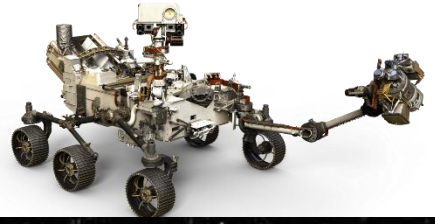
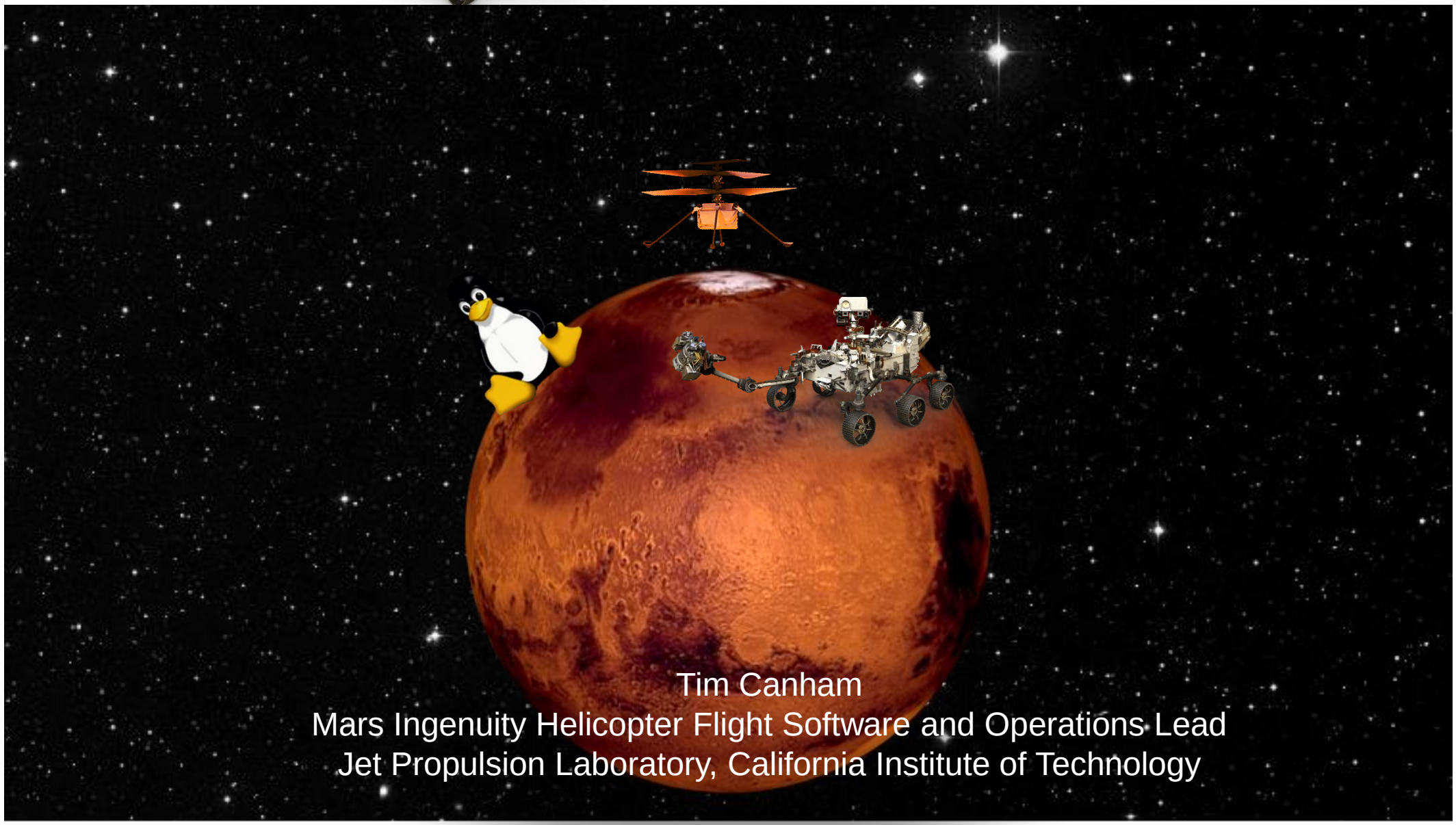


Jet Propulsion Laboratory
California Institute of Technology

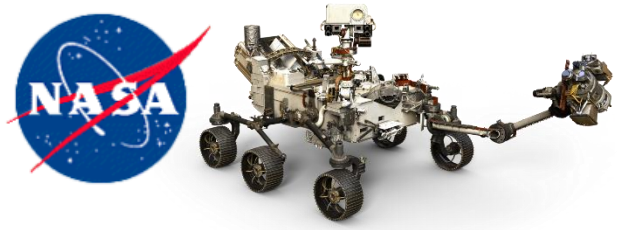


# Linux on Mars





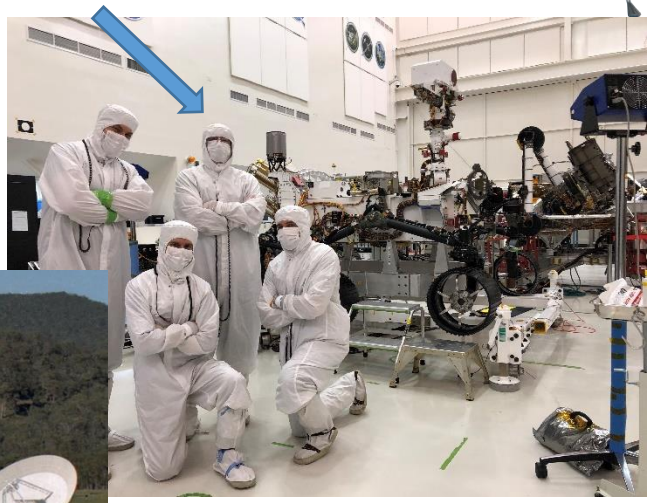
Tim Canham
Mars Ingenuity Helicopter Flight Software and Operations Lead
Jet Propulsion Laboratory, California Institute of Technology

