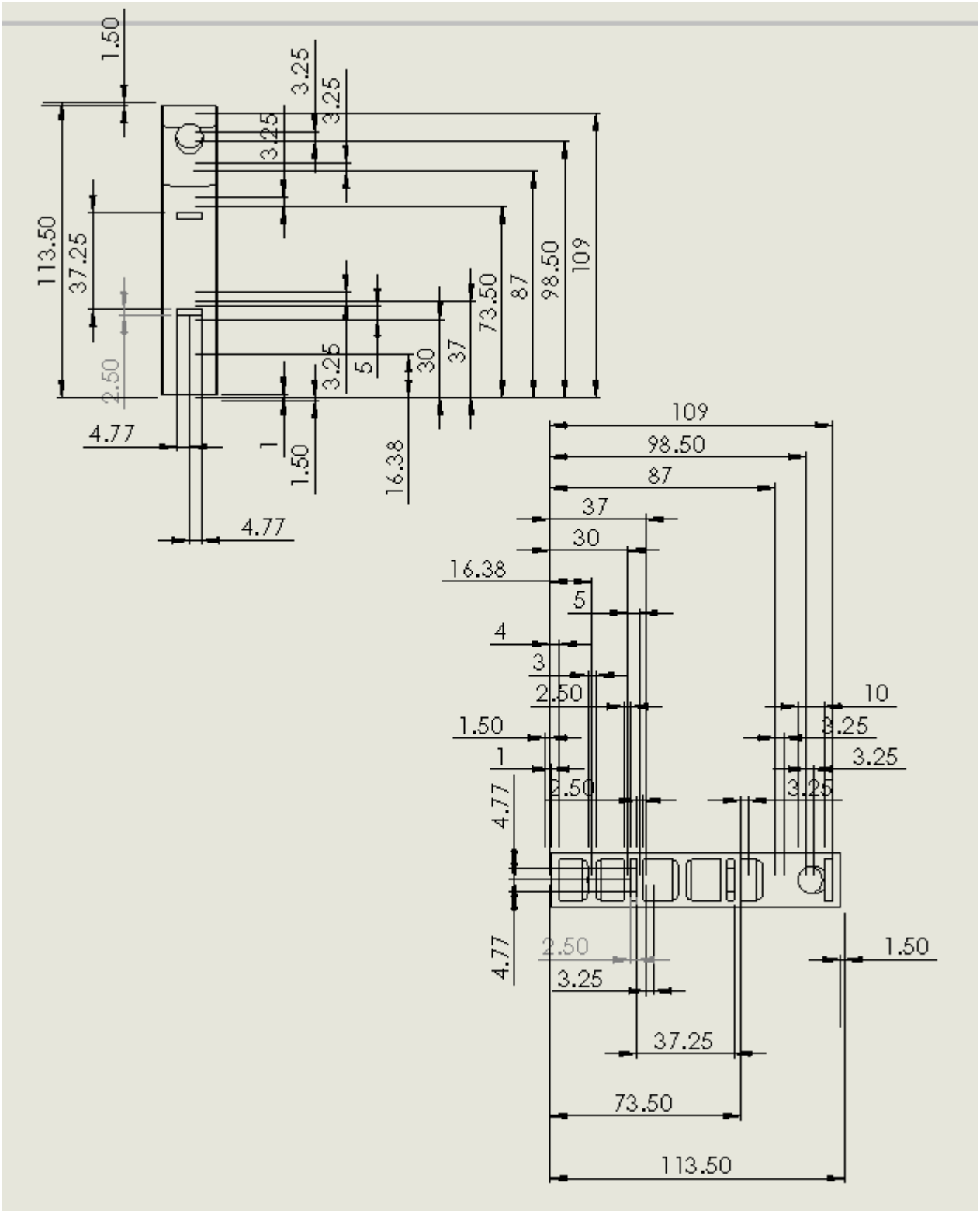


13.1.7 - Post

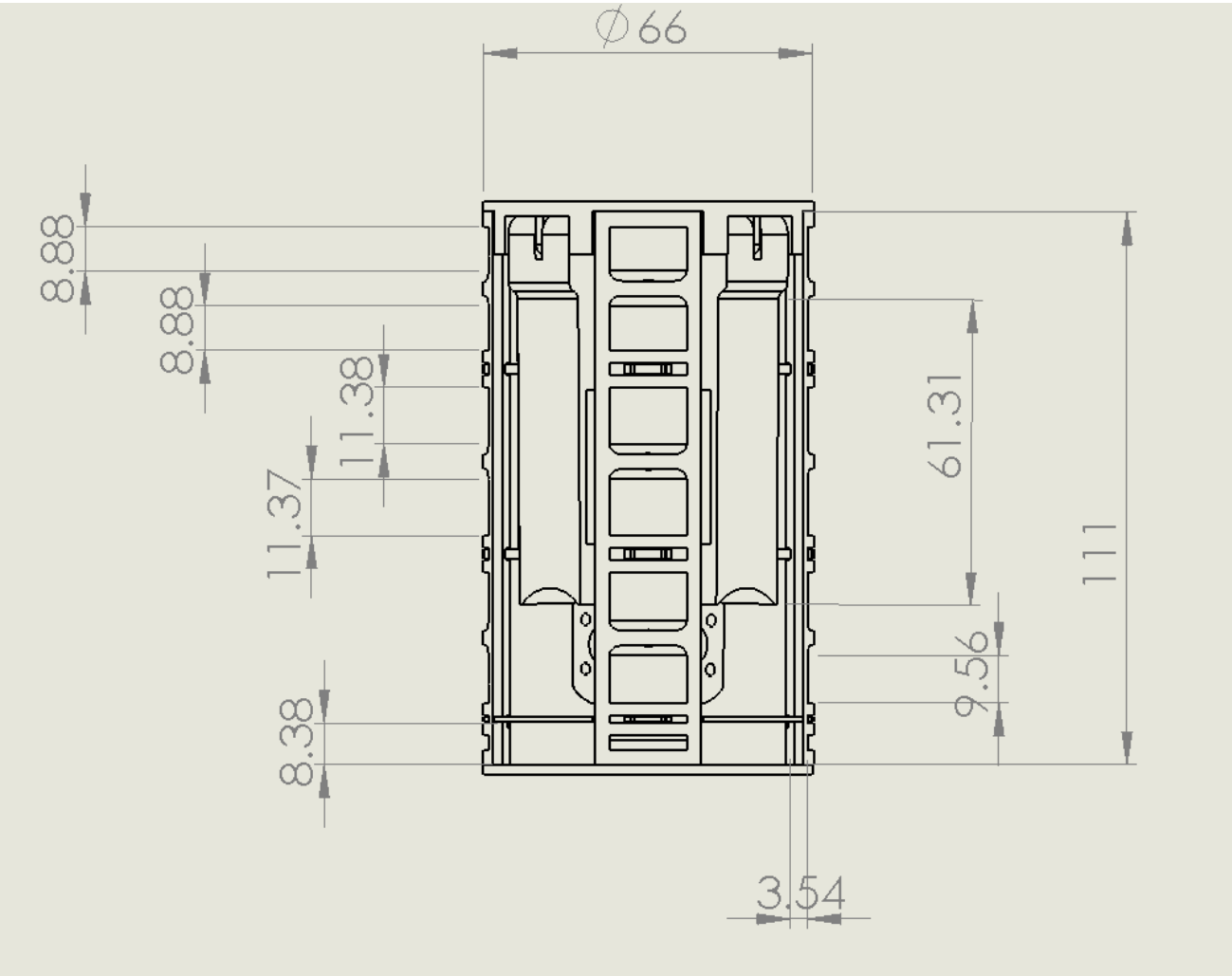

1.50
1
3.25
3.25
3.25
113.50
70.87
37.25
2.50
4.77
4.77
1
1.50
3.25
5
30
37
73.50
87
98.50
109
16.38
4.77
4.77
1
1.50
2.50
2.50
3
3.25
4
5
16.38
30
37
73.50
87
98.50
109
113.50
70.87
70.19
37.25
34.76
2.50
1.50
1
1
3.25
3.25


