Linux In Space
Birds-of-a-feather Meeting
SmallSat 2023

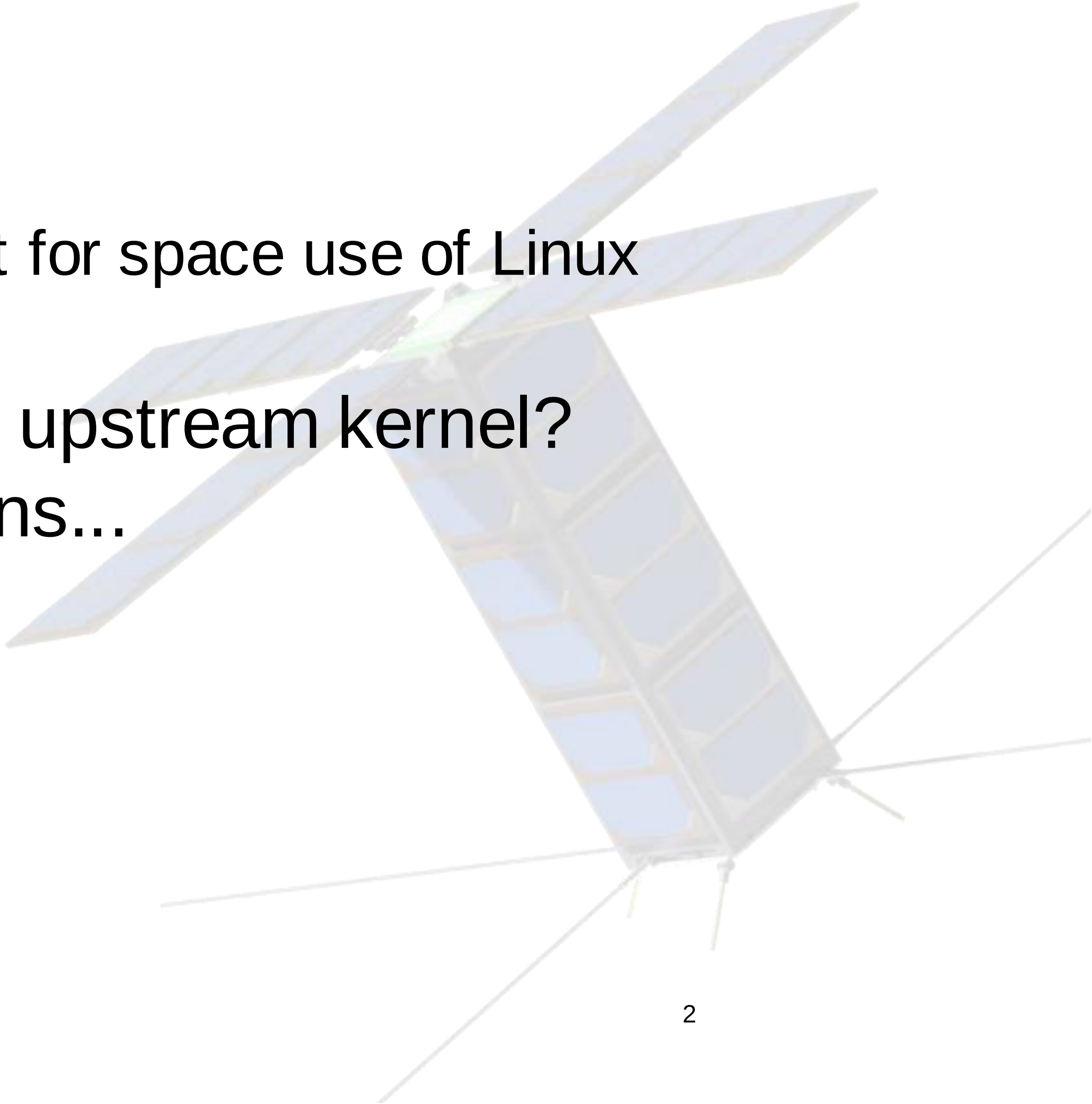
Tim Bird

Principal Software Engineer, Sony Electronics

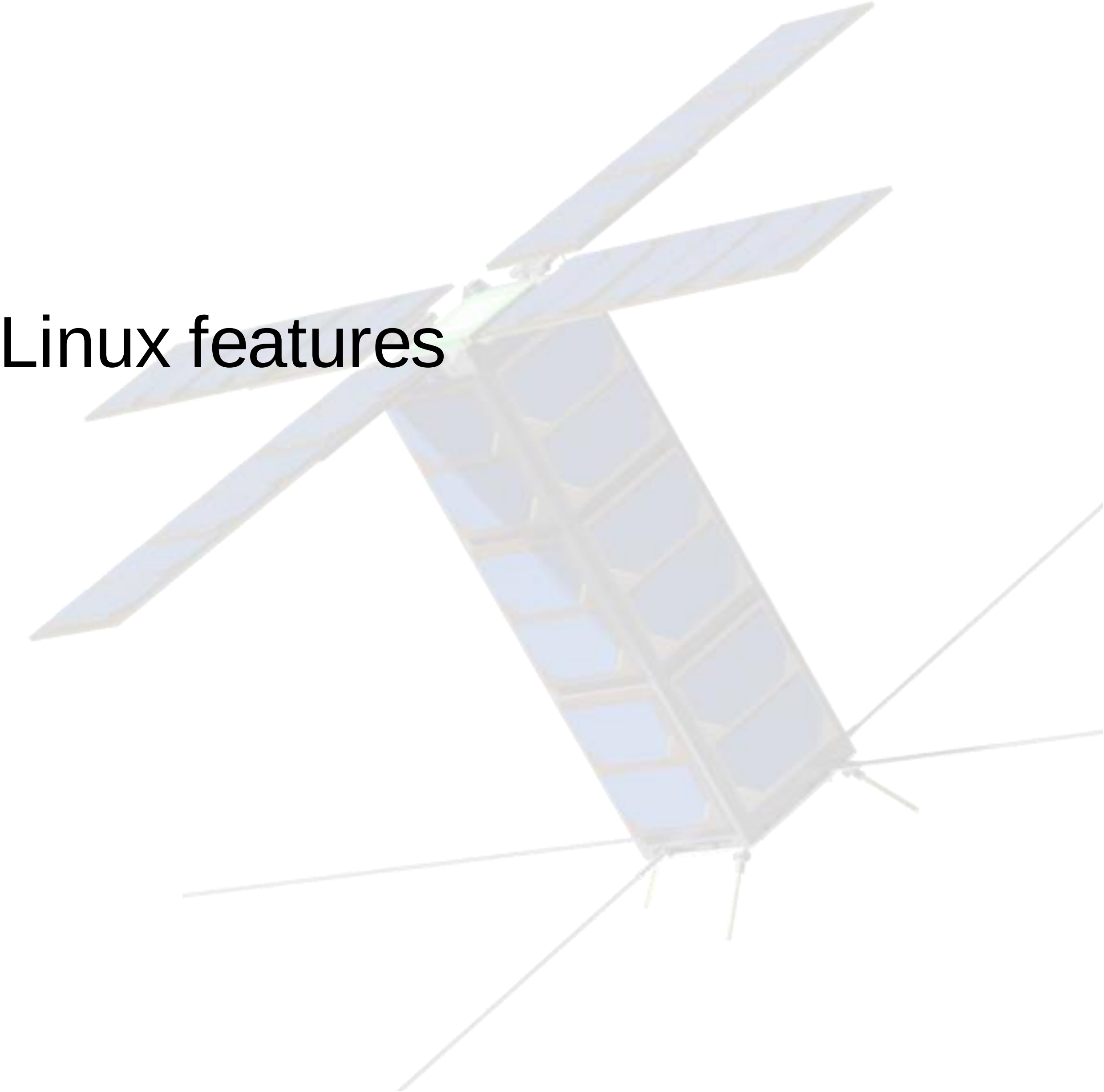
Director, Linux Foundation

Agenda

- Status of Embedded Linux
  - Features that might be of interest for space use of Linux
- Linux4Space project
- What features are desired from upstream kernel?
- Status of specific space missions...

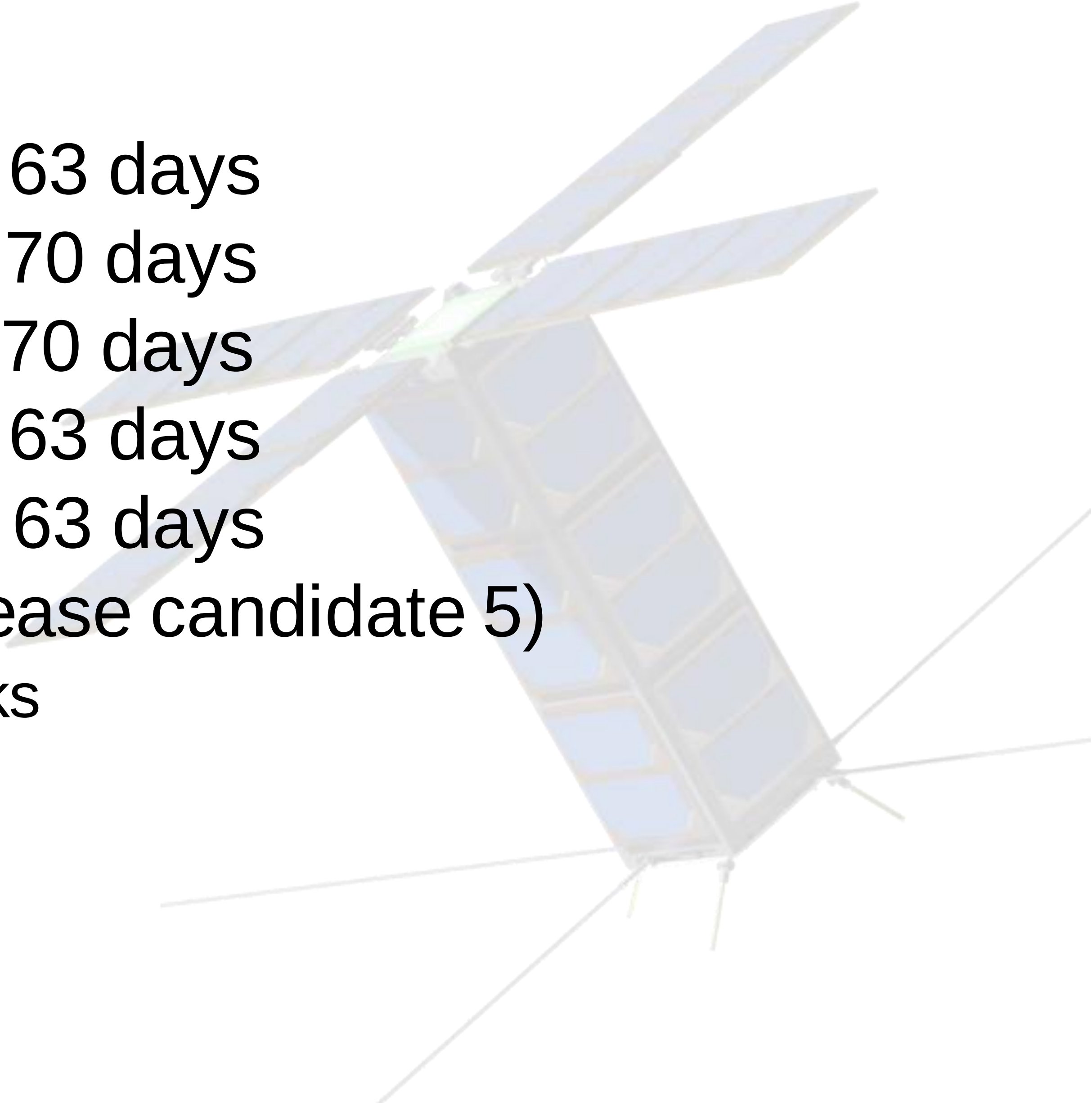


Linux Kernel



- Release cadence
- Some relatively new features
- Status of historical embedded Linux features

