



# Flight Software System Engineering

Timothy Canham  
NASA Jet Propulsion Laboratory  
October 16, 2023

Copyright © 2023 California Institute of Technology.  
Government sponsorship acknowledged.

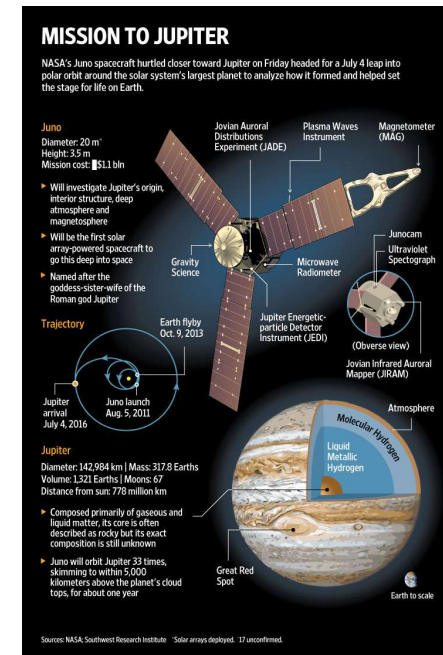


Jet Propulsion Laboratory  
California Institute of Technology



# Flight Software System Engineering

- **Definition** – Designing the various functions of the flight software and their interactions with the spacecraft and each other
- Understanding the mission objectives
  - What does the mission want to accomplish?
- Understanding the spacecraft system
  - How is the spacecraft design going to enable accomplishing the objectives?
- Concept of Operations
  - How are you going to operate the spacecraft to achieve the objectives?





# Flight Software System Engineering

- Understanding the interfaces to the system
  - How does it interact with the environment?
  - How does it interact with payloads?
  - What is the avionics platform?
  - What is the execution environment?
    - RTOS, Non-RTOS, bare metal



Orion



# Flight Software System Engineering

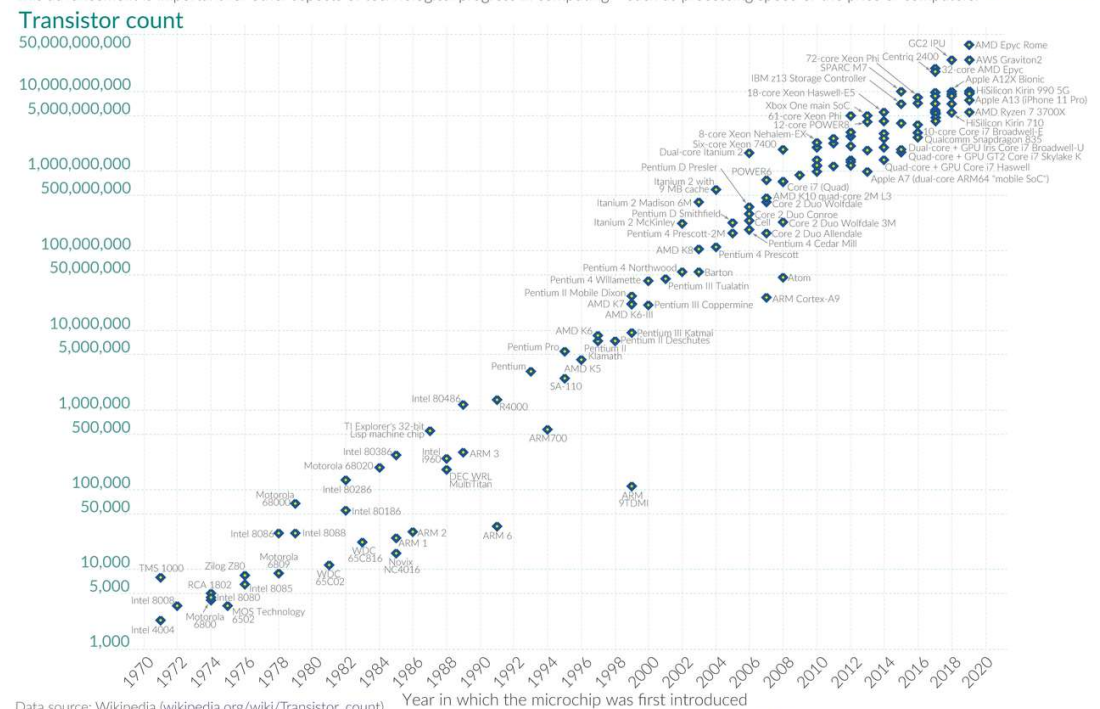
- Understanding the requirements

- Performance
  - Deadlines
  - Computing load
  - Data throughput
- Resources
  - Energy
  - Memory
  - Storage
- Activities
  - Behaviors to complete mission
- Testability

## Moore's Law: The number of transistors on microchips doubles every two years

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.

Our World  
in Data



Data source: Wikipedia ([wikipedia.org/wiki/Transistor\\_count](https://wikipedia.org/wiki/Transistor_count))

OurWorldinData.org – Research and data to make progress against the world's largest problems.

Licensed under CC-BY by the authors Hannah Ritchie and Max Roser.

