SAPPHO: Spectroscopic Analysis of Phobos Plus HD Orbital survey

Dakota Richline (Solo)

Department of Aerospace, Physics, and Space Sciences, Florida Tech

AEE 4806: Space Mission Engineering

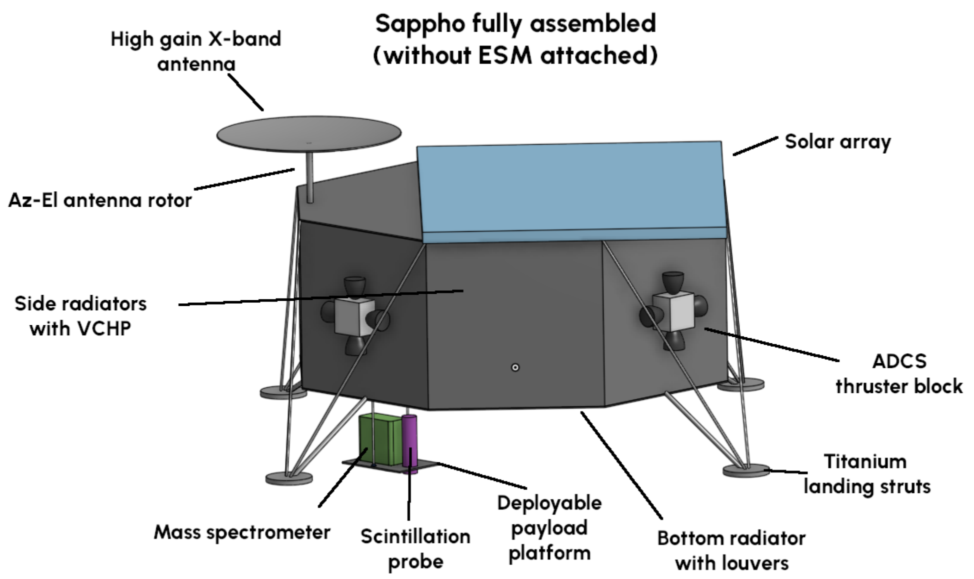
Dr. Paula do Vale Pereira

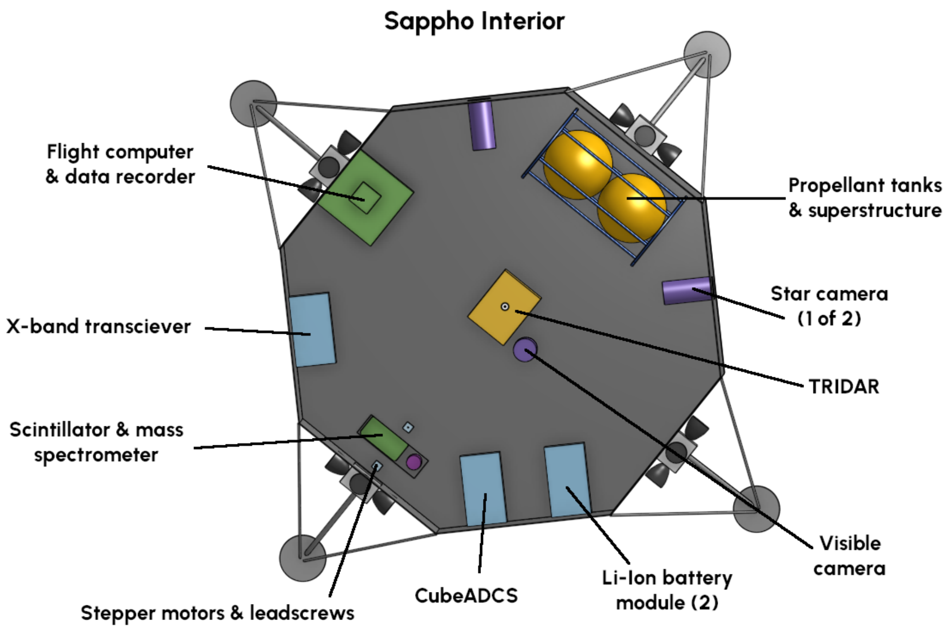
SAPPHO: Spectroscopic Analysis of Phobos Plus HD Orbital survey