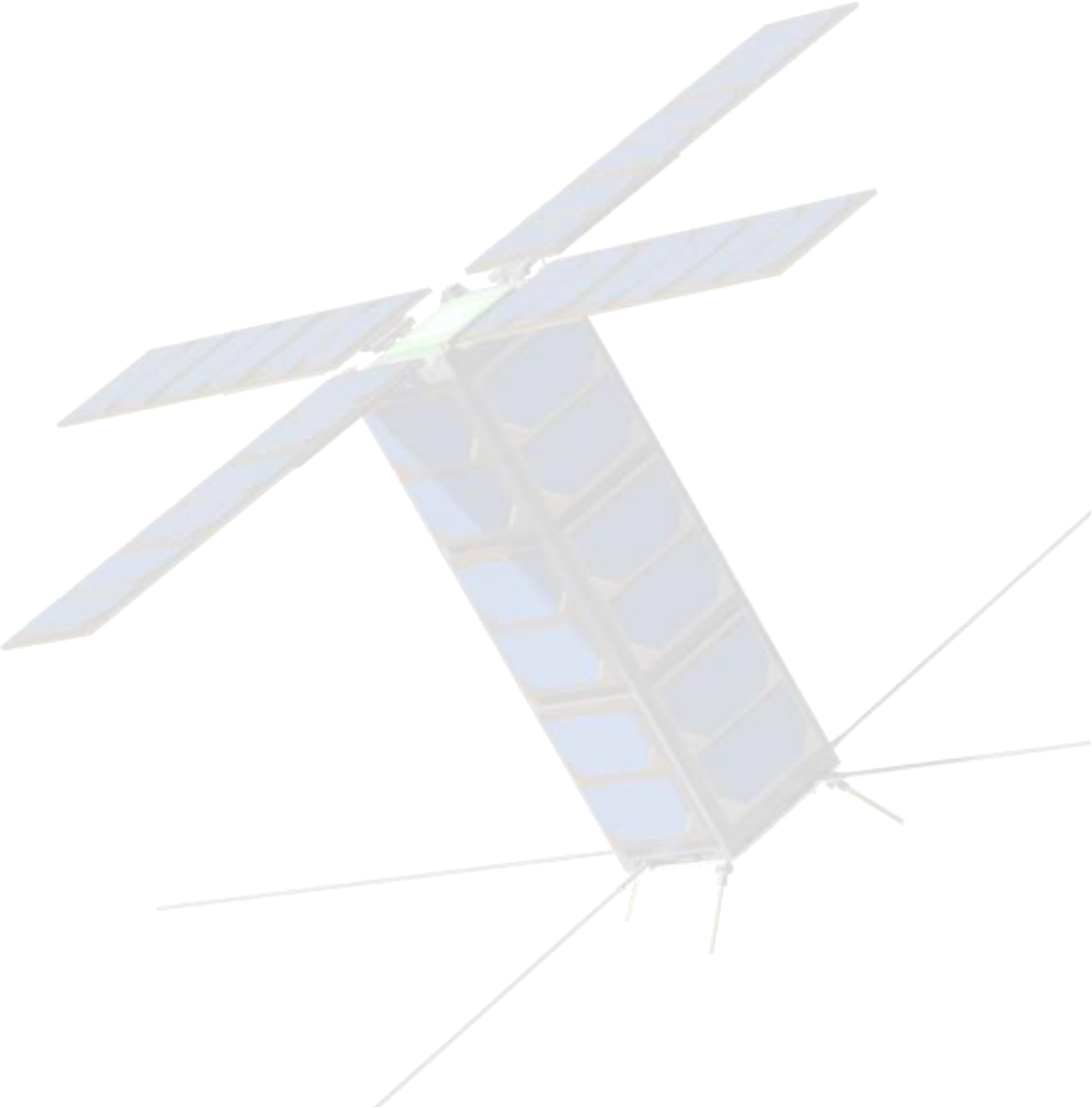
Kernel Versions



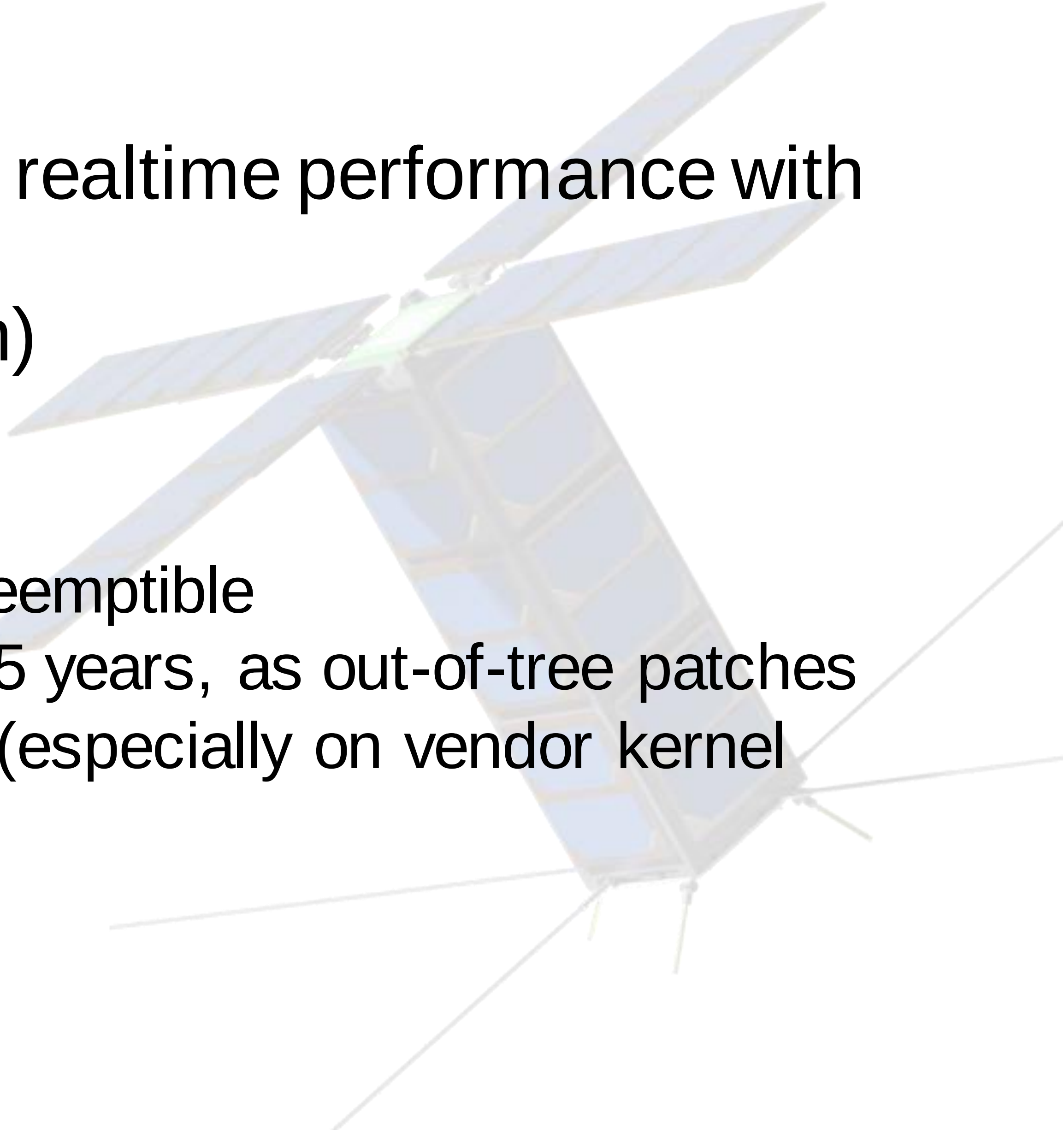
- Linux v6.0 – 2 Oct 2022 – 63 days
- Linux v6.1 – 11 Dec 2022 – 70 days
- Linux v6.2 – 19 Feb 2023 – 70 days
- Linux v6.3 – 23 Apr 2023 – 63 days
- Linux v6.4 – 25 Jun 2023 – 63 days
- Currently at Linux v6.5-rc5 (release candidate 5)
  - Expect 6.5 release in 2 to 3 weeks

Some interesting features

- Realtime status
- eBPF
- Rust
- io_uring



Real-Time



- There are several approaches to realtime performance with the Linux kernel
- Xenomai ("dual-kernel" approach)
  - RTOS hypervisor with Linux guest
- PREEMPT_RT patch set
  - Make all kernel code paths fully preemptible
  - Have been working on it for over 15 years, as out-of-tree patches
  - Large patch set was hard to apply (especially on vendor kernel trees)

PREEMPT_RT mainlining

- RT-enabling stuff has trickled in over a period of 10 years
  - lock cycle detection
  - priority inheritance handling
  - re-organization of IRQ front and backend handling
  - Lots of algorithm improvements
    - ie. shortening of non-preemptble sections
- Key feature (preemptible spinlocks) was mainlined in v5.16 (January, 2022)
- Patches have been going in continuously – through 6.4 (June 2023)

PREEMPT_RT - What’s left

- What’s left in PREEMPT_RT patches out of mainline:
  - last year (2022):
    - About 1300 lines of code, affecting 92 files (in 51 patches) (!!)
  - this year (2023) (patches-6.4-rc5-rt4):
    - About 3100 lines of code, affecting 93 files (in 85 patches)
      - Number of lines could be off due to printk work-in-progress
    - Some changes to the printk, 8250 serial driver, the core scheduler, some locking and timer tweaks, and a few other places.
  - People are still anxious for Linux RT without having to apply a patch
    - Thomas said in June that printk changes are the blocker to the patch that allows enabling PREEMPT_RT in the mainline (Torvalds) kernel
- See https://mirrors.edge.kernel.org/pub/linux/kernel/projects/rt/6.4/

PREEMPT_RT – out of mainline status

- IMHO – the patches are small enough that it doesn't matter that there's still residual bits out-of-tree
  - Every board and vendor's kernel has out-of-tree patches!
    - e.g. Most Sony products have several hundred out-of-tree patches
  - It's easy to apply the RT patch set, and unlikely to be complicated by vendor patches
- Bigger concern is whether in-kernel preemption latencies can be kept low
  - Requires onoing vigilance by the kernel realtime developers
    - e.g. To prevent driver developers from making RT-affecting mistakes

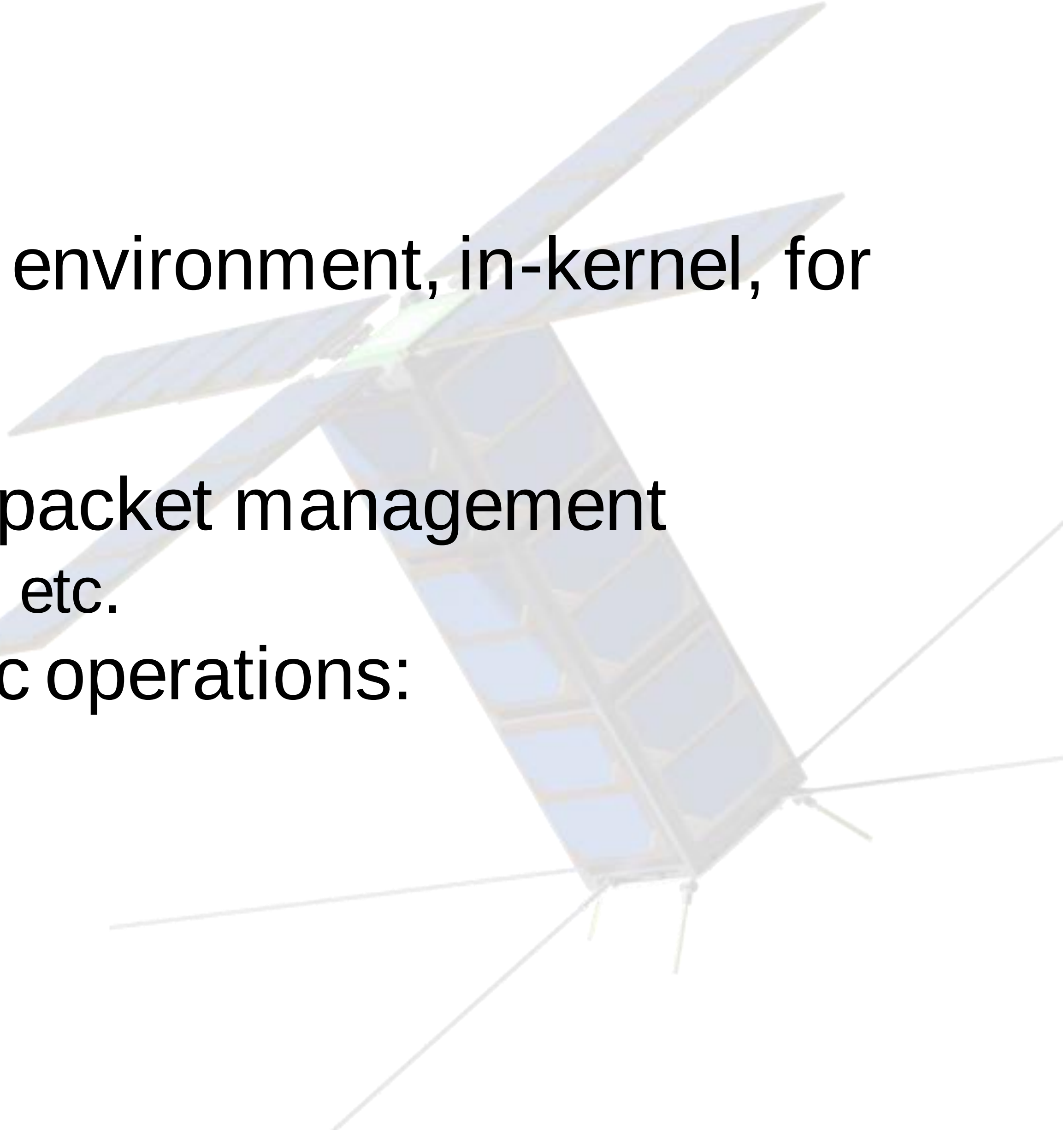
Status of realtime performance

- Worst-case wakeup latency for a realtime task, with a stress workload = 50 microseconds
  - UNLESS you see massive L1 cache interference from another processor
  - Then it goes to about 600 microseconds worst-case latency
  - This is workload and cache-size dependent
    - ie You still have to do time and space partitioning of your workloads
- Source:
  - "Evaluation of PREEMPT_RT in Virtualized Environments", Jan Altenberg, Open Source Automation Development Lab (OSADL), Embedded Linux Conference, June 2023
  - Slides: https://elinux.org/images/0/05/Preempt_rt_virtualization.pdf
  - Video: https://youtu.be/yOuQ4opLkQo