113.50
109
98.50
87
73.50
70.87
37.25
37
30
16.38
5
3.25
3
2.50
2.50
2.50
1.50
3.25
3.25
3.25
1
4
R2
1
1
1.50

13.1.8 - Front Post

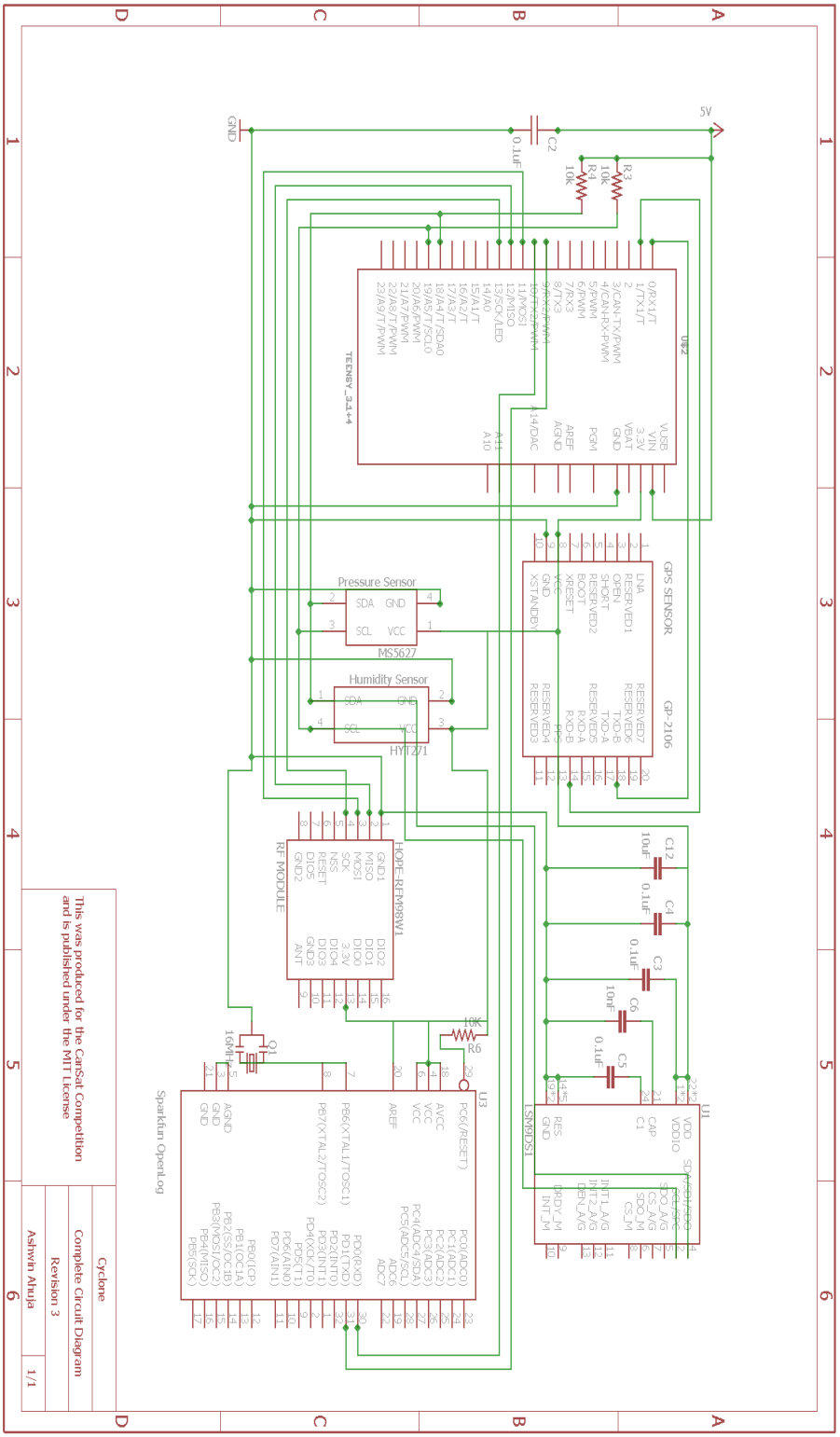


13.1.9 - Stack Assembly

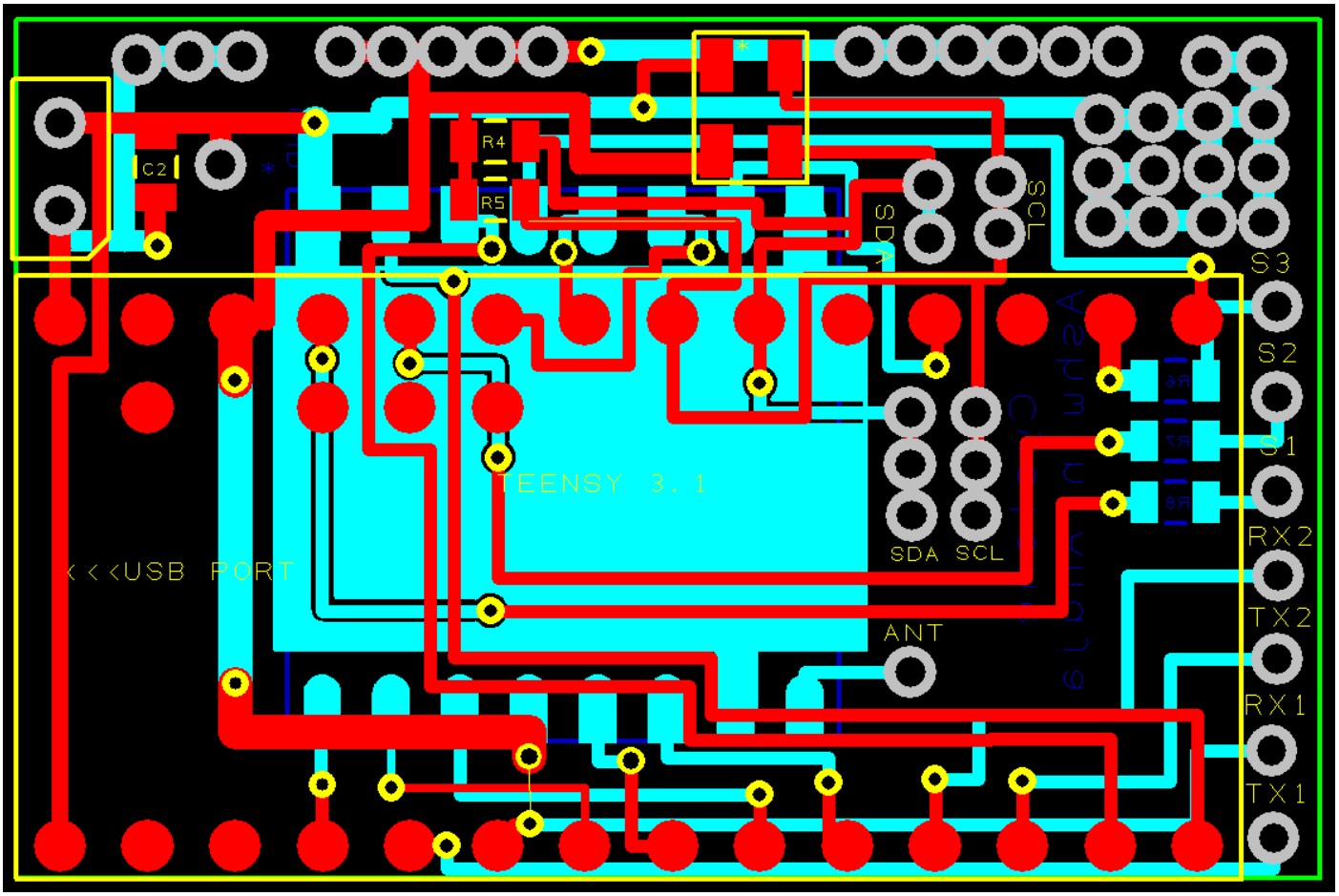


13.2 – Electronics Design

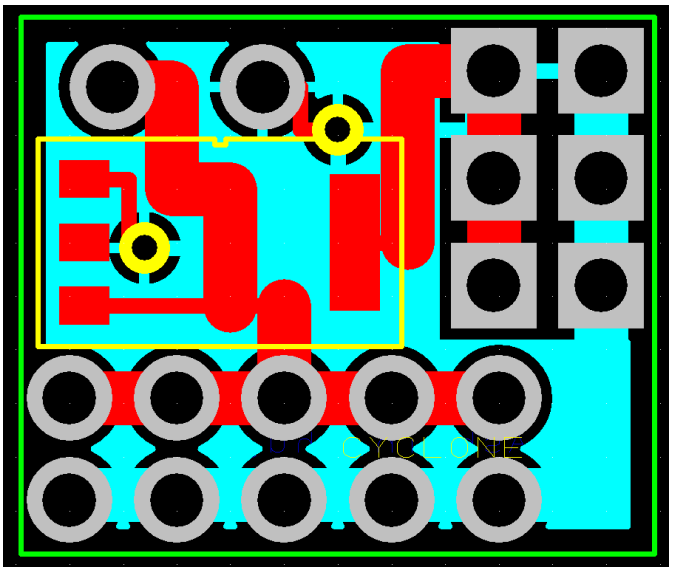
Sensor System Schematic



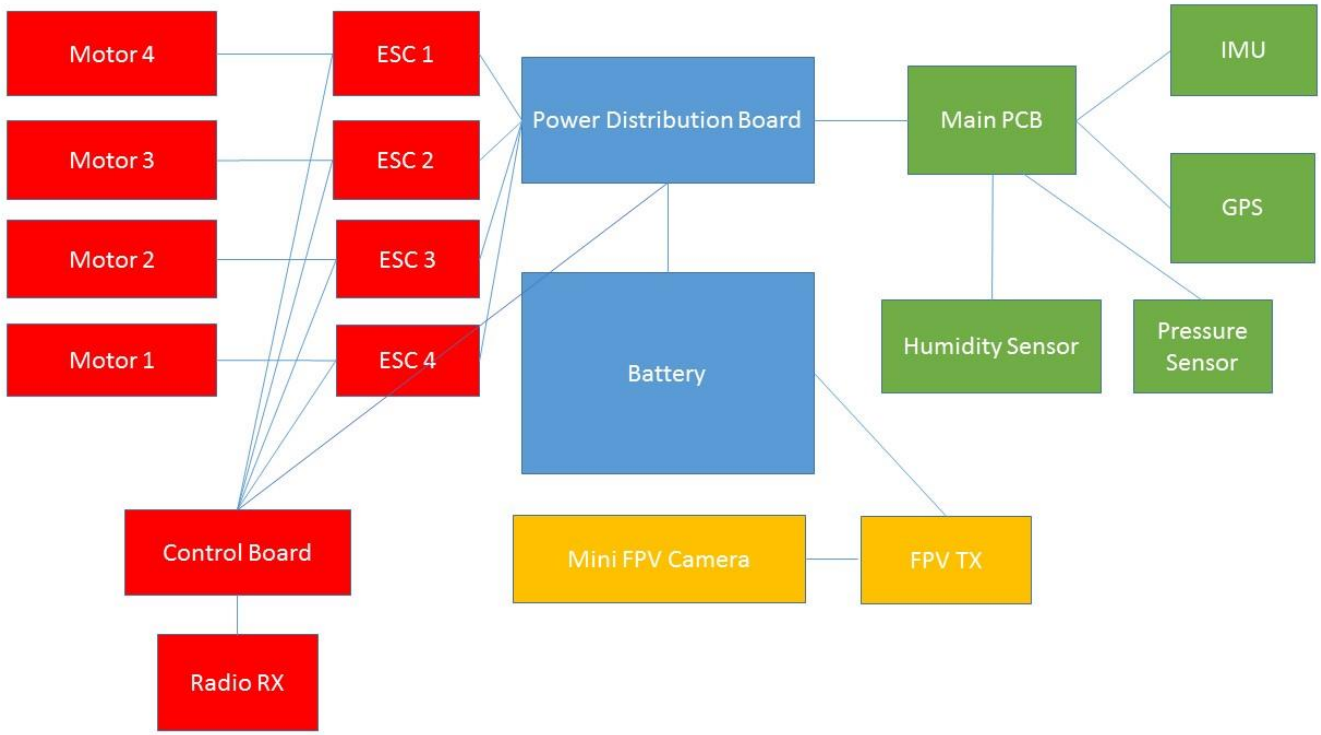
Sensor System PCB



Power Distribution PCB



Wiring Diagram



13.3 – Code

13.3.1 – Can Code

DISCOVERY.ino

```
// This is designed for the Advanced Category of the United Kingdom CanSat
// Competition 2016, as part of Cyclone, a team from St Paul's School,
// Barnes, London.
//
// For more information about the entry, including the Electronics Design
// Take a look on the GitHub Page - http://github.com/cyclonecansat or our
// website at: http://teamcycl.one
//
// Any questions, contact me at: ashwin.ahuja@gmail.com
/*
   INCLUDES

   Sensor Library - Library for the reading and manipulation of the MS5637
   and HYT271 - the pressure, internal temperature (MS5637) and humidity
   and external temperature (HYT271) sensors

   RFM98W Library - Library for the RFM98W Transceiver Board, using the LoRa
```

```
   transmission protocol - including dealing with creation of the Packet and
   the transmission and receiving of data.

   Servo Library - Library used for the release of the parachute, currently
   unused, since it will likely not be used.

   Wire Library - Library required for all I2C communication.

   TinyGPS++ Library - Library used for receiving and interpreting data from
   the GP-2106 module (available from Sparkfun) and used alongside the board
   produced by Sparkfun - the GP-2106 Evaluation Board

   SPI - Library used for all SPI communications - currently unused, but could
   be used for the 9DOF sensor.
*/
#include <sensor_library.h>
#include <RFM98W_library.h>
#include <Servo.h>
#include <Wire.h>
#include <TinyGPS++.h>
#include <SPI.h>
#include <LSM9DS1_Registers.h>
#include <LSM9DS1_Types.h>
#include <SparkFunLSM9DS1.h>
/*
   DEFINITIONS

   Pins for the RFM98W - as implied
   Hardware Serial declarations for the GPS module and the OpenLog
   Number of bytes implied by each command
   Parachute Release Servo Pin
   Boolean - parachute released
   Boolean - parachute armed
   Parachute Servo - move from (minimum)
   Parachute Servo - move to (maximum)
   Baud Rate for GPS communication
   Baud Rate for Computer Serial communications
   Verbosity - 0 for none, 1 for all
   Boolean - Longitude
   Boolean - Latitude
   Time between transmissions (during other times, the system will be receiving
data from the RFM98W
   Various pieces of information about the program
*/
//LoRa Pins
#define nss 20
#define dio0 7
#define dio5 16
#define rfm_rst 21

// Serial Declarations
#define gpsSerial Serial1
#define openLog Serial3
```

```
//radio command lengths
const byte cmd_lengths[8] = {0, 8, 2, 1, 1, 2, 2, 2};

// Servo Info
const int ServoPin = 0; // NOT CORRECT
bool parachuteReleased = false;
bool parachuteArmed = false;
const int servoMin = 0;
const int servoMax = 180;

// Baud Rate Declarations
#define GPS_BaudRate 4800
#define Computer_BaudRate 115200
#define OpenLogBaudRate 9600

//Verbosity
#define verbosity 1

//LSM9DS1 definitions
#define LSM9DS1_M  0x1E // Would be 0x1C if SDO_M is LOW
#define LSM9DS1_AG  0x6B // Would be 0x6A if SDO_AG is LOW
#define PRINT_CALCULATED

//GPS Declarations
bool latitudePositive = false;
bool longitudePositive = true;

//Time between transmissions
#define timeBetweenTransmissions 250
#define sensorReadingPeriod 250


//Various other info
#define SoftwareVersionNumber "1.5.0"
#define Author "Ashwin Ahuja"

/*
   OBJECT DECLARATIONS

   TinyGPSPlus Object - receives and decodes GPS data
   SensLib - Communication with MS5637 and HYT271
   RFMLib - Communication with Hope RFM98W
   LSM9DS1 -  Communication with 9DOF
   Servo - Parachite release
   Radio transmission timer.
   Sensor reading timer
   Packet received boolean
   Has MS5637 been read?
   Is it time to read?
*/

TinyGPSPlus gps; // GPS
```

```
SensLib sns; // Sensor Object
RFMLib radio = RFMLib(nss, dio0, dio5, rfm_rst); // Radio Object
LSM9DS1 imu; // LSM9DS1 Object
Servo ParaRelease; // Parachute Release Servo Object
uint32_t radioTransmitTimer;
uint32_t sensorReadTimer;
boolean pkt_rx = false;
boolean msRead = false;
boolean readSens = false;

/*
   Miscellaneous Declarations
*/
float gyx;
float gyy;
float gyz;
float acx;
float acy;
float acz;
float magx;
float magy;
float magz;
float roll;
float pitch;
float heading;
int sampleNumber;

void assemblePacket (RFMLib:: Packet &pkt);
void decodePacket (RFMLib:: Packet pkt);
void transmission();
void RFFinished();
void finishRFM();
void printToOpenLog();
void calculateHeadingAndRoll();
void readIMU();

void setup() {
  ParaRelease.attach(ServoPin);
  ParaRelease.write(servoMin);
  Serial.begin(Computer_BaudRate);
  gpsSerial.begin(GPS_BaudRate);
  openLog.begin(OpenLogBaudRate);
  openLog.println("SAMPLE NUMBER, INTERNAL TEMPERATURE, PRESSURE, EXTERNAL
TEMPERATURE, HUMIDITY, TIME(HOURS), TIME(MINUTES), TIME(SECONDS), GPS FIX (SATS),
LONGITUDE, LATITUDE, ALTITUDE, ACCELERATION IN X, ACCELERATION IN Y, ACCELERATION
IN Z, ROTATION IN X, ROTATION IN Y, ROTATION IN Z, MAGNETIC FIELD STRENGTH IN X,
MAGNETIC FIELD STRENGTH IN Y, MAGNETIC FIELD STRENGTH IN Z, HEADING, PITCH, ROLL,
AGRICULTURAL VIABILITY, ALTITUDE (USING PRESSURE SENSOR), DEW POINT");
  SPI.begin();
  byte my_config[6] = {0x44,0x84,0x88,0xAC,0xCD, 0x08};
  radio.configure(my_config);//Radio configuration
  Wire.begin();//join the I2C bus
  sns.initialise();//initialise the sensors connected over I2C
```

```
  imu.settings.device.commInterface = IMU_MODE_I2C;
  imu.settings.device.mAddress = LSM9DS1_M;
  imu.settings.device.agAddress = LSM9DS1_AG;
  radioTransmitTimer = millis();
#if verbosity != 0
  Serial.print("Cyclone CanSat Firmware Version Number: ");
  Serial.print(SoftwareVersionNumber);
  Serial.print(" (Discovery) Verbosity is equal to ");
  Serial.println(verbosity);
#endif
  if (imu.begin())
  {
      #if verbosity == 1
          Serial.println("9DOF Initialised");
      #endif
  }
  sampleNumber = 0;
}

void loop() {
  // READ FROM IMU
  readIMU();
  if (millis() - sensorReadTimer >= sensorReadingPeriod)
    readSens = true;
  // READ FROM SENSORS
  if (readSens)
  {
    if (msRead)
    {
      sns.pollHYT271();
      readSens = false;
      sensorReadTimer = millis();
      msRead = false;
    }
    else
    {
      sns.pollMS5637();
      msRead = true;
    }
  }
  if (radio.rfm_done)
    finishRFM();
  while (gpsSerial.available())
    gps.encode(gpsSerial.read());
  if ((millis() - radioTransmitTimer) > timeBetweenTransmissions &&
radio.rfm_status != 1)
    transmission();
}
void transmission()
{
  if (radio.rfm_status == 2)
  {
    RFMLib::Packet p;
```

```
    radio.endRX(p);
  }
  RFMLib::Packet p;
  assemblePacket(p);
  radio.beginTX(p);
  attachInterrupt(7, RFFinished, RISING);
  radioTransmitTimer = millis();
  sensorReadTimer = millis();
}
void RFFinished()
{
  radio.rfm_done = true;
}
void finishRFM()
{
  switch (radio.rfm_status) {
    case 1:
      radio.endTX();
      radio.beginRX();
      radio.rfm_done = false;
      attachInterrupt(7, RFFinished, RISING);
      break;
    case 2:
#if verbosity != 0
      Serial.println("Ending reception.");
#endif
      RFMLib::Packet rx;
      radio.endRX(rx);
      decodePacket(rx);
      break;
  }
}
void decodePacket(RFMLib::Packet pkt)
{
  byte j = 0;
  if (verbosity != 0)
  {
    Serial.println("Decode: ");
    for (int i = 0; i <= pkt.len; i++)
    {
      Serial.print(pkt.data[i]);
    }
    Serial.println();
  }
  while (j < pkt.len)
  {
    switch (pkt.data[j]) {
      case 7:
        if (pkt.data[j + 1] == 255 && pkt.data[j + 2] == 255)
        {
          Serial.println("RELEASE");
          ParaRelease.write(servoMax);
        }
```

```
        j = j + 2;
        break;
    }
  }
}
void printToOpenLog()
{
  float gamma = log(sns.humidity/100.0) + ((17.67 *
sns.external_temperature)/(243.5 + sns.external_temperature));
    float dewPoint = ((243.5 * gamma)/(17.67 - gamma));
    // float dewPoint = (((log((sns.humidity)/100)) + ((17.62 *
sns.external_temperature)/(243.12+sns.external_temperature)))/(17.62 -
(log((sns.humidity)/(100)) +
((17.62*sns.external_temperature)/(243.12+sns.external_temperature)))));
    float epress = sns.pressure / 1013.2501;
    float tempViability = 1-(pow((sns.external_temperature-27),2))/(972);
    float presViability = (-(pow(epress, 2))+(3*epress)-1)/epress;
    float humViability = -1*pow((sns.humidity/100),2)+((2*sns.humidity)/100);
    float agriViability = tempViability * presViability * humViability;
    float qfe = 1010.0;
    float h = ((sns.external_temperature+273)/(-0.0065)) *
(pow((sns.pressure/qfe),((-8.31432 * -0.0065)/(9.80665*0.0289644)))-1);

  openLog.print(sampleNumber);
  openLog.print(",");
  openLog.print(sns.internal_temperature);
  openLog.print(",");
  openLog.print(sns.pressure);
  openLog.print(",");
  openLog.print(sns.external_temperature);
  openLog.print(",");
  openLog.print(sns.humidity);
  int time3 = gps.time.hour() * 3600 + gps.time.minute() * 60 +
gps.time.second();
  openLog.print(",");
  openLog.print(gps.time.hour());
  openLog.print(",");
  openLog.print(gps.time.minute());
  openLog.print(",");
  openLog.print(gps.time.second());
  openLog.print(",");
  openLog.print(gps.satellites.value());
  openLog.print(",");
  openLog.print(gps.location.lng());
  openLog.print(",");
  openLog.print(gps.location.lat());
  openLog.print(",");
  openLog.print(gps.altitude.meters());
  openLog.print(",");
  openLog.print(acx);
  openLog.print(",");
  openLog.print(acy);
  openLog.print(",");
```

```
  openLog.print(acz);
  openLog.print(",");
  openLog.print(gyx);
  openLog.print(",");
  openLog.print(gyy);
  openLog.print(",");
  openLog.print(gyz);
  openLog.print(",");
  openLog.print(magx);
  openLog.print(",");
  openLog.print(magy);
  openLog.print(",");
  openLog.print(magz);
  openLog.print(",");
  openLog.print(heading);
  openLog.print(",");
  openLog.print(pitch);
  openLog.print(",");
  openLog.print(roll);
  openLog.print(",");
  openLog.print(agriViability);
  openLog.print(",");
  openLog.print(h);
  openLog.print(",");
  openLog.println(dewPoint);
}
void assemblePacket(RFMLib::Packet &pkt)
{
  pkt.len = 26;
  int32_t pr_calc = sns.pressure;
  byte round_byte = ((pr_calc % 10) > 4) ? 1 : 0;
  pr_calc /= 10;
  pr_calc += (int16_t) round_byte;
  uint16_t small_pressure = (uint16_t) pr_calc;
  sampleNumber++;
  pkt.data[0] = (byte)(sampleNumber >> 8);
  pkt.data[1] = (byte)(sampleNumber & 255);
  pkt.data[2] = (byte)(sns.internal_temperature >> 8);
  pkt.data[3] = sns.internal_temperature & 255;
  pkt.data[4] = (byte)(small_pressure >> 8);
  pkt.data[5] = small_pressure & 255;
  pkt.data[6] = (byte)(sns.external_temperature >> 8);
  pkt.data[7] = sns.external_temperature & 255;
  pkt.data[8] = (byte)(sns.humidity >> 8);
  pkt.data[9] = sns.humidity & 255;
  int time2 = gps.time.hour() * 3600 + gps.time.minute() * 60 +
gps.time.second();
  pkt.data[10] = (byte)(time2 >> 8);
  pkt.data[11] = time2 & 255;
  float rawGPSSats = gps.satellites.value(); // I really don't know why I get
number of connected satellites as a float!!
  int transmittableGPSSats = (int)rawGPSSats;
  pkt.data[12] = (byte)transmittableGPSSats;
```

```
  float rawlongitude = gps.location.lng();
  rawlongitude = rawlongitude * 1000000000;
  int longitudeToSend = (int)rawlongitude;
  pkt.data[13] = (byte)(longitudeToSend >> 24);
  pkt.data[14] = (byte)(longitudeToSend >> 16);
  pkt.data[15] = (byte)(longitudeToSend >> 8);
  pkt.data[16] = longitudeToSend & 255;
  float rawlatitude = gps.location.lat();
  rawlatitude = rawlatitude * 10000000;
  int latitudeToSend = (int)rawlatitude;
  pkt.data[17] = (byte)(latitudeToSend >> 24);
  pkt.data[18] = (byte)(latitudeToSend >> 16);
  pkt.data[19] = (byte)(latitudeToSend >> 8);
  pkt.data[20] = latitudeToSend & 255;
  float raw_alt = gps.altitude.meters();
  int alt = (int)raw_alt;
  pkt.data[21] = (byte)(alt >> 8);
  pkt.data[22] = alt & 255;
  pkt.data[23] = (byte)((int)heading);
  pkt.data[24] = (byte)((int)pitch);
  pkt.data[25] = (byte)((int)roll);

  if (verbosity > 0)
  {
    Serial.print("Sample Number = ");
    Serial.println(sampleNumber);
    Serial.print("Internal Temp = ");
    Serial.println(sns.internal_temperature);
    Serial.print("Pressure = ");
    Serial.println(sns.pressure);
    Serial.print("External Temp = ");
    Serial.println(sns.external_temperature);
    Serial.print("Humidity = ");
    Serial.println(sns.humidity);
    int time3 = gps.time.hour() * 3600 + gps.time.minute() * 60 +
gps.time.second();
    Serial.print("Second = ");
    Serial.println(time3);
    Serial.print("GPS Fix = ");
    Serial.println(gps.satellites.value(), 10);
    Serial.print("Longitude = ");
    Serial.println(gps.location.lng(), 10);
    Serial.print("Latitude = ");
    Serial.println(gps.location.lat(), 10);
    Serial.print("Altitude = ");
    Serial.println(gps.altitude.meters(), 5);
    Serial.print("Acceleration in x = ");
    Serial.println(acx, 5);
    Serial.print("Acceleration in y = ");
    Serial.println(acy, 5);
    Serial.print("Acceleration in z = ");
    Serial.println(acz, 5);
    Serial.print("Rotation in x = ");
```

```
    Serial.println(gyx, 5);
    Serial.print("Rotation in y = ");
    Serial.println(gyy, 5);
    Serial.print("Rotation in z = ");
    Serial.println(gyz, 5);
    Serial.print("Heading = ");
    Serial.println(heading, 5);
    Serial.print("Pitch = ");
    Serial.println(pitch, 5);
    Serial.print("Roll = ");
    Serial.println(roll, 5);
    Serial.println();
    Serial.println();
    Serial.println();
    Serial.println();
    Serial.println();
  }
  printToOpenLog();
}


void calculateHeadingAndRoll()
{
#define DECLINATION -8.58
  roll = atan2(imu.ay, imu.az);
  pitch = atan2(-imu.ax, sqrt(imu.ay * imu.ay + imu.az * imu.az));
  if (imu.my == 0)
    heading = (imu.mx < 0) ? 180.0 : 0;
  else
    heading = atan2(imu.mx, imu.my);

  heading -= DECLINATION * PI / 180;

  if (heading > PI) heading -= (2 * PI);
  else if (heading < -PI) heading += (2 * PI);
  else if (heading < 0) heading += 2 * PI;

  // Convert everything from radians to degrees:
  heading *= 180.0 / PI;
  pitch *= 180.0 / PI;
  roll  *= 180.0 / PI;
}
void readIMU()
{
  imu.readGyro();
  imu.readAccel();
  imu.readMag();
  gyx = imu.calcGyro(imu.gx);
  gyy = imu.calcGyro(imu.gy);
  gyz = imu.calcGyro(imu.gz);
  acx = imu.calcAccel(imu.ax);
  acy = imu.calcAccel(imu.ay);
  acz = imu.calcAccel(imu.az);
```

```
  magx = imu.calcMag(imu.mx);
  magy = imu.calcMag(imu.my);
  magz = imu.calcMag(imu.mz);
  calculateHeadingAndRoll();
}
```

Sensor Library CPP File {adapted from http://bit.ly/1LO3H1x}

```
#include "sensor_library.h"


SensLib::SensLib(){
}
void SensLib::initialise(){
 Wire.beginTransmission(ms5637_addr);
 Wire.write(0x1E);
 Wire.endTransmission();

 for(int i = 0;i<7;i++){
  Wire.beginTransmission(ms_addr);
  int reg = 0xA0;
  reg += (2*i);
  Wire.write(reg);
  Wire.endTransmission();
  Wire.requestFrom(ms_addr,2);
  long timer = millis();
  while(!Wire.available()){
   if((millis() - timer)>1000)
     break;
  }
  calib[i]=(Wire.read()<<8) | Wire.read();//store in calibration value array
 }
}
```

```
void SensLib::pollMS5637(){
  uint32_t d1,d2;
  //read data
  Wire.beginTransmission(ms_addr);

  Wire.write(0x4A);//d1 convert
  Wire.endTransmission();
  delay(25);//typ time is 16.44ms for conversion at 8192 bit oversampling ratio - would guess that this is more or less constant, but doesn't hurt to allow an extra 9ms!
  Wire.beginTransmission(ms_addr);
  Wire.write(0x00);//ADC read
  Wire.endTransmission();
  Wire.requestFrom(ms_addr,3);//24 bits
  while(!Wire.available());
  d1 = (Wire.read()<<16) | (Wire.read()<<8) | Wire.read();//read in all three bits and get D1

  Wire.beginTransmission(ms_addr);//now for d2
  Wire.write(0x5A);//d2 convert
  Wire.endTransmission();
  delay(25);//typ time is 16.44ms for conversion at 8192 bit oversampling ratio - would guess that this is more or less constant, but doesn't hurt to allow an extra 9ms!
  Wire.beginTransmission(ms_addr);
  Wire.write(0x00);//ADC read
  Wire.endTransmission();
  Wire.requestFrom(ms_addr,3);//24 bits
  while(!Wire.available());

  d2 = (Wire.read()<<16) | (Wire.read()<<8) | Wire.read();
  int32_t dt = (int32_t)(d2- (int32_t)calib[5] * 256);
  int32_t temp =(int32_t)(2000+(int32_t)(dt*(int32_t)calib[6])/(int32_t)8388608);
```

```
  int64_t off = (int64_t)((int64_t)calib[2] * (int64_t)131072 + ((int64_t)calib[4]*(int64_t)dt) /(int64_t)64);
  int64_t sens =(int64_t)((int64_t)calib[1] * (int64_t)65536 + ((int64_t)calib[3] * (int64_t)dt)/(int64_t)128);
  int64_t sens2 = 0;
  int64_t off2 = 0;
  int64_t t2;
  if(temp>2000){//"second order temperature compensation"
   t2 = (int64_t)((int64_t)5 * (int64_t)((int64_t)dt*(int64_t)dt) / (int64_t)274877906944);
   //high temp
  }
  else{
   //low temp
   t2 = (int64_t)((int64_t)3 * (int64_t)((int64_t)dt*(int64_t)dt) / (int64_t)8589934592);
   off2 = (int64_t)temp - (int64_t)2000;
   off2 = (int64_t)(off2 * off2);
   sens2 = off2;
   off2 = (int64_t)((int64_t)61 * off2 / 16);
   sens2 = (int64_t)((int64_t)(29 * sens2) / 16);
  }
  //won't be less than -15degC with electronics, so don't need to account for this case.
  temp = (int32_t)((int64_t)temp - t2);//apply compensation
  off = off-off2;
  sens = sens-sens2;

  pressure = (int32_t)((d1 * sens / 2097152 - off) / 32768);//calculate final pressure
  internal_temperature = temp;
}

void SensLib::pollHYT271(){
 byte data[4];
 Wire.beginTransmission(hyt_addr);
 Wire.endTransmission();//send measurement request
```

```
 delay(80);//spec 60ms.
 Wire.requestFrom(hyt_addr,4);//V Important - we don't wait for wire available,
 //in case this causes a hang.
 for(byte i = 0;i<4;i++){
   data[i] = Wire.read();//read the expected five bytes into an array.
 }

 Wire.endTransmission();



 int rawH = ((data[0] << 8) & 0x3FFF) | data[1];
 int rawT = data[2] << 6;
 rawT = rawT | (data[3] >> 2);
 external_temperature = (int16_t)((((float)(rawT) * 165.0 / 16384.0) - 40.0)*100);
 humidity = (int16_t)(((float)rawH * 100.0 / 16384.0)*100);
}
```

Sensor Library Header File adapted from http://bit.ly/1LO3H1x}

```
/* (C) Cyclone 2015
 * Made available under a modification of the MIT license - see repository root.
 * Sensor library; MS5637 and HYT-271.
 */

 #ifndef SensLib_h
 #define SensLib_h

 #include <Arduino.h>
 #include <Wire.h>
 #define ms_addr 0x76
 #define hyt_addr 0x28
```

```
class SensLib {
public:
  int16_t internal_temperature;
  int16_t external_temperature;
  int16_t humidity;
  int32_t pressure;
  SensLib();
  void pollMS5637();
  void pollHYT271();.
  void initialise();
private:
 uint16_t calib[7];

};

#endif
```

RFM98W Library Header File

```
/* (C) Cyclone 2015
 * Made available under a modification of the MIT license - see repository root.
 * Comms library: RFM98W
 */

#ifndef RFMLib_h
#define RFMLib_h

#include <Arduino.h>
#include <SPI.h>
```

```
class RFMLib {
public:

  RFMLib(byte nss, byte dio0, byte dio5, byte rfm_rst);
  typedef struct Packet{//data structure for storing a packet
    byte len;
    byte data[256];
    int snr; // Signal to noise ratio
    int rssi; // Signal success indication - should be read in
    bool crc;
  } Packet;
  void configure(byte config[5]);
  void beginTX(Packet tx);
  void endTX();
  void beginRX();
  void endRX(Packet &received);

  volatile bool rfm_done;
    byte rfm_status;//0=idle,1=tx,2=rx


private:
  void radioMode(byte m);//set the mode of the radio

  void wRFM(byte ad, byte val);//IO functions
  void bwRFM(byte ad, byte vals[], int n);
  byte rRFM(byte ad);
  void brRFM(byte ad, byte vals[], byte len);
  byte nss, dio0, dio5, rfm_rst;
```

```
};

#endif
```