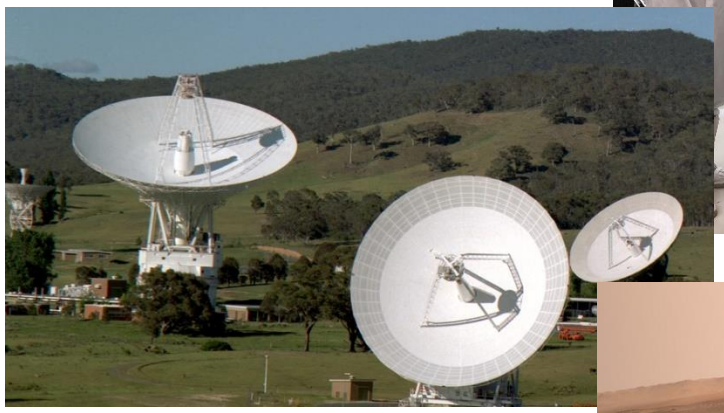


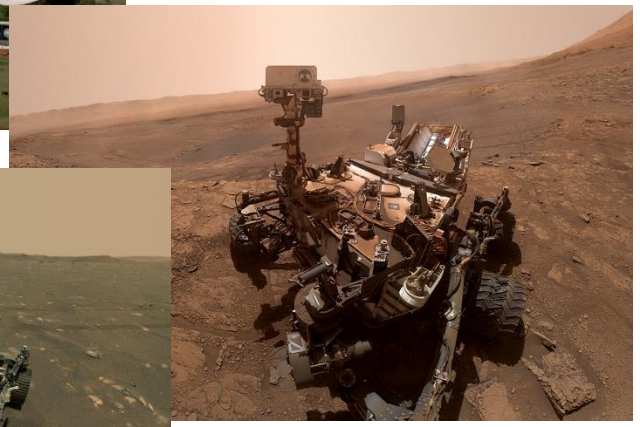
# Who I am

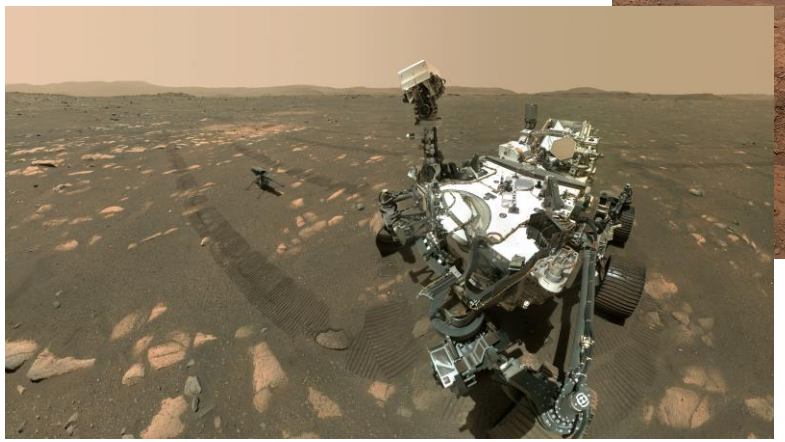


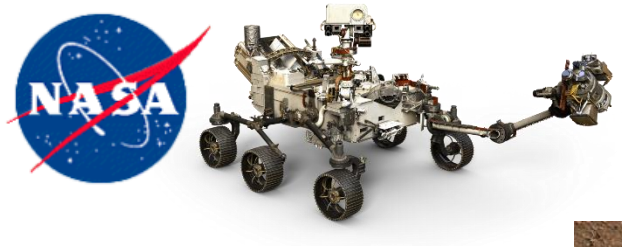
- Senior Software Engineer at NASA's Jet Propulsion Laboratory
- Projects
  - Deep Space Network
  - Cassini
  - Curiosity
  - Ingenuity
    - FSW lead
    - Operations lead
- Architect of F Prime
- Helped advance Linux at JPL