Realtime resources

- For latest presentations on realtime status see:
  - Presentations from realtime workshop (June 2023)
    - Youtube playlist of talks:
      - https://www.youtube.com/watch?v=NWVWXtfOzXM&list=PLbzoR-pLrL6oEVSWhTJHb8fYaL88tACo8&pp=iAQB
  - Presentations from Embedded Linux Conference (June 2023)
    - https://elinux.org/ELC_Europe_2023_Presentations

eBPF



- extended Berkely Packet Filters
- Provides a sandboxed execution environment, in-kernel, for dynamically loaded code
    - ie, an in-kernel virtual machine
- Developed originally for network packet management
    - For tunnels, bridges, routers, VPN, etc.
- Now used for all kinds of dynamic operations:
    - Security checks, tracing, and more

eBPF resources

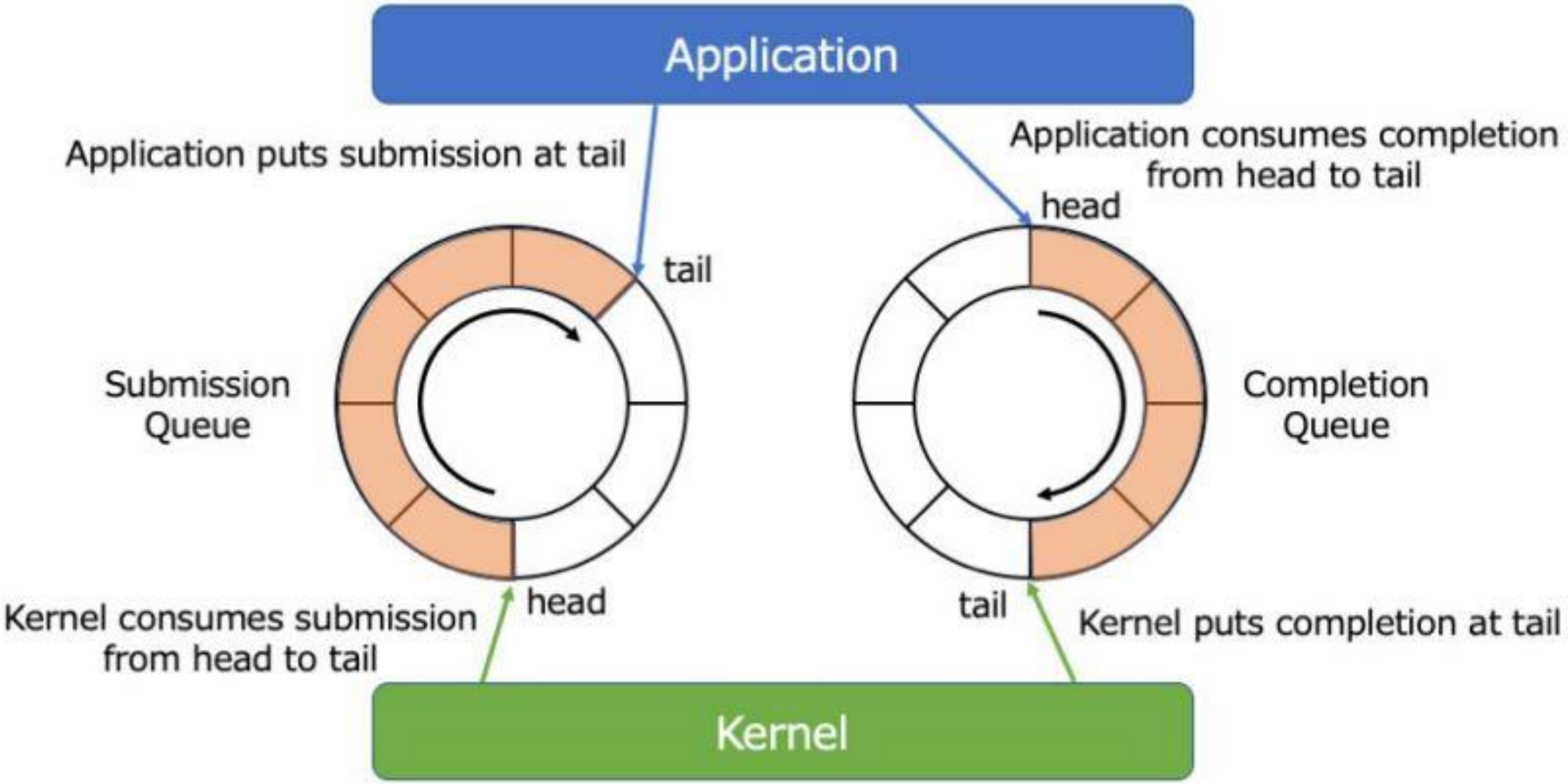
- "A Thorough introduction to eBPF", Matt Fleming, LWN.net, December 2 2017
  - https://lwn.net/Articles/740157/
- LWN.net kernel index of BPF articles:
  - https://lwn.net/Kernel/Index/#BPF

Rust

- Initial Rust support added to kernel in 6.1 (December 2022)
- Rust support continues to go into the mainline kernel
  - But it has not been used for a “real” driver yet
- My own impression:
  - Rust is being used more and more throughout the industry
    - ex: KataOS = secure operating system written by Google
    - See https://www.phoronix.com/news/Google-KataOS
- It will remain experimental for at least 2 more years
  - Then, there will be delays until complex drivers are written with it

io_uring

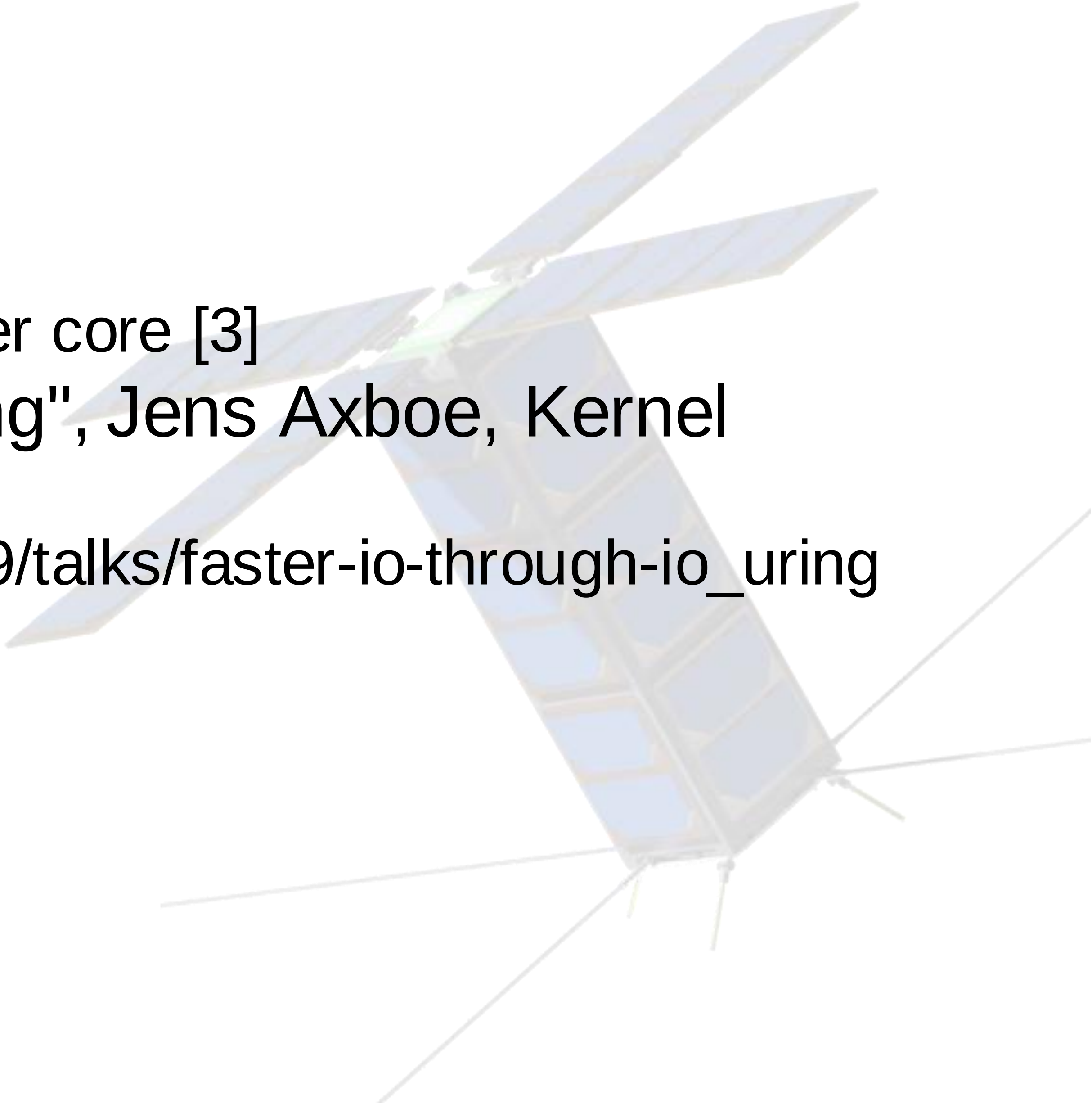
- io_uring is a relatively new system for asynchronous I/O
  - First showed up in kernel v5.1 (May 2019)
- Improves I/O performance by:
  - eliminating system call overhead for sequential operations
  - performing I/O asynchronously with user-space process
- Overview:
  - User-space process fills a queue with I/O operations
  - Kernel operates on the queue, and provides notifications of completion (on a separate "completion queue"), without waiting for user-space
- Work with both file systems and networking I/O


Application
Application puts submission at tail
Application consumes completion from head to tail
head
tail
Submission Queue
Completion Queue
head
tail
Kernel consumes submission from head to tail
Kernel puts completion at tail
Kernel