RFM98W CPP File

```
#include "RFM98W_library.h"

RFMLib::RFMLib(byte a, byte b, byte c, byte d){
   nss = (byte)a;
   dio0 = (byte)b;
   dio5 = (byte)c;
   rfm_rst = (byte)d;
   rfm_done = false;
   rfm_status =0;
}

void RFMLib::configure(byte config[5]){
   pinMode(nss,OUTPUT);
   pinMode(rfm_rst,OUTPUT);
   digitalWrite(nss,HIGH);
   digitalWrite(rfm_rst,HIGH);
   pinMode(dio0,INPUT);
   pinMode(dio5,INPUT);
   radioMode(0);
   wRFM(0x1D,config[0]);
   wRFM(0x1E,config[1]);//modem config registers
   wRFM(0x09,config[2]);//use PA_BOOST - even at the same power setting
   //it seems to give a stronger RSSI.
   wRFM(0x07,config[3]);//freq to 434.7MHz - mid SB
   wRFM(0x08,config[4]);//freq -LSB
```

```
}

void RFMLib::beginRX(){
   rfm_status = 2;
   rfm_done = false;
   radioMode(1);
   wRFM(0x12,255);//reset IRQ
   wRFM(0x0D,rRFM(0x0F));//set RX base address
   wRFM(0x40,0);//set up DIO0 interrupt
   radioMode(2);
   //You need to attach an interrupt function which sets this object's "rfm_done" bool to TRUE on a RISING interrupt on DIO0
}

void RFMLib::endRX(Packet& received){//function to be called on, or soon after, reception of RX_DONE interrupt
   rfm_done = false;
   rfm_status = 0;
   radioMode(1);//stby
   byte len = rRFM(0x13);//length of packet
   received.len = len;
   byte packet[(int)len];
   if(bitRead(rRFM(0x12),5)){
      received.crc = false;
   }
   else{
      wRFM(0x0D,0);
      brRFM(0x00,received.data,len);
   }
   received.rssi = (int)(rRFM(0x1B)-137);
   byte rawSNR = rRFM(0x19);
```

```
    if(bitRead(rawSNR,7)){
      received.snr = 0-(255-rawSNR);
    }
    else{
      received.snr = rawSNR;
    }
    received.snr /= 4;
    radioMode(0);//now sleeps
    wRFM(0x12,255);//clear IRQ again.
}

void RFMLib::endTX(){//function to be called at the end of transmission; cleans up.
    rfm_status = 0;
    rfm_done = false;
    radioMode(1);//stby
    wRFM(0x12,255);//clear IRQ
    radioMode(0);//sleep
}

void RFMLib::beginTX(Packet transmit_pkt){
    rfm_status = 1;
    rfm_done = false;
    radioMode(1);//stby
    wRFM(0x12,255);//clear IRQ
    wRFM(0x22,transmit_pkt.len);//set payload length;
    byte base_addr = rRFM(0x0E);
    wRFM(0x0D,base_addr);//Put transmit base FIFO addr in FIFO pointer
    byte new_data[transmit_pkt.len];
    for(int i = 0;i<transmit_pkt.len;i++){
      new_data[i] = transmit_pkt.data[i];
    }
```

```
  bwRFM(0x00,new_data,transmit_pkt.len);
  wRFM(0x0D,base_addr);//reset FIFO pointer
  wRFM(0x0D,base_addr);//and again...
  wRFM(0x40,0x40);//arm DIO0 interrupt
  radioMode(4);//begin transmit
  //you need to attach a rising interrupt on DIO0.
}

void RFMLib::radioMode(byte m){//set specified mode
  switch(m){
    case 0://sleep
      wRFM(0x01,0x80);
      break;
    case 1://stby
      wRFM(0x01,0x81);
      break;
    case 2://rx cont
      wRFM(0x01,0x85);
      break;
    case 3://rx single
      wRFM(0x01,0x86);
      break;
    case 4://tx
      wRFM(0x01,0x83);
      break;
  }

}

//Low-level IO functions beyond this point. ==========================================
void RFMLib::wRFM(byte ad, byte val){//single byte write
```

```
  digitalWrite(nss,LOW);
  SPI.transfer(ad | 128);//set wrn bit - WRITE = 1
  SPI.transfer(val);
  digitalWrite(nss,HIGH);
}

void RFMLib::bwRFM(byte ad, byte vals[], int n){//burst write - less efficient but faster
  //for multiple bits
  //(less efficient for singles due to array overhead etc)
  digitalWrite(nss,LOW);
  SPI.transfer(ad | 128);//set wrn bit - WRITE = 1
  for(int i = 0;i<n;i++)
    SPI.transfer(vals[i]);

  digitalWrite(nss,HIGH);
}

byte RFMLib::rRFM(byte ad){//single byte read
  digitalWrite(nss,LOW);
  SPI.transfer(ad & B01111111);//wrn bit low
  byte val = SPI.transfer(0);//read, but we still have to spec a value?
  digitalWrite(nss,HIGH);
  return val;
}

void RFMLib::brRFM(byte ad, byte vals[], byte len){//burst read - slower for singles due to
  digitalWrite(nss,LOW);   //overhead of working with arrays
  SPI.transfer(ad & 0x7F);//wrn bit low
  for(int i = 0;i<len;i++){
    vals[i] = SPI.transfer(0);
  }
```

```
  digitalWrite(nss,HIGH);
}
```

TinyGPS++ Library Header File (Taken from: http://arduiniana.org/libraries/tinygpsplus/)

```
/*
TinyGPS - a small GPS library for Arduino providing basic NMEA parsing
Based on work by and "distance_to" and "course_to" courtesy of Maarten Lamers.
Suggestion to add satellites(), course_to(), and cardinal(), by Matt Monson.
Copyright (C) 2008-2012 Mikal Hart
All rights reserved.

This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301  USA
*/

#ifndef TinyGPS_h
#define TinyGPS_h

#if defined(ARDUINO) && ARDUINO >= 100
```

```
#include "Arduino.h"
#else
#include "WProgram.h"
#endif

#define _GPS_VERSION 12 // software version of this library
#define _GPS_MPH_PER_KNOT 1.15077945
#define _GPS_MPS_PER_KNOT 0.51444444
#define _GPS_KMPH_PER_KNOT 1.852
#define _GPS_MILES_PER_METER 0.00062137112
#define _GPS_KM_PER_METER 0.001
// #define _GPS_NO_STATS

class TinyGPS
{
public:
  enum {
    GPS_INVALID_AGE = 0xFFFFFFFF,      GPS_INVALID_ANGLE = 999999999,
    GPS_INVALID_ALTITUDE = 999999999,  GPS_INVALID_DATE = 0,
    GPS_INVALID_TIME = 0xFFFFFFFF,                  GPS_INVALID_SPEED = 999999999,
    GPS_INVALID_FIX_TIME = 0xFFFFFFFF, GPS_INVALID_SATELLITES = 0xFF,
    GPS_INVALID_HDOP = 0xFFFFFFFF
  };

  static const float GPS_INVALID_F_ANGLE, GPS_INVALID_F_ALTITUDE, GPS_INVALID_F_SPEED;

  TinyGPS();
  bool encode(char c); // process one character received from GPS
  TinyGPS &operator << (char c) {encode(c); return *this;}

  // lat/long in hundred thousandths of a degree and age of fix in milliseconds
```

```
void get_position(long *latitude, long *longitude, unsigned long *fix_age = 0);

// date as ddmmyy, time as hhmmsscc, and age in milliseconds
void get_datetime(unsigned long *date, unsigned long *time, unsigned long *age = 0);

// signed altitude in centimeters (from GPGGA sentence)
inline long altitude() { return _altitude; }

// course in last full GPRMC sentence in 100th of a degree
inline unsigned long course() { return _course; }

// speed in last full GPRMC sentence in 100ths of a knot
inline unsigned long speed() { return _speed; }

// satellites used in last full GPGGA sentence
inline unsigned short satellites() { return _numsats; }

// horizontal dilution of precision in 100ths
inline unsigned long hdop() { return _hdop; }

void f_get_position(float *latitude, float *longitude, unsigned long *fix_age = 0);
void crack_datetime(int *year, byte *month, byte *day,
  byte *hour, byte *minute, byte *second, byte *hundredths = 0, unsigned long *fix_age = 0);
float f_altitude();
float f_course();
float f_speed_knots();
float f_speed_mph();
float f_speed_mps();
float f_speed_kmph();

static int library_version() { return _GPS_VERSION; }
```

```
  static float distance_between (float lat1, float long1, float lat2, float long2);
  static float course_to (float lat1, float long1, float lat2, float long2);
  static const char *cardinal(float course);

#ifndef _GPS_NO_STATS
  void stats(unsigned long *chars, unsigned short *good_sentences, unsigned short *failed_cs);
#endif

private:
  enum {_GPS_SENTENCE_GPGGA, _GPS_SENTENCE_GPRMC, _GPS_SENTENCE_OTHER};

  // properties
  unsigned long _time, _new_time;
  unsigned long _date, _new_date;
  long _latitude, _new_latitude;
  long _longitude, _new_longitude;
  long _altitude, _new_altitude;
  unsigned long  _speed, _new_speed;
  unsigned long  _course, _new_course;
  unsigned long  _hdop, _new_hdop;
  unsigned short _numsats, _new_numsats;

  unsigned long _last_time_fix, _new_time_fix;
  unsigned long _last_position_fix, _new_position_fix;

  // parsing state variables
  byte _parity;
  bool _is_checksum_term;
  char _term[15];
  byte _sentence_type;
```

```
  byte _term_number;
  byte _term_offset;
  bool _gps_data_good;

#ifndef _GPS_NO_STATS
  // statistics
  unsigned long _encoded_characters;
  unsigned short _good_sentences;
  unsigned short _failed_checksum;
  unsigned short _passed_checksum;
#endif

  // internal utilities
  int from_hex(char a);
  unsigned long parse_decimal();
  unsigned long parse_degrees();
  bool term_complete();
  bool gpsisdigit(char c) { return c >= '0' && c <= '9'; }
  long gpsatol(const char *str);
  int gpsstrcmp(const char *str1, const char *str2);
};

#if !defined(ARDUINO)
// Arduino 0012 workaround
#undef int
#undef char
#undef long
#undef byte
#undef float
#undef abs
#undef round
```

```
#endif

#endif
```

TinyGPS++ Library CPP File

```
/*
TinyGPS - a small GPS library for Arduino providing basic NMEA parsing
Based on work by and "distance_to" and "course_to" courtesy of Maarten Lamers.
Suggestion to add satellites(), course_to(), and cardinal(), by Matt Monson.
Copyright (C) 2008-2012 Mikal Hart
All rights reserved.

This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301  USA
*/

#include "TinyGPS.h"

#define _GPRMC_TERM   "GPRMC"
```

```
#define _GPGGA_TERM   "GPGGA"


TinyGPS::TinyGPS()
  : _time(GPS_INVALID_TIME)
  , _date(GPS_INVALID_DATE)
  , _latitude(GPS_INVALID_ANGLE)
  , _longitude(GPS_INVALID_ANGLE)
  , _altitude(GPS_INVALID_ALTITUDE)
  , _speed(GPS_INVALID_SPEED)
  , _course(GPS_INVALID_ANGLE)
  , _hdop(GPS_INVALID_HDOP)
  , _numsats(GPS_INVALID_SATELLITES)
  , _last_time_fix(GPS_INVALID_FIX_TIME)
  , _last_position_fix(GPS_INVALID_FIX_TIME)
  , _parity(0)
  , _is_checksum_term(false)
  , _sentence_type(_GPS_SENTENCE_OTHER)
  , _term_number(0)
  , _term_offset(0)
  , _gps_data_good(false)
#ifndef _GPS_NO_STATS
  , _encoded_characters(0)
  , _good_sentences(0)
  , _failed_checksum(0)
#endif
{
  _term[0] = '\0';
}

//
// public methods
```

```
//

bool TinyGPS::encode(char c)
{
  bool valid_sentence = false;

#ifndef _GPS_NO_STATS
  ++_encoded_characters;
#endif
  switch(c)
  {
  case ',': // term terminators
    _parity ^= c;
  case '\r':
  case '\n':
  case '*':
    if (_term_offset < sizeof(_term))
    {
      _term[_term_offset] = 0;
      valid_sentence = term_complete();
    }
    ++_term_number;
    _term_offset = 0;
    _is_checksum_term = c == '*';
    return valid_sentence;

  case '$': // sentence begin
    _term_number = _term_offset = 0;
    _parity = 0;
    _sentence_type = _GPS_SENTENCE_OTHER;
    _is_checksum_term = false;
```

```
    _gps_data_good = false;
    return valid_sentence;
  }

  // ordinary characters
  if (_term_offset < sizeof(_term) - 1)
    _term[_term_offset++] = c;
  if (!_is_checksum_term)
    _parity ^= c;

  return valid_sentence;
}

#ifndef _GPS_NO_STATS
void TinyGPS::stats(unsigned long *chars, unsigned short *sentences, unsigned short *failed_cs)
{
  if (chars) *chars = _encoded_characters;
  if (sentences) *sentences = _good_sentences;
  if (failed_cs) *failed_cs = _failed_checksum;
}
#endif

//
// internal utilities
//
int TinyGPS::from_hex(char a)
{
  if (a >= 'A' && a <= 'F')
    return a - 'A' + 10;
  else if (a >= 'a' && a <= 'f')
    return a - 'a' + 10;
```

```
  else
    return a - '0';
}

unsigned long TinyGPS::parse_decimal()
{
  char *p = _term;
  bool isneg = *p == '-';
  if (isneg) ++p;
  unsigned long ret = 100UL * gpsatol(p);
  while (gpsisdigit(*p)) ++p;
  if (*p == '.')
  {
    if (gpsisdigit(p[1]))
    {
      ret += 10 * (p[1] - '0');
      if (gpsisdigit(p[2]))
        ret += p[2] - '0';
    }
  }
  return isneg ? -ret : ret;
}

unsigned long TinyGPS::parse_degrees()
{
  char *p;
  unsigned long left = gpsatol(_term);
  unsigned long tenk_minutes = (left % 100UL) * 10000UL;
  for (p=_term; gpsisdigit(*p); ++p);
  if (*p == '.')
  {
```

```
   unsigned long mult = 1000;
   while (gpsisdigit(*++p))
   {
    tenk_minutes += mult * (*p - '0');
    mult /= 10;
   }
  }
  return (left / 100) * 100000 + tenk_minutes / 6;
}

#define COMBINE(sentence_type, term_number) (((unsigned)(sentence_type) << 5) | term_number)

// Processes a just-completed term
// Returns true if new sentence has just passed checksum test and is validated
bool TinyGPS::term_complete()
{
  if (_is_checksum_term)
  {
   byte checksum = 16 * from_hex(_term[0]) + from_hex(_term[1]);
   if (checksum == _parity)
   {
    if (_gps_data_good)
    {
#ifndef _GPS_NO_STATS
      ++_good_sentences;
#endif
      _last_time_fix = _new_time_fix;
      _last_position_fix = _new_position_fix;

      switch(_sentence_type)
      {
```

```
      case _GPS_SENTENCE_GPRMC:
        _time      = _new_time;
        _date      = _new_date;
        _latitude  = _new_latitude;
        _longitude = _new_longitude;
        _speed     = _new_speed;
        _course    = _new_course;
        break;
      case _GPS_SENTENCE_GPGGA:
        _altitude  = _new_altitude;
        _time      = _new_time;
        _latitude  = _new_latitude;
        _longitude = _new_longitude;
        _numsats   = _new_numsats;
        _hdop      = _new_hdop;
        break;
      }

      return true;
    }
  }

#ifndef _GPS_NO_STATS
  else
    ++_failed_checksum;
#endif
  return false;
 }

 // the first term determines the sentence type
 if (_term_number == 0)
```

```
{
  if (!gpsstrcmp(_term, _GPRMC_TERM))
    _sentence_type = _GPS_SENTENCE_GPRMC;
  else if (!gpsstrcmp(_term, _GPGGA_TERM))
    _sentence_type = _GPS_SENTENCE_GPGGA;
  else
    _sentence_type = _GPS_SENTENCE_OTHER;
  return false;
}

if (_sentence_type != _GPS_SENTENCE_OTHER && _term[0])
  switch(COMBINE(_sentence_type, _term_number))
{
  case COMBINE(_GPS_SENTENCE_GPRMC, 1): // Time in both sentences
  case COMBINE(_GPS_SENTENCE_GPGGA, 1):
    _new_time = parse_decimal();
    _new_time_fix = millis();
    break;
  case COMBINE(_GPS_SENTENCE_GPRMC, 2): // GPRMC validity
    _gps_data_good = _term[0] == 'A';
    break;
  case COMBINE(_GPS_SENTENCE_GPRMC, 3): // Latitude
  case COMBINE(_GPS_SENTENCE_GPGGA, 2):
    _new_latitude = parse_degrees();
    _new_position_fix = millis();
    break;
  case COMBINE(_GPS_SENTENCE_GPRMC, 4): // N/S
  case COMBINE(_GPS_SENTENCE_GPGGA, 3):
    if (_term[0] == 'S')
      _new_latitude = -_new_latitude;
    break;
```

```
  case COMBINE(_GPS_SENTENCE_GPRMC, 5): // Longitude
  case COMBINE(_GPS_SENTENCE_GPGGA, 4):
    _new_longitude = parse_degrees();
    break;
  case COMBINE(_GPS_SENTENCE_GPRMC, 6): // E/W
  case COMBINE(_GPS_SENTENCE_GPGGA, 5):
    if (_term[0] == 'W')
      _new_longitude = -_new_longitude;
    break;
  case COMBINE(_GPS_SENTENCE_GPRMC, 7): // Speed (GPRMC)
    _new_speed = parse_decimal();
    break;
  case COMBINE(_GPS_SENTENCE_GPRMC, 8): // Course (GPRMC)
    _new_course = parse_decimal();
    break;
  case COMBINE(_GPS_SENTENCE_GPRMC, 9): // Date (GPRMC)
    _new_date = gpsatol(_term);
    break;
  case COMBINE(_GPS_SENTENCE_GPGGA, 6): // Fix data (GPGGA)
    _gps_data_good = _term[0] > '0';
    break;
  case COMBINE(_GPS_SENTENCE_GPGGA, 7): // Satellites used (GPGGA)
    _new_numsats = (unsigned char)atoi(_term);
    break;
  case COMBINE(_GPS_SENTENCE_GPGGA, 8): // HDOP
    _new_hdop = parse_decimal();
    break;
  case COMBINE(_GPS_SENTENCE_GPGGA, 9): // Altitude (GPGGA)
    _new_altitude = parse_decimal();
    break;
}
```

```
  return false;
}

long TinyGPS::gpsatol(const char *str)
{
  long ret = 0;
  while (gpsisdigit(*str))
    ret = 10 * ret + *str++ - '0';
  return ret;
}

int TinyGPS::gpsstrcmp(const char *str1, const char *str2)
{
  while (*str1 && *str1 == *str2)
    ++str1, ++str2;
  return *str1;
}

/* static */
float TinyGPS::distance_between (float lat1, float long1, float lat2, float long2)
{
  // returns distance in meters between two positions, both specified
  // as signed decimal-degrees latitude and longitude. Uses great-circle
  // distance computation for hypothetical sphere of radius 6372795 meters.
  // Because Earth is no exact sphere, rounding errors may be up to 0.5%.
  // Courtesy of Maarten Lamers
  float delta = radians(long1-long2);
  float sdlong = sin(delta);
  float cdlong = cos(delta);
  lat1 = radians(lat1);
```

```
  lat2 = radians(lat2);
  float slat1 = sin(lat1);
  float clat1 = cos(lat1);
  float slat2 = sin(lat2);
  float clat2 = cos(lat2);
  delta = (clat1 * slat2) - (slat1 * clat2 * cdlong);
  delta = sq(delta);
  delta += sq(clat2 * sdlong);
  delta = sqrt(delta);
  float denom = (slat1 * slat2) + (clat1 * clat2 * cdlong);
  delta = atan2(delta, denom);
  return delta * 6372795;
}

float TinyGPS::course_to (float lat1, float long1, float lat2, float long2)
{
  // returns course in degrees (North=0, West=270) from position 1 to position 2,
  // both specified as signed decimal-degrees latitude and longitude.
  // Because Earth is no exact sphere, calculated course may be off by a tiny fraction.
  // Courtesy of Maarten Lamers
  float dlon = radians(long2-long1);
  lat1 = radians(lat1);
  lat2 = radians(lat2);
  float a1 = sin(dlon) * cos(lat2);
  float a2 = sin(lat1) * cos(lat2) * cos(dlon);
  a2 = cos(lat1) * sin(lat2) - a2;
  a2 = atan2(a1, a2);
  if (a2 < 0.0)
  {
    a2 += TWO_PI;
  }
```

```
  return degrees(a2);
}

const char *TinyGPS::cardinal (float course)
{
  static const char* directions[] = {"N", "NNE", "NE", "ENE", "E", "ESE", "SE", "SSE", "S", "SSW", "SW", "WSW",
"W", "WNW", "NW", "NNW"};

  int direction = (int)((course + 11.25f) / 22.5f);
  return directions[direction % 16];
}

// lat/long in hundred thousandths of a degree and age of fix in milliseconds
void TinyGPS::get_position(long *latitude, long *longitude, unsigned long *fix_age)
{
  if (latitude) *latitude = _latitude;
  if (longitude) *longitude = _longitude;
  if (fix_age) *fix_age = _last_position_fix == GPS_INVALID_FIX_TIME ?
GPS_INVALID_AGE : millis() - _last_position_fix;
}

// date as ddmmyy, time as hhmmsscc, and age in milliseconds
void TinyGPS::get_datetime(unsigned long *date, unsigned long *time, unsigned long *age)
{
  if (date) *date = _date;
  if (time) *time = _time;
  if (age) *age = _last_time_fix == GPS_INVALID_FIX_TIME ?
GPS_INVALID_AGE : millis() - _last_time_fix;
}

void TinyGPS::f_get_position(float *latitude, float *longitude, unsigned long *fix_age)
```

```
{
  long lat, lon;
  get_position(&lat, &lon, fix_age);
  *latitude = lat == GPS_INVALID_ANGLE ? GPS_INVALID_F_ANGLE : (lat / 100000.0);
  *longitude = lat == GPS_INVALID_ANGLE ? GPS_INVALID_F_ANGLE : (lon / 100000.0);
}

void TinyGPS::crack_datetime(int *year, byte *month, byte *day,
  byte *hour, byte *minute, byte *second, byte *hundredths, unsigned long *age)
{
  unsigned long date, time;
  get_datetime(&date, &time, age);
  if (year)
  {
    *year = date % 100;
    *year += *year > 80 ? 1900 : 2000;
  }
  if (month) *month = (date / 100) % 100;
  if (day) *day = date / 10000;
  if (hour) *hour = time / 1000000;
  if (minute) *minute = (time / 10000) % 100;
  if (second) *second = (time / 100) % 100;
  if (hundredths) *hundredths = time % 100;
}

float TinyGPS::f_altitude()
{
  return _altitude == GPS_INVALID_ALTITUDE ? GPS_INVALID_F_ALTITUDE : _altitude / 100.0;
}

float TinyGPS::f_course()
```

```
{
  return _course == GPS_INVALID_ANGLE ? GPS_INVALID_F_ANGLE : _course / 100.0;
}

float TinyGPS::f_speed_knots()
{
  return _speed == GPS_INVALID_SPEED ? GPS_INVALID_F_SPEED : _speed / 100.0;
}

float TinyGPS::f_speed_mph()
{
  float sk = f_speed_knots();
  return sk == GPS_INVALID_F_SPEED ? GPS_INVALID_F_SPEED : _GPS_MPH_PER_KNOT * f_speed_knots();
}

float TinyGPS::f_speed_mps()
{
  float sk = f_speed_knots();
  return sk == GPS_INVALID_F_SPEED ? GPS_INVALID_F_SPEED : _GPS_MPS_PER_KNOT * f_speed_knots();
}

float TinyGPS::f_speed_kmph()
{
  float sk = f_speed_knots();
  return sk == GPS_INVALID_F_SPEED ? GPS_INVALID_F_SPEED : _GPS_KMPH_PER_KNOT * f_speed_knots();
}

const float TinyGPS::GPS_INVALID_F_ANGLE = 1000.0;
const float TinyGPS::GPS_INVALID_F_ALTITUDE = 1000000.0;
const float TinyGPS::GPS_INVALID_F_SPEED = -1.0;
```