Mission Concept

Sappho is a Phobos lander that will use a European Service Module for its ballistic (no aerobraking) transfer to Mars. After capture, Hohmann transfer, and rendezvous with Phobos, Sappho will separate from the service module and begin station-keeping with Phobos. An “orbital” survey will be conducted via visible and TRIDAR imaging. Since Phobos orbit is impossible due to very low gravity, Sappho will “dock” with the sunlit side of Phobos and abort back to rendezvous if docking fails. Phobos is tidally locked, so its upper surface is always pointed zenith and has two Sun/Earth passes per Sol. The payload platform is deployed after landing, pressing the mass spectrometer and scintillation probe firmly against the regolith; the seismometer will detect seismic activity through the landing legs. Sappho will remain on Phobos for the duration of its mission; Sun/Earth passes will be spent charging batteries and transmitting/recieving data via the Deep Space Network (DSN).

Since Sappho will remain connected to the service module for the first 260 days of its mission, it will hibernate between system checkouts and remain passive until separation. The service module is derived from the ATV cargo vehicle, which is designed for long-duration missions in LEO or Orion missions beyond Earth orbit.

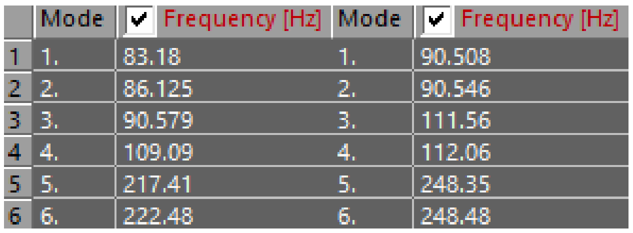

Sappho fully assembled (without ESM attached)
High gain X-band antenna
Solar array
Az-El antenna rotor
Side radiators with VCHP
ADCS thruster block
Titanium landing struts
Mass spectrometer
Scintillation probe
Deployable payload platform
Bottom radiator with louvers


Sappho Interior
Flight computer & data recorder
Propellant tanks & superstructure
Star camera (1 of 2)
X-band transciever
TRIDAR
Scintillator & mass spectrometer
Visible camera
Stepper motors & leadscrews
CubeADCS
Li-Ion battery module (2)

Structure

The main structure of Sappho is a simple 3-piece 2 meter by 75 cm hollow aluminum octagon with 1/8 inch walls and four titanium landing struts that project 30 cm in the z-direction. The octogon's aluminum base is used as a radiator for components mounted to it with louvers; its sides are also used as radiators, but through Variable Conductance Heat Pipes (VCHP). Components are mounted to either the top, bottom, or sides. I wanted to take advantage of the ESM's 4 meter lander and ended up making Sappho larger volumetrically than necessary. However, the requirements call for conformal solar panels, which barely fit on Sappho at its current size. The oversized walls are just barely large enough to dissipate heat in the hot case, and the intermediate-cold case is very well balanced with louvers and heat pipes. The large interior allows components to be evenly spaced without clustering them together by subsystem; I tried to place things logically, e.g. the transceiver is very close to the antenna. Axial sensors (cameras and TRIDAR) are roughly centered and integrated parts are toward the sides.

The landing legs are also used for mounting to the service module, which should help constrain them and raise the natural frequency as shown by the vibration analysis comparing the constrained and free struts. The landing struts very thin, albeit overbuilt and can support a 500 kg spacecraft during an 8g acceleration (Max 6g per SpaceX) with a factor of safety of over 15. After launch, the maximum acceleration ranges from 0.04 g to 0.1 g. The SpaceX Falcon User's Guide prescribes a maximum of 100 Hz axial vibration at 1g; only the first two natural frequencies are below 100 Hz. If necessary, elastomer could be used to isolate and dampen vibrations of the feet, legs, and antenna rotor during launch. All models are wrong, but I hope mine are useful.