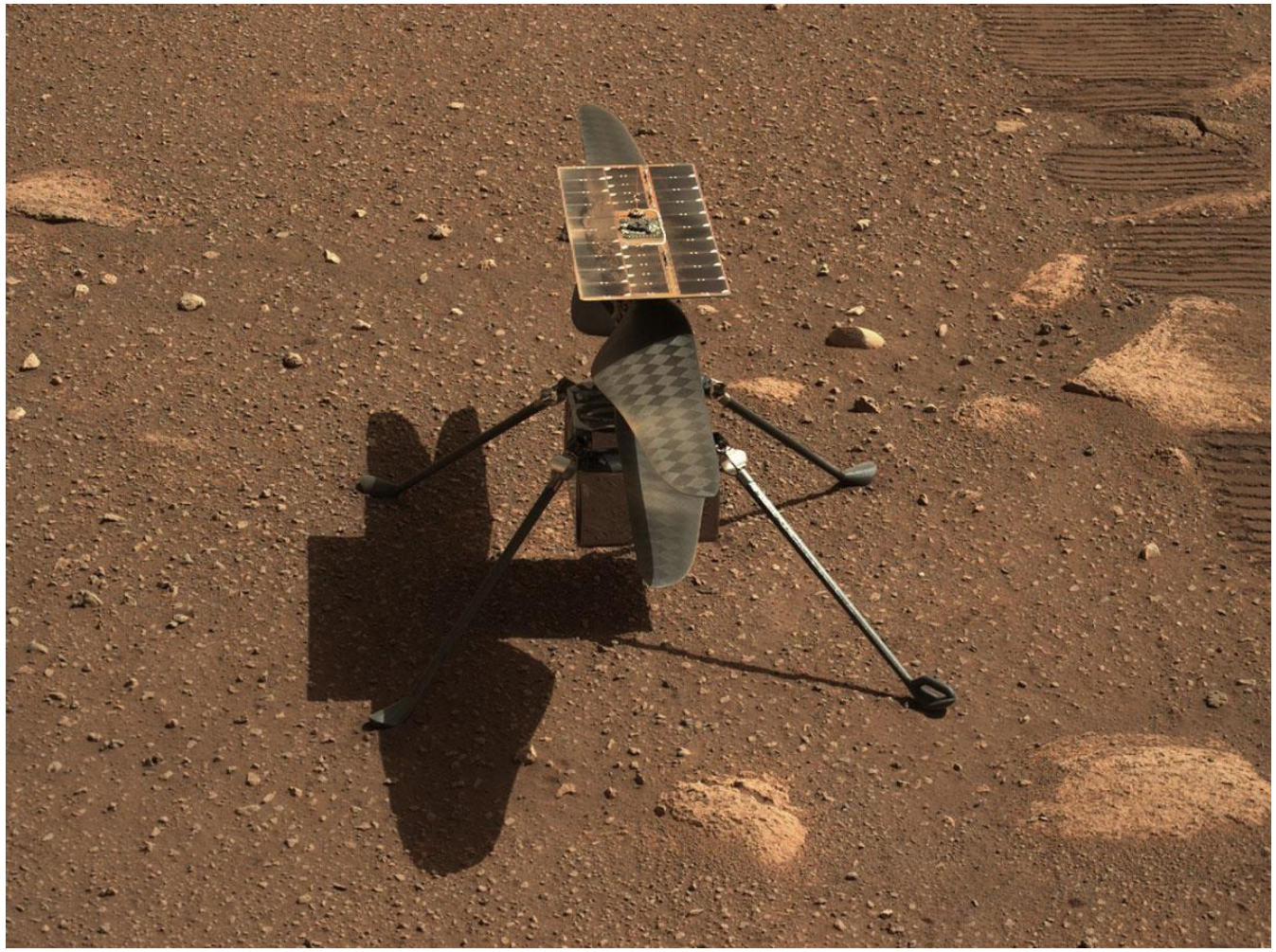






# Ingenuity Mars Helicopter



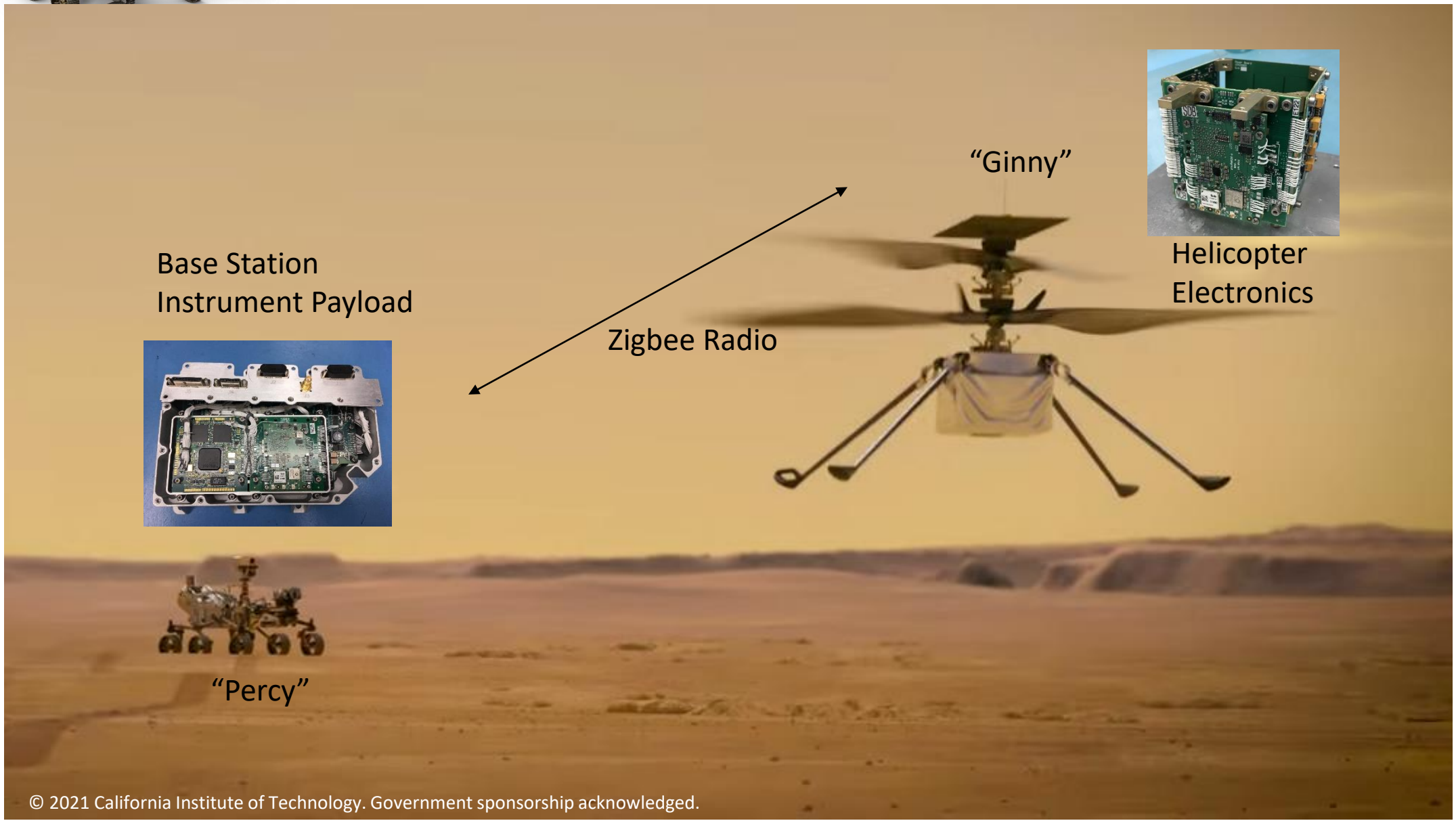


© 2021 California Institute of Technology. Government sponsorship acknowledged.

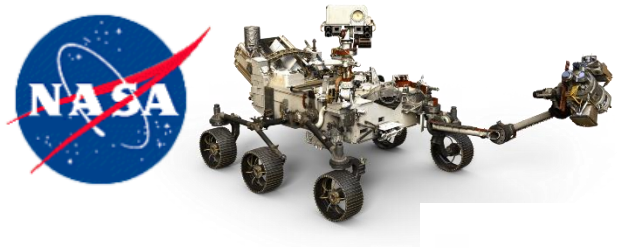


# Mars Helicopter System






© 2021 California Institute of Technology. Government sponsorship acknowledged.