Source: https://medium.com/nttlabs/rust-async-with-io-uring-db3fa2642dd4

io_uring performance

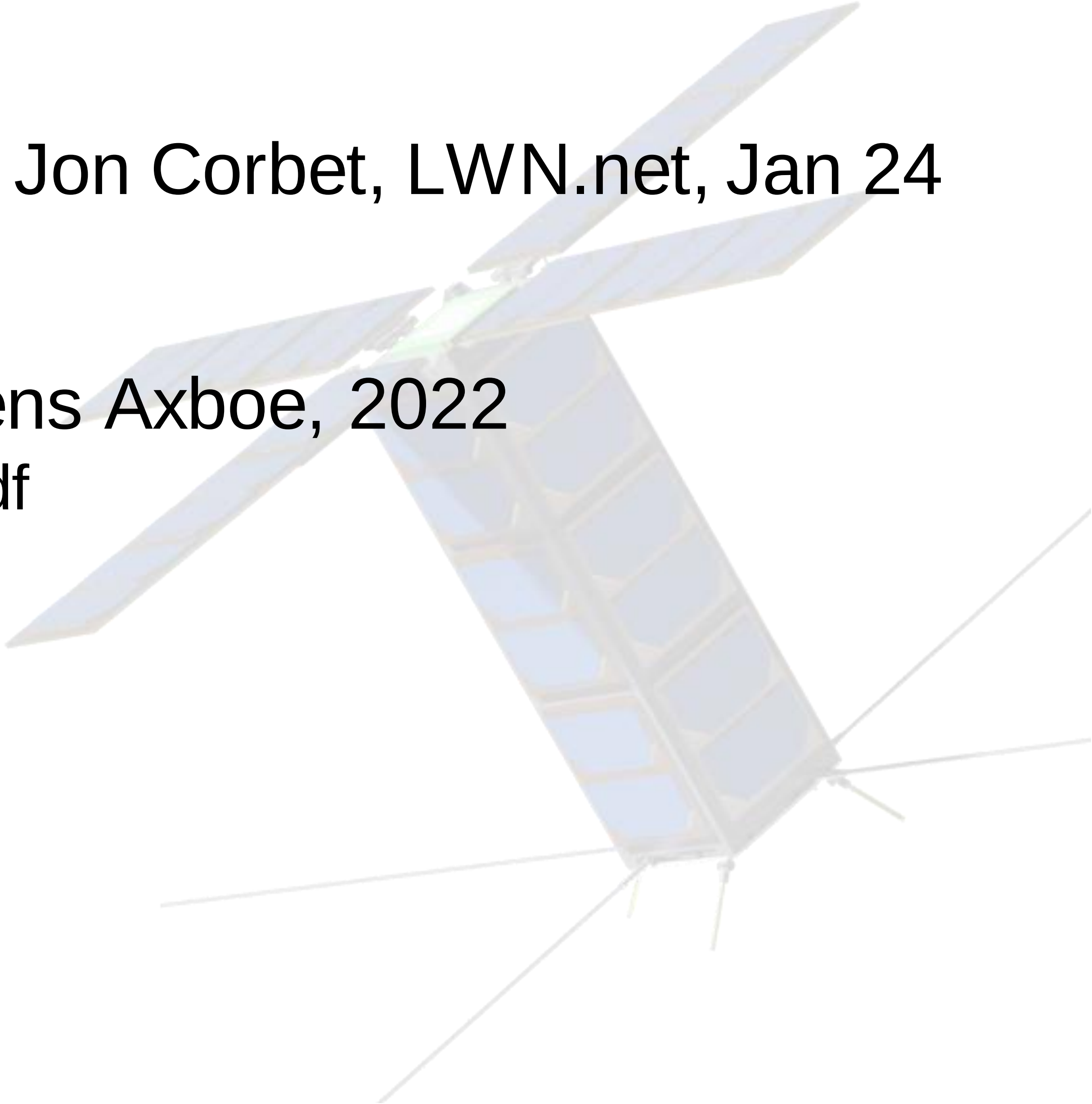
- One performance data point:
  - AOI -> 500K IOPS per core
  - io_uring -> 1 to 2 million IOPS per core [3]
- See: "Faster IO through io_uring", Jens Axboe, Kernel Recipes talk, 2019
  - https://kernel-recipes.org/en/2019/talks/faster-io-through-io_uring



io_uring work in progress

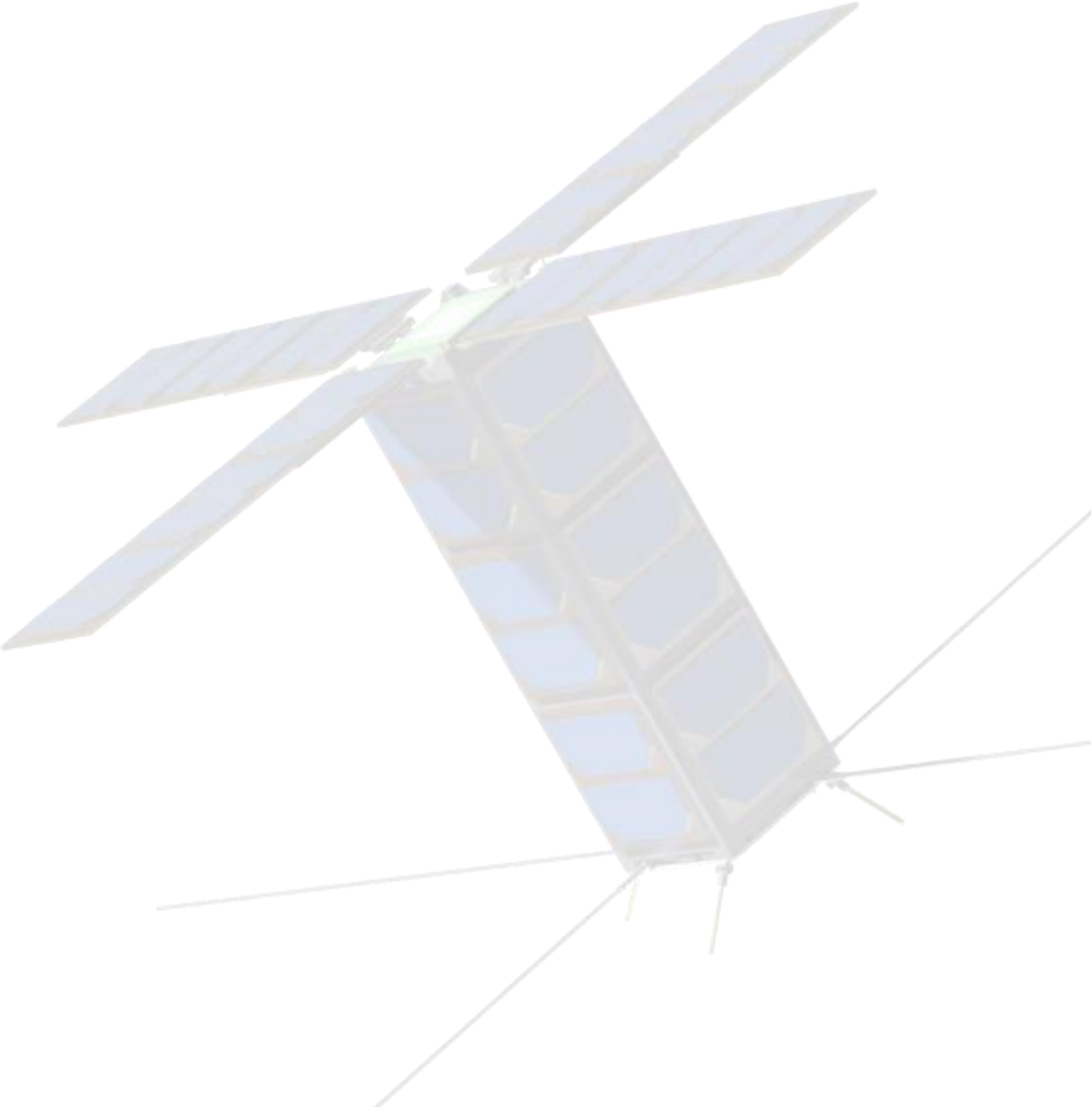
- Work being done on io_uring_spawn
  - Is more efficient than traditional user-space based 'fork & exec'
    - fork & exec almost always discards the parent’s code immediately
  - Can do IORING_OP_CLONE followed by IORING_OP_EXEC
    - All inside the kernel, without interaction with user space
  - 6-10% faster than vfork() and 30+% faster than posix_spawn()
  - See https://lwn.net/Articles/908268/

io_uring resources

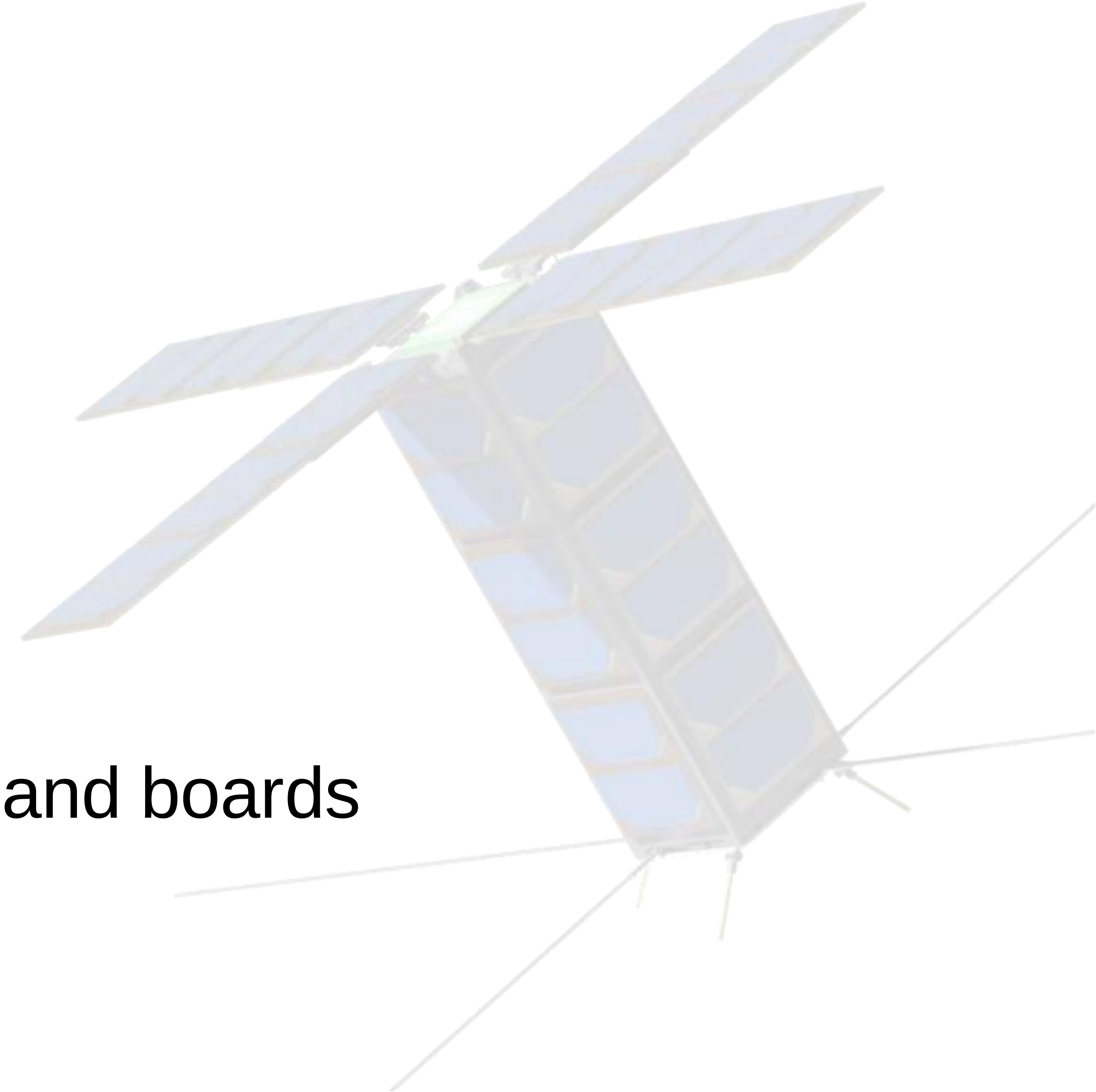


- "The rapid growth of io_uring", Jon Corbet, LWN.net, Jan 24 2020
  - https://lwn.net/Articles/810414/
- "What's new with io_uring"?, Jens Axboe, 2022
  - https://kernel.dk/axboe-kr2022.pdf