	Mode	✓ Frequency [Hz]	Mode	✓ Frequency [Hz]
1	1.	83.18	1.	90.508
2	2.	86.125	2.	90.546
3	3.	90.579	3.	111.56
4	4.	109.09	4.	112.06
5	5.	217.41	5.	248.35
6	6.	222.48	6.	248.48

Figure 1. Modal analysis for unconstrained (left) and constrained (right) landing strut

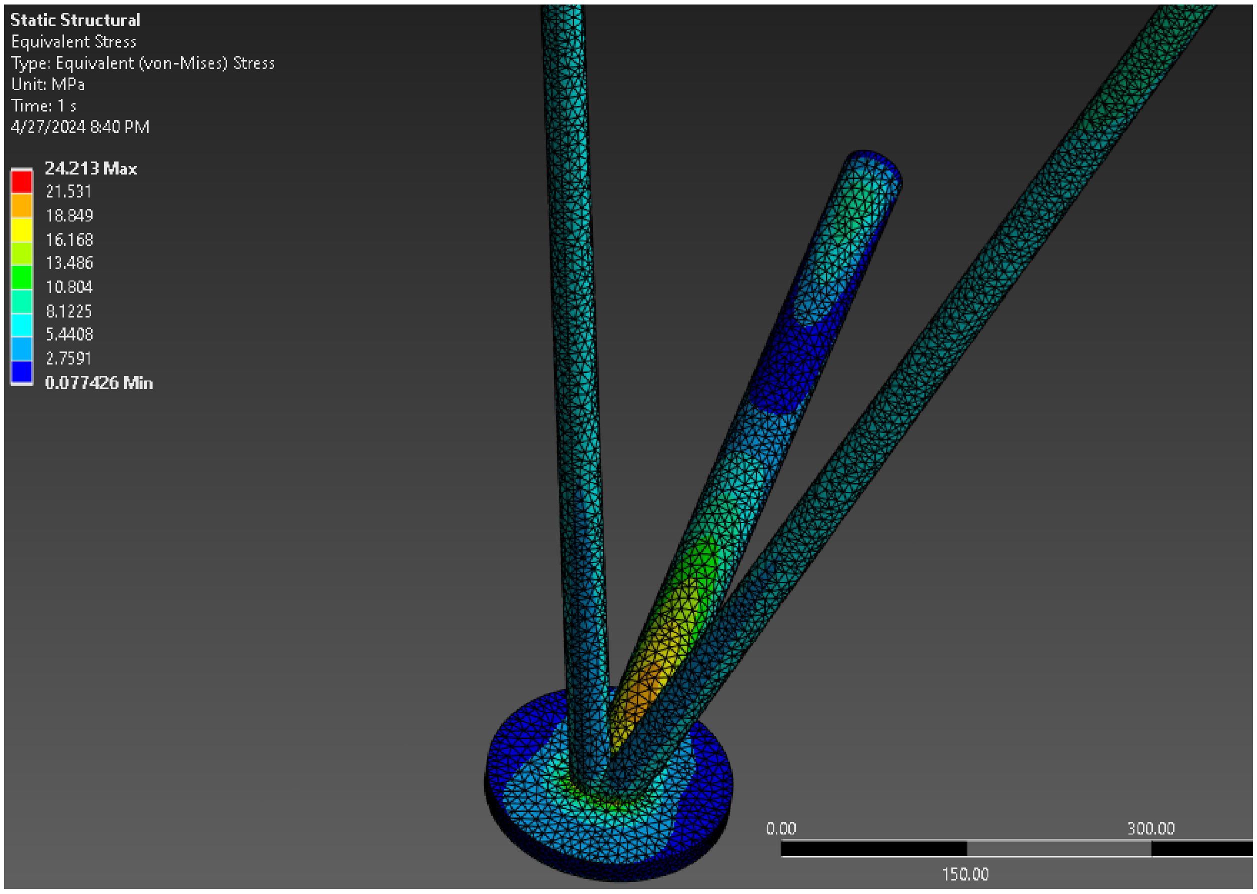


Figure 2. Von Mises stress for 8g acceleration

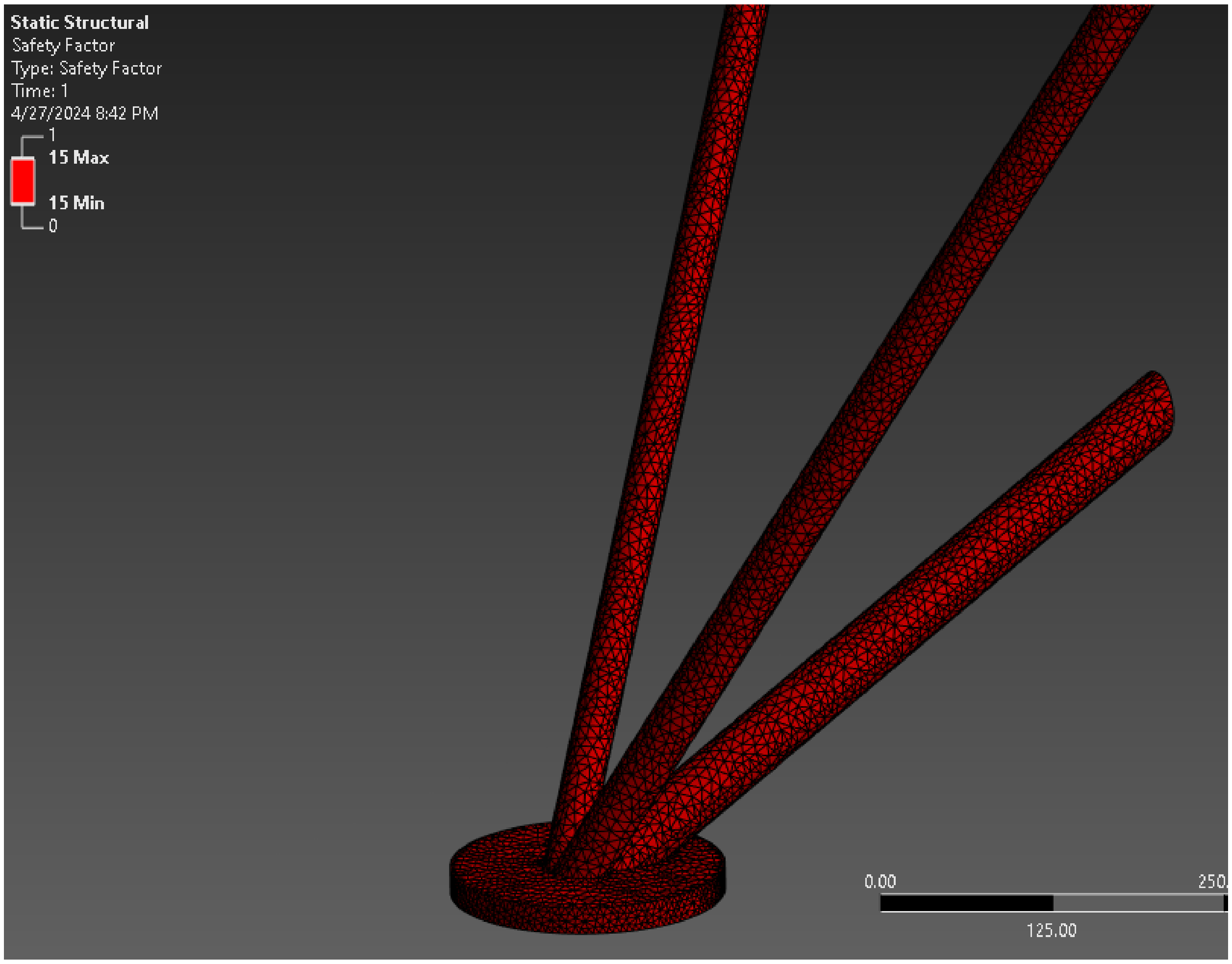


Figure 3. Factor of Safety for 8g acceleration

All components besides the antenna, conformal solar panels, and payload platform will remain inside the vehicle for shielding and temperature control. The original requirements did not allow deployable mechanisms, but I couldn't find another way to make contact with regolith while also protecting the instruments during landing. As a compromise, the mechanism only deploys once and cannot interfere with other spacecraft components. It uses two NEMA-14 stepper motors with 12-inch passthrough leadscrews to precisely lower the platform, retaining the leadscrews inside Sappho until deployment.

