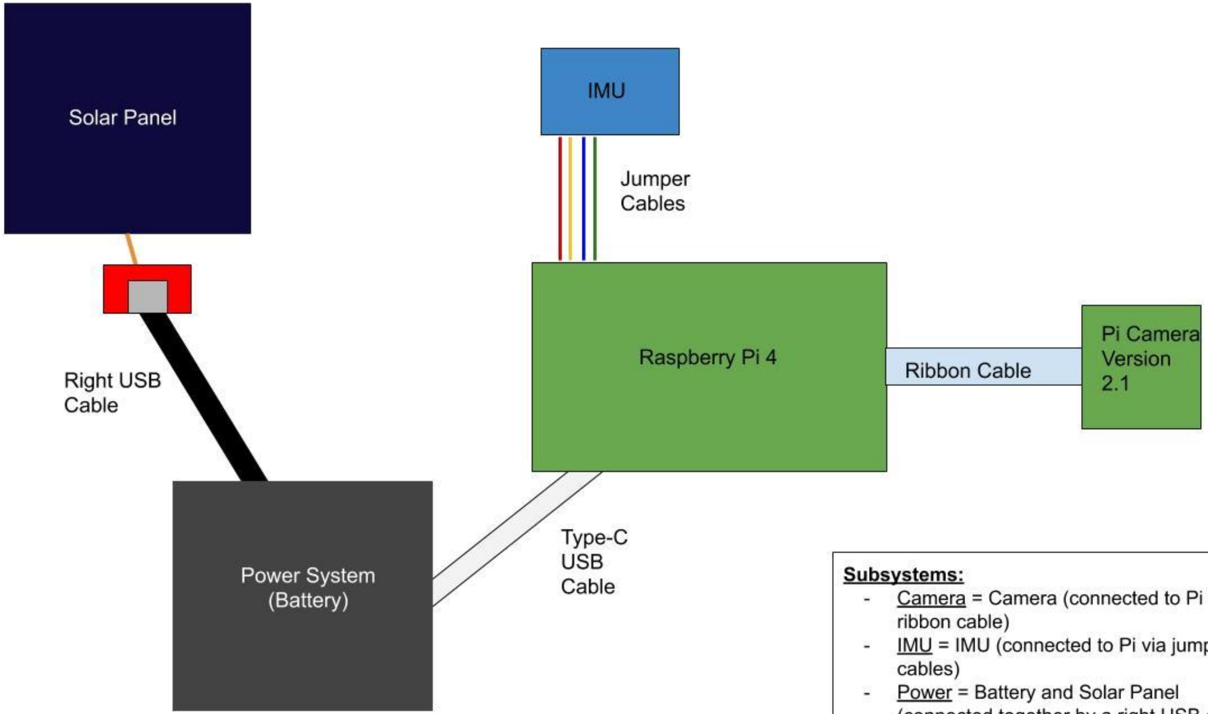
Design Review

Avik Banerjee, Hannah Epstein, Nico Moldovean, Emily McCarthy, Albert Chu

Mission Requirements

Requirements	Verification
The CubeSat shall capture images of each poster board from waist height at arms length out (~2ft) in order to simulate LEO	Verify by viewing the images on the Raspberry Pi
The CubeSat shall be within 1U in dimension	Verify using measurements from a ruler
The CubeSat's mass shall be within 1.33 kg.	Verify using measurements from a scale and predicted values from a final mass budget
The CubeSat shall record the time and location of each image it captures during its orbit.	Verify by testing and viewing image filenames on the ground station and github.
The CubeSat shall send one telemetry packet--containing current RPY, subsystem status, current orbit, and number of pictures captured--to the ground station once per orbit.	Verify by viewing the packet on the ground station.
The CubeSat shall downlink images and telemetry packets to the ground station through bluetooth.	Verify by checking for new images and telemetry packets on the ground station
The images shall be processed to determine the percentage of plastic on the poster board, the percentage of plastic relative to each other, and the size of each piece of plastic on the board.	Verify by comparing the processed values to true values
The ground station shall upload images of each unique poster board, telemetry packets, and plastic analysis to a Github repository accessible to instructors.	Verify through testing and viewing files in a shared Github repository
The CubeSat shall have sufficient power to last 10 minutes of orbit.	Verify through stress tests, experimentation, and a final power budget