Status of historical focus areas

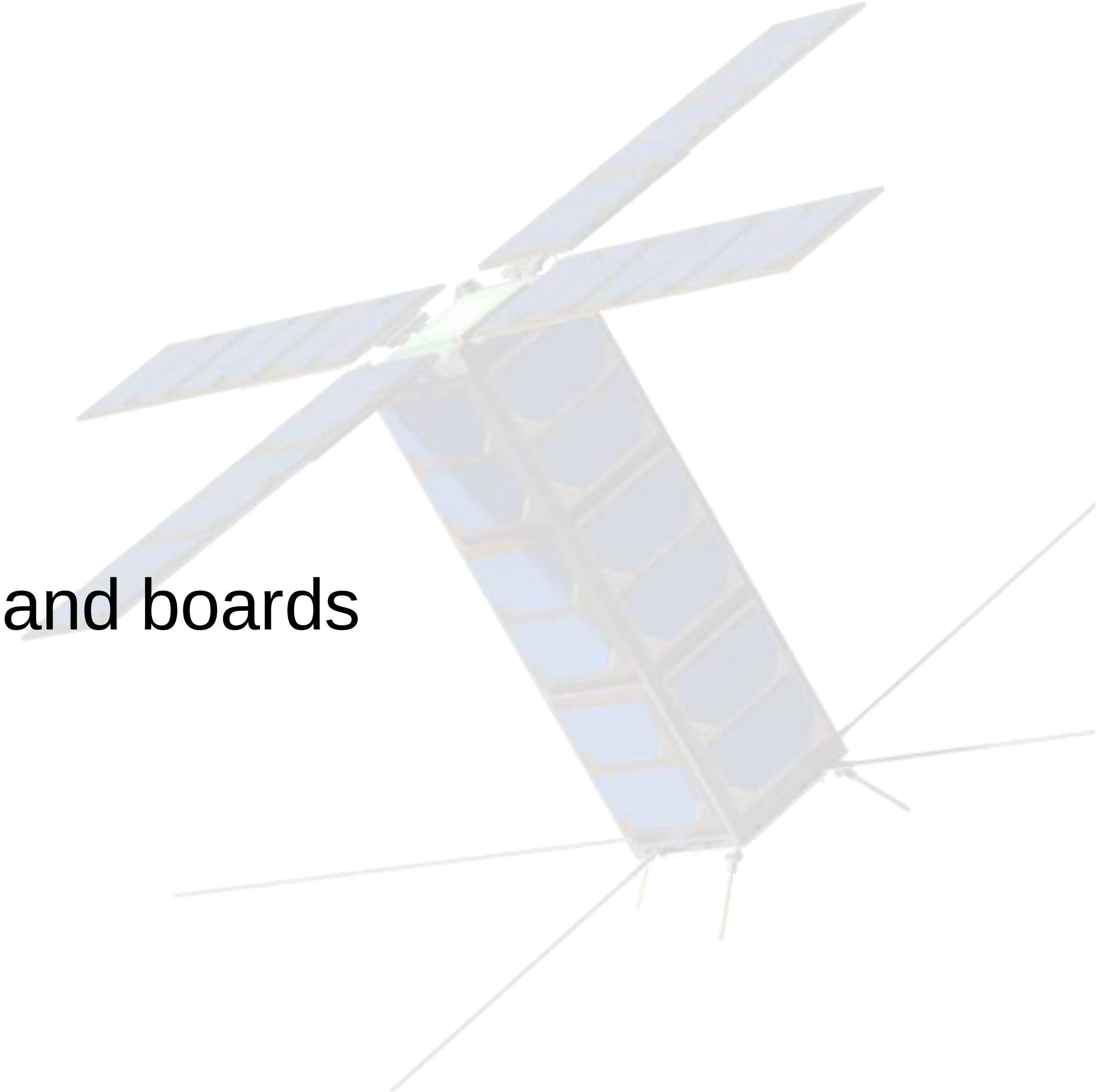


Historical Embedded Linux Focus areas



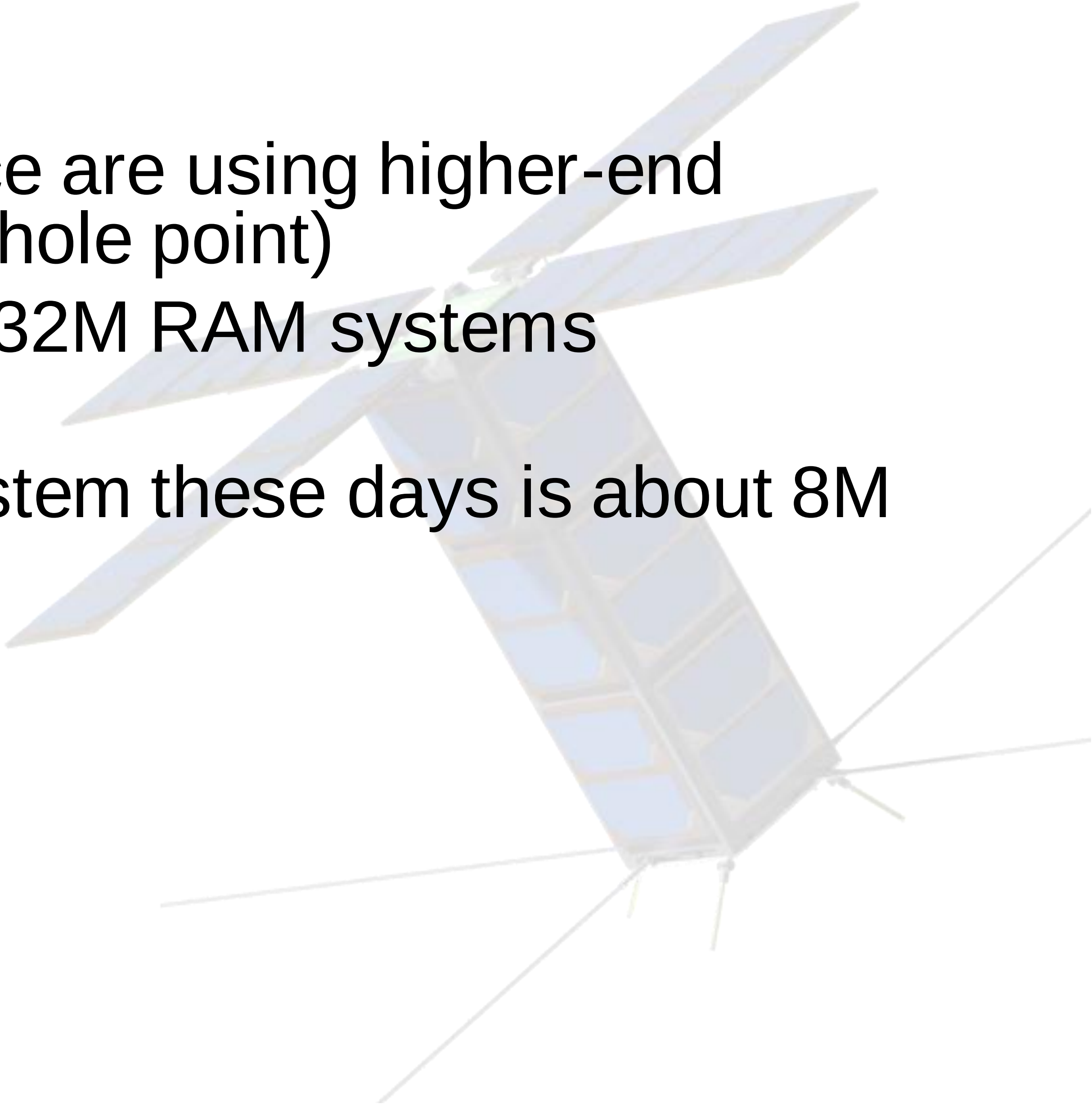
- System size
- Boot time
- Power management
- Realtime
- Security
- Audio drivers
- Video Drivers
- Flash filesystems (MTD)
- Support for processors, SOCs and boards
  - Arch support and drivers

Focus areas for space



- System size ?
- Boot time ?
- Power management
- Realtime
- Security?
- Support for processors, SOCs and boards
  - Arch support and drivers
- Drivers for space hardware

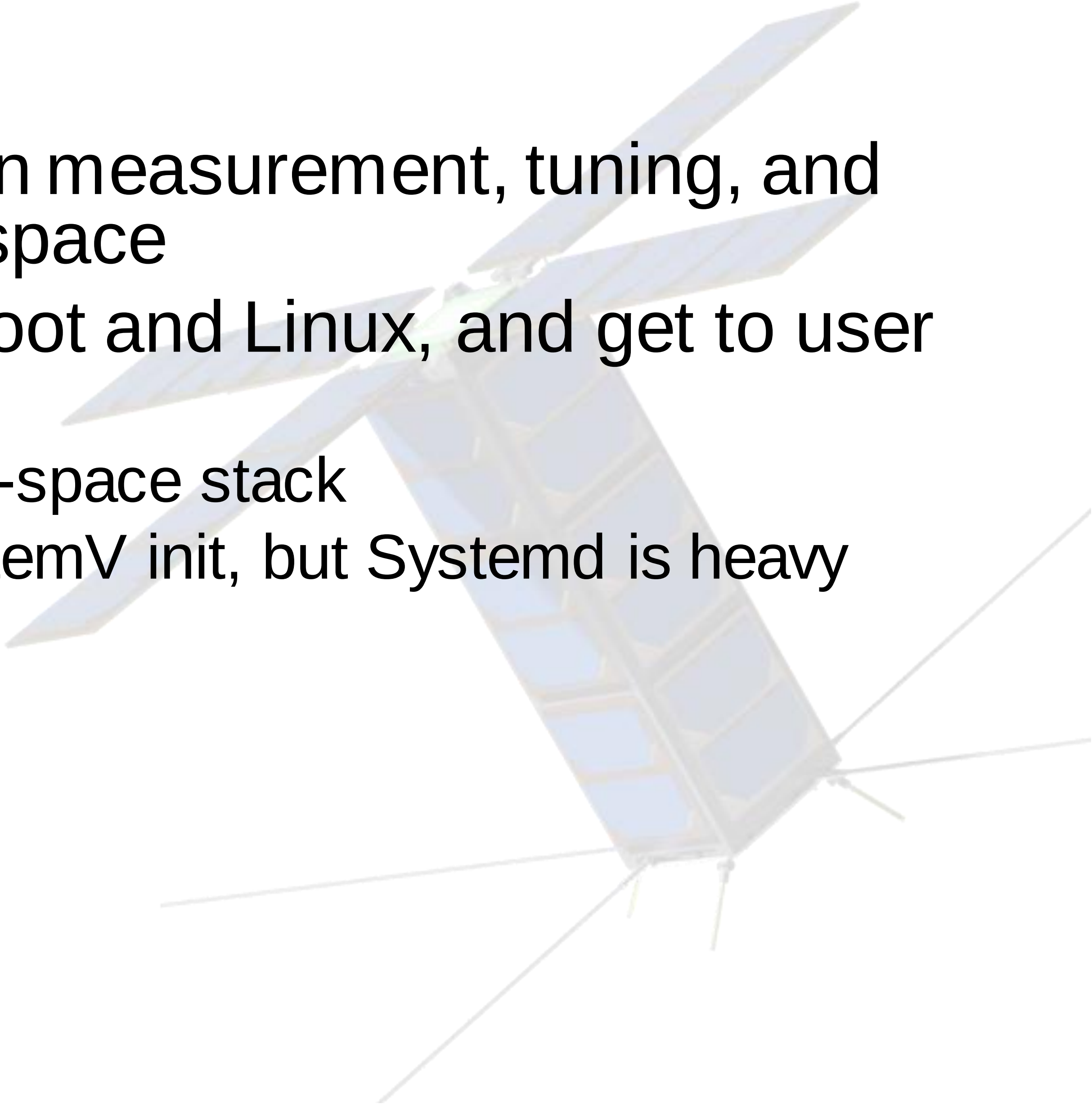
System Size



- Developers using Linux in space are using higher-end processors (that's kind of the whole point)
- No need to shrink Linux below 32M RAM systems
  - Is there a need?
- Realistic low-footprint Linux system these days is about 8M