13.3.2 – Base Station

Base Station XAML File

```
<Window x:Class="Halley__Version_1_.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:m="clr-
namespace:Microsoft.Maps.MapControl.WPF;assembly=Microsoft.Maps.MapControl.WPF"
        xmlns:local="clr-namespace:Halley__Version_1_"
        xmlns:cam="clr-namespace:WebcamControl;assembly=WebcamControl"
        xmlns:lvc="clr-namespace:LiveCharts;assembly=LiveCharts"
        mc:Ignorable="d"
        Title="Cyclone CanSat - Halley" Height="900" Width="1500">
    <Viewbox>
        <Grid Width="1524" Height="900" Background="WhiteSmoke">
            <Grid.RowDefinitions>
                <RowDefinition/>
                <RowDefinition Height="224*"/>
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition/>
                <ColumnDefinition Width="0*"/>
            </Grid.ColumnDefinitions>
            <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="43"
Margin="651,256.308,0,0" Stroke="Black" VerticalAlignment="Top" Width="395" Grid.Row="1"/>
            <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="43"
Margin="651,214.308,0,0" Stroke="Black" VerticalAlignment="Top" Width="395" Grid.Row="1"/>
            <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="43"
Margin="651,172.308,0,0" Stroke="Black" VerticalAlignment="Top" Width="395" Grid.Row="1"/>
            <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="43"
Margin="651,130.308,0,0" Stroke="Black" VerticalAlignment="Top" Width="395" Grid.Row="1"/>
            <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="43"
Margin="651,89.308,0,0" Stroke="Black" VerticalAlignment="Top" Width="395" Grid.Row="1"/>
            <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="43"
Margin="651,47.308,0,0" Stroke="Black" VerticalAlignment="Top" Width="395" Grid.Row="1"/>
            <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="46"
Margin="651,3.308,0,0" Stroke="Black" VerticalAlignment="Top" Width="395" Grid.Row="1"/>
            <Grid Margin="1096,77.308,10,781" Grid.Row="1">
                <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="38"
Stroke="Black" VerticalAlignment="Top" Width="418" RenderTransformOrigin="0.5,0.5">
                    <Rectangle.RenderTransform>
                        <TransformGroup>
                            <ScaleTransform ScaleY="-1"/>
                            <SkewTransform/>
                            <RotateTransform/>
                            <TranslateTransform/>
                        </TransformGroup>
                    </Rectangle.RenderTransform>
```

```
                </Rectangle>
                <Label x:Name="label2_Copy8" Content="Longitude / °N"
HorizontalAlignment="Left" Margin="2,6,0,0" VerticalAlignment="Top" Height="26" Width="101"/>
                <TextBox x:Name="longBox" HorizontalAlignment="Left" Height="26"
Margin="296,6,0,0" TextWrapping="Wrap" Text="0.000000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnly="True"/>
            </Grid>
            <Grid Margin="1096,3.308,10,855" Grid.Row="1">
                <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="38"
Stroke="Black" VerticalAlignment="Top" Width="418" RenderTransformOrigin="0.5,0.5">
                    <Rectangle.RenderTransform>
                        <TransformGroup>
                            <ScaleTransform ScaleY="-1"/>
                            <SkewTransform/>
                            <RotateTransform/>
                            <TranslateTransform/>
                        </TransformGroup>
                    </Rectangle.RenderTransform>
                </Rectangle>
                <Label x:Name="label2_Copy6" Content="Number of Satellites (GPS Fix)"
HorizontalAlignment="Left" Margin="2,6,0,0" VerticalAlignment="Top" Height="26" Width="167"/>

                <TextBox x:Name="satsBox" HorizontalAlignment="Left" Height="26"
Margin="296,6,0,0" TextWrapping="Wrap" Text="0" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnly="True"/>
            </Grid>
            <GridSplitter x:Name="gridSplitter" HorizontalAlignment="Left" Height="900"
Margin="614,0,0,0" VerticalAlignment="Top" Width="12" RenderTransformOrigin="2.339,0.496"
Grid.RowSpan="2"/>
            <m:Map x:Name="map" CredentialsProvider="Au-zGYgVQzzAWdnCldQiy8--
n5H8r95_FWHm4byx7I2ns-Bundb0xWiTgCDEI_xS" Margin="305,411,910,186" Mode="AerialWithLabels"
Grid.Row="1" />
            <GridSplitter x:Name="gridSplitter1" HorizontalAlignment="Left" Height="10"
Margin="4,387.308,0,0" VerticalAlignment="Top" Width="621" RenderTransformOrigin="0.5,0.5"
Grid.Row="1">
                <GridSplitter.RenderTransform>
                    <TransformGroup>
                        <ScaleTransform ScaleY="-1"/>
                        <SkewTransform/>
                        <RotateTransform/>
                        <TranslateTransform/>
                    </TransformGroup>
                </GridSplitter.RenderTransform>
            </GridSplitter>
            <GridSplitter x:Name="gridSplitter2" HorizontalAlignment="Left" Height="14"
Margin="296,720.308,0,0" VerticalAlignment="Top" Width="328" RenderTransformOrigin="0.5,0.5"
Grid.Row="1">
                <GridSplitter.RenderTransform>
                    <TransformGroup>
                        <ScaleTransform ScaleY="-1"/>
                        <SkewTransform/>
                        <RotateTransform/>
                        <TranslateTransform/>
                    </TransformGroup>
```

```
                </GridSplitter.RenderTransform>
            </GridSplitter>
            <GridSplitter x:Name="gridSplitter3" HorizontalAlignment="Left" Height="496"
Margin="293,397.308,0,0" VerticalAlignment="Top" Width="13" RenderTransformOrigin="0.5,0.5"
Grid.Row="1">
                <GridSplitter.RenderTransform>
                    <TransformGroup>
                        <ScaleTransform ScaleX="-1"/>
                        <SkewTransform/>
                        <RotateTransform/>
                        <TranslateTransform/>
                    </TransformGroup>
                </GridSplitter.RenderTransform>
            </GridSplitter>
            <Label x:Name="samplesReceivedLabel" Content="Samples Received"
HorizontalAlignment="Left" Margin="305,734.308,0,0" VerticalAlignment="Top"
BorderThickness="0" FontFamily="Calibri Light" FontSize="18" Height="32" Width="139"
Grid.Row="1"/>
            <TextBox x:Name="dataReceivedCounter" HorizontalAlignment="Left" Height="29"
Margin="494,737.308,0,0" BorderThickness="1" TextWrapping="Wrap" Width="102"
RenderTransformOrigin="0.446,0.621" TextAlignment="Center" Text="0" IsReadOnly="True"
VerticalAlignment="Top" TextOptions.TextHintingMode="Animated" FontFamily="Calibri"
FontSize="16" Grid.Row="1"/>
            <Label x:Name="samplesReceivedLabel_Copy" Content="Signal to Noise Ratio"
HorizontalAlignment="Left" Margin="305,791.308,0,0" VerticalAlignment="Top"
BorderThickness="0" FontFamily="Calibri Light" FontSize="18" Height="32" Width="157"
Grid.Row="1"/>
            <TextBox x:Name="snrBox" HorizontalAlignment="Left" Height="29"
Margin="494,794.308,0,0" TextWrapping="Wrap" Width="102" RenderTransformOrigin="0.446,0.621"
TextAlignment="Center" Text="0" IsReadOnly="True" VerticalAlignment="Top"
TextOptions.TextHintingMode="Animated" FontFamily="Calibri" FontSize="16" Grid.Row="1"/>
            <TextBox x:Name="rssiBox" HorizontalAlignment="Left" Height="29"
Margin="494,848.308,0,0" TextWrapping="Wrap" Width="102" RenderTransformOrigin="0.446,0.621"
TextAlignment="Center" Text="0" IsReadOnly="True" VerticalAlignment="Top"
TextOptions.TextHintingMode="Animated" FontFamily="Calibri" FontSize="16" Grid.Row="1"/>
            <Label x:Name="samplesReceivedLabel_Copy1" Content="RSSI"
HorizontalAlignment="Left" Margin="305,842.308,0,0" VerticalAlignment="Top"
BorderThickness="0" FontFamily="Calibri Light" FontSize="18" Height="32" Width="40"
Grid.Row="1"/>
            <GridSplitter x:Name="gridSplitter4" HorizontalAlignment="Stretch" Height="8"
Margin="619,622.308,0,0" VerticalAlignment="Top" Grid.Row="1"/>
            <Label x:Name="dataNoDataLabel" Content="NO DATA" HorizontalContentAlignment
="Center" HorizontalAlignment="Left" Margin="0,593.308,0,0" VerticalAlignment="Top"
FontSize="20" Foreground="Red" FontWeight="Bold" Height="37" Width="286" Grid.Row="1"/>
            <Label x:Name="label1" Content="Longitude" HorizontalAlignment="Left"
Margin="631,649.308,0,0" VerticalAlignment="Top" Height="46" Width="120" FontSize="17.333"
Grid.Row="1"/>
            <TextBox x:Name="longSent" HorizontalAlignment="Left" Height="30"
Margin="756,649.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="195"
FontSize="18.667" TextAlignment="Center" Grid.Row="1"/>
            <Label x:Name="label1_Copy" Content="Latitude" HorizontalAlignment="Left"
Margin="631,708.308,0,0" VerticalAlignment="Top" Height="46" Width="120" FontSize="17.333"
Grid.Row="1"/>
```

```
            <TextBox x:Name="latSent" HorizontalAlignment="Left" Height="29"
Margin="756,710.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="195"
FontSize="18.667" TextAlignment="Center" Grid.Row="1"/>
            <Button x:Name="sendCoordinates" Content="Set Coordinates"
HorizontalAlignment="Left" Margin="636,754.308,0,0" VerticalAlignment="Top" Width="310"
Height="57" FontSize="18.667" FontWeight="Bold" Click="sendCoordinates_Click" Grid.Row="1" />
            <Button x:Name="returnToHome" Content="Return to Home" HorizontalAlignment="Left"
Margin="636,829.308,0,0" VerticalAlignment="Top" Width="310" Height="57" FontSize="18.667"
FontWeight="Bold" Click="returnToHome_Click" Grid.Row="1"/>
            <GridSplitter x:Name="gridSplitter4_Copy" HorizontalAlignment="Stretch"
Height="274" Margin="958,622.308,553,0" VerticalAlignment="Top" Grid.Row="1"/>
            <TextBox x:Name="qfeInput" HorizontalAlignment="Left" Height="46"
Margin="981,653.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="147"
FontSize="18.667" TextAlignment="Center" Grid.Row="1"/>
            <Label x:Name="label1_Copy1" Content="mBars" HorizontalAlignment="Left"
Margin="1133,651.308,0,0" VerticalAlignment="Top" Height="46" Width="66" FontSize="17.333"
Grid.Row="1"/>
            <Button x:Name="setQFE" Content="Set QFE" HorizontalAlignment="Left"
Margin="1204,646.308,0,0" VerticalAlignment="Top" Width="96" Height="57" FontSize="18.667"
FontWeight="Bold" Click="setQFE_Click" Grid.Row="1"/>
            <Button x:Name="setQFECurrent" Content="Set QFE as current"
HorizontalAlignment="Left" Margin="1320,646.308,0,0" VerticalAlignment="Top" Width="170"
Height="57" FontSize="18.667" FontWeight="Bold" Click="setQFECurrent_Click" Grid.Row="1"/>
            <GridSplitter x:Name="gridSplitter4_Copy1" HorizontalAlignment="Stretch"
Height="14" Margin="971,725.308,0,0" VerticalAlignment="Top" Grid.Row="1"/>
            <Button x:Name="servoArm" Margin="981,754.308,290,10" Click="servoArm_Click"
Grid.Row="1">
                <TextBlock x:Name="t1" TextWrapping="Wrap" Text="Arm Parachute Release
Mechanism" FontSize="18.667" TextAlignment="Center" FontWeight="Bold"/>
            </Button>
            <Button x:Name="releaseServo" Margin="1281,754.308,2,10"
Click="releaseServo_Click" Grid.Row="1">
                <TextBlock x:Name ="t2" TextWrapping="Wrap" Text="Release Parachute"
FontSize="18.667" TextAlignment="Center" FontWeight="Bold"/>
            </Button>
            <GridSplitter x:Name="gridSplitter5" HorizontalAlignment="Left" Height="16"
Margin="626,324.308,0,0" VerticalAlignment="Top" Width="898" Grid.Row="1"/>
            <Label x:Name="label2" Content="Internal Temperature / °C"
HorizontalAlignment="Left" Margin="651,14.308,0,0" VerticalAlignment="Top" Height="26"
Width="144" Grid.Row="1"/>
            <Label x:Name="label2_Copy" Content="Barometric Pressure / mBar (hPa)"
HorizontalAlignment="Left" Margin="651,56.308,0,0" VerticalAlignment="Top" Height="26"
Width="184" Grid.Row="1"/>
            <Label x:Name="label2_Copy1" Content="External Temperature / °C"
HorizontalAlignment="Left" Margin="651,97.308,0,0" VerticalAlignment="Top" Height="26"
Width="146" Grid.Row="1"/>
            <Label x:Name="label2_Copy2" Content="Relative Humidity / %"
HorizontalAlignment="Left" Margin="651,139.308,0,0" VerticalAlignment="Top" Height="26"
Width="123" Grid.Row="1"/>
            <Label x:Name="label2_Copy3" Content="Dew Point / °C" HorizontalAlignment="Left"
Margin="651,222.308,0,0" VerticalAlignment="Top" Height="26" Width="86" Grid.Row="1"/>
            <Label x:Name="label2_Copy4" Content="Relative Agricultural Potential / Arbitrary
Units" HorizontalAlignment="Left" Margin="651,265.308,0,0" VerticalAlignment="Top"
Height="26" Width="253" Grid.Row="1"/>
```

```
            <Label x:Name="label2_Copy5" Content="Altitude / m" HorizontalAlignment="Left"
Margin="651,181.308,0,0" VerticalAlignment="Top" Height="26" Width="74" Grid.Row="1"/>
            <lvc:LineChart Series="{Binding Series}" Hoverable="True"
Margin="1003,411,283,279" Grid.Row="1">
                <lvc:LineChart.AxisX>
                    <lvc:Axis LabelFormatter="{Binding XFormatter}" Separator="{x:Static
lvc:DefaultAxes.CleanSeparator}"/>
                </lvc:LineChart.AxisX>
                <lvc:LineChart.AxisY>
                    <lvc:Axis LabelFormatter="{Binding YFormatter}"></lvc:Axis>
                </lvc:LineChart.AxisY>
            </lvc:LineChart>
            <lvc:LineChart Series="{Binding Series2}" Hoverable="True"
Margin="1246,411,10,279" Grid.Row="1">
                <lvc:LineChart.AxisX>
                    <lvc:Axis LabelFormatter="{Binding XFormatter2}" Separator="{x:Static
lvc:DefaultAxes.CleanSeparator}"/>
                </lvc:LineChart.AxisX>
                <lvc:LineChart.AxisY>
                    <lvc:Axis LabelFormatter="{Binding YFormatter2}"></lvc:Axis>
                </lvc:LineChart.AxisY>
            </lvc:LineChart>
            <GridSplitter x:Name="gridSplitter6" HorizontalAlignment="Left" Height="304"
Margin="985,324.308,0,0" VerticalAlignment="Top" Width="12" RenderTransformOrigin="0.5,0.5"
Grid.Row="1">
                <GridSplitter.RenderTransform>
                    <TransformGroup>
                        <ScaleTransform ScaleX="-1"/>
                        <SkewTransform/>
                        <RotateTransform/>
                        <TranslateTransform/>
                    </TransformGroup>
                </GridSplitter.RenderTransform>
            </GridSplitter>
            <TextBox x:Name="rapBox" HorizontalAlignment="Left" Height="26"
Margin="928,265.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnlyCaretVisible="True" IsReadOnly="True" Grid.Row="1"/>
            <TextBox x:Name="dewPointBox" HorizontalAlignment="Left" Height="26"
Margin="928,222.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnlyCaretVisible="True" IsReadOnly="True" Grid.Row="1"/>
            <TextBox x:Name="altitudeBox" HorizontalAlignment="Left" Height="26"
Margin="928,181.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnlyCaretVisible="True" IsReadOnly="True" Grid.Row="1"/>
            <TextBox x:Name="humidityBox" HorizontalAlignment="Left" Height="26"
Margin="928,139.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnlyCaretVisible="True" IsReadOnly="True" Grid.Row="1"/>
            <TextBox x:Name="externalTempBox" HorizontalAlignment="Left" Height="26"
Margin="928,98.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnlyCaretVisible="True" IsReadOnly="True" Grid.Row="1"/>
            <TextBox x:Name="pressureBox" HorizontalAlignment="Left" Height="26"
Margin="928,57.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnlyCaretVisible="True" IsReadOnly="True" Grid.Row="1"/>
```

```
            <TextBox x:Name="internalTempBox" HorizontalAlignment="Left" Height="26"
Margin="928,14.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnly="True" Grid.Row="1"/>
            <Grid Margin="1096,40.308,10,818" Grid.Row="1">
                <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="38"
Stroke="Black" VerticalAlignment="Top" Width="418" RenderTransformOrigin="0.5,0.5">
                    <Rectangle.RenderTransform>
                        <TransformGroup>
                            <ScaleTransform ScaleY="-1"/>
                            <SkewTransform/>
                            <RotateTransform/>
                            <TranslateTransform/>
                        </TransformGroup>
                    </Rectangle.RenderTransform>
                </Rectangle>
                <Label x:Name="label2_Copy7" Content="Latitude / °N"
HorizontalAlignment="Left" Margin="2,6,0,0" VerticalAlignment="Top" Height="26" Width="78"/>
                <TextBox x:Name="latBox" HorizontalAlignment="Left" Height="26"
Margin="296,6,0,0" TextWrapping="Wrap" Text="0.000000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnly="True"/>
            </Grid>
            <Grid Margin="1096,114.308,10,744" Grid.Row="1">
                <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="38"
Stroke="Black" VerticalAlignment="Top" Width="418" RenderTransformOrigin="0.5,0.5"
Margin="0,-1,0,0">
                    <Rectangle.RenderTransform>
                        <TransformGroup>
                            <ScaleTransform ScaleY="-1"/>
                            <SkewTransform/>
                            <RotateTransform/>
                            <TranslateTransform/>
                        </TransformGroup>
                    </Rectangle.RenderTransform>
                </Rectangle>
                <Label x:Name="label2_Copy9" Content="Altitude / m"
HorizontalAlignment="Left" Margin="2,4,0,0" VerticalAlignment="Top" Height="26" Width="74"/>
                <TextBox x:Name="altitudeGPSBox" HorizontalAlignment="Left" Height="26"
Margin="296,6,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnly="True"/>
            </Grid>
            <Grid Margin="1096,152.308,10,706" Grid.Row="1">
                <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="38"
Stroke="Black" VerticalAlignment="Top" Width="418" RenderTransformOrigin="0.5,0.5"
Margin="0,-2,0,0">
                    <Rectangle.RenderTransform>
                        <TransformGroup>
                            <ScaleTransform ScaleY="-1"/>
                            <SkewTransform/>
                            <RotateTransform/>
                            <TranslateTransform/>
                        </TransformGroup>
                    </Rectangle.RenderTransform>
                </Rectangle>
```

```
                <Label x:Name="label2_Copy10" Content="Heading / °"
HorizontalAlignment="Left" Margin="2,5,0,0" VerticalAlignment="Top" Height="26" Width="70"/>
                <TextBox x:Name="headingBox" HorizontalAlignment="Left" Height="26"
Margin="296,5,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnly="True"/>
            </Grid>
            <Grid Margin="1096,189.308,10,669" Grid.Row="1">
                <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="38"
Stroke="Black" VerticalAlignment="Top" Width="418" RenderTransformOrigin="0.5,0.5"
Margin="0,-2,0,0">
                    <Rectangle.RenderTransform>
                        <TransformGroup>
                            <ScaleTransform ScaleY="-1"/>
                            <SkewTransform/>
                            <RotateTransform/>
                            <TranslateTransform/>
                        </TransformGroup>
                    </Rectangle.RenderTransform>
                </Rectangle>
                <Label x:Name="label2_Copy11" Content="Pitch / °" HorizontalAlignment="Left"
Margin="2,5,0,0" VerticalAlignment="Top" Height="26" Width="70"/>
                <TextBox x:Name="pitchBox" HorizontalAlignment="Left" Height="26"
Margin="296,5,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnly="True"/>
            </Grid>
            <Grid Margin="1096,226.308,10,632" Grid.Row="1">
                <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="38"
Stroke="Black" VerticalAlignment="Top" Width="418" RenderTransformOrigin="0.5,0.5"
Margin="0,-3,0,0">
                    <Rectangle.RenderTransform>
                        <TransformGroup>
                            <ScaleTransform ScaleY="-1"/>
                            <SkewTransform/>
                            <RotateTransform/>
                            <TranslateTransform/>
                        </TransformGroup>
                    </Rectangle.RenderTransform>
                </Rectangle>
                <Label x:Name="label2_Copy12" Content="Roll / °" HorizontalAlignment="Left"
Margin="2,4,0,0" VerticalAlignment="Top" Height="26" Width="45"/>
                <TextBox x:Name="rollBox" HorizontalAlignment="Left" Height="26"
Margin="296,5,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnly="True"/>
            </Grid>
            <Grid Margin="1096,264.308,10,597" Grid.Row="1">
                <Rectangle Fill="#FFFFFF" HorizontalAlignment="Left" Height="38"
Stroke="Black" VerticalAlignment="Top" Width="418" RenderTransformOrigin="0.5,0.5"
Margin="0,-4,0,0">
                    <Rectangle.RenderTransform>
                        <TransformGroup>
                            <ScaleTransform ScaleY="-1"/>
                            <SkewTransform/>
                            <RotateTransform/>
                            <TranslateTransform/>
```

```
                    </TransformGroup>
                </Rectangle.RenderTransform>
            </Rectangle>
            <Label x:Name="label2_Copy13" Content="Time / hours : minutes : seconds"
HorizontalAlignment="Left" Margin="2,4,0,0" VerticalAlignment="Top" Height="26" Width="181"/>
            <TextBox x:Name="timeBox" HorizontalAlignment="Left" Height="26"
Margin="296,4,0,0" TextWrapping="Wrap" Text="00:00:00" VerticalAlignment="Top" Width="108"
FontSize="13.333" IsReadOnly="True"/>
        </Grid>
        <Label x:Name="label2_Copy14" Content="Ground Temperature:"
HorizontalAlignment="Left" Margin="1003,345.308,0,0" VerticalAlignment="Top" Height="26"
Width="125" Grid.Row="1"/>
        <TextBox x:Name="groundTempBox" HorizontalAlignment="Left" Height="26"
Margin="1133,347.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top"
Width="108" FontSize="13.333" IsReadOnlyCaretVisible="True" IsReadOnly="True" Grid.Row="1"/>
        <Label x:Name="label2_Copy15" Content="Ground Air Pressure:"
HorizontalAlignment="Left" Margin="1275,345.308,0,0" VerticalAlignment="Top" Height="26"
Width="131" Grid.Row="1"/>
        <GridSplitter x:Name="gridSplitter5_Copy" HorizontalAlignment="Left" Height="16"
Margin="991,390.308,0,0" VerticalAlignment="Top" Width="533" Grid.Row="1"/>
        <ComboBox x:Name="serialPortSelectionBox" HorizontalAlignment="Left"
Margin="10,397.308,0,0" VerticalAlignment="Top" Width="128" Height="47" Grid.Row="1"/>
        <Button x:Name="connectSerialButton" Content="Connect" HorizontalAlignment="Left"
Margin="10,449.308,0,0" VerticalAlignment="Top" Width="259" Height="57" FontSize="18.667"
FontWeight="Bold" Click="connectSerialButton_Click" Grid.Row="1"/>
        <Button x:Name="disconnectSerialButton" Content="Disconnect"
HorizontalAlignment="Left" Margin="10,511.308,0,0" VerticalAlignment="Top" Width="259"
Height="57" FontSize="18.667" FontWeight="Bold" Click="disconnectSerialButton_Click"
Grid.Row="1"/>
        <GridSplitter x:Name="gridSplitter3_Copy" HorizontalAlignment="Left" Height="12"
Margin="1,573.308,0,0" VerticalAlignment="Top" Width="299" RenderTransformOrigin="0.5,0.5"
Grid.Row="1">
            <GridSplitter.RenderTransform>
                <TransformGroup>
                    <ScaleTransform ScaleX="-1"/>
                    <SkewTransform/>
                    <RotateTransform/>
                    <TranslateTransform/>
                </TransformGroup>
            </GridSplitter.RenderTransform>
        </GridSplitter>
        <TextBox x:Name="groundPressure" HorizontalAlignment="Left" Height="26"
Margin="1406,347.308,0,0" TextWrapping="Wrap" Text="0.000" VerticalAlignment="Top"
Width="108" FontSize="13.333" IsReadOnlyCaretVisible="True" IsReadOnly="True" Grid.Row="1"/>
        <ComboBox x:Name="serialPortBaudRate" HorizontalAlignment="Left"
Margin="143,397.308,0,0" VerticalAlignment="Top" Width="126" Height="47" Grid.Row="1"/>
        <cam:Webcam Name="WebcamCtrl" Margin="10,6,908,514" HorizontalAlignment="Stretch"
VerticalAlignment="Stretch" Grid.Row="1"/>

        <StackPanel Margin="0,327,1236,0" Orientation="Vertical"
HorizontalAlignment="Center" VerticalAlignment="Top" Grid.Row="1">
            <StackPanel Orientation="Horizontal" HorizontalAlignment="Center"
Height="22"/>
```

```
                <StackPanel Orientation="Horizontal" HorizontalAlignment="Center" Height="23"
Margin="0,10,0,0"/>
            </StackPanel>

            <Grid Margin="643,0,553,284" HorizontalAlignment="Center"
VerticalAlignment="Bottom" Height="154" Width="328" RenderTransformOrigin="0.5,0.667"
Grid.Row="1">
                <Button Content="Start Recording" Height="43" Width="164" Margin="0,59,0,0"
HorizontalAlignment="Left"
                    VerticalAlignment="Top" x:Name="StartRecordingButton"
Click="StartRecordingButton_Click"/>
                <Button Content="Stop Recording" Height="43" Width="151" Margin="0,59,0,0"
HorizontalAlignment="Right"
                    VerticalAlignment="Top" x:Name="StopRecordingButton"
Click="StopRecordingButton_Click"/>
                <Button Content="Stop Capture" Height="44" Width="151"
HorizontalAlignment="Right"
                    VerticalAlignment="Top" x:Name="StopCaptureButton"
Click="StopCaptureButton_Click" Margin="0,10,0,0"/>
                <Button Content="Start Capture" Height="44" Width="164"
HorizontalAlignment="Left"
                    VerticalAlignment="Top" x:Name="StartCaptureButton"
Click="StartCaptureButton_Click" Margin="0,10,0,0"/>