The final mass of Sappho is 232 kg; less than half the 500 kg mass budget.

Thermal

Since Sappho is essentially a box with all components located inside, thermal calculations are relatively simple. The maximum projected area is just the area of the octagonal face: $2.8\,\mathrm{m}^2$. Including the sides, the total area available for radiating is $7.28\,\mathrm{m}^2$. There is a hole in the base when the payload platform is deployed, but it can be closed by retracting the platform; the payload shouldn't need heaters since it's stowed on the warm aluminum base and only outside Sappho for a few minutes to take its measurements. Kapton heaters can be added if necessary. Since subsystem components are mounted indiscriminately to the bottom plate, my goal is to keep the bottom plate at 20°C using heaters and active/passive cooling. At 20°, it can radiate 234 W. Using the sides and bottom, Sappho can radiate 610 W maximum.

The large difference between the hot case (588 W) and the cold case (52 W) does make keeping a constant temperature difficult. Specifically, the hot case is orbital survey during Sun passes with the transmitter and all sensors active; the cold case is in Mars/Phobos shade with only flight computers and avionics running. The hot case includes direct solar radiation and albedo + IR from Mars. The batteries can be isolated and have built-in temperature control, but other components are mounted directly to the radiating surface. Placing VCHPs between the bottom and side walls, and louvers on the bottom surface that can reduce radiation by 6x, Sappho radiates a minimum of 39 W. Since the cold case is 52 W, the heaters do not need to be active; they produce up to $50\,\mathrm{W/in^2}$ and can be placed strategically. The VCHPs can be cycled to regulate temperature during low-medium power modes, with louvers opening in the hot case (orbital survey, sunlit). The power consumption during cruise is 40 W with heaters, which should be a perfect match for the main radiator with louvers closed (assuming the ESM is shielding Sappho from the Sun).

Electrical

Sappho was originally designed for a power budget of 200 W in Sun, and 150 W in shade. The lithium-ion batteries can easily handle 150 W for half a Sol (which is significantly higher than the actual power draw in shade), and the solar cells produce 250 W in direct sunlight (designed for 600 W on Earth). There are three solar panels, and two battery modules that share the loads for redundancy. I determined electrical power states from the four stages of conops: cruise, survey/landing, post-landing, and shade; shade applies both on-orbit and on Phobos. Sappho is significantly power-positive in all modes. During the cruise phase Sappho is hibernating and draws power from the ESM, which generates an effectively unlimited power budget of 10 kW. Survey and landing are the most energy-intensive states because cameras, TRIDAR, and the transmitter are active. I neglected the power draw of the scintillator and spectrometer because they are only on for a very brief period, are low-power, and can take their measurements in the shade when the transmitter is off.

ADCS/Propulsion

For the first 260 days, Sappho relies on its service module for propulsion and attitude control: an AJ-10 engine and its 24 secondary ADCS/docking thrusters (MON3/MMH). Assuming the ESM can propel itself forward with 6 thrusters, it has an average acceleration of $0.14\,\mathrm{m/s^2}$. Since the ESM was originally designed for docking, I wanted to make sure Sappho could achieve a similar level of acceleration using its four forward thrusters: $0.38\,\mathrm{m/s^2}$ average. Sappho uses 16 of the same Frontier 10-lbf MON3/MMH engines as the Peregrine lander, mounted “Apollo style;” they are designed for very short PWM pulses and relatively low operating temperature (-40°C). The tanks hold 15 kg of propellant each, based on similar missions.