Bootup time

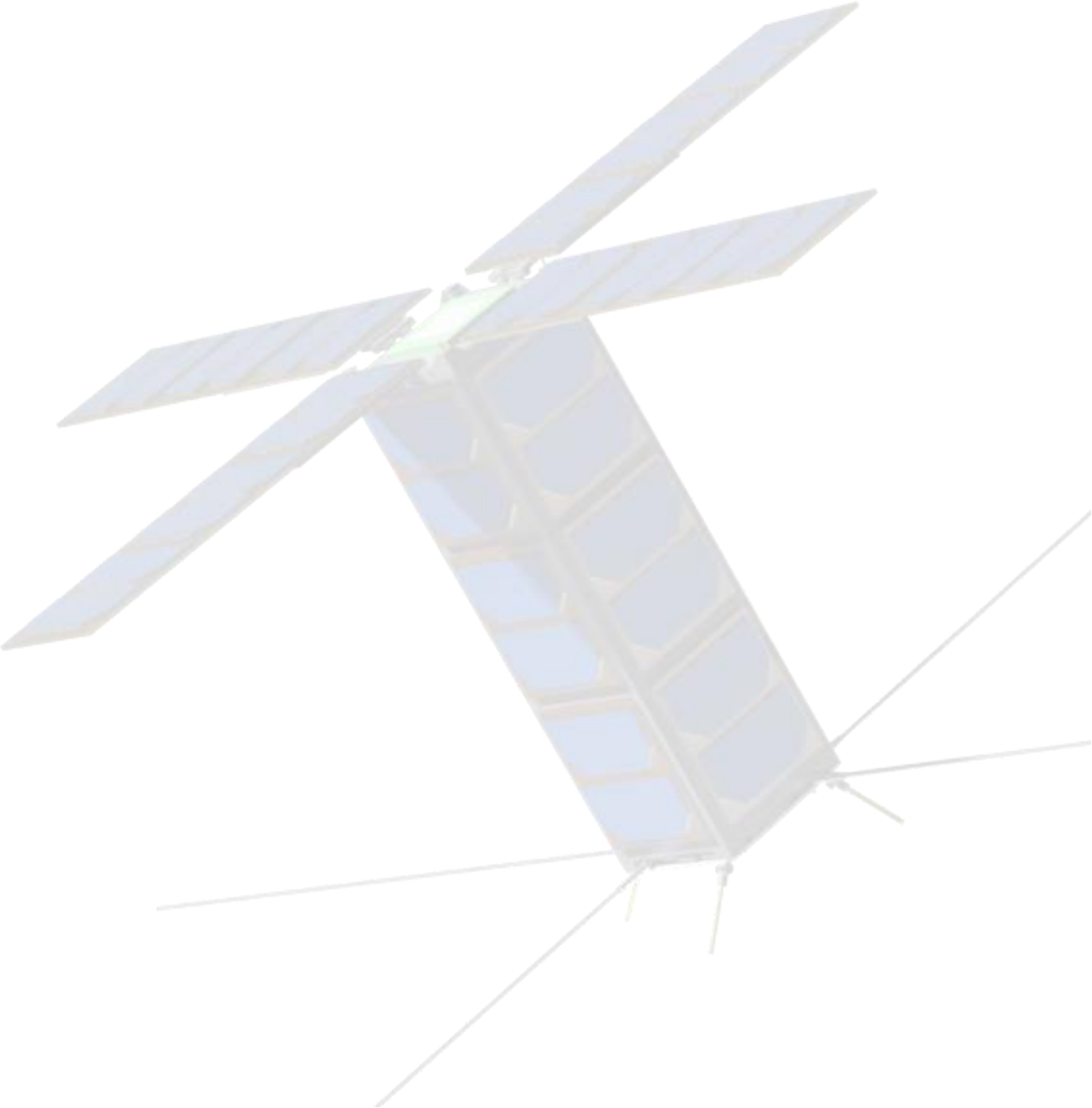
- Requires a LOT of configuration measurement, tuning, and sometimes refactoring of user space
- With a little work, can boot U-boot and Linux, and get to user space in about 2 seconds
    - What happens then is up to user-space stack
    - Systemd is better than older systemV init, but Systemd is heavy (about 2 M)



Bootloaders

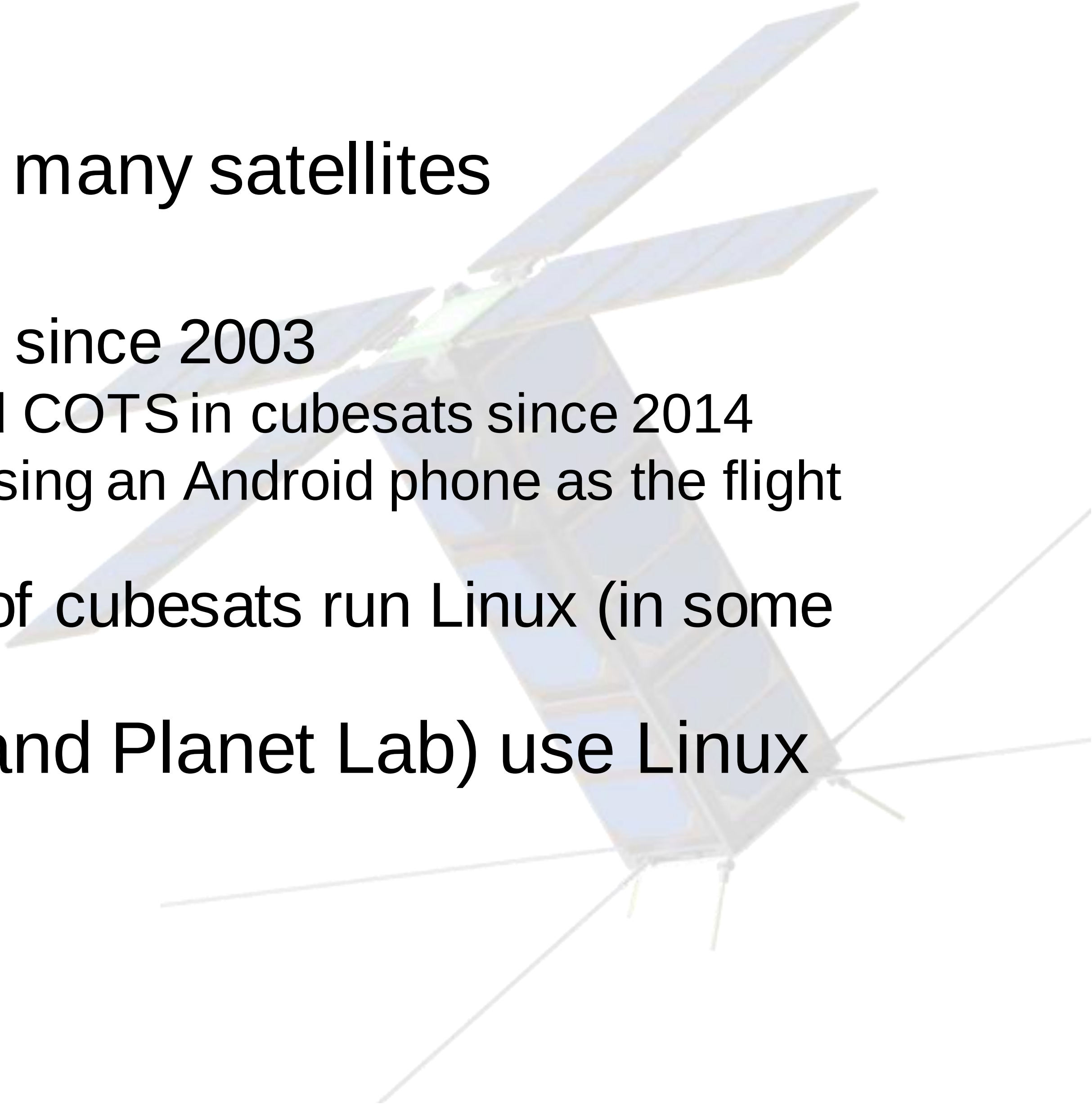
- U-Boot
  - Now supports loading images over HTTP
    - Previously only supported the UDP protocol
    - Could only use NFS or TFTP as servers
    - Now can download kernel and other images (dtb, initrd, etc.) from a web server
    - See https://www.linaro.org/blog/http-now-supported-in-u-boot/

Interesting embedded Linux uses



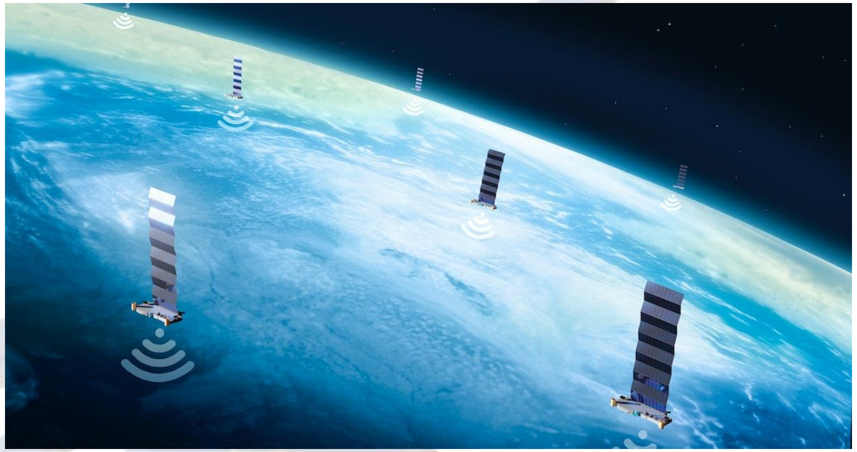
- Satellites
  - CubeSats
  - Starlink satellite constellation
- Mars Ingenuity helicopter

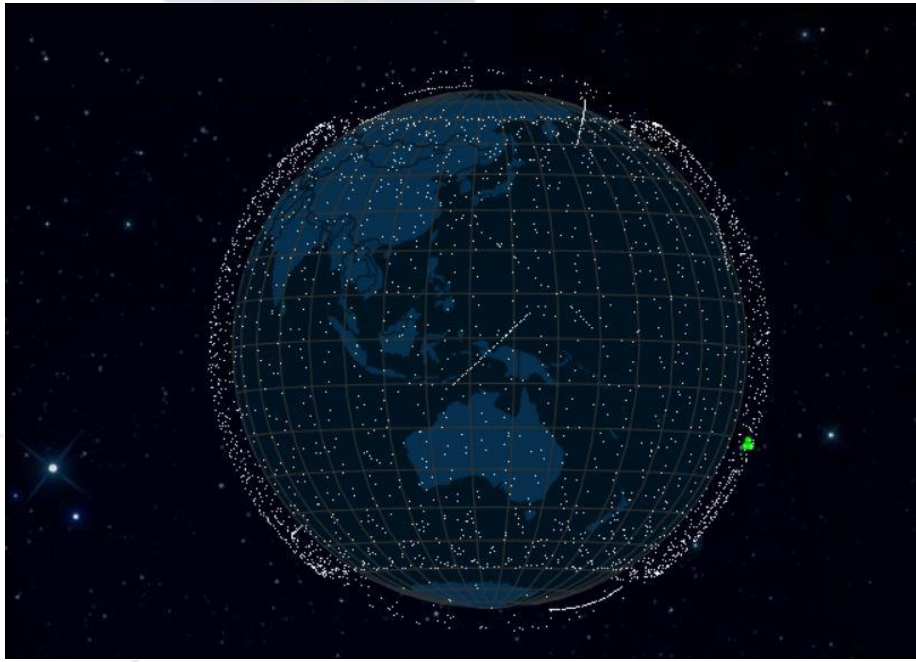
Linux in Satellites



- Linux is currently being used in many satellites
- CubeSats
  - Linux has been used in satellites since 2003
    - Lots of experiments with Linux and COTS in cubesats since 2014
    - One example: NASA PhoneSat (using an Android phone as the flight computer for a cubesat)
  - One estimate is that about 50% of cubesats run Linux (in some part of the flight stack)
- Major constellations (StarLink and Planet Lab) use Linux
  - (see next slide)