                <Button Content="Take Snapshot" Height="47" Width="328" Margin="0,107,0,0"
HorizontalAlignment="Center"
                    VerticalAlignment="Top" x:Name="TakeSnapshotButton"
Click="TakeSnapshotButton_Click"/>
            </Grid>
            <TextBox x:Name="SerialOutputs" VerticalScrollBarVisibility="Auto"
HorizontalScrollBarVisibility="Auto" HorizontalAlignment="Left" Height="240"
Margin="10,646.308,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="259"
Grid.Row="1"/>
            <ComboBox Height="22" Width="200" HorizontalAlignment="Left"
VerticalAlignment="Top"
              x:Name="VideoDevicesComboBox" Margin="756,359,0,0" Grid.Row="1"/>
            <TextBlock Height="22" Width="78" HorizontalAlignment="Left"
VerticalAlignment="Top"
              Text="Video Device" TextAlignment="Left" FlowDirection="LeftToRight"
Margin="647,360,0,0" Grid.Row="1" />
            <TextBlock Height="22" Width="78" HorizontalAlignment="Left"
VerticalAlignment="Top"
              Text="Audio Device" TextAlignment="Left" FlowDirection="LeftToRight"
Margin="647,411,0,0" Grid.Row="1"/>
            <ComboBox HorizontalAlignment="Left" Height="23" Width="200"
VerticalAlignment="Bottom"
              x:Name="AudioDevicesComboBox" Margin="756,0,0,462" Grid.Row="1" />


        </Grid>
    </Viewbox>
</Window>
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using System.IO;
using System.IO.Ports;
using System.Timers;
using System.Threading;
using System.ComponentModel;
using Microsoft.Maps.MapControl.WPF;
using AForge;
using WebcamControl;
using Microsoft.Expression.Encoder;
using Microsoft.Expression.Encoder.Devices;
using LiveCharts;
using System.Windows.Threading;

namespace Halley__Version_1_
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public class tempGraph
    {
        public double temperature { get; set; }
        public DateTime time { get; set; }
    }
    public class pressureGraph
    {
        public double pressure { get; set; }
        public DateTime time { get; set; }
    }
    public class data
    {
        public string input {set; get;}
        public int snr { set; get; }
        public int rssi { set; get; }
        public int sampleNumber { set; get; }
        public double internal_temperature { set; get; }
        public double barometric_pressure { set; get; }
```

```
    public double external_temperature { set; get; }
    public double humidity { set; get; }
    public int hours { set; get; }
    public int minutes { set; get; }
    public int seconds { set; get; }
    public int gpsFix { set; get; }
    public double longitude { set; get; }
    public double latitude { set; get; }
    public double altitude { set; get; }
    public double heading { set; get; }
    public double pitch { set; get; }
    public double roll { set; get; }
    public double agriculturalViability { set; get; }
    public double pressureAtms { set; get; }
    public double altitudeFromPressure { set; get; }
    public double dewPoint { set; get; }
    public double groundBarometricPressure { set; get; }
    public double groundTemperature { set; get; }
    public double groundLatitude { set; get; }
    public double groundLongitude { set; get; }
    public double groundAltitude { set; get; }
}

public partial class MainWindow : Window
{
    List<data> allData = new List<data>();
    const int expectedLength = 26;
    StreamWriter writer = File.AppendText("CANSAT-DATA.txt");
    public SerialPort serial = new SerialPort();
    public bool connected = false;
    public double numberOfItems = 0;
    public double longAverage = 0;
    public double latAverage = 0;
    public int counter = 0;
    public int counter2;
    public bool newAvailable = false;
    System.Timers.Timer timer = new System.Timers.Timer();
    public bool elapsed = false;
    public const int defaultBaudRate = 115200;
    public readonly BackgroundWorker bw = new BackgroundWorker();

    private readonly DispatcherTimer _timer2;
    public Func<double, string> XFormatter2 { get; set; }
    public Func<double, string> YFormatter2 { get; set; }
    public SeriesCollection Series2 { get; set; }

    private readonly DispatcherTimer _timer;
    private DateTime _currentDate = DateTime.Now;
    public SeriesCollection Series { get; set; }
    public Func<double, string> YFormatter { get; set; }
    public Func<double, string> XFormatter { get; set; }
    private void TimerOnTick(object sender, EventArgs eventArgs)
    {
        _currentDate= DateTime.Now;
```

```
    if(Series[0].Values.Count > 10)
    {
        Series[0].Values.RemoveAt(0);
    }
    if (Series[1].Values.Count > 10)
    {
        Series[1].Values.RemoveAt(0);
    }
    if (Series[2].Values.Count > 10)
    {
        Series[2].Values.RemoveAt(0);
    }
    if (allData.Count() > 0)
    {
        Series[0].Values.Add(new tempGraph
        {
            temperature = allData[allData.Count() - 1].external_temperature,
            time = _currentDate
        }
        );
        Series[1].Values.Add(new tempGraph
        {
            temperature = allData[allData.Count() - 1].internal_temperature,
            time = _currentDate
        }
        );
        Series[2].Values.Add(new tempGraph
        {
            temperature = allData[allData.Count() - 1].groundTemperature,
            time = _currentDate
        }
        );
    }
    /*foreach (var series in Series)
    {
        if (series.Values.Count > 10) series.Values.RemoveAt(0);
        series.Values.Add(new tempGraph
        {
            temperature = _,
            time = _currentDate
        });
    }*/
}
private void Timer2OnTick(object sender, EventArgs eventArgs)
{
    _currentDate = DateTime.Now;
    if (Series2[0].Values.Count > 10)
    {
        Series2[0].Values.RemoveAt(0);
    }
    if (Series2[1].Values.Count > 10)
    {
        Series2[1].Values.RemoveAt(0);
    }
```

```
            if (allData.Count() > 0)
            {
                Series2[0].Values.Add(new pressureGraph
                {
                    pressure = allData[allData.Count() - 1].barometric_pressure,
                    time = _currentDate
                }
                );
                Series2[1].Values.Add(new pressureGraph
                {
                    pressure = allData[allData.Count() - 1].groundBarometricPressure,
                    time = _currentDate
                }
                );
            }
        }
        public MainWindow()
        {
            InitializeComponent();
            timer.Start();
            setUpBaudRates();

            var config = new SeriesConfiguration<tempGraph>();
            config.Y(model => model.temperature);
            config.X(model => model.time.ToOADate());
            Series = new SeriesCollection(config) { new LineSeries { Values = new
ChartValues<tempGraph>() } };
            Series.Add(new LineSeries { Values = new ChartValues<tempGraph>() });
            Series.Add(new LineSeries { Values = new ChartValues<tempGraph>() });
            Series[0].Title = "External Temperature";
            Series[1].Title = "Internal Temperature";
            Series[2].Title = "Ground Temperature";
            XFormatter = val => DateTime.FromOADate(val).ToString("mm.ss");
            YFormatter = val => val + " °";
            DataContext = this;
            _timer = new DispatcherTimer { Interval = TimeSpan.FromMilliseconds(1000) };
            _timer.Tick += TimerOnTick;
            _timer.Start();

            var config2 = new SeriesConfiguration<pressureGraph>();
            config2.Y(model => model.pressure);
            config2.X(model => model.time.ToOADate());
            Series2 = new SeriesCollection(config2) { new LineSeries { Values = new
ChartValues<pressureGraph>() } };
            Series2.Add(new LineSeries { Values = new ChartValues<pressureGraph>() });
            Series2[0].Title = "Barometric Pressure";
            Series2[1].Title = "Ground Barometric Pressure";
            XFormatter2 = val => DateTime.FromOADate(val).ToString("mm.ss");
            YFormatter2 = val => val + "hPa";
            DataContext = this;
            _timer2 = new DispatcherTimer { Interval = TimeSpan.FromMilliseconds(1000) };
            _timer2.Tick += Timer2OnTick;
            _timer2.Start();
            this.Show();
```

```
        setUpPossibleInputsForSerialPort();
        ///AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        Binding binding_1 = new Binding("SelectedValue");
        binding_1.Source = VideoDevicesComboBox;
        WebcamCtrl.SetBinding(Webcam.VideoDeviceProperty, binding_1);

        Binding binding_2 = new Binding("SelectedValue");
        binding_2.Source = AudioDevicesComboBox;
        WebcamCtrl.SetBinding(Webcam.AudioDeviceProperty, binding_2);

        // Create directory for saving video files
        string videoPath = @"C:\VideoClips";

        if (!Directory.Exists(videoPath))
        {
            Directory.CreateDirectory(videoPath);
        }
        // Create directory for saving image files
        string imagePath = @"C:\WebcamSnapshots";

        if (!Directory.Exists(imagePath))
        {
            Directory.CreateDirectory(imagePath);
        }
        //AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        // Set some properties of the Webcam control
        WebcamCtrl.VideoDirectory = videoPath;
        WebcamCtrl.ImageDirectory = imagePath;
        WebcamCtrl.FrameRate = 72;
        WebcamCtrl.FrameSize = new System.Drawing.Size(1080, 720);

        // Find available a/v devices
        var vidDevices = EncoderDevices.FindDevices(EncoderDeviceType.Video);
        var audDevices = EncoderDevices.FindDevices(EncoderDeviceType.Audio);
        VideoDevicesComboBox.ItemsSource = vidDevices;
        AudioDevicesComboBox.ItemsSource = audDevices;
        VideoDevicesComboBox.SelectedIndex = 0;
        AudioDevicesComboBox.SelectedIndex = 0;
        releaseServo.IsEnabled = false;
        servoArm.IsEnabled = false;
        t1.IsEnabled = false;
        t2.IsEnabled = false;
        bw.DoWork += Bw_DoWork;
        bw.RunWorkerCompleted += Bw_RunWorkerCompleted;
        bw.RunWorkerAsync();

    }
    private void StartCaptureButton_Click(object sender, RoutedEventArgs e)
    {
        try
        {
            // Display webcam video
            WebcamCtrl.StartPreview();
        }
```

```
            catch
            {
                MessageBox.Show("Device is in use by another application");
            }
        }

        private void StopCaptureButton_Click(object sender, RoutedEventArgs e)
        {
            // Stop the display of webcam video.
            WebcamCtrl.StopPreview();
        }

        private void StartRecordingButton_Click(object sender, RoutedEventArgs e)
        {
            // Start recording of webcam video to harddisk.
            WebcamCtrl.StartRecording();
        }

        private void StopRecordingButton_Click(object sender, RoutedEventArgs e)
        {
            // Stop recording of webcam video to harddisk.
            WebcamCtrl.StopRecording();
        }

        private void TakeSnapshotButton_Click(object sender, RoutedEventArgs e)
        {
            // Take snapshot of webcam video.
            WebcamCtrl.TakeSnapshot();
        }

        public void setUpPossibleInputsForSerialPort()
        {
            bool doIt = true;
            if (serialPortSelectionBox.IsDropDownOpen == true || serialPortSelectionBox.Text
!= "")
                doIt = false;
            if (doIt)
            {
                serialPortSelectionBox.Items.Clear();
                string[] portsAvailable;
                portsAvailable = SerialPort.GetPortNames();
                for (int i = 0; i < portsAvailable.Length; i++)
                    serialPortSelectionBox.Items.Add(portsAvailable[i]);

                if(portsAvailable.Length == 1)
                {
                    serialPortSelectionBox.Text = portsAvailable[0];
                    serialPortBaudRate.Text = Convert.ToString(defaultBaudRate);
                    connectToSerial();
                }
            }
        }
        public void setUpBaudRates()
        {
```

```
            bool doIt = true;
            if (serialPortBaudRate.Text != "" || serialPortBaudRate.IsDropDownOpen == true)
                doIt = false;
            if (doIt)
            {
                serialPortBaudRate.Items.Clear();
                if (SerialPort.GetPortNames().Length > 0)
                {
                    int[] possibleBaudRates =
                    {
                4800, 9600, 19200, 38400, 57600, 115200
                 };
                    for (int i = 0; i < possibleBaudRates.Length; i++)
                    {
                        serialPortBaudRate.Items.Add(possibleBaudRates[i]);
                    }
                }
                else
                {
                    serialPortBaudRate.IsEnabled = false;
                    serialPortSelectionBox.IsEnabled = false;
                    connectSerialButton.IsEnabled = false;
                }
            }
        }
        public void connectToSerial()
        {
            if (connected)
                MessageBox.Show("Please disconnect from other devices before connecting to
this one");
            if (serialPortSelectionBox.Text == "")
                MessageBox.Show("You must choose a valid port");
            else
            {
                serial.PortName = serialPortSelectionBox.Text;
                serial.BaudRate = Convert.ToInt32(serialPortBaudRate.Text);
                serial.Open();
                connected = true;
                connectSerialButton.IsEnabled = false;
                changeButtons();
                _timer.Start();
                _timer2.Start();
            }
            serial.DataReceived += new
System.IO.Ports.SerialDataReceivedEventHandler(receive);
            changeButtons();
        }

        private void connectSerialButton_Click(object sender, RoutedEventArgs e)
        {
            connectToSerial();
        }
        private void Bw_DoWork(object sender, DoWorkEventArgs e)
        {
```

```
    Thread.Sleep(50);
}
/*
// TEST SYSTEM - changing something in the UI
public void changeStuff()
{
    dataNoDataLabel.Content = Convert.ToString(counter);
    counter++;
}
*/
private void Bw_RunWorkerCompleted(object sender, RunWorkerCompletedEventArgs e)
{
    // DO STUFF HERE
    if(!connected)
    {
        setUpPossibleInputsForSerialPort();

        Thread.Sleep(50);
        setUpBaudRates();
        dataNoDataLabel.Content = "NO DATA";
        dataNoDataLabel.Foreground = new SolidColorBrush(Colors.Red);
        Thread.Sleep(50);
        disableButtons();

        Thread.Sleep(50);
    }
    else
    {
        string input = "";
        bool newReceived = false;
        //Check for receiving data
        /*try
        {
            input = serial.ReadLine();
            writer.WriteLine(input);
            newReceived = true;
        }
        catch (TimeoutException)
        {
            MessageBox.Show("HELLO");
            newReceived = false;
            counter2++;
            dataNoDataLabel.Content = "NO DATA";
            dataNoDataLabel.Foreground = new SolidColorBrush(Colors.Red);
            if (counter2 > 10)
            {
                dataNoDataLabel.Content = "NO DATA";
                dataNoDataLabel.Foreground = new SolidColorBrush(Colors.Red);
            }
        }
        */
        if(newAvailable)
        {
            input = serial.ReadLine();
```

```
            writer.WriteLine(input);
            newReceived = true;
        }
        else
        {

            if (elapsed)
            {
                dataNoDataLabel.Content = "NO DATA";
                dataNoDataLabel.Foreground = new SolidColorBrush(Colors.Red);
            }

        }
        newAvailable = false;
        Thread.Sleep(50);
        //DECODE DATA
        bool valid = false;
        if(newReceived)
        {
            SerialOutputs.AppendText("\n");
            SerialOutputs.AppendText("\n");
            SerialOutputs.AppendText(input);
            SerialOutputs.CaretIndex = SerialOutputs.Text.Length;
            SerialOutputs.ScrollToEnd();
            valid = checkValid(input);
            Thread.Sleep(50);
            if (valid)
            {
                decodeData(input);
                Thread.Sleep(50);
                counter++;
                dataNoDataLabel.Content = "RECEIVING DATA";
                dataNoDataLabel.Foreground = new SolidColorBrush(Colors.Green);
                elapsed = false;
                timer.Stop();
                timer.Start();
                timer.Interval = 2000;
                timer.AutoReset = true;
                timer.Elapsed += Timer_Elapsed;
            }
            else
            {
                dataNoDataLabel.Content = "NO DATA";
            }
        }
        if (allData.Count() > 0)
        {
            updateThings();
        }
    }
    bw.RunWorkerAsync();
}

private void Timer_Elapsed(object sender, ElapsedEventArgs e)
```

```
{
    elapsed = true;
}

public bool checkValid(string input)
{
    string[] inputs = input.Split(',');
    if (inputs.Length != 26)
        return false;
    else
        return true;
}
public void updateThings()
{
    /*
    for reference....
     Location newTest = new Location();
     newTest.Latitude = 0;
     newTest.Longitude = 0;
     Pushpin pin = new Pushpin();
     map.Children.Add(pin);
     */
    int temp = allData.Count() - 1;
    internalTempBox.Text = Convert.ToString(allData[temp].internal_temperature);
    pressureBox.Text = Convert.ToString(allData[temp].barometric_pressure);
    externalTempBox.Text = Convert.ToString(allData[temp].external_temperature);
    humidityBox.Text = Convert.ToString(allData[temp].humidity);
    altitudeBox.Text = Convert.ToString(allData[temp].altitudeFromPressure);
    dewPointBox.Text = Convert.ToString(allData[temp].dewPoint);
    rapBox.Text = Convert.ToString(allData[temp].agriculturalViability);
    satsBox.Text = Convert.ToString(allData[temp].gpsFix);
    bool temp2 = false;
    if (satsBox.Text != "0")
    {
        longBox.IsEnabled = true;
        latBox.IsEnabled = true;
        altitudeGPSBox.IsEnabled = true;
        temp2 = true;
        longBox.Text = Convert.ToString(allData[temp].longitude);
        latBox.Text = Convert.ToString(allData[temp].latitude);
        altitudeGPSBox.Text = Convert.ToString(allData[temp].altitude);
    }
    else
    {
        longBox.IsEnabled = false;
        latBox.IsEnabled = false;
        altitudeGPSBox.IsEnabled = false;
    }
    headingBox.Text = Convert.ToString(allData[temp].heading);
    pitchBox.Text = Convert.ToString(allData[temp].pitch);
    rollBox.Text = Convert.ToString(allData[temp].roll);
    string hours = allData[temp].hours.ToString("D2");
    string minutes = allData[temp].minutes.ToString("D2");
    string seconds = allData[temp].seconds.ToString("D2");
```

```
        string time = hours + ":" + minutes + ":" + seconds;
        timeBox.Text = time;
        groundTempBox.Text = Convert.ToString(allData[temp].groundTemperature);
        groundPressure.Text = Convert.ToString(allData[temp].groundBarometricPressure);
        snrBox.Text = Convert.ToString(allData[temp].snr);
        rssiBox.Text = Convert.ToString(allData[temp].rssi);
        dataReceivedCounter.Text = Convert.ToString(counter);
        if (temp2)
        {
            Location newLocation = new Location();
            newLocation.Latitude = allData[temp].latitude;
            newLocation.Longitude = allData[temp].longitude;
            Pushpin pin = new Pushpin();
            pin.Background = new SolidColorBrush(Colors.AliceBlue);
            map.Children.Add(pin);
            Location groundLocation = new Location();
            groundLocation.Latitude = allData[temp].groundLatitude;
            groundLocation.Longitude = allData[temp].groundLongitude;
            Pushpin pin2 = new Pushpin();
            pin2.Background = new SolidColorBrush(Colors.Red);
            map.Children.Add(pin2);
            double longSum = longAverage * numberOfItems;
            longSum += newLocation.Longitude + groundLocation.Longitude;
            numberOfItems += 2;
            longAverage = longSum / (double)numberOfItems;
            double latSum = latAverage * (numberOfItems - 2);
            latSum += newLocation.Latitude + groundLocation.Latitude;
            latAverage = latSum / numberOfItems;
            Location center = new Location();
            center.Latitude = latAverage;
            center.Longitude = longAverage;
            map.SetView(center, 12);
        }
    }
    public void decodeData(string input)
    {
        string[] inputs = input.Split(',');
        data newData = new data();
        newData.snr = Convert.ToInt32(inputs[0]);
        newData.rssi = Convert.ToInt32(inputs[1]);
        newData.sampleNumber = Convert.ToInt32(inputs[2]);
        newData.internal_temperature = Convert.ToDouble(inputs[3]);
        newData.barometric_pressure = Convert.ToDouble(inputs[4]);
        newData.external_temperature = Convert.ToDouble(inputs[5]);
        newData.humidity= Convert.ToDouble(inputs[6]);
        newData.hours = Convert.ToInt32(inputs[7]);
        newData.minutes = Convert.ToInt32(inputs[8]);
        newData.seconds = Convert.ToInt32(inputs[9]);
        newData.gpsFix = Convert.ToInt32(inputs[10]);
        newData.longitude = Convert.ToDouble(inputs[11]);
        newData.latitude = Convert.ToDouble(inputs[12]);
        newData.altitude = Convert.ToDouble(inputs[13]);
        newData.heading = Convert.ToDouble(inputs[14]);
        newData.pitch = Convert.ToDouble(inputs[15]);
```

```
        newData.roll = Convert.ToDouble(inputs[16]);
        newData.agriculturalViability = Convert.ToDouble(inputs[17]);
        newData.pressureAtms = Convert.ToDouble(inputs[18]);
        newData.altitudeFromPressure = Convert.ToDouble(inputs[19]);
        newData.dewPoint = Convert.ToDouble(inputs[20]);
        newData.groundTemperature = Convert.ToDouble(inputs[22]);
        newData.groundBarometricPressure = Convert.ToDouble(inputs[21]);
        newData.groundLatitude = Convert.ToDouble(inputs[23]);
        newData.groundLongitude = Convert.ToDouble(inputs[24]);
        newData.groundAltitude = Convert.ToDouble(inputs[25]);
        allData.Add(newData);
    }
    public void disableButtons()
    {
        connectSerialButton.IsEnabled = true;
        serialPortBaudRate.IsEnabled = true;
        serialPortSelectionBox.IsEnabled = true;
        disconnectSerialButton.IsEnabled = false;
        dataReceivedCounter.IsEnabled = false;
        snrBox.IsEnabled = false;
        rssiBox.IsEnabled = false;
        samplesReceivedLabel.IsEnabled = false;
        samplesReceivedLabel_Copy.IsEnabled = false;
        samplesReceivedLabel_Copy1.IsEnabled = false;
        map.IsEnabled = true;
        label1.IsEnabled = false;
        label1_Copy.IsEnabled = false;
        qfeInput.IsEnabled = false;
        setQFE.IsEnabled = false;
        setQFECurrent.IsEnabled = false;
        longSent.IsEnabled = false;
        latSent.IsEnabled = false;
        sendCoordinates.IsEnabled = false;
        returnToHome.IsEnabled = false;
        servoArm.IsEnabled = false;
        t1.IsEnabled = false;
        releaseServo.IsEnabled = false;
        t2.IsEnabled = false;
    }
    public void changeButtons()
    {
        disconnectSerialButton.IsEnabled = true;
        dataReceivedCounter.IsEnabled = true;
        snrBox.IsEnabled = true;
        rssiBox.IsEnabled = true;
        samplesReceivedLabel.IsEnabled = true;
        samplesReceivedLabel_Copy.IsEnabled = true;
        samplesReceivedLabel_Copy1.IsEnabled = true;
        map.IsEnabled = true;
        label1.IsEnabled = true;
        label1_Copy.IsEnabled = true;
        qfeInput.IsEnabled = true;
        setQFE.IsEnabled = true;
        setQFECurrent.IsEnabled = true;
```

```
        longSent.IsEnabled = true;
        latSent.IsEnabled = true;
        sendCoordinates.IsEnabled = true;
        returnToHome.IsEnabled = true;
        servoArm.IsEnabled = true;
        releaseServo.IsEnabled = false;
        t1.IsEnabled = true;
        t2.IsEnabled = false;
    }

    private void disconnectSerialButton_Click(object sender, RoutedEventArgs e)
    {
        serial.Close();
        connected = false;
        _timer.Stop();
        _timer2.Stop();
    }

    private void sendCoordinates_Click(object sender, RoutedEventArgs e)
    {
        string toBeSent = "3," + longSent.Text + "," + latSent.Text;
        serial.WriteLine(toBeSent);
    }

    private void returnToHome_Click(object sender, RoutedEventArgs e)
    {
        double tempLat = allData[allData.Count() - 1].groundLatitude;
        double tempLong = allData[allData.Count() - 1].groundLongitude;
        string toBeSent = "4," + tempLong + "," + tempLat;
    }

    private void setQFE_Click(object sender, RoutedEventArgs e)
    {
        string toBeSent = "1" + qfeInput.Text;
        serial.WriteLine(toBeSent);
    }

    private void setQFECurrent_Click(object sender, RoutedEventArgs e)
    {
        string toBeSent = "0";
        serial.WriteLine(toBeSent);
    }

    private void servoArm_Click(object sender, RoutedEventArgs e)
    {
        //serial.Write("256akq20a4");
        servoArm.IsEnabled = false;
        releaseServo.IsEnabled = true;
        t1.IsEnabled = false;
        t2.IsEnabled = true;
    }

    private void releaseServo_Click(object sender, RoutedEventArgs e)
    {
```

```
            //serial.Write("256akq20a4");
            releaseServo.IsEnabled = false;
            t2.IsEnabled = false;
        }
        public void receive(object sender, System.IO.Ports.SerialDataReceivedEventArgs e)
        {
            newAvailable = true;
        }
    }
}
```