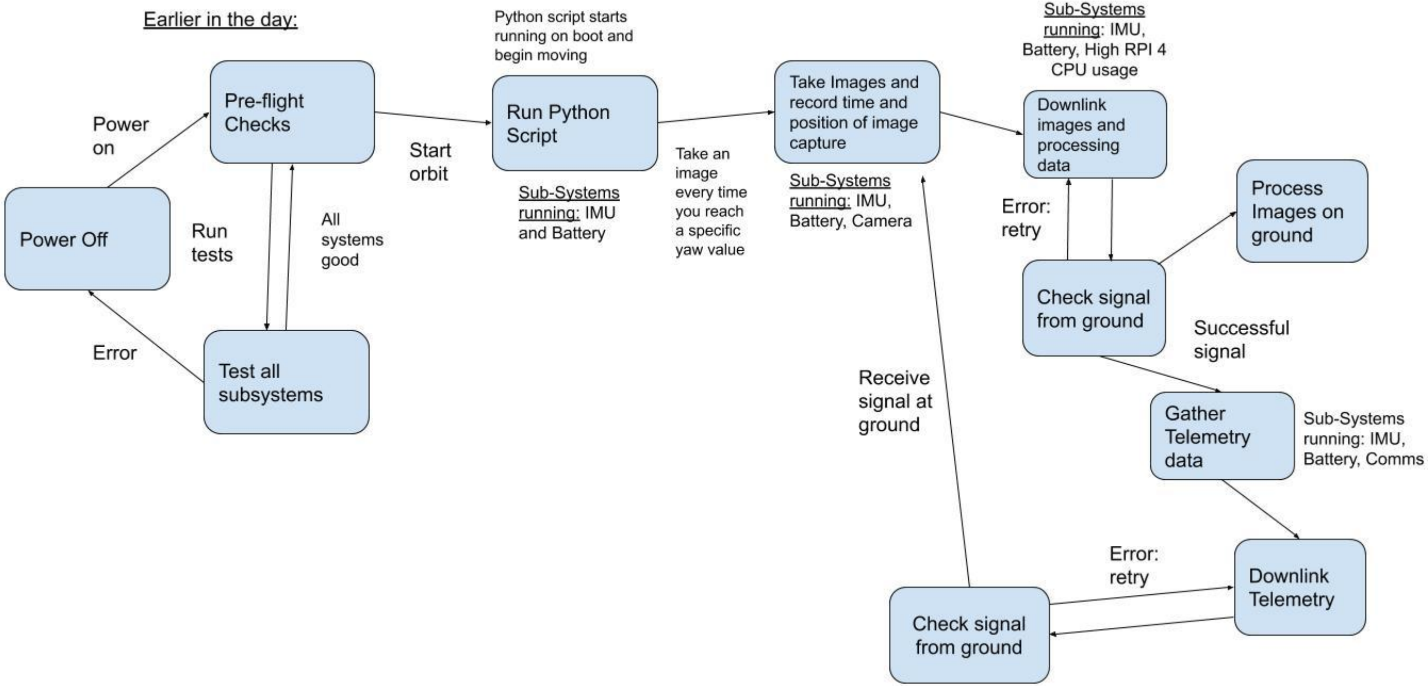
Block Diagram



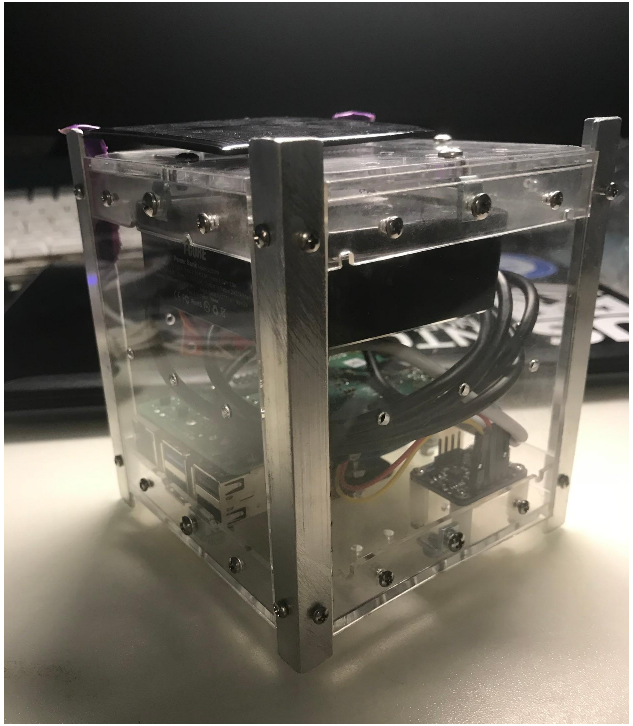
Subsystems:

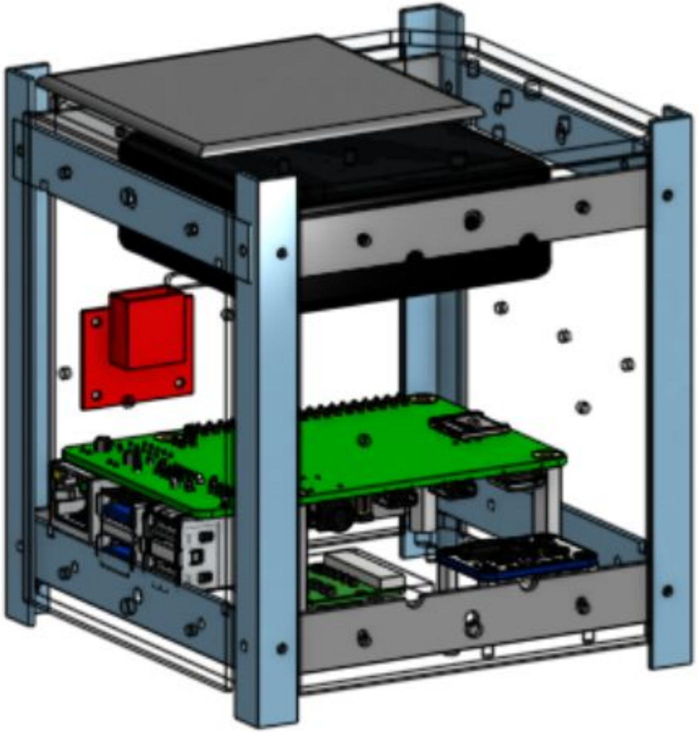
- Camera = Camera (connected to Pi via a ribbon cable)
- IMU = IMU (connected to Pi via jumper cables)
- Power = Battery and Solar Panel (connected together by a right USB cable and connected to Pi via a Type-C cable)
- Comm = Raspberry Pi

Concept of Operations (CONOPS)



Mechanical Design - CAD Model





Power Budget

Mode	Current (Amps)	Voltage (V)	Power (W)	Runtime(h) using	1% drain (h)	1% drain (min)	1% drain (sec)	% Contingency	1% drain (sec) with contingency
Idle	0.4	5	2	25	0.25	15	900	20	720
Camera	0.6	5	3	16.66666667	0.1666666667	10	600	20	480
Downlink	0.43	5	2.15	23.25581395	0.2325581395	13.95348837	837.2093023	20	669.7674419
IMU	0.5	5	2.5	20	0.2	12	720	20	576

Mode	IMU	Camera	Downlink	Downlink	Camera	Downlink	Camera	Downlink	Camera	Downlink	Downlink
Time in flight	Full time	5 sec	10 sec	20 sec	25 sec	30sec	45 sec	50 sec	65 sec	70 sec	80 sec
Time total	10 min	5 sec	5 sec	5 sec	5 sec	5 sec	5 sec	5 sec	5 sec	5 sec	5 sec
Power used (W)	2.5	3	2.15	2.15	3	2.15	3	2.15	3	2.15	2.15
Battery percentage difference	1.041666667	0.01041666667	0.007465277778	0.007465277778	0.01041666667	0.007465277778	0.01041666667	0.007465277778	0.01041666667	0.007465277778	0.007465277778
Battery percentage	98.95833333	98.94791667	98.94045139	98.93298611	98.92256944	98.91510417	98.9046875	98.89722222	98.88680556	98.87934028	98.871875

- Final battery percentage: 98.7% (assuming a start of 100% battery)
- Will lose ~1.3% of battery throughout flight

Mass Budget

- Mass determined from our own measurements and manufacturer's specifications
- Contingency was 10%