Starlink Satellite constellation





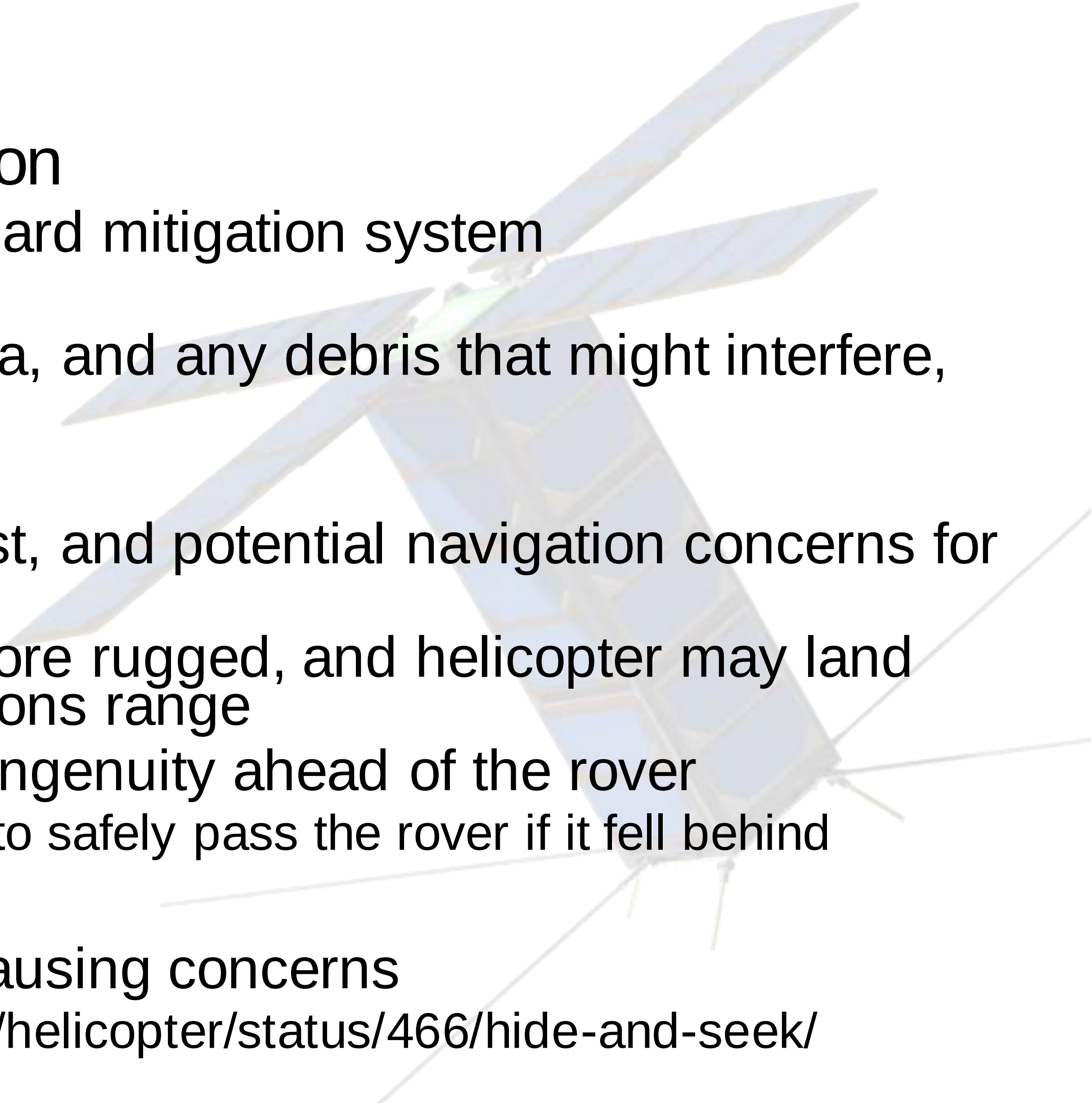
- SpaceX uses Linux in their rockets, space capsules and satellites
- Each Starlink satellite uses between 60 and 200 Linux processors:
  - Uses clusters for fault tolerance
    - Voting algorithms
    - Sub-component reboot capabilities
    - Redundant failover
- There are now (as of June 2023) over 4600 Starlinks satellites currently in orbit
  - https://satellitemap.space/ is quite interesting!!

Mars Helicopter - Ingenuity

Mars Helicopter

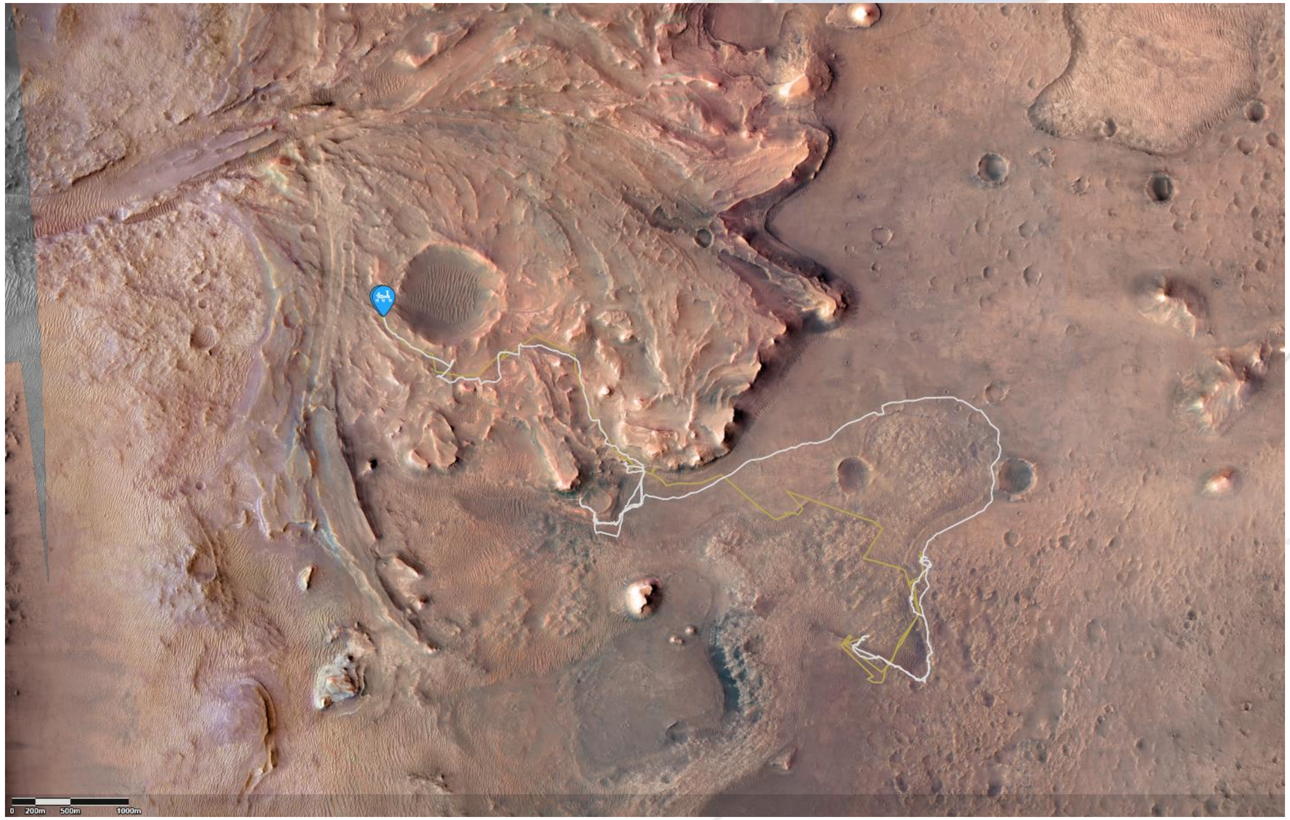
- Mars Ingenuity Helicopter landed in February, 2021 on Mars
- Performed tests and demonstrations in April & May (2021)
  - First 5 flights were part of “Technology Demonstration”
- After demo, NASA created a plan for continued flights
- Is still flying...
  - Has performed 51 flights so far
- Updates:
  - Autonomous landing site selection

Ingenuity Helicopter Update (June 2023)



- Autonomous Landing Site Selection
  - NASA uploaded a new Landing hazard mitigation system
    - First used in flight 39
  - Detects the slope of the landing area, and any debris that might interfere, and adjusts landing position
- Scouting for Perseverance
  - Provides pictures of areas of interest, and potential navigation concerns for the rover Team
  - Now off the crater floor, terrain is more rugged, and helicopter may land somewhere outside of communications range
  - Flights 41-46 consisted of keeping Ingenuity ahead of the rover
    - Canyon was too narrow for Ingenuity to safely pass the rover if it fell behind
- Playing hide and seek
  - Lost radio contact for a few days, causing concerns
    - See https://mars.nasa.gov/technology/helicopter/status/466/hide-and-seek/

Ingenuity flights on Mars



Sources for Mars helicopter

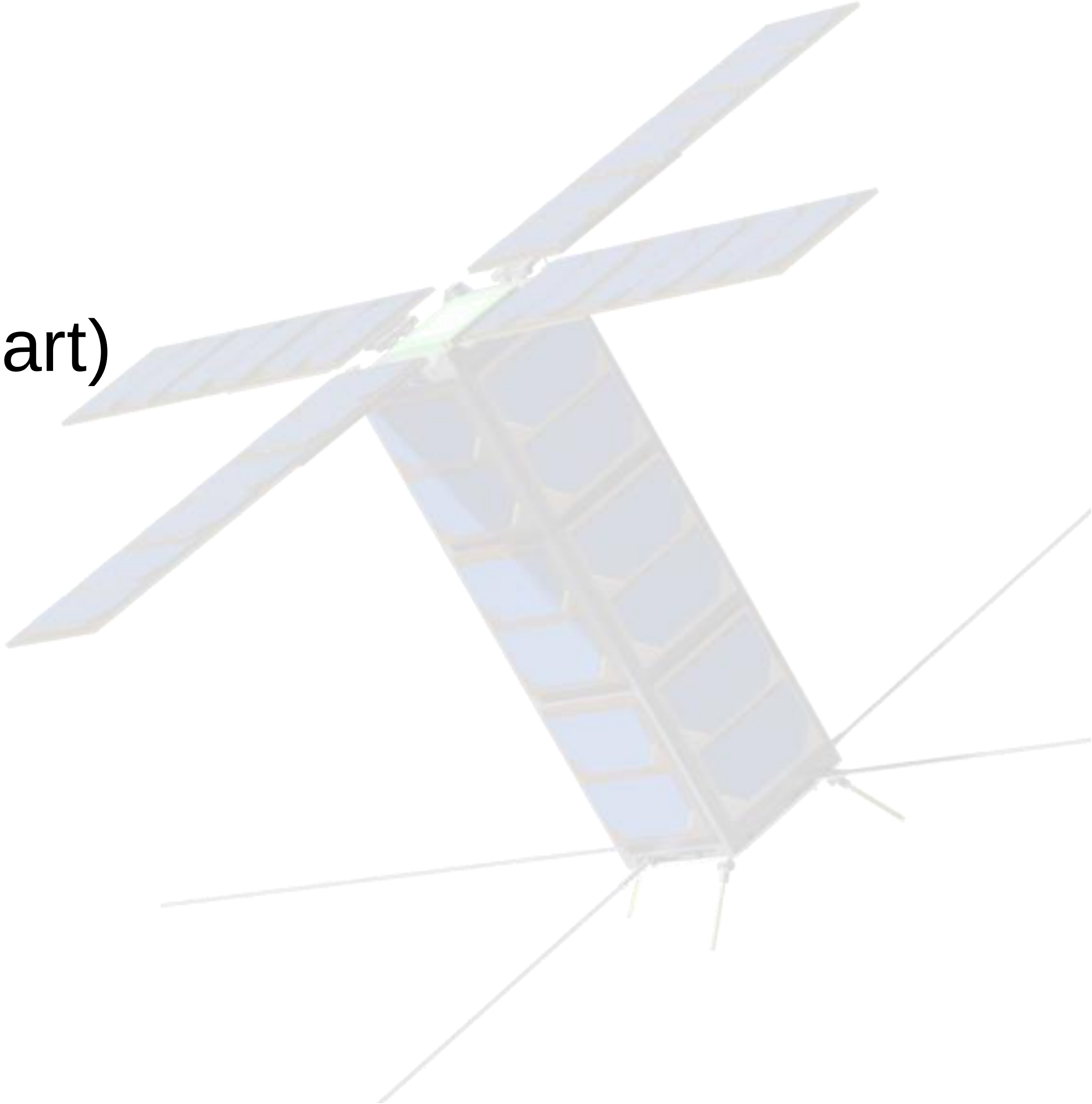
- Talk by Tim Canham at ELC 2021
  - Slides:https://elinux.org/images/5/5a/1._TIMOTHY_CANHAM.pdf
  - Video: https://youtu.be/0_GfMcBmbCg

- https://mars.nasa.gov/technology/helicopter/
- https://en.wikipedia.org/wiki/Ingenuity_(helicopter)
- https://thenewstack.io/how-the-first-helicopter-on-mars-uses-off-the-shelf-hardware-and-linux/
- https://www.pcmag.com/news/4-android-smartphones-with-as-much-power-as-nasas-mars-helicopter