13.3.3 – Website

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta name="description" content="This is the website of Cyclone, the Advanced Entry from St Paul's School in London for the UK CanSat Competition of 2015-2016">
  <meta name="author" content="Ashwin Ahuja">
  <title>Cyclone CanSat</title
  <link href="main.css" rel="stylesheet" type="text/css">
  <link href="main2.css" rel="stylesheet">
      <link href="main3.css" rel="stylesheet">
      <link rel="apple-touch-icon" sizes="57x57" href="/apple-touch-icon-57x57.png">
<link rel="apple-touch-icon" sizes="60x60" href="/apple-touch-icon-60x60.png">
<link rel="apple-touch-icon" sizes="72x72" href="/apple-touch-icon-72x72.png">
<link rel="apple-touch-icon" sizes="76x76" href="/apple-touch-icon-76x76.png">
<link rel="apple-touch-icon" sizes="114x114" href="/apple-touch-icon-114x114.png">
<link rel="apple-touch-icon" sizes="120x120" href="/apple-touch-icon-120x120.png">
<link rel="apple-touch-icon" sizes="144x144" href="/apple-touch-icon-144x144.png">
<link rel="apple-touch-icon" sizes="152x152" href="/apple-touch-icon-152x152.png">
```

```
<link rel="apple-touch-icon" sizes="180x180" href="/apple-touch-icon-180x180.png">
<link rel="icon" type="image/png" href="/favicon-32x32.png" sizes="32x32">
<link rel="icon" type="image/png" href="/android-chrome-192x192.png" sizes="192x192">
<link rel="icon" type="image/png" href="/favicon-96x96.png" sizes="96x96">
<link rel="icon" type="image/png" href="/favicon-16x16.png" sizes="16x16">
<link rel="manifest" href="/manifest.json">
<meta name="msapplication-TileColor" content="#da532c">
<meta name="msapplication-TileImage" content="/mstile-144x144.png">
<meta name="theme-color" content="#ffffff">
<link rel="apple-touch-icon" sizes="57x57" href="/apple-touch-icon-57x57.png"/>
<link rel="apple-touch-icon" sizes="60x60" href="/apple-touch-icon-60x60.png"/>
<link rel="apple-touch-icon" sizes="72x72" href="/apple-touch-icon-72x72.png"/>
<link rel="apple-touch-icon" sizes="76x76" href="/apple-touch-icon-76x76.png"/>
<link rel="apple-touch-icon" sizes="114x114" href="/apple-touch-icon-114x114.png"/>
<link rel="apple-touch-icon" sizes="120x120" href="/apple-touch-icon-120x120.png"/>
<link rel="apple-touch-icon" sizes="144x144" href="/apple-touch-icon-144x144.png"/>
<link rel="apple-touch-icon" sizes="152x152" href="/apple-touch-icon-152x152.png"/>
<link rel="apple-touch-icon" sizes="180x180" href="/apple-touch-icon-180x180.png"/>
<link rel="icon" type="image/png" href="/favicon-32x32.png" sizes="32x32"/>
<link rel="icon" type="image/png" href="/android-chrome-192x192.png" sizes="192x192"/>
<link rel="icon" type="image/png" href="/favicon-96x96.png" sizes="96x96"/>
<link rel="icon" type="image/png" href="/favicon-16x16.png" sizes="16x16"/>
<link rel="manifest" href="/manifest.json"/>
<meta name="msapplication-TileColor" content="#da532c"/>
<meta name="msapplication-TileImage" content="/mstile-144x144.png"/>
<meta name="theme-color" content="#ffffff"/>
        <style>
        table.table1 {
border: 1px solid #ddd;
border-collapse: separate;
border-left: 0;
```

```
border-radius: 10px;
border-spacing: 0px;
}

table.table2 td a { display: block; width: 100%; height: 100%; }

table.table1 thead {
display: table-header-group;
vertical-align: middle;
border-color: inherit;
border-collapse: separate;
}

table.table2 thead {
display: table-header-group;
vertical-align: middle;
border-color: inherit;
border-collapse: separate;
}

table.table1 tr {
display: table-row;
vertical-align: inherit;
border-color: inherit;
}

table.table2 tr {
display: table-row;
vertical-align: inherit;
border-color: inherit;
```

```
}

table.table1 th, td {
padding: 5px 4px 6px 4px;
text-align: center;
vertical-align: middle;

        background-color: #619BFF;
}

table.table2 th, td {
padding: 5px 4px 6px 4px;
text-align: center;
vertical-align: middle;

        background-color: #619BFF;
}

table.table1 td {

opacity:0.62;
}

table.table2 td {

opacity:0.8;
}
 table.table1 thead:first-child tr:first-child th:first-child, tbody:first-child tr:first-child td:first-child {

}
table.table2 thead:last-child tr:last-child th:first-child, tbody:last-child tr:last-child td:first-child {
```

```
}
table.table1 td:hover{
opacity:1;
}

table.table2 td:hover{
opacity:1;
}
video_background {
  position: fixed;
  top: 0;
  left: 0;
  bottom: 0;
  right: 0;
  z-index: -1000;
}

            </style>

<script type="text/javascript">

  if (screen.width <= 800) {
    window.location = "http://teamcycl.one/mobile/mindex.html";
  }
</script>

</head>

<body id="page-top" data-spy="scroll" data-target=".navbar-custom">
```

```
<nav class="navbar navbar-custom navbar-fixed-top" role="navigation">
  <div class="container">
    <div class="navbar-header page-scroll">
      <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-main-collapse">
        <i class="fa fa-bars"></i>
      </button>
      <a class="navbar-brand" href="index.html">

        <h1>CYCLONE</h1>
      </a>
    </div>

    <div class="collapse navbar-collapse navbar-right navbar-main-collapse">
  <ul class="nav navbar-nav">
   <li class="active"><a href="index.html">Home</a></li>
   <li>
    <a href="team.html#intro" class="dropdown-toggle" data-toggle="dropdown">About Us <b class="caret"></b></a>
    <ul class="dropdown-menu">
     <li><a href="team.html#mission">CanSat and Our Mission</a></li>
     <li><a href="team.html#team">Our Team</a></li>
     <li><a href="team.html#contact">Contact Us</a></li>
    </ul>
   </li>
            <li>
    <a href="news.html">News</a>
   </li>
            <li class="dropdown">
    <a href="media.html#intro" class="dropdown-toggle" data-toggle="dropdown">Media<b class="caret"></b></a>
    <ul class="dropdown-menu">
```

```
        <li><a href="media.html#photos">Photographs and Videos</a></li>
        <li><a href="media.html#docs">Documentation</a></li>
       </ul>
      </li>
     </ul>
        </div>
      </div>
   </nav>

          <div>

<table width="100%" class="table1">
<tr height="100%">
<td width = "33%" height = "900" onclick="location.href='team.html'" style="background:
url('http://teamcycl.one/team.jpg'); background-position: center; background-size:auto 100%; no-repeat
center" onmouseover=bigCell(this) onmouseout=smallCell(this)>
<table align="center" class="table2" width="190px">

<tr>
<td>
<a style ="opacity: 1.0" href="team.html">
<font color="white" size="6">
<b style="font-variant: small-caps;">
About Us
</b>
</font>
</a>
</td>
</tr>
<!--<tr>
<td>
```


CanSat and Our Mission

</td>

</tr>

<tr>

<td>

Our Team

</td>

</tr>

<tr>

<td>

Contact Us

</td>

</tr>-->

</table>

</td>

<td width = "33%" height = "900" onclick="location.href='news.html'" style = "background: url('http://teamcycl.one/news.jpg'); background-position: center; background-size:auto 100%; no-repeat center" onmouseover=bigCell(this) onmouseout=smallCell(this)>

```
<table align = "center" class="table2" width="190px">
<tr>
<td>
<a align="center" href="news.html">
<font color="white" size = "6">
<b style="font-variant: small-caps;">
News
</b>
</font>
</h1>
</td>
</tr>

</table>

</td>
<td width = "33%" height = "900"  onclick="location.href='media.html'"style="background: url('http://teamcycl.one/media.jpg'); background-position: center; background-size:auto 100%; no-repeat center" onmouseover=bigCell(this) onmouseout=smallCell(this)>
<table align="center" class="table2" width="190px">
<tr>
<td>
<a href="media.html">
<font color="white" size="6">
<b style="font-variant: small-caps;">
Media
</b>
</font>
</a>
</td>
</tr>
```

```
<!--
<tr>
<td>
<a href="photos.html">
<font color="white">
Photographs and Videos
</font>
</a>
</td>
</tr>
<tr>
<td>
<a href="documentation.html">
<font color="white">
Documentation
</font>
</a>
</td>
</tr>
-->
</table>
</td>

<!--<td width = "33%"> <img onmouseover="bigImg(this)" onmouseout="normalImg(this)" border="0"
src="http://stylonica.com/wp-content/uploads/2014/02/nature.jpg"width = "300" height = "200"></td>
<td width = "33%"> <img onmouseover="bigImg(this)" onmouseout="normalImg(this)" border="0"
src="http://images6.alphacoders.com/316/316963.jpg"width = "300" height = "200"></td>
-->
</tr>
</div>
<script>
```

```
function makeOpaque(x){
	opacity = 1.0;
}
function makeTrans(x){
	opacity = 0.6;
}
function bigImg(x){
	x.style.height = "800px"
	x.style.width = "1100px"
	}
function bigCell(x){
	x.style.width = "60%"
	}
function smallCell(x){
	x.style.width = "20%"
	}
function normalImg(x){
	x.style.height = "800px"
	x.style.width = "500px"
	}
</script>

<!--

	<footer>
		<div class="container">
			<div class="row">
				<div class="col-md-12 col-lg-12">

					<p>Designed by Ashwin Ahuja - July 2015</p>
				</div>
```

```
                        </div>
                    </div>
            </footer>
-->

    <script src="js1.js"></script>
    <script src="js2.js"></script>
            <script src="js3.js"></script>
    <script src="js4.js"></script>
<script>
  (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
  (i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
  m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
  })(window,document,'script','//www.google-analytics.com/analytics.js','ga');

  ga('create', 'UA-65679213-1', 'auto');
  ga('send', 'pageview');

</script>
</body>
</html>
```

Team.html

```
<!DOCTYPE html>
<html lang="en">

<head>
<meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <meta name="description" content="This is the website of Cyclone, the Advanced Entry from St Paul's School in London for the UK CanSat Competition of 2015-2016">
    <meta name="author" content="Ashwin Ahuja">