Moore's Law

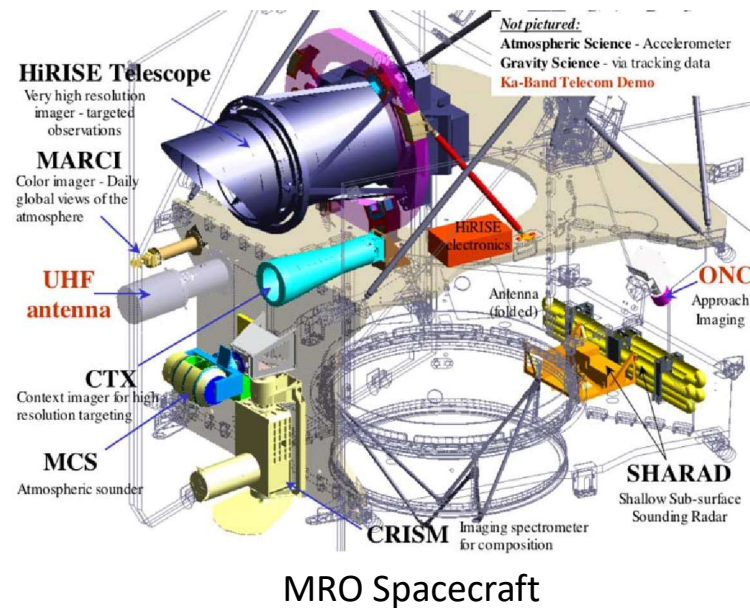


# Spacecraft Subsystems – Hardware Layering

Peripheral  
(Payload, Power,  
Telecom, etc)

I/O  
Device

Processor



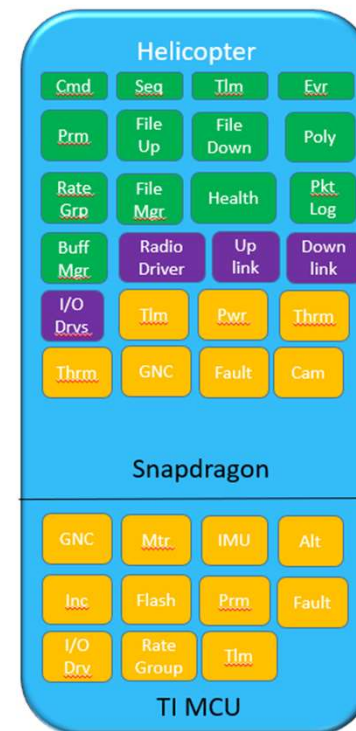
# Spacecraft Subsystems – Hardware Layering

- Hardware layers are designed (for the most part) to be modular and reusable
  - Peripherals can be reused
  - I/O can be reused
  - Processors can be reused
  - Layers can be tested independently
  - Clear interfaces
- Software
  - Software should be layered for the same purpose
  - Modularity
  - Reusability
  - Testability

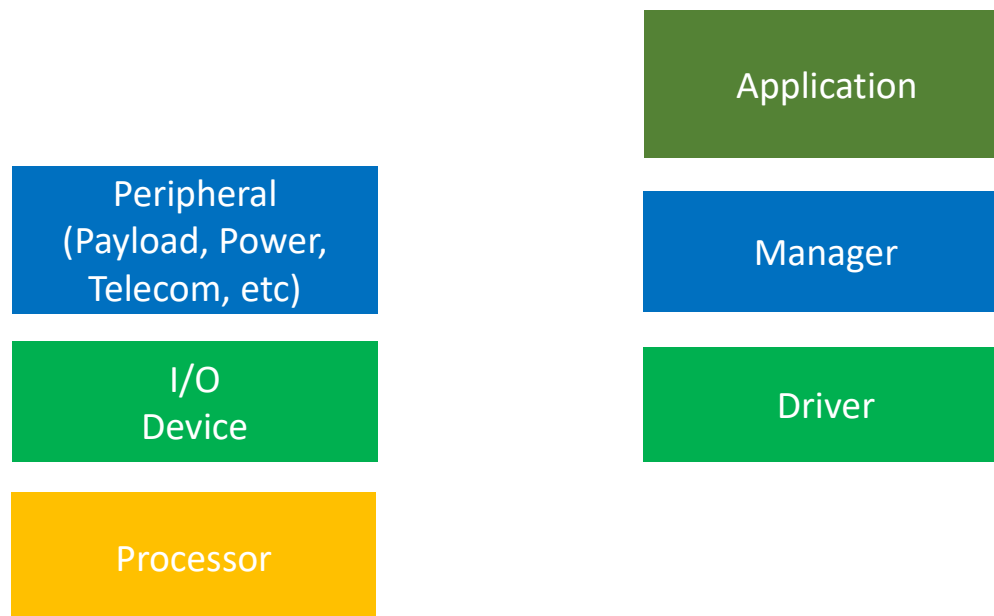


# Software System Architecture

- Decompose software into modules
  - Separation of function
  - Definition of interfaces
  - Behavioral characteristics
    - Rate groups
    - Concurrency
    - Data flow
  - Test at the unit level
  - Ownership