# Mars Helicopter Block Diagram







Sony IMX214 13MP Color

MIPI

Omnivision OV7251 B&W

Qualcomm Snapdragon 801
4 core, 2.2GHz
2GB RAM,
32 GB eMMC
(Linux)

SPI

5MHz

GPIOs

MicroSemi ProASIC 3L FPGA

Motors

I2C

Garmin Lidar Lite LRF

SPI

8MHz

Bosch BMI-160 IMU

SPI

500 KHz

muRata SCA100T inclinometer

SPI

8MHz

GPIOs

Zigbee Radio

UART

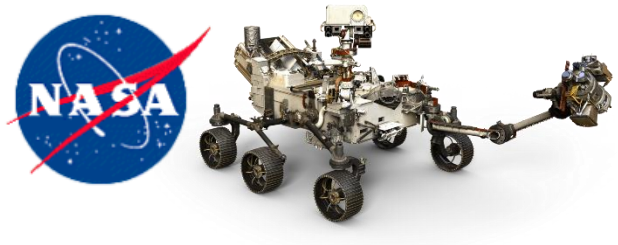
460Kb

UART

920Kb

TI TMS570LC4357 MCU
350MHz, 4MB flash,
512KB RAM
(Bare Metal)

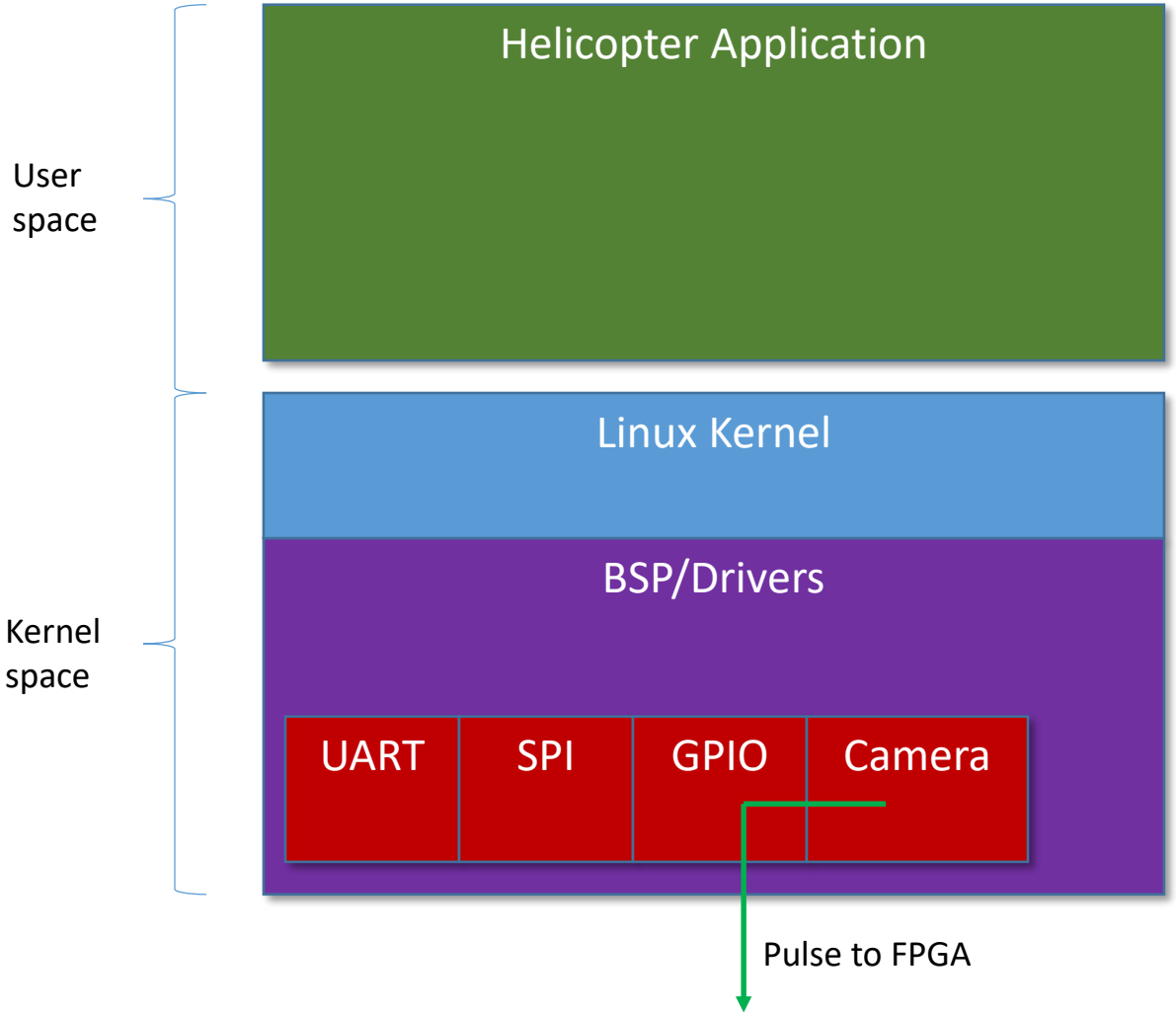
© 2021 California Institute of Technology. Government sponsorship acknowledged.



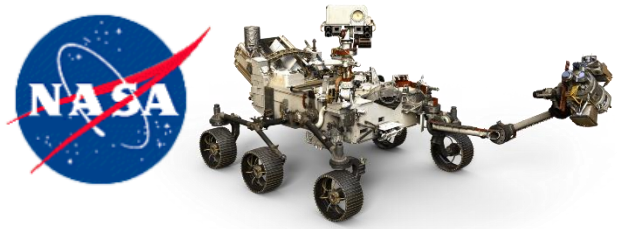
# Mars Helicopter Operating System and Software



- Linux
  - Linaro 3.4.0
  - Linux/Android hybrid
  - PREEMPT patch (No RT patch!)
  - BSP provided by Qualcomm/Intrinsyc
  - Camera drivers included with BSP
    - Modified to “pulse” camera interface with FPGA to time-stamp images
  - Linux kernel driver interface for I/O in BSP
  - Helicopter application is fully userspace
    - Runs as root