Component	Quantity	Mass (g)	Total Comp Mas	% Contingency	Mass + Contingency (g)
Structure					
Side Panel	3	14	42	10	46.2
Top/Bottom Panel	2	28	56	10	61.6
Side Panel(extra holes)	1	13	13	10	14.3
Side Bracket	8	2	16	10	17.6
Corner Rail	4	8	32	10	35.2
Standoff (2.5cm)	4	0.4	1.6	10	1.76
Standoff (1 cm)	4	0.1	0.4	10	0.44
1/4" screws	22	0.4	8.8	10	9.68
3/16" screws	36	0.3	10.8	10	11.88
4-40 L bracket	8	1	8	10	8.8
Camera					
Camera (with case)	1	11	11	10	12.1
IMU					
IMU	1	2	2	10	2.2
Jumper Cable	4	0.2	0.8	10	0.88
Power					
Battery	1	175	175	10	192.5
Solar Panel	1	19	19	10	20.9
USB-C Cord	1	10	10	10	11
microUSB Cord	1	25	25	10	27.5
Comm/Power					
Raspberry Pi (with SD car	1	48	48	10	52.8

- Total mass of 527g, taking contingency into account
- Only 40% of the total mass allowed (1,330 g)
- This potentially leaves room for additional hardware, but space constraints are more limiting

	No contingency	10% contingency
Total (g)	479.4	527.34
Total Mass Allowed (g)	1330	1330
Mass Remaining (g)	850.6	802.66
Mass Remaining %	63.95	60.35

Note: If necessary, duct tape will be added. That would add another ~0.5 g.

Bill of Materials

- Total cost: $182.94

Component	Quantity	Mass (g)	Cost (USD)
Structure			
Side Panel	3	14	$5.49
Top Panel	1	28	$1.83
Bottom Panel	1	28	$1.83
Side Panel(extra holes)	1	13	$1.83
Side Bracket	8	2	$4.18
Corner Rail	4	8	$12.00
Standoff (2.5cm)	4	0.4	$10.24
Standoff (1 cm)	4	0.1	$6.28
1/4" screws	22	0.4	$0.25
3/16" screws	36	0.3	$1.84
4-40 L bracket	8	1	$4.00
Duct tape	5 cm²	0.5	$4.00
Camera			
Camera (with case)	1	11	$33.00
IMU			
IMU	1	2	$14.95
Jumper Cable	4	0.2	$3.00
Power			
Battery	1	175	$28.00
Solar Panel	1	19	$2.28
USB-C Cord	1	10	*came with batter
microUSB Cord	1	25	$5.99
Comm/Power			
Micro SD Card 32 GB	1	9	$6.95
Raspberry Pi	1	48	$35.00
TOTAL:			$182.94

Link and Data Budgets

Paramters	Value	Calculate:	Value
Image Comm Time for Orbits 1-3 (in seconds)	15	Bits per Image	320166
Telemetry Packet Comm Time for Orbits 1-3 (in seconds)	5	Bytes per Image	40020.8
Telemetry Packet Comm Time for Orbits 4-10 (in seconds)	5	Bits per Telemetry Packet	24000.0
Maximum downlink Serial Port BAUD Rate (bits per second)	256,000	Seconds to transmit 1 telemetry packet	0.1
Bits per pixel	24	Seconds to transmit 1 image	1.3
Image resolution (pixels^2)	13340.3	Images that may be transmitted during Orbits 1-3	12.0
Image width (in pixels)	115.5	Telemetry Packets that may be transmitted during Orbits 1-3	53.3
Image height (in pixels)	115.5	Telemetry Packets that may be transmitted during Orbits 4-10	53.3
How the link will change:			
- At the beginning of the orbit we should be able to close the link very quickly			
- As the Pi begins trying to downlink more images, the link rate will likely slow down			
- If the Pi gets overwhelmed by other processes occuring, the time it takes to close the link will also increase			

Integration and Test Plan

Some of the things we want to do:

-Within the main pi script:

-Continuously running our IMU function to keep our position updated

-Sending telemetry packets every orbit

-Taking and downlinking pictures based on position

-At the ground station:

-Send commands to the pi if needed

-Appropriately receive data sent from the pi and upload it to Github

How do we get there?

-Integrate Critical Functions into on-board and ground scripts through clear communication

-i.e. Discussing what the IMU output looks like and how we can directly send it via the serial port/use it to trigger an image capture; Streamlining file paths so we send/upload the correct files; Initiating serial reception at the ground station, etc.

-Have each group member individually test components to help isolate bugs and test each phase of implementation of the main software