Sappho has a very high-throughput flight computer for processing sensor data in real-time during landing (c.f. C&DH), as well a low-power CubeADCS avionics package. The cubesat avionics provide an IMU and star cameras, but it will need to interface with the flight computer’s FPGA to control thrusters and receive navigation data during landing.

Communication

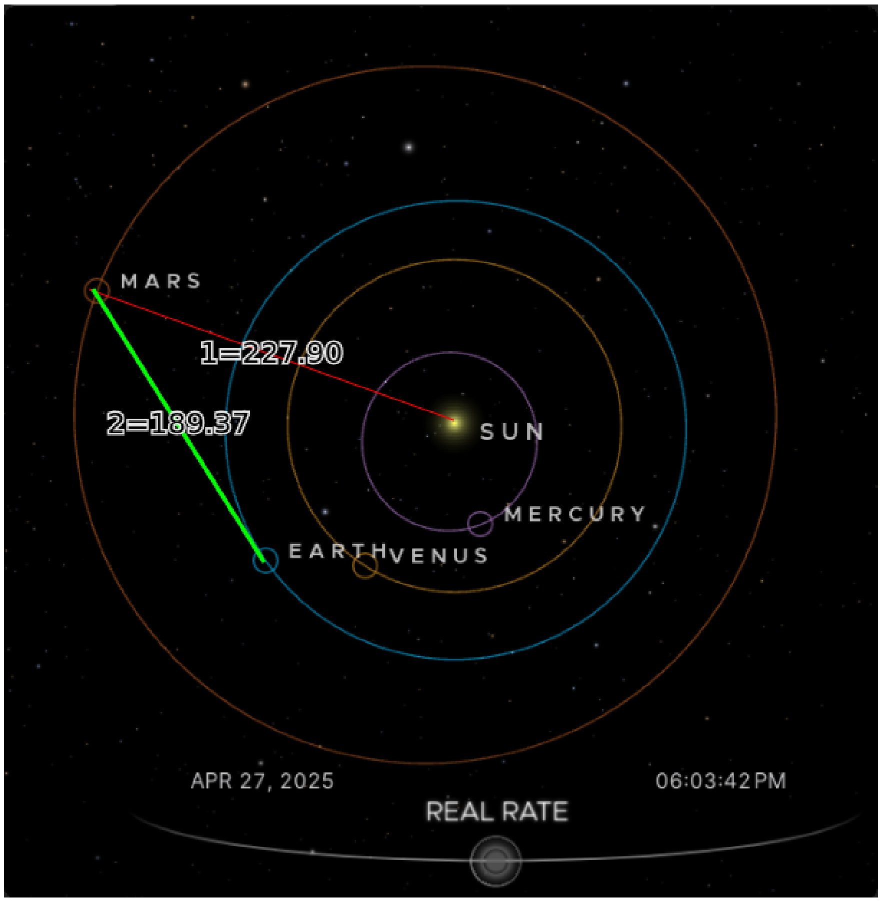


Figure 4. [1×10^9 m] Distance from Earth to Mars 260 days after 2024 launch window

The ESM has its own (unknown) high-gain communications system—and it only needs very small TT&C links—so I focused on Sappho's link budget. Communications for Sappho use the X-band per requirements and heritage in Martian spacecraft. I was able to find specs for the DSN antennas that support X-band: 68.3 dBi of gain, 17 K system temperature (excluding sky and atmosphere temperature), and 110 dBw transmit power. Since the $\frac{E_b}{N_0}$ is hard to achieve, I chose the largest dish antenna (800 mm diameter) I could fit without impinging on the solar panels.

I had a lot of trouble closing the link for the downlink budget; significantly changing requirements was the only way to reasonably achieve it. I set up the equations in Julia and just kept iterating until it worked. Specifically, I had to increase transmit power from 10 to 50 W, increase the transmit duty cycle from 25% to 50% of each Earth pass, and reduce the lifetime data budget from 6 to 2 GB. ${\frac{E_b}{N_0}}_{\text{down}} = 6.5\,\text{dB}$, which corresponds to an increased bit error rate from 1e-5 to 1e-3 (1 MB/GB). Comparing the results to real spacecraft seems to indicate that either 2 GB is too ambitious, or I did something incorrectly. The Pathfinder lander has a total power budget of 35 W and an antenna 40% the size of Sappho's, so it probably has a similar $\frac{E_b}{N_0}$ with much more modest hardware.