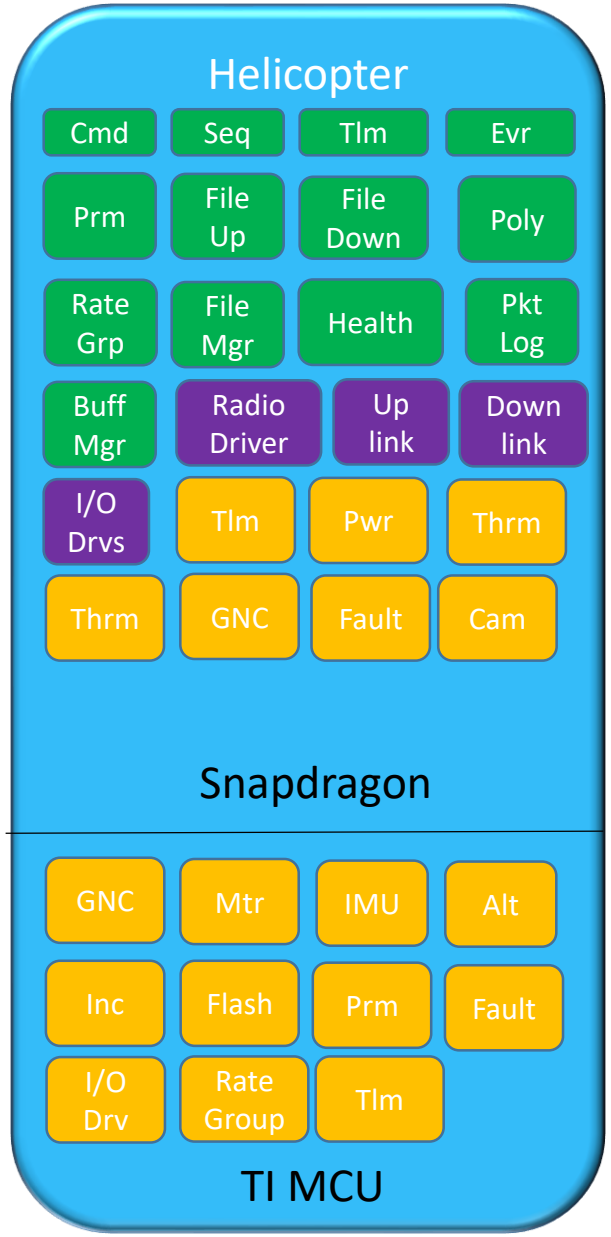

© 2021 California Institute of Technology. Government sponsorship acknowledged.

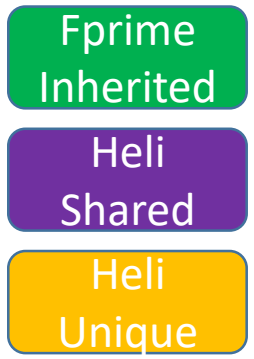


# Helicopter Application

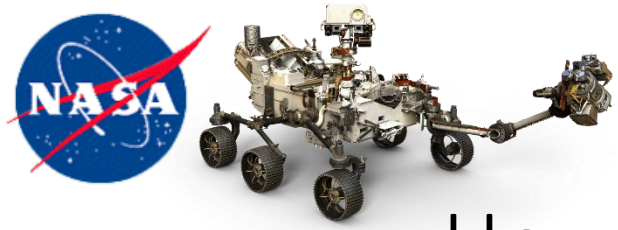


- Uses F Prime open-source flight software framework
  - https://github.com/nasa/fprime
- Tinker-toy style component architecture
- Inherits code from previous JPL missions
- Shares code internally
- Broadcasts real-time data via radio and stores higher rate telemetry to file after each flight
- 6 redundant copies with checksums started by upstart scripts




© 2021 California Institute of Technology. Government sponsorship acknowledged.





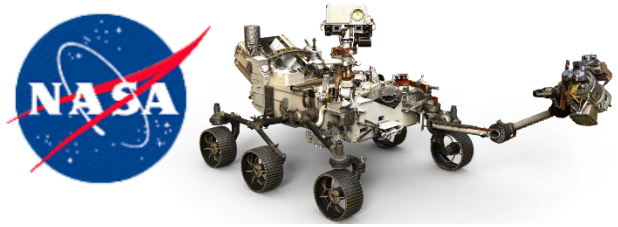
# How do we use Linux besides the application?

- We have a command to invoke arbitrary commands on the Linux command line.
  - Uses stdlib system() API call
- We have used it to:
  - Compress log files (bzip2)
  - Checksum files (md5sum)
  - List files (ls)
  - Remove older files (rm)
  - Run bash shell for various cleanup tasks
- Use “taskset –c” to select which core to use