    <title>Cyclone CanSat</title>

    <link href="css/bootstrap.min.css" rel="stylesheet" type="text/css">

    <link href="font-awesome/css/font-awesome.min.css" rel="stylesheet" type="text/css">
        <link href="css/animate.css" rel="stylesheet" />
    <link href="css/style.css" rel="stylesheet">
        <link href="color/default.css" rel="stylesheet">
        <link rel="apple-touch-icon" sizes="57x57" href="/apple-touch-icon-57x57.png">
<link rel="apple-touch-icon" sizes="60x60" href="/apple-touch-icon-60x60.png">
<link rel="apple-touch-icon" sizes="72x72" href="/apple-touch-icon-72x72.png">
<link rel="apple-touch-icon" sizes="76x76" href="/apple-touch-icon-76x76.png">
<link rel="apple-touch-icon" sizes="114x114" href="/apple-touch-icon-114x114.png">
<link rel="apple-touch-icon" sizes="120x120" href="/apple-touch-icon-120x120.png">
<link rel="apple-touch-icon" sizes="144x144" href="/apple-touch-icon-144x144.png">
<link rel="apple-touch-icon" sizes="152x152" href="/apple-touch-icon-152x152.png">
<link rel="apple-touch-icon" sizes="180x180" href="/apple-touch-icon-180x180.png">
<link rel="icon" type="image/png" href="/favicon-32x32.png" sizes="32x32">
<link rel="icon" type="image/png" href="/android-chrome-192x192.png" sizes="192x192">
<link rel="icon" type="image/png" href="/favicon-96x96.png" sizes="96x96">
<link rel="icon" type="image/png" href="/favicon-16x16.png" sizes="16x16">
<link rel="manifest" href="/manifest.json">
<meta name="msapplication-TileColor" content="#da532c">
<meta name="msapplication-TileImage" content="/mstile-144x144.png">
<meta name="theme-color" content="#ffffff">
<link rel="apple-touch-icon" sizes="57x57" href="/apple-touch-icon-57x57.png"/>
<link rel="apple-touch-icon" sizes="60x60" href="/apple-touch-icon-60x60.png"/>
```

```
<link rel="apple-touch-icon" sizes="72x72" href="/apple-touch-icon-72x72.png"/>
<link rel="apple-touch-icon" sizes="76x76" href="/apple-touch-icon-76x76.png"/>
<link rel="apple-touch-icon" sizes="114x114" href="/apple-touch-icon-114x114.png"/>
<link rel="apple-touch-icon" sizes="120x120" href="/apple-touch-icon-120x120.png"/>
<link rel="apple-touch-icon" sizes="144x144" href="/apple-touch-icon-144x144.png"/>
<link rel="apple-touch-icon" sizes="152x152" href="/apple-touch-icon-152x152.png"/>
<link rel="apple-touch-icon" sizes="180x180" href="/apple-touch-icon-180x180.png"/>
<link rel="icon" type="image/png" href="/favicon-32x32.png" sizes="32x32"/>
<link rel="icon" type="image/png" href="/android-chrome-192x192.png" sizes="192x192"/>
<link rel="icon" type="image/png" href="/favicon-96x96.png" sizes="96x96"/>
<link rel="icon" type="image/png" href="/favicon-16x16.png" sizes="16x16"/>
<link rel="manifest" href="/manifest.json"/>
<meta name="msapplication-TileColor" content="#da532c"/>
<meta name="msapplication-TileImage" content="/mstile-144x144.png"/>
<meta name="theme-color" content="#ffffff"/>

        <style>
            @import url(http://weloveiconfonts.com/api/?family=brandico|fontawesome|zocial);

/* brandico */
[class*="brandico-"]:before {
  font-family: 'brandico', sans-serif;
}

/* fontawesome */
[class*="fontawesome-"]:before {
  font-family: 'FontAwesome', sans-serif;
}

/* zocial */
[class*="zocial-"]:before {
```

```
  font-family: 'zocial', sans-serif;
}
ul.social-buttons {
  text-align: center;
  margin-top: 50px;
}

ul.social-buttons li {
  display: inline-block;
  margin: 0 10px;
}

ul.social-buttons li a {
  width: 60px;
  height: 60px;
  display: block;
  border-radius: 50px;
  text-decoration: none;
  font-size: 30px;
  line-height: 60px;
  color: white;
}

ul.social-buttons li a.brandico-twitter-bird {
  background-color: #4099FF;
}

ul.social-buttons li a.brandico-facebook {
  background-color: #3B5998;
}
```

```
ul.social-buttons li a.brandico-instagram {
  background-color: #3f729b;
}

ul.social-buttons li a.brandico-vimeo {
  background-color: #4EBBFF;
}

ul.social-buttons li a.brandico-linkedin {
  background-color: #0e76a8;
}

/*Demo 1*/
ul#demo1 li a {
  transition: transform 0.2s linear;
}

ul#demo1 li:hover a {
  transform: translateY(-10px);
}
</style>
<script src="js5.js"></script>
<script type="text/javascript">
  <!--
  if (screen.width <= 800) {
    window.location = "http://teamcycl.one/mobile/mindex.html";
  }
  //-->
</script>
</head>
```

```
<body id="page-top" data-spy="scroll" data-target=".navbar-custom">


  <nav class="navbar navbar-custom navbar-fixed-top" role="navigation">
    <div class="container">
      <div class="navbar-header page-scroll">
        <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-main-collapse">
          <i class="fa fa-bars"></i>
        </button>
        <a class="navbar-brand" href="index.html">
          <h1>CYCLONE</h1>
        </a>
      </div>


      <div class="collapse navbar-collapse navbar-right navbar-main-collapse">
   <ul class="nav navbar-nav">
    <li><a href="index.html">Home</a></li>
    <li class="active-dropdown">
     <a href="#" class="dropdown-toggle" data-toggle="dropdown">About Us<b class="caret"></b></a>
     <ul class="dropdown-menu">
      <li><a href="team.html#mission">CanSat and Our Mission</a></li>
      <li><a href="team.html#team">Our Team</a></li>
      <li><a href="team.html#contact">Contact Us</a></li>
     </ul>
    </li>
                  <li><a href="news.html#intro">News</a>
    </li>
                  <li class="dropdown">
     <a href="#" class="dropdown-toggle" data-toggle="dropdown">Media<b class="caret"></b></a>
     <ul class="dropdown-menu">
      <li><a href="media.html#photos">Photographs and Videos</a></li>
```

```
      <li><a href="media.html#docs">Documentation</a></li>
    </ul>
   </li>
  </ul>
      </div>
   </div>
</nav>


<section id="intro" class="intro">

                <div class="slogan">
                        <h2>About Us</h2>
                </div>
                <div class="page-scroll">
                        <a href="#mission" class="btn btn-circle">
                                <i class="fa fa-angle-double-down animated"></i>
                        </a>
                </div>
</section>




        <!-- Section: services -->
<section id="mission" class="home-section text-center bg-gray">

                <div class="heading-about">
                        <div class="container">
                        <div class="row">
```

```
                    <div class="col-lg-8 col-lg-offset-2">
                    <div class="wow bounceInDown" data-wow-delay="0.05s">
                            <div class="section-heading">
                            <h2>CanSat and Our Mission</h2>
                            <i class="fa fa-2x fa-angle-down"></i>

                            </div>
                    </div>
                    </div>
            </div>
            </div>
</div>
<div class="container">
<div class="row">
            <div class="col-lg-2 col-lg-offset-5">
                    <hr class="marginbot-50">
            </div>
</div>
<table width="100%">
<tr>
<td width="45%">
<img src="mission1.png" width= "60%" style="float: left"> </img>
</td>
<td width="55%">
                    <img src="mission2.jpg" width= "100%"> </img>
</td>
</tr>
</table>
<br>
<br>
```

 A CanSat is a simulation of a real satellite, integrated within the volume and shape of a soft drink can (with a diameter of 66mm and a height of 115mm). The challenge for us is to fit all the major subsystems found in a satellite, such as power, sensors and a communication system, into this minimal volume. The CanSat is then launched to an altitude of a few hundred metres by a captive balloon and its mission begins: to carry out a scientific experiment and achieve a safe landing.

Team Cyclone is aiming to, as its primary mission, measure the temperature and pressure of the atmosphere it is passing through for the duration of its descent. This primary mission is the same for all CanSats in the competition; however, where Team Cyclone will be looking to impress the judges is with its secondary mission, which defines and differentiates each entrant.

 For the secondary mission we will be aiming to produce a quadcopter capable of surveying an extraterrestrial planet, using a live-feed camera, thus being able to analyse the topography and investigating the presence of any flora and fauna. The drone should be capable of investigating the air through which it is passing: finding the dew point of the air, useful for finding the quality (combining with the temperature) of the area for agriculture, thus human inhabitability. Finally, the drone should be able to navigate autonomously, in order that it can investigate specific sites.

<figure>

<figcaption style="font-size:115%"> Preliminary Ideas for CanSat Structure </figcaption>

</figure>

<figure>

<figcaption style="font-size:115%"> Preliminary Ideas for CanSat Electronics Flowchart and Components List </figcaption>

</figure>

 If you have any questions regarding the quadcopter, please contact either the software and electronics team at <u> sande@teamcycl.one</u> or the structural design team at <u> structure@teamcycl.one</u>

</section>

```
<section id="team" class="home-section text-center">
                <div class="heading-about">
                        <div class="container">
                        <div class="row">
                                <div class="col-lg-8 col-lg-offset-2">
                                        <div class="wow bounceInDown" data-wow-delay="0.05s">
                                        <div class="section-heading">
                                        <h2>About us</h2>
                                        <i class="fa fa-2x fa-angle-down"></i>

                                        </div>
                                        </div>
                                </div>
                        </div>
                        </div>
                </div>
                <div class="container">

                <div class="row">
                        <div class="col-lg-2 col-lg-offset-5">
                                <hr class="marginbot-50">
                        </div>
                </div>
                <br>
    <div class="row">
      <div class="col-xs-6 col-sm-3 col-md-3">
                                <div class="wow bounceInUp" data-wow-delay="0s">
        <div class="team boxed-grey">
          <div class="inner">
                                                <h5>Benjamin Yass</h5>
            <p class="subtitle">Co-Team Leader. Head of Structure and Mechanical Design
```

```
                                                    <br>
                                                    <u>
                                                    <a href="mailto:ben@teamcycl.one"
target="_top">ben@teamcycl.one</a></p></u><br>
                <div class="avatar"><img src="ben.jpg" alt="" class="img-responsive img-circle" /></div>
              </div>
            </div>
                                    </div>
        </div>
                            <div class="col-xs-6 col-sm-3 col-md-3">
                                    <div class="wow bounceInUp" data-wow-delay="0s">
            <div class="team boxed-grey">
              <div class="inner">
                                                    <h5>Ashwin Ahuja</h5>
                <p class="subtitle">Co-Team Leader. Head of Software and Electronics.<br> <u><a
href="http://twitter.com/ashwin_ahuja"> @Ashwin_Ahuja</a><br>

                                                    <a href="mailto:ashwin@teamcycl.one"
target="_top">ashwin@teamcycl.one</a></p></u>
                <div class="avatar"><img src="ashwin.jpg" alt="" class="img-responsive img-circle" /></div>

              </div>
            </div>
                                    </div>
        </div>


    </div>
                <br>

    <div class="row">
      <div class="col-xs-6 col-sm-3 col-md-3">
```

```
                                <div class="wow bounceInUp" data-wow-delay="0s">
        <div class="team boxed-grey">
          <div class="inner">
                                                <h5>Quentin Gueroult</h5>
            <p class="subtitle">Head of Outreach. <br><u> <a
href="http://twitter.com/guerouq">@guerouq</a></u>
            <u>
                                                <a href="mailto:quentin@teamcycl.one"
target="_top">quentin@teamcycl.one</a></p></u>
                                                <br>
                                                <div class="avatar"><img src="q.jpg" alt="" class="img-
responsive img-circle" /></div>
          </div>
        </div>
                                </div>
      </div>
                        <div class="col-xs-6 col-sm-3 col-md-3">
                                <div class="wow bounceInUp" data-wow-delay="0s">
        <div class="team boxed-grey">
          <div class="inner">
                                                <h5>William Eustace</h5>
            <p class="subtitle">Working as part of the Software and Electronics Team, with specific
responsibilities for the electronics and software of the base station<u><a
href="http://twitter.com/williameustace"> @WilliamEustace</a>
                                                <br>
                                                <a href="mailto:wiliam@teamcycl.one"
target="_top">william@teamcycl.one</a></u></p>
            <div class="avatar"><img src="william.jpeg" alt="" class="img-responsive img-circle" /></div>


          </div>
        </div>
                                </div>
      </div>
```

```
</div>
                <br>
                    <div class="row">
    <div class="col-xs-6 col-sm-3 col-md-3">
                                <div class="wow bounceInUp" data-wow-delay="0s">
      <div class="team boxed-grey">
        <div class="inner">
                                                <h5>Nick Palmer</h5>
          <p class="subtitle">Working in the Structural Team and in charge of Data Analysis


                                                <br><u><a href="mailto:nick@teamcycl.one"
target="_top">nick@teamcycl.one</a></p></u>
                                                <center><br><br>
          <div class="avatar"><img src="nick.jpg" alt="" class="img-responsive img-circle"
style="float:center" /></div></center>
        </div>
      </div>
                                </div>
    </div>
                        <div class="col-xs-6 col-sm-3 col-md-3">
                                <div class="wow bounceInUp" data-wow-delay="0s">
      <div class="team boxed-grey">
        <div class="inner">
                                                <h5>Daniel Halstead</h5>
          <p class="subtitle">Working in the Software and Electronics Team, with specific responsibility
for flight managment
                                                <br><u><a href="mailto:dan@teamcycl.one"
target="_top">dan@teamcycl.one</a></p></u>
          <div class="avatar"><img src="dan.jpg" alt="" class="img-responsive img-circle" /></div>
```

```
            </div>
          </div>
                                    </div>
        </div>


    </div>
<br>
                    <div class="row">
        <div class="col-xs-6 col-sm-3 col-md-3">
                                    <div class="wow bounceInUp" data-wow-delay="0s">
          <div class="team boxed-grey">
            <div class="inner">
                                                <h5>Philip Fernandes</h5>
              <p class="subtitle">Working in the Outreach Team, with specific responsibility for Social Media Presence
                                                <br><u><a href="mailto:phil@teamcycl.one" target="_top">phil@teamcycl.one</a></p></u>
                                                <br>
                                                <br>
              <div class="avatar"><img src="phil.jpg" alt="" class="img-responsive img-circle" /></div>
                                                <br>
            </div>
          </div>
                                    </div>
        </div>
                        <div class="col-xs-6 col-sm-3 col-md-3">
                                    <div class="wow bounceInUp" data-wow-delay="0s">
          <div class="team boxed-grey">
            <div class="inner">
                                                <h5>Hugo Aaronson</h5>
```

Working in the Software and Electronics Team, with a specific responsibility for Ground Processing Software

<u>hugo@teamcycl.one </u>

<h5>James Crompton</h5>

Working in the Structure and Outreach Teams, with specific responsibility for documentation, including progress reports and presentations.

<u>james@teamcycl.one </u>


```
    </div>
                </div>
            </section>
<section id="contact" class="home-section text-center">
                <div class="heading-contact">
                        <div class="container">
                        <div class="row">
                                <div class="col-lg-8 col-lg-offset-2">
                                        <div class="wow bounceInDown" data-wow-delay="0.05s">
                                        <div class="section-heading">
                                        <h2>Contact Us</h2>
                                        <i class="fa fa-2x fa-angle-down"></i>

                                        </div>
                                        </div>
                                </div>
                        </div>
                        </div>
                </div>
                <div class="container">

                <div class="row">
                        <div class="col-lg-2 col-lg-offset-5">
                                <hr class="marginbot-50">
                        </div>
                </div>
  <div class="row">
    <div class="col-lg-8">
```

```
<div class="boxed-grey">
  <form id="contact-form">
  <div class="row">
    <div class="col-md-6">
      <div class="form-group">
        <label for="name">
          Name</label>
        <input type="text" class="form-control" id="name" placeholder="Enter name"
required="required" />
      </div>
      <div class="form-group">
        <label for="email">
          Email Address</label>
        <div class="input-group">
          <span class="input-group-addon"><span class="glyphicon glyphicon-envelope"></span>
          </span>
          <input type="email" class="form-control" id="email" placeholder="Enter email"
required="required" /></div>
      </div>

    </div>
    <div class="col-md-6">
      <div class="form-group">
        <label for="name">
          Message</label>
        <textarea name="message" id="message" class="form-control" rows="9" cols="10"
required="required"
          placeholder="Message"></textarea>
      </div>
    </div>
    <div class="col-md-12">
      <button type="submit" class="btn btn-skin pull-right" id="btnContactUs">
```

```
                Send Message</button>
            </div>
          </div>
          </form>
        </div>
      </div>

                <div class="col-lg-4">
                    <div class="widget-contact">
                        <h5>Location</h5>

                        <address>
                          <strong>St Paul's School</strong><br>
                          Lonsdale Road, Barnes<br>
                          London, SW13 8JT<br>
                          <abbr title="Phone"></abbr>+447597711495
                        </address>

                        <address>
                          <strong>Email</strong><br>
                          <a href="mailto:#">hello@teamcycl.one</a>
                        </address>


                    </div>
                </div>
</div>

                </div>
        </section>
        </div>
```

```
</section>


<footer>
    <div class="container">
        <div class="row">
            <div class="col-md-12 col-lg-12">

                <div class="page-scroll marginbot-30">
                    <a href="#intro" id="totop" class="btn btn-circle">
                        <i class="fa fa-angle-double-up animated"></i>
                    </a>
                </div>
                                    <ul class="social-buttons"
id="demo1">
                                    <table width="100%">
                                    <tr>
                                    <td width="50%">
                                    <div align="center">
                                    <a class="twitter-timeline"
href="https://twitter.com/spscyclone" style="text-align:center;display:block;" data-widget-
id="624754089604345856">Tweets by @spscyclone</a>
                                    </div>
                                    </td>
                                    <td width="50%">

                                    <!-- HTML Codes by Quackit.com -->
                                    <!-- HTML Codes by Quackit.com -->
<span style="font-family:Verdana;font-size:22px;font-style:italic;font-weight:normal;text-
decoration:none;text-transform:none;font-variant:small-caps;color:#FFFFFF;">Sponsored by:</span><br>
<img src='hobbyking.png' height="auto" width="50%"><br>
<img src='pcbtrain.png' height="auto" width="90%">
```

```
</td>
</tr>
</table>

                                                                    <script>!function(d,s,id){var
js,fjs=d.getElementsByTagName(s)[0],p=/^http:/.test(d.location)?'http':'https';if(!d.getElementById(id)){js=d.
createElement(s);js.id=id;js.src=p+"://platform.twitter.com/widgets.js";fjs.parentNode.insertBefore(js,fjs);}}(
document,"script","twitter-wjs");</script>
<br>
<br>
 <li>
   <a href="http://twitter.com/spscyclone" class="brandico-twitter-bird"></a>
 </li>
 <li>
   <a href="http://on.fb.me/1OrQvAZ" class="brandico-facebook"></a>
 </li>
<li>
   <a href="https://vimeo.com/user42195003" class="brandico-vimeo"></a>
 </li>
   <li>
   <a href="https://github.com/ashwinahuja" class="brandico-github"></a>
 </li>
<li>
   <a href="http://cyclonecansat.wordpress.com" class="brandico-wordpress"></a>
 </li>

</ul>
                                        <p>Designed by Ashwin Ahuja - August 2015</p>
                                </div>
                        </div>
                </div>
        </footer>
```

```
        <script src="js1.js"></script>
   <script src="js2.js"></script>
        <script src="js3.js"></script>
   <script src="js4.js"></script>
<!-- Start of StatCounter Code for Default Guide -->
<script type="text/javascript">
var sc_project=10546202;
var sc_invisible=1;
var sc_security="0989a23a";
var scJsHost = (("https:" == document.location.protocol) ?
"https://secure." : "http://www.");
document.write("<sc"+"ript type='text/javascript' src='" +
scJsHost+
"statcounter.com/counter/counter.js'></"+"script>");
</script>
<noscript><div class="statcounter"><a title="shopify traffic
stats" href="http://statcounter.com/shopify/"
target="_blank"><img class="statcounter"
src="http://c.statcounter.com/10546202/0/0989a23a/1/"
alt="shopify traffic stats"></a></div></noscript>
<!-- End of StatCounter Code for Default Guide -->
<script>
  (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
```

```
(i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
})(window,document,'script','//www.google-analytics.com/analytics.js','ga');

ga('create', 'UA-65679213-1', 'auto');
ga('send', 'pageview');

</script>
</body>

</html>
```

News.html

```
<!DOCTYPE html>
<html lang="en">

<head>
<meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta name="description" content="This is the website of Cyclone, the Advanced Entry from St Paul's School in London for the UK CanSat Competition of 2015-2016">
  <meta name="author" content="Ashwin Ahuja">

  <title>Cyclone CanSat</title>

  <link href="css/bootstrap.min.css" rel="stylesheet" type="text/css">

  <link href="font-awesome/css/font-awesome.min.css" rel="stylesheet" type="text/css">
      <link href="css/animate.css" rel="stylesheet" />
  <link href="css/style.css" rel="stylesheet">
```