The uplink budget was much easier: substituting the appropriate transmitter and receiver values resulted in an ${\frac{E_b}{N_0}}_{\text{up}}$ of 102 dB at DSN's maximum transmit power of 110 dBw. Unfortunate since uplink is only used for TT&C.

Command and Data Handling

Sappho's scientific payloads create a very small amount of textual data, which only needs to be uplinked once; the reduced 2 GB data budget will be spent mostly on TRIDAR and visible images. Raw images from the Curiosity rover are approximately 100 kB each, which corresponds to 10,000 images per gigabyte downlinked. The visible camera also has on-board storage for 128 GB of images. The TRIDAR 3d maps of Phobos would be very enlightening, but the data budget places a hard limit on the size and precision of the resulting pointclouds; they will be processed and stored on board during landing but not necessarily downlinked. The scintillator and mass spectrometer only need to take a single reading, both of which are small amounts of tabular data. The scintillator creates a plot of relative decay counts versus decay energy, whose size can probably be reduced further by normalizing the data or mapping peaks to isotopes. The mass spectrometer produces a similar plot of intensity versus mass/charge ratio, which can be reduced to a table of elements and their relative abundance. TRIDAR is NASA's 3d camera that uses infrared and several LIDAR sensors to create a 3d map of an unknown surface without any targets, e.g. the ISS Pressurized Mating Adapter from within the Space Shuttle payload bay. I chose TRIDAR over RADAR for imaging and landing because it is an integrated system designed for docking and has been used on prototype rovers.

I started selecting a C&DH package by looking at the Peregrine lander's computer because it also processes sensor data real-time during landing: Peregrine uses a GR712RC dual-core processor that runs at 100 MHz. To ensure Sappho has more than enough compute and interfaces for components, I chose a Moog SpaceVPX single-board computer that runs at 2 GHz (20x Peregrine) with extremely high data throughput (1-10 Gb/s). It also includes FPGAs for interfacing with external components. A Moog 400 GB data recorder with the same throughput will save all sensor data for processing; irrelevant pointclouds can be deleted if storage fills up during landing.

Calculations

```julia
md"""# Calculations"""
```

Interface with GNU Units

```julia
md"""## Interface with GNU Units"""
```

units (generic function with 1 method)

```julia
units(from) = run(pipeline(`/usr/bin/units $from`, `/usr/bin/head -n 1`))
```

units (generic function with 2 methods)

```julia
units(from, to) = run(pipeline(`/usr/bin/units "$from" "$to"`, `/usr/bin/head -n 1`))
```

```julia
using Printf
```

Thermal

```julia
md"""# Thermal"""
```

```julia
let Aₚ = 2.8, ϕ = 588, albedo_frac = 0.16, α = 0.2, ε = α, ϕ_IR = 239 * 588 / 1360
    solar = Aₚ * ϕ * α
    albedo = albedo_frac * Aₚ * ϕ * α
    IR = Aₚ * ϕ_IR * ε
    @printf "Solar: %i W\nAlbedo: %i W\nIR: %i W\nWorst case: %i W" solar albedo IR solar+albedo+IR
end
```

```
Solar: 329 W
Albedo: 53 W
IR: 58 W
Worst case: 440 W
```