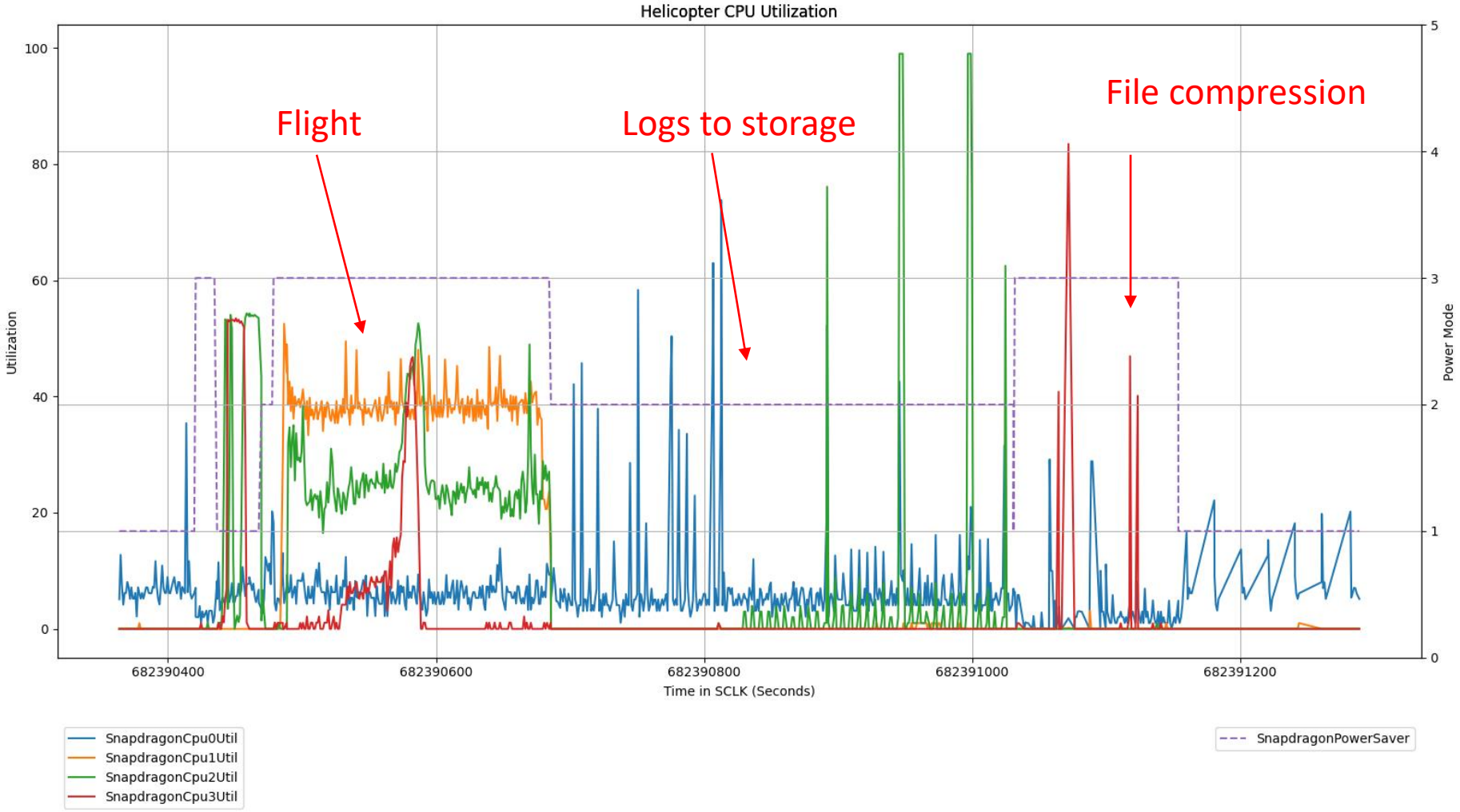


© 2021 California Institute of Technology. Government sponsorship acknowledged.



# How busy is the system?






- Core 0
  - Data handling and logging
  - Telecom
  - Device I/O
- Core 1
  - Cameras
- Core 2
  - Visual processing
  - Image logging
  - Data routing to MCU
- Core 3
  - Guidance/Naviga-tion processing

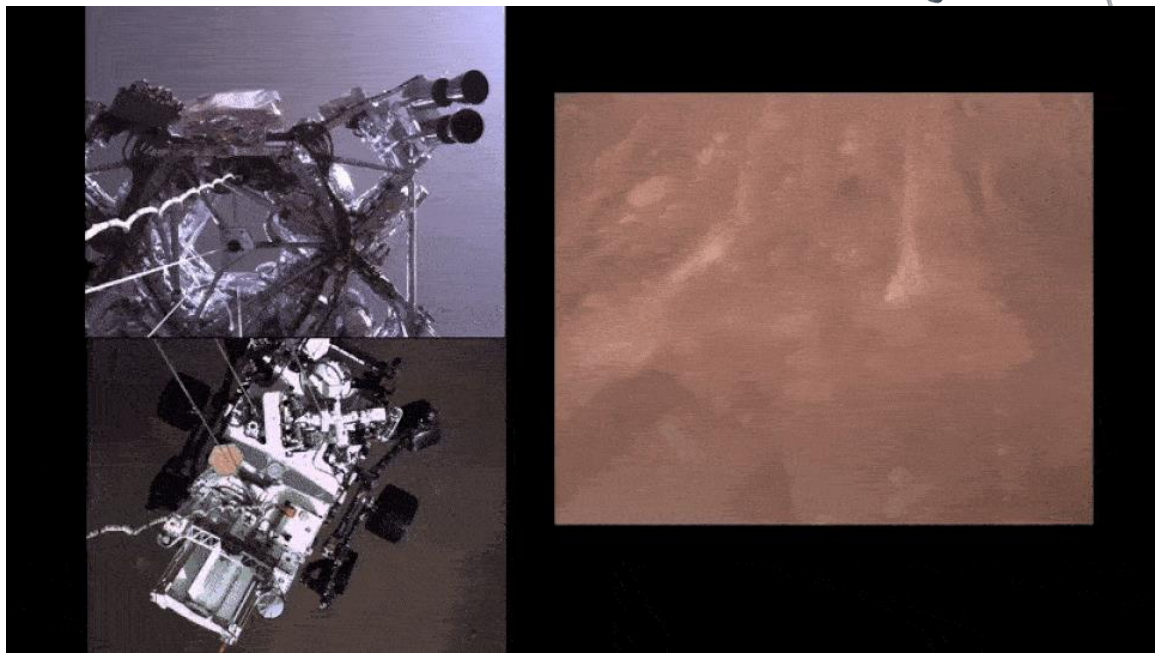
© 2021 California Institute of Technology. Government sponsorship acknowledged.



# Perseverance Rover EDL Cameras



- Perseverance Rover also had Linux-based landing camera system
  - Not involved in guidance, just recorded landing
- Ruggedized Intel Atom PC
  - More like a conventional PC
- USB cameras
  - USB cabling with hubs throughout vehicle
  - FTDI to rover interface UART
- Linux x86 kernel 4.15.7
- Used much open-source including ffmpeg and Python