Linux Foundation projects

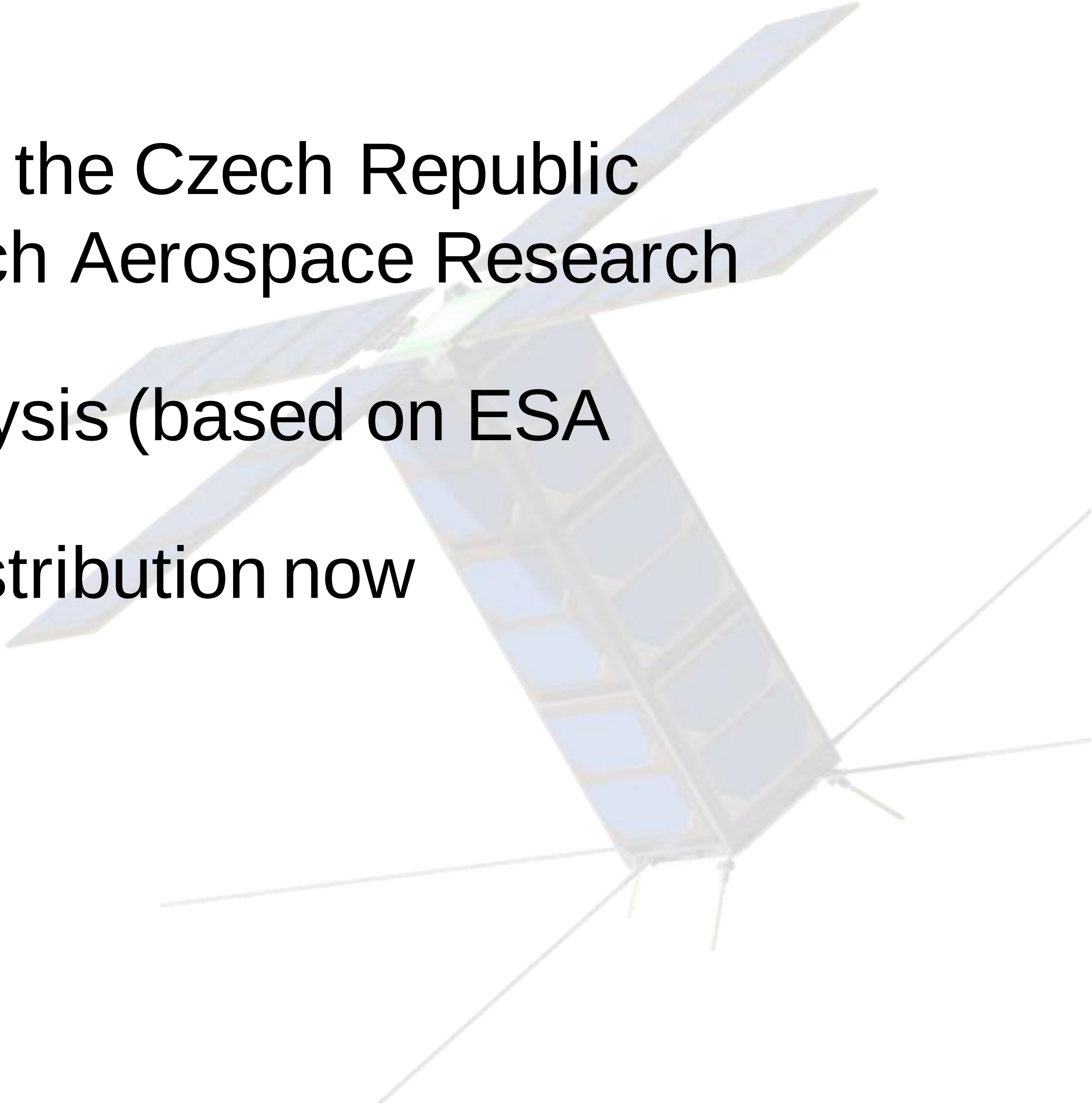
- Linux Foundation
  - Core Infrastructure Project (CIP) – handles support longevity
  - ELISA – handles issues with safety certification and standards
  - OpenChain – handles issues with supply chain
  - SPDX – Deals with licensing issues and SBOMs
  - Automotive Grade Linux (AGL) – handles automotive vertical
  - KernelCI – handles automated testing (for upstream)
  - Yocto Project – build system for embedded OSS (not just Linux)
  - DroneCode – handles drone vertical
  - Core Embedded Linux Project – is shutting down

Pre-build distributions of Linux for space



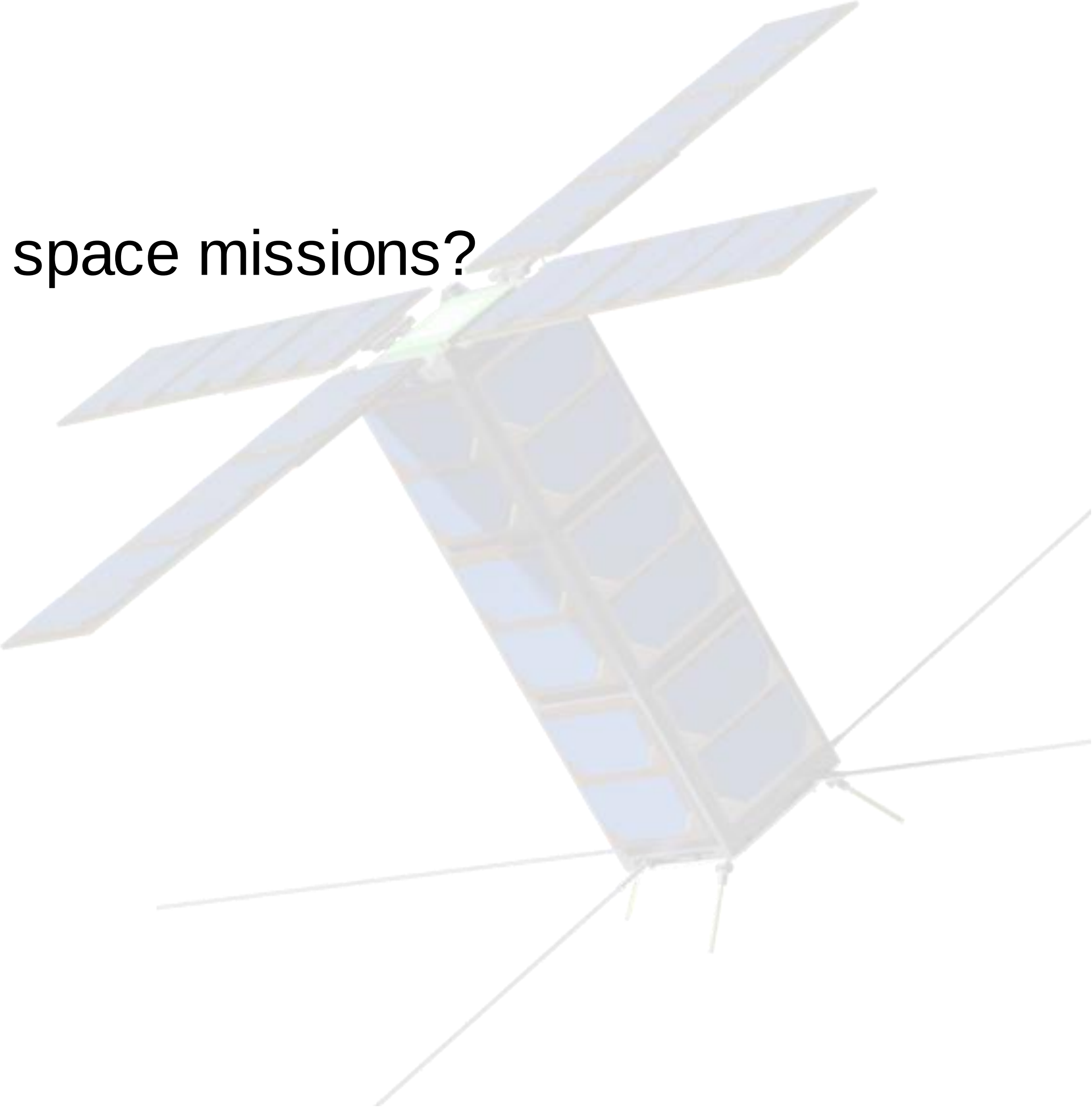
- FlightLinux – 1996 to 2002
- Kubos
- Linux4Space – 2022 (project start)

Linux4Space project



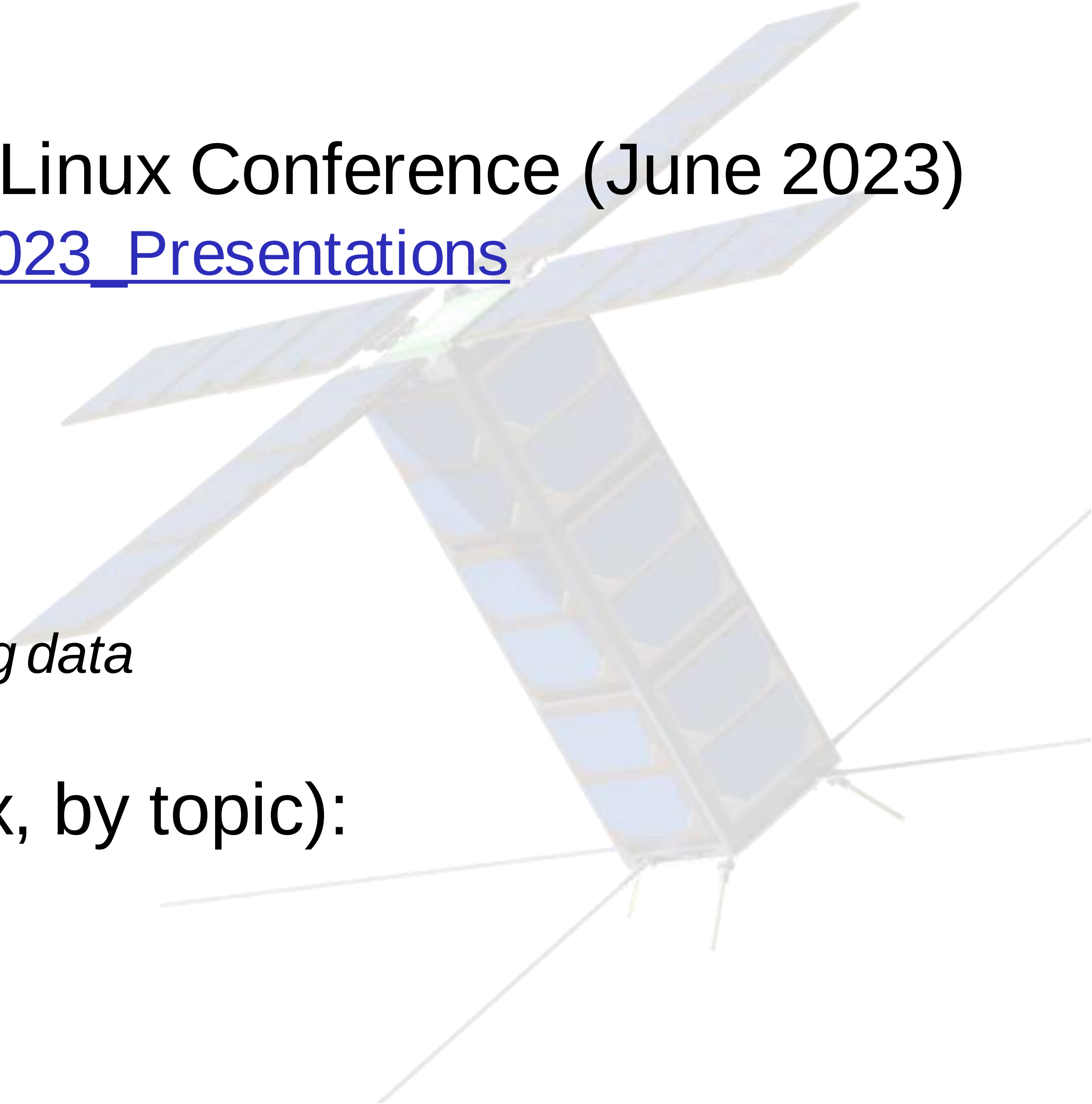
- Project by Liberec University in the Czech Republic
- In conjunction with VZLU (Czech Aerospace Research Center)
- Started with requirements analysis (based on ESA requirements)
- Are building a Yocto Project distribution now

What do *you* need in the Linux kernel?



- Discuss here
  - What barriers remain for Linux in space missions?

Embedded Linux Resources



- Presentations from Embedded Linux Conference (June 2023)
  - https://elinux.org/ELC_Europe_2023_Presentations
- eLinux wiki:
  - https://elinux.org/
- Linux in space wiki:
  - *Under Construction!*
    - *We're just getting started collecting data*
  - https://linux4space.org/
- Linux kernel news (kernel index, by topic):
  - https://lwn