Communication

```julia
md"""# Communication"""
```

```
6.476428656664837
```

```julia
# Downlink
let r = 189.9e9, r_antenna = 0.4, Pₜ = 10log10(50), Gᵣ = 68.3, Lₗ = 0.3, Lc = 0.3, R = 10log10(4e3), λ = 3e8 / 8e9, Lₐ = 5, T₀ = 55 + 273.15, NF = 1.3

    A = pi * r_antenna^2
    Gₜ = 10log10(7A / λ^2)
    Lₛ = 10log10((4pi * r / λ)^2)

    T_amp = (10^(NF / 10) - 1) * T₀ # Worst case
    T_sky = 120 # From graph
    T_sys = 10log10(T_sky + T_amp)

    P_EIRP = Pₜ - Lₗ + Gₜ
    Eb_No = P_EIRP - Lₛ - Lₐ + Gᵣ - Lc + 228.6 - T_sys - R

end
```

```
101.82117679361716
```

```julia
# Uplink
let r = 189.9e9, r_antenna = 0.4, Pₜ = 110, Gₜ = 68.3, Lₗ = 0.3, Lc = 0.3, R = 10log10(4e3), λ = 3e8 / 8e9, Lₐ = 5, T₀ = 55 + 273.15, NF = 1.3

    A = pi * r_antenna^2
    Gᵣ = 10log10(7A / λ^2)
    Lₛ = 10log10((4pi * r / λ)^2)

    T_amp = 17 # From DSN
    T_sky = 120 # From graph
    T_sys = 10log10(T_sky + T_amp)

    P_EIRP = Pₜ - Lₗ + Gₜ
    Eb_No = P_EIRP - Lₛ - Lₐ + Gᵣ - Lc + 228.6 - T_sys - R

end
```

```
▸ProcessChain([Process(`/usr/bin/units '2 GB / (1|4 * (444 day -260 day))'`, ProcessExit
```

```julia
# X-band data budget assuming 25% duty cycle
units("2 GB / (1|4 * (444 day -260 day))")
```

```
        Definition: 4025.7649 bit / s
```

```
ProcessChain([Process(`/usr/bin/units 1e-3 MB/GB`, ProcessExited(0)), Process(`/usr/bin
```

```julia
# Bit error rate
units("1e-3", "MB/GB")
```

```
    * 1
```

EPS

```
md"""# EPS"""
```

```
▶ ProcessChain([Process(`/usr/bin/units '3 200 W 588/1360' W`, ProcessExited(0)), Process
```

```
# Solar power
units("3 200 W 588/1360", "W")
```

```
	* 259.41176
```

```
▶ ProcessChain([Process(`/usr/bin/units '(2 37.8 Ah 28 V) / (1|2 sol 150 W)'`, ProcessExi
```

```
# Margin of battery capacity for continuous 150 W in shadow
units("(2 37.8 Ah 28 V) / (1|2 sol 150 W)")
```

```
	Definition: 1.1445353
```

Propulsion

```
md"""# Propulsion"""
```

```
ProcessChain([Process(`/usr/bin/units '4|3 pi (150 mm)^3 waterdensity'`, ProcessExited(
```

```
# Hypergolic mass per tank (two total)
units("4|3 pi (150 mm)^3 waterdensity")
```

```
	Definition: 14.137167 kg
```

```
ProcessChain([Process(`/usr/bin/units '24|4 220 N / (13.5 tonne + 500 kg - 8.6|2 tonne)
```

```
# Max acceleration of ATV on 1/4 its docking thrusters
units("24|4 220 N / (13.5 tonne + 500 kg - 8.6|2 tonne)")
```

```
	Definition: 0.13608247 m / s^2
```

```
ProcessChain([Process(`/usr/bin/units '4 10 lbf/(500 kg - 30 kg)'`, ProcessExited(0)),
```

```
# Average acceleration of Sappho on four thrusters
units("4 10 lbf/(500 kg - 30 kg)")
```

```
	Definition: 0.37857205 m / s^2
```

Payload

```
md"""# Payload"""
```

```
Process(`/usr/bin/units '5 250 mm pi ((31.5 mm)^2 - (30.5 mm)^2) aluminumdensity'`, Proce
```

```
# Assuming that the scintillator has a mass 5x that of its housing
run(`/usr/bin/units "5 250 mm pi ((31.5 mm)^2 - (30.5 mm)^2) aluminumdensity"`)
```

```
	Definition: 0.65737826 kg
```