Computer

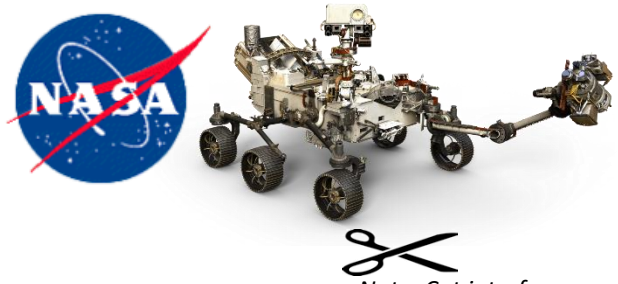





USB Hub



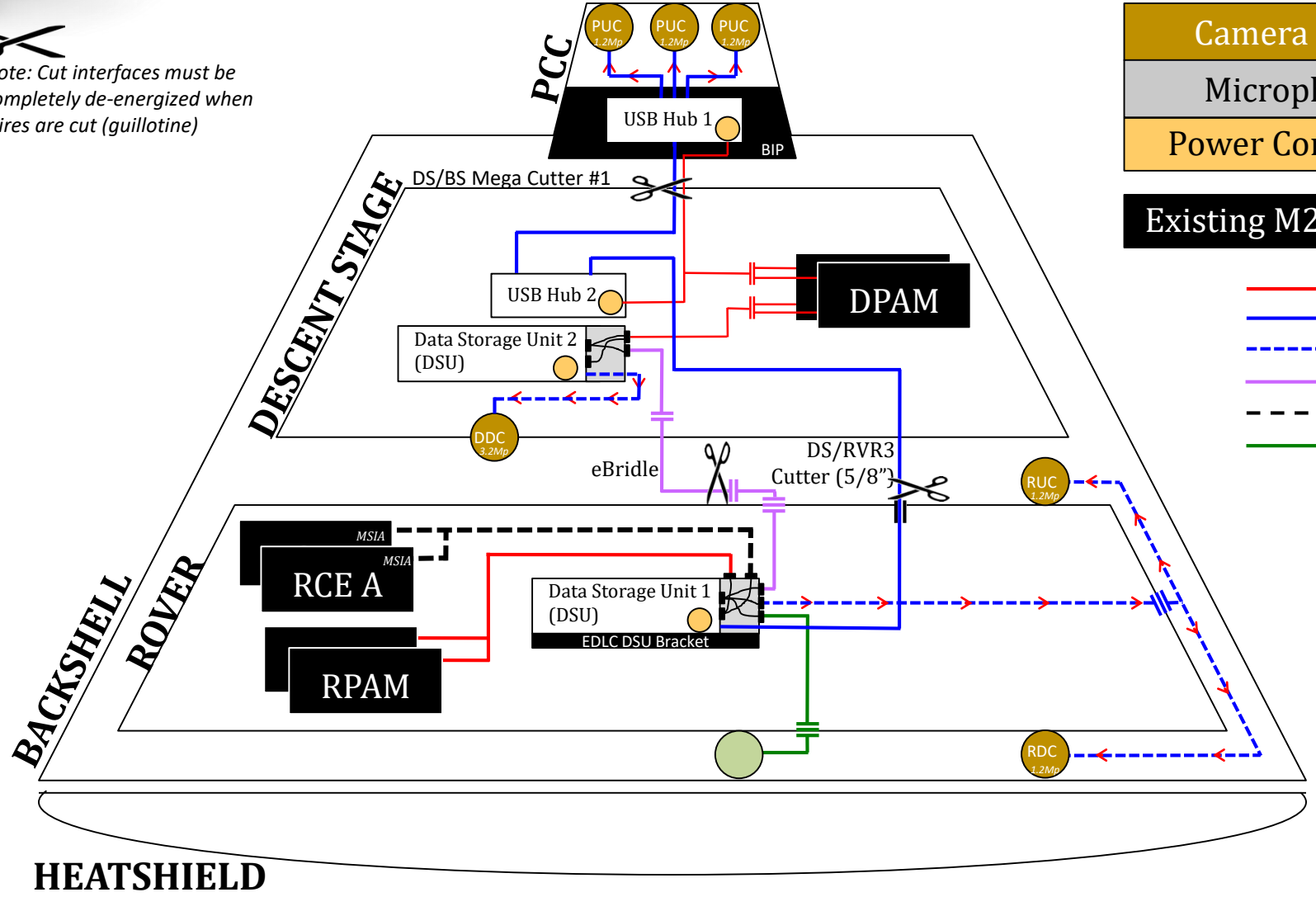
USB Cameras



# EDL Camera Functional Block Diagram



*Note: Cut interfaces must be completely de-energized when wires are cut (guillotine)*




Camera Head

Microphone

Power Converter

Existing M2020 HW

Power

USB3

USB2

ENET

RS422

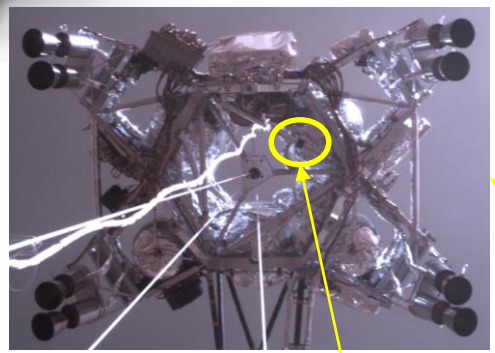
Analog

© 2021 California Institute of Technology. Government sponsorship acknowledged.



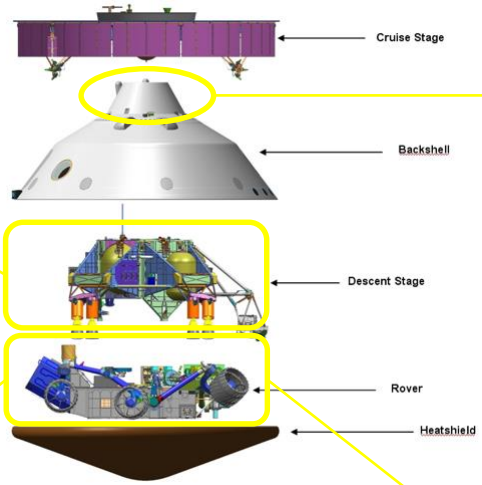
# EDLCAM Sensors

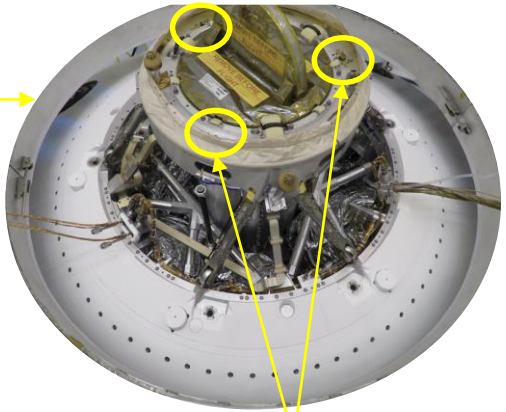




Descent Stage Downlook Camera (DDC)