```
        <link href="color/default.css" rel="stylesheet">
        <link rel="apple-touch-icon" sizes="57x57" href="/apple-touch-icon-57x57.png">
<link rel="apple-touch-icon" sizes="60x60" href="/apple-touch-icon-60x60.png">
<link rel="apple-touch-icon" sizes="72x72" href="/apple-touch-icon-72x72.png">
<link rel="apple-touch-icon" sizes="76x76" href="/apple-touch-icon-76x76.png">
<link rel="apple-touch-icon" sizes="114x114" href="/apple-touch-icon-114x114.png">
<link rel="apple-touch-icon" sizes="120x120" href="/apple-touch-icon-120x120.png">
<link rel="apple-touch-icon" sizes="144x144" href="/apple-touch-icon-144x144.png">
<link rel="apple-touch-icon" sizes="152x152" href="/apple-touch-icon-152x152.png">
<link rel="apple-touch-icon" sizes="180x180" href="/apple-touch-icon-180x180.png">
<link rel="icon" type="image/png" href="/favicon-32x32.png" sizes="32x32">
<link rel="icon" type="image/png" href="/android-chrome-192x192.png" sizes="192x192">
<link rel="icon" type="image/png" href="/favicon-96x96.png" sizes="96x96">
<link rel="icon" type="image/png" href="/favicon-16x16.png" sizes="16x16">
<link rel="manifest" href="/manifest.json">
<meta name="msapplication-TileColor" content="#da532c">
<meta name="msapplication-TileImage" content="/mstile-144x144.png">
<meta name="theme-color" content="#ffffff">
<link rel="apple-touch-icon" sizes="57x57" href="/apple-touch-icon-57x57.png"/>
<link rel="apple-touch-icon" sizes="60x60" href="/apple-touch-icon-60x60.png"/>
<link rel="apple-touch-icon" sizes="72x72" href="/apple-touch-icon-72x72.png"/>
<link rel="apple-touch-icon" sizes="76x76" href="/apple-touch-icon-76x76.png"/>
<link rel="apple-touch-icon" sizes="114x114" href="/apple-touch-icon-114x114.png"/>
<link rel="apple-touch-icon" sizes="120x120" href="/apple-touch-icon-120x120.png"/>
<link rel="apple-touch-icon" sizes="144x144" href="/apple-touch-icon-144x144.png"/>
<link rel="apple-touch-icon" sizes="152x152" href="/apple-touch-icon-152x152.png"/>
<link rel="apple-touch-icon" sizes="180x180" href="/apple-touch-icon-180x180.png"/>
<link rel="icon" type="image/png" href="/favicon-32x32.png" sizes="32x32"/>
<link rel="icon" type="image/png" href="/android-chrome-192x192.png" sizes="192x192"/>
<link rel="icon" type="image/png" href="/favicon-96x96.png" sizes="96x96"/>
<link rel="icon" type="image/png" href="/favicon-16x16.png" sizes="16x16"/>
```

```
<link rel="manifest" href="/manifest.json"/>
<meta name="msapplication-TileColor" content="#da532c"/>
<meta name="msapplication-TileImage" content="/mstile-144x144.png"/>
<meta name="theme-color" content="#ffffff"/>

        <style>
            @import url(http://weloveiconfonts.com/api/?family=brandico);

/* brandico */
[class*="brandico-"]:before {
  font-family: 'brandico', sans-serif;
}

ul.social-buttons {
  text-align: center;
  margin-top: 50px;
}

ul.social-buttons li {
  display: inline-block;
  margin: 0 10px;
}

ul.social-buttons li a {
  width: 60px;
  height: 60px;
  display: block;
  border-radius: 50px;
  text-decoration: none;
  font-size: 30px;
  line-height: 60px;
```

```
  color: white;
}

ul.social-buttons li a.brandico-twitter-bird {
  background-color: #4099FF;
}

ul.social-buttons li a.brandico-facebook {
  background-color: #3B5998;
}

ul.social-buttons li a.brandico-instagram {
  background-color: #3f729b;
}

ul.social-buttons li a.brandico-vimeo {
  background-color: #4EBBFF;
}

ul.social-buttons li a.brandico-linkedin {
  background-color: #0e76a8;
}

/*Demo 1*/
ul#demo1 li a {
  transition: transform 0.2s linear;
}

ul#demo1 li:hover a {
  transform: translateY(-10px);
}
```

```
</style>
<script src="js5.js"></script>
<script type="text/javascript">
 <!--
 if (screen.width <= 800) {
  window.location = "http://teamcycl.one/mobile/mindex.html";
 }
 //-->
</script>
</head>

<body id="page-top" data-spy="scroll" data-target=".navbar-custom">

  <nav class="navbar navbar-custom navbar-fixed-top" role="navigation">
    <div class="container">
      <div class="navbar-header page-scroll">
        <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-main-collapse">
          <i class="fa fa-bars"></i>
        </button>
        <a class="navbar-brand" href="index.html">
          <h1>CYCLONE</h1>
        </a>
      </div>

      <div class="collapse navbar-collapse navbar-right navbar-main-collapse">
    <ul class="nav navbar-nav">
     <li><a href="index.html">Home</a></li>
     <li class="active-dropdown">
       <a href="#" class="dropdown-toggle" data-toggle="dropdown">About Us<b class="caret"></b></a>
       <ul class="dropdown-menu">
```

```
      <li><a href="team.html#mission">CanSat and Our Mission</a></li>
      <li><a href="team.html#team">Our Team</a></li>
      <li><a href="team.html#contact">Contact Us</a></li>
    </ul>
  </li>
                <li><a href="news.html#intro">News</a>
  </li>
                <li class="dropdown">
    <a href="#" class="dropdown-toggle" data-toggle="dropdown">Media<b class="caret"></b></a>
    <ul class="dropdown-menu">
      <li><a href="media.html#photos">Photographs and Videos</a></li>
      <li><a href="media.html#docs">Documentation</a></li>
    </ul>
  </li>
 </ul>
     </div>
  </div>
</nav>


<section id="intro" class="intro">

                <div class="slogan">
                        <h2>News</h2>
                </div>
                <div class="page-scroll">
                        <a href="#mission" class="btn btn-circle">
                                <i class="fa fa-angle-double-down animated"></i>
                        </a>
                </div>
</section>
```

```
        <!-- Section: services -->
    <section id="mission" class="home-section text-center bg-gray">


                <div class="heading-about">
                        <div class="container">
                        <div class="row">
                                <div class="col-lg-8 col-lg-offset-2">
                                <div class="wow bounceInDown" data-wow-delay="0.05s">
                                        <div class="section-heading">
                                        <h2>Blog - available on <a href="cyclonecansat.wordpress.com">
cyclonecansat.wordpress.com</a></h2>
                                        <i class="fa fa-2x fa-angle-down"></i>


                                        </div>
                                </div>
                                </div>
                        </div>
                        </div>
                </div>
                <div class="container">
                <div class="row">
                        <div class="col-lg-2 col-lg-offset-5">
                                <hr class="marginbot-50">
                        </div>
                </div>
        <iframe src="http://cyclonecansat.wordpress.com" width="100%" height="600px">
```

```
        <p>Your browser does not support iframes.</p>
</iframe>
        </div>
        </section>




        <footer>
                <div class="container">
                        <div class="row">
                                <div class="col-md-12 col-lg-12">


                                        <div class="page-scroll marginbot-30">
                                                <a href="#intro" id="totop" class="btn btn-circle">
                                                        <i class="fa fa-angle-double-up animated"></i>
                                                </a>
                                        </div>
                                                                        <ul class="social-buttons"
id="demo1">
                                                                        <table width="100%">
                                                                        <tr>
                                                                        <td width="50%">
                                                                        <div align="center">
                                                                        <a class="twitter-timeline"
href="https://twitter.com/spscyclone" style="text-align:center;display:block;" data-widget-
id="624754089604345856">Tweets by @spscyclone</a>
                                                                        </div>
                                                                        </td>
                                                                        <td width="50%">

                                                                        <!-- HTML Codes by Quackit.com -->
                                                                        <!-- HTML Codes by Quackit.com -->
```

Sponsored by:

<img src='hobbyking.png' height="auto" width="50%">

<img src='pcbtrain.png' height="auto" width="90%">

</td>

</tr>

</table>

<script>!function(d,s,id){var js,fjs=d.getElementsByTagName(s)[0],p=/^http:/.test(d.location)?'http':'https';if(!d.getElementById(id)){js=d.createElement(s);js.id=id;js.src=p+"://platform.twitter.com/widgets.js";fjs.parentNode.insertBefore(js,fjs);}}(document,"script","twitter-wjs");</script>

Designed by Ashwin Ahuja - August 2015

```
                    </div>
                </div>
            </div>
        </footer>

        <script src="js1.js"></script>
    <script src="js2.js"></script>
        <script src="js3.js"></script>
    <script src="js4.js"></script>
<!-- Start of StatCounter Code for Default Guide -->
<script type="text/javascript">
var sc_project=10546202;
var sc_invisible=1;
var sc_security="0989a23a";
var scJsHost = (("https:" == document.location.protocol) ?
"https://secure." : "http://www.");
document.write("<sc"+"ript type='text/javascript' src='" +
scJsHost+
"statcounter.com/counter/counter.js'></"+"script>");
</script>
<script>
  (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
  (i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
```

```
 m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
 })(window,document,'script','//www.google-analytics.com/analytics.js','ga');

 ga('create', 'UA-65679213-1', 'auto');
 ga('send', 'pageview');

</script>
<noscript><div class="statcounter"><a title="shopify traffic
stats" href="http://statcounter.com/shopify/"
target="_blank"><img class="statcounter"
src="http://c.statcounter.com/10546202/0/0989a23a/1/"
alt="shopify traffic stats"></a></div></noscript>
<!-- End of StatCounter Code for Default Guide -->
</body>

</html>
```

Media.html

```
<!DOCTYPE html>
<html lang="en">

<head>
<meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <meta name="description" content="This is the website of Cyclone, the Advanced Entry from St Paul's
School in London for the UK CanSat Competition of 2015-2016">
   <meta name="author" content="Ashwin Ahuja">

   <title>Cyclone CanSat</title>
```

```
    <link href="css/bootstrap.min.css" rel="stylesheet" type="text/css">

    <link href="font-awesome/css/font-awesome.min.css" rel="stylesheet" type="text/css">
        <link href="css/animate.css" rel="stylesheet" />
    <link href="css/style.css" rel="stylesheet">
        <link href="color/default.css" rel="stylesheet">
        <link rel="apple-touch-icon" sizes="57x57" href="/apple-touch-icon-57x57.png">
<link rel="apple-touch-icon" sizes="60x60" href="/apple-touch-icon-60x60.png">
<link rel="apple-touch-icon" sizes="72x72" href="/apple-touch-icon-72x72.png">
<link rel="apple-touch-icon" sizes="76x76" href="/apple-touch-icon-76x76.png">
<link rel="apple-touch-icon" sizes="114x114" href="/apple-touch-icon-114x114.png">
<link rel="apple-touch-icon" sizes="120x120" href="/apple-touch-icon-120x120.png">
<link rel="apple-touch-icon" sizes="144x144" href="/apple-touch-icon-144x144.png">
<link rel="apple-touch-icon" sizes="152x152" href="/apple-touch-icon-152x152.png">
<link rel="apple-touch-icon" sizes="180x180" href="/apple-touch-icon-180x180.png">
<link rel="icon" type="image/png" href="/favicon-32x32.png" sizes="32x32">
<link rel="icon" type="image/png" href="/android-chrome-192x192.png" sizes="192x192">
<link rel="icon" type="image/png" href="/favicon-96x96.png" sizes="96x96">
<link rel="icon" type="image/png" href="/favicon-16x16.png" sizes="16x16">
<link rel="manifest" href="/manifest.json">
<meta name="msapplication-TileColor" content="#da532c">
<meta name="msapplication-TileImage" content="/mstile-144x144.png">
<meta name="theme-color" content="#ffffff">
<link rel="apple-touch-icon" sizes="57x57" href="/apple-touch-icon-57x57.png"/>
<link rel="apple-touch-icon" sizes="60x60" href="/apple-touch-icon-60x60.png"/>
<link rel="apple-touch-icon" sizes="72x72" href="/apple-touch-icon-72x72.png"/>
<link rel="apple-touch-icon" sizes="76x76" href="/apple-touch-icon-76x76.png"/>
<link rel="apple-touch-icon" sizes="114x114" href="/apple-touch-icon-114x114.png"/>
<link rel="apple-touch-icon" sizes="120x120" href="/apple-touch-icon-120x120.png"/>
<link rel="apple-touch-icon" sizes="144x144" href="/apple-touch-icon-144x144.png"/>
<link rel="apple-touch-icon" sizes="152x152" href="/apple-touch-icon-152x152.png"/>
```

```
<link rel="apple-touch-icon" sizes="180x180" href="/apple-touch-icon-180x180.png"/>
<link rel="icon" type="image/png" href="/favicon-32x32.png" sizes="32x32"/>
<link rel="icon" type="image/png" href="/android-chrome-192x192.png" sizes="192x192"/>
<link rel="icon" type="image/png" href="/favicon-96x96.png" sizes="96x96"/>
<link rel="icon" type="image/png" href="/favicon-16x16.png" sizes="16x16"/>
<link rel="manifest" href="/manifest.json"/>
<meta name="msapplication-TileColor" content="#da532c"/>
<meta name="msapplication-TileImage" content="/mstile-144x144.png"/>
<meta name="theme-color" content="#ffffff"/>

        <style>
            @import url(http://weloveiconfonts.com/api/?family=brandico);

/* brandico */
[class*="brandico-"]:before {
  font-family: 'brandico', sans-serif;
}

ul.social-buttons {
  text-align: center;
  margin-top: 50px;
}

ul.social-buttons li {
  display: inline-block;
  margin: 0 10px;
}

ul.social-buttons li a {
  width: 60px;
  height: 60px;
```

```
  display: block;
  border-radius: 50px;
  text-decoration: none;
  font-size: 30px;
  line-height: 60px;
  color: white;
}

ul.social-buttons li a.brandico-twitter-bird {
  background-color: #4099FF;
}

ul.social-buttons li a.brandico-facebook {
  background-color: #3B5998;
}

ul.social-buttons li a.brandico-instagram {
  background-color: #3f729b;
}

ul.social-buttons li a.brandico-vimeo {
  background-color: #4EBBFF;
}

ul.social-buttons li a.brandico-linkedin {
  background-color: #0e76a8;
}

/*Demo 1*/
ul#demo1 li a {
  transition: transform 0.2s linear;
```

```
}

ul#demo1 li:hover a {
 transform: translateY(-10px);
}
</style>
<script src="js5.js"></script>
<script type="text/javascript">
 <!--
 if (screen.width <= 800) {
   window.location = "http://teamcycl.one/mobile/mindex.html";
 }
 //-->
</script>
</head>

<body id="page-top" data-spy="scroll" data-target=".navbar-custom">

   <nav class="navbar navbar-custom navbar-fixed-top" role="navigation">
      <div class="container">
         <div class="navbar-header page-scroll">
            <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-main-collapse">
               <i class="fa fa-bars"></i>
            </button>
            <a class="navbar-brand" href="index.html">
               <h1>CYCLONE</h1>
            </a>
         </div>

         <div class="collapse navbar-collapse navbar-right navbar-main-collapse">
```

```
 <ul class="nav navbar-nav">
  <li><a href="index.html">Home</a></li>
  <li class="active-dropdown">
   <a href="#" class="dropdown-toggle" data-toggle="dropdown">About Us<b class="caret"></b></a>
   <ul class="dropdown-menu">
    <li><a href="team.html#mission">CanSat and Our Mission</a></li>
    <li><a href="team.html#team">Our Team</a></li>
    <li><a href="team.html#contact">Contact Us</a></li>
   </ul>
  </li>
                <li><a href="news.html#intro">News</a>
  </li>
                <li class="dropdown">
   <a href="#" class="dropdown-toggle" data-toggle="dropdown">Media<b class="caret"></b></a>
   <ul class="dropdown-menu">
    <li><a href="media.html#photos">Photographs and Videos</a></li>
    <li><a href="media.html#docs">Documentation</a></li>
   </ul>
  </li>
 </ul>
    </div>
  </div>
</nav>


<section id="intro" class="intro">

                <div class="slogan">
                        <h2>Media</h2>
                </div>
                <div class="page-scroll">
```

```
                    <a href="#photos" class="btn btn-circle">
                            <i class="fa fa-angle-double-down animated"></i>
                    </a>
            </div>
</section>




    <!-- Section: services -->
<section id="photos" class="home-section text-center bg-gray">

            <div class="heading-about">
                    <div class="container">
                    <div class="row">
                            <div class="col-lg-8 col-lg-offset-2">
                            <div class="wow bounceInDown" data-wow-delay="0.05s">
                                    <div class="section-heading">
                                    <h2>Photographs and Videos</h2>
                                    <i class="fa fa-2x fa-angle-down"></i>

                                    </div>
                            </div>
                            </div>
                    </div>
                    </div>
            </div>
            <div class="container">
            <div class="row">
                    <div class="col-lg-2 col-lg-offset-5">
```

```
                    <hr class="marginbot-50">
                </div>
                </div>
                </div>

                </section>
                <table cellpadding="10" width="100%" border="1">
                <tr>
                <td width="50%">
                <object width="100%" height="600" style="float:middle;"> <param
name="flashvars" value="offsite=true&lang=en-
us&page_show_url=%2Fphotos%2F134309143%40N08%2Fshow%2F&page_show_back_url=%2Fphotos%2F
134309143%40N08%2F&user_id=134309143@N08&jump_to="></param> <param name="movie"
value="https://www.flickr.com/apps/slideshow/show.swf?v=1811922554"></param> <param
name="allowFullScreen" value="true"></param><embed type="application/x-shockwave-flash"
src="https://www.flickr.com/apps/slideshow/show.swf?v=1811922554" allowFullScreen="true"
flashvars="offsite=true&lang=en-
us&page_show_url=%2Fphotos%2F134309143%40N08%2Fshow%2F&page_show_back_url=%2Fphotos%2F
134309143%40N08%2F&user_id=134309143@N08&jump_to=" width="800"
height="600"></embed></object>
                </td>
                <td width="50%">
                <iframe
src="http://player.vimeo.com/hubnut/user/user42195003/uploaded_videos?color=44bbff&amp;backgroun
d=000000&amp;slideshow=1&amp;video_title=1&amp;video_byline=1" width="100%" height="600px"
frameborder="0" webkitAllowFullScreen mozallowfullscreen allowFullScreen style="float:center;"></iframe>
                </td>

                </tr>
                </table>


        <section id="docs" class="home-section text-center bg-gray">

            <div class="heading-about">
```

```
                    <div class="container">
                    <div class="row">
                            <div class="col-lg-8 col-lg-offset-2">
                            <div class="wow bounceInDown" data-wow-delay="0.05s">
                                    <div class="section-heading">
                                    <h2>Documentation</h2>
                                    <i class="fa fa-2x fa-angle-down"></i>


                                    </div>
                            </div>
                            </div>
                    </div>
                    </div>
            </div>
            <div class="container">
            <div class="row">
                    <div class="col-lg-2 col-lg-offset-5">
                            <hr class="marginbot-50">
                    </div>
            </div>
            <iframe
src="https://app.box.com/embed_widget/s/hbo8nu2nnb78rs27ooh45e2x65n18sb6?view=icon&sort=name
&direction=ASC&theme=gray" width="800" height="550" frameborder="0"allowfullscreen
webkitallowfullscreen msallowfullscreen></iframe>
            </div>
    </section>



    <footer>
            <div class="container">
                    <div class="row">
                            <div class="col-md-12 col-lg-12">
```

```
<div class="page-scroll marginbot-30">
<a href="#intro" id="totop" class="btn btn-circle">
<i class="fa fa-angle-double-up animated"></i>
</a>
</div>
<ul class="social-buttons" id="demo1">
<table width="100%">
<tr>
<td width="50%">
<div align="center">
<a class="twitter-timeline" href="https://twitter.com/spscyclone" style="text-align:center;display:block;" data-widget-id="624754089604345856">Tweets by @spscyclone</a>
</div>
</td>
<td width="50%">

<!-- HTML Codes by Quackit.com -->
<!-- HTML Codes by Quackit.com -->
<span style="font-family:Verdana;font-size:22px;font-style:italic;font-weight:normal;text-decoration:none;text-transform:none;font-variant:small-caps;color:#FFFFFF;">Sponsored by:</span><br>
<img src='hobbyking.png' height="auto" width="50%"><br>
<img src='pcbtrain.png' height="auto" width="90%">
</td>
</tr>
</table>

<script>!function(d,s,id){var js,fjs=d.getElementsByTagName(s)[0],p=/^http:/.test(d.location)?'http':'https';if(!d.getElementById(id)){js=d.createElement(s);js.id=id;js.src=p+"://platform.twitter.com/widgets.js";fjs.parentNode.insertBefore(js,fjs);}}(document,"script","twitter-wjs");</script>
```

```
<br>
<br>
 <li>
  <a href="http://twitter.com/spscyclone" class="brandico-twitter-bird"></a>
 </li>
 <li>
  <a href="http://on.fb.me/1OrQvAZ" class="brandico-facebook"></a>
 </li>
<li>
  <a href="https://vimeo.com/user42195003" class="brandico-vimeo"></a>
 </li>
  <li>
  <a href="https://github.com/ashwinahuja" class="brandico-github"></a>
 </li>
<li>
  <a href="http://cyclonecansat.wordpress.com" class="brandico-wordpress"></a>
 </li>

</ul>
                                        <p>Designed by Ashwin Ahuja - August 2015</p>
                                    </div>
                                </div>
                            </div>
                        </footer>
```

```
        <script src="js1.js"></script>
   <script src="js2.js"></script>
        <script src="js3.js"></script>
   <script src="js4.js"></script>
<!-- Start of StatCounter Code for Default Guide -->
<script type="text/javascript">
var sc_project=10546202;
var sc_invisible=1;
var sc_security="0989a23a";
var scJsHost = (("https:" == document.location.protocol) ?
"https://secure." : "http://www.");
document.write("<sc"+"ript type='text/javascript' src='" +
scJsHost+
"statcounter.com/counter/counter.js'></"+"script>");
</script>
<noscript><div class="statcounter"><a title="shopify traffic
stats" href="http://statcounter.com/shopify/"
target="_blank"><img class="statcounter"
src="http://c.statcounter.com/10546202/0/0989a23a/1/"
alt="shopify traffic stats"></a></div></noscript>
<!-- End of StatCounter Code for Default Guide -->
<script>
  (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
  (i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
  m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
  })(window,document,'script','//www.google-analytics.com/analytics.js','ga');

  ga('create', 'UA-65679213-1', 'auto');
  ga('send', 'pageview');
```

```
</script>
</body>

</html>
```