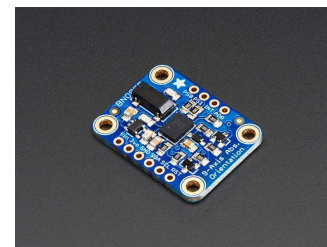
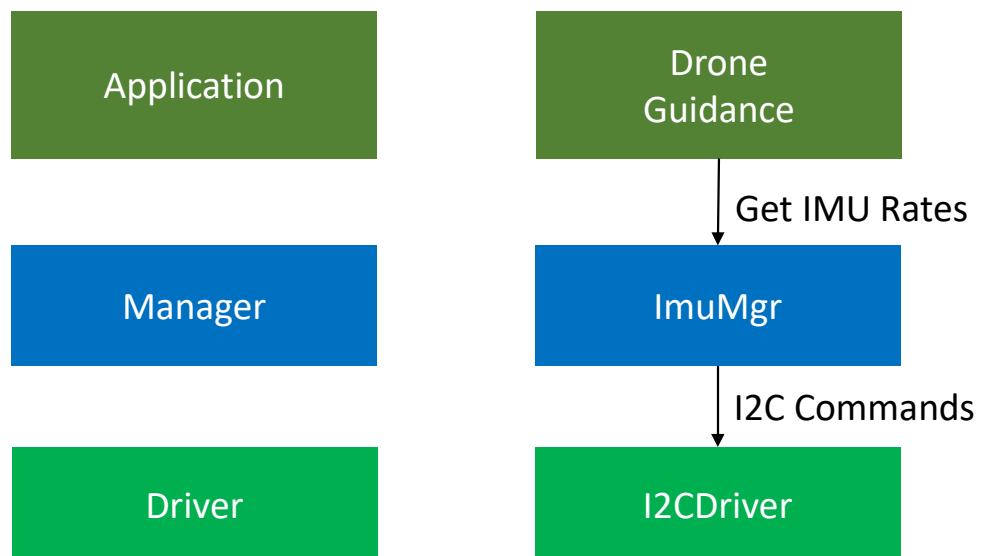
# Spacecraft Subsystems – Software Layering







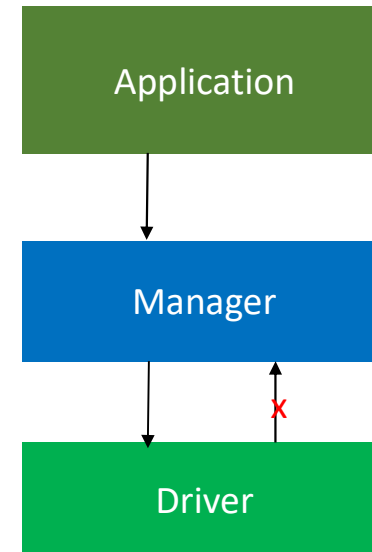
# Example - IMU



Adafruit I2C IMU

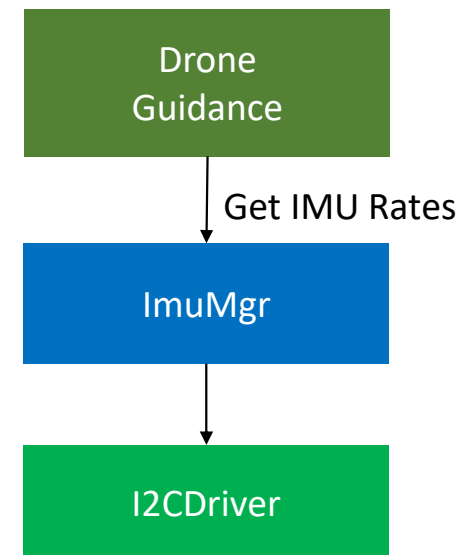
# Software Layers

- Driver
  - Dedicated to a particular hardware interface
  - Written to interact with that device only
    - Interrupt driven, polling
  - Should not need to know use of hardware
  - Provide abstracted interface
- Manager
  - Manages a particular peripheral
  - Uses the interface driver at abstract level
  - Should not know the use of the peripheral
  - Also provide abstracted interface
- Application
  - Implements mission specific use of the peripheral (and others)
  - Should not know about the driver to get to the peripheral
- Calls should be down the layers; no direct calls up
  - Registration for callbacks for asynchronous invocation



# Software Layers

- Layering allows:
  - Separation of concerns
    - Only implement logic for that layer
  - Reusability
    - Reuse software modules on other projects
  - Replacement
    - Abstraction allows layers to be replaced with other implementations with same interface
    - New driver or manager
    - Simulation
  - Testing
    - Layers can be tested at their level with stubs for lower levels
  - Fault protection
    - Faults can be handled at the appropriate layers





## References (shortened via bitly.com)

- Juno - <https://yhoo.it/2QmFedE>
- Orion - <https://bit.ly/2YVga0B>
- Moore's Law - <https://bit.ly/2K4nZgc>
- MRO - <https://bit.ly/2VUQH5D>
- IMU - <https://bit.ly/2jF4rlo>