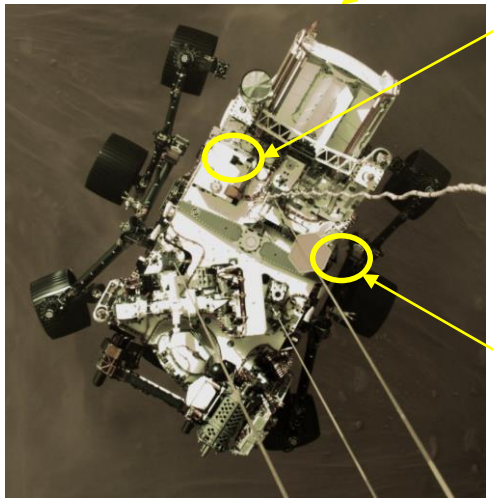






Parachute Uplook Cameras (PUC x3)





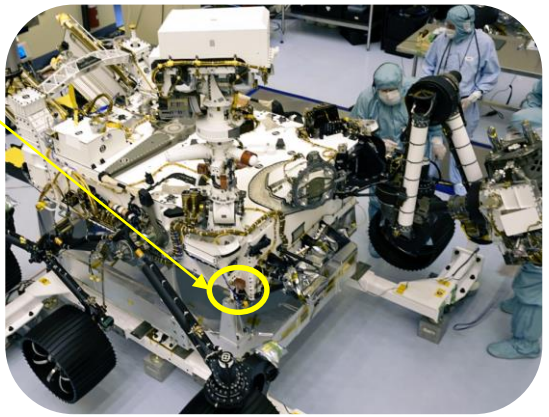
Rover Uplook Camera (RDC)



Microphone Capsule



Rover Downlook Camera (RDC)

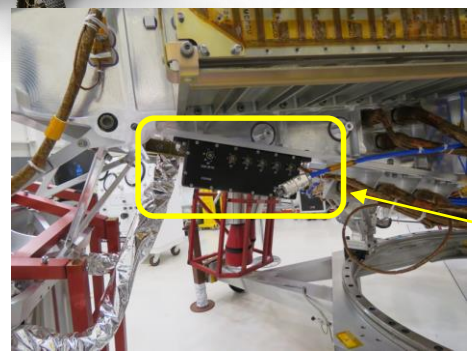


© 2021 California Institute of Technology. Government sponsorship acknowledged.

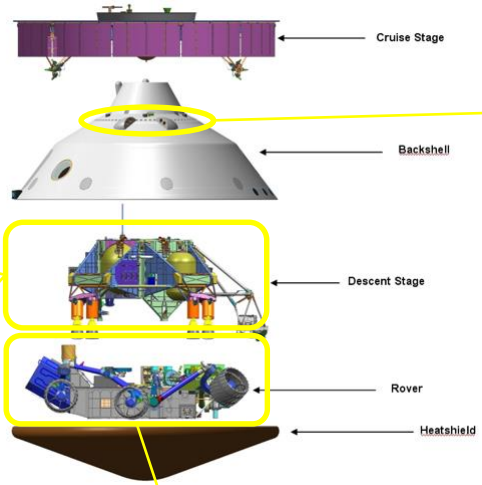


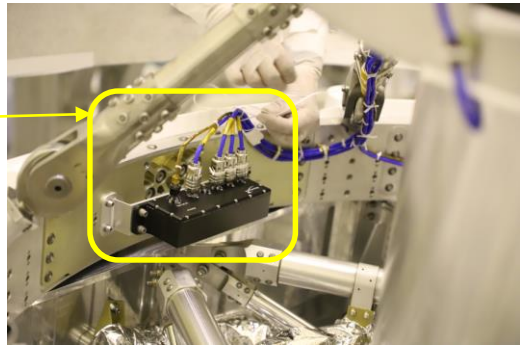
# EDLCAM Support Hardware





**Descent Stage Mounted USB Hub**

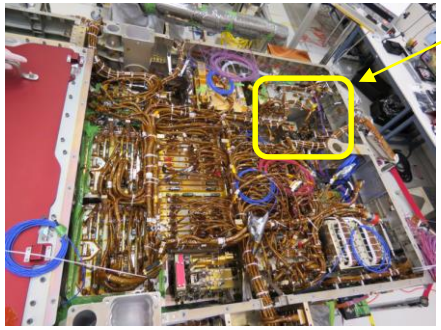


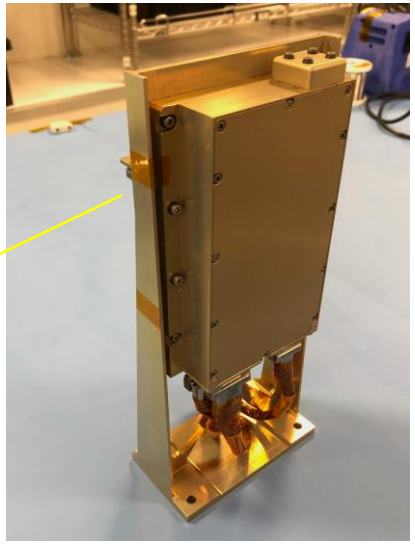



**Backshell Mounted USB Hub**



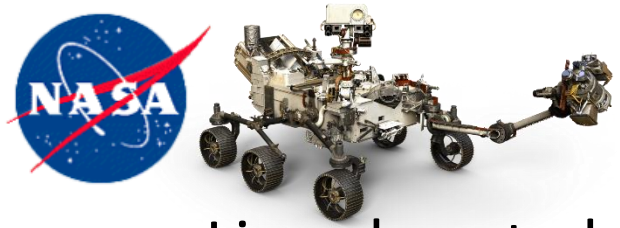
**Descent Stage Data Storage Unit**





**Rover Data Storage Unit**

© 2021 California Institute of Technology. Government sponsorship acknowledged.



# Conclusions



- Linux boosted our ability to develop quickly
  - We had standard I/O drivers
  - Manufacturer BSP was available
  - Shell/adb interface made testing much easier
  - COTS facilities like Wi-Fi, USB and standard I/O made test support equipment *much* easier
  - Allowed early prototyping on other platforms like Raspberry Pi
- Linux did very well, as long as you were aware of its limitations
  - Not real time, so built in robustness to slips
    - RT patch probably would have been better, but not available on our kernel
  - Avoid file I/O during performance critical times
  - Build in file-system level protections (ex. read-only partitions for software/Linux executables)
- Future of Linux in space exploration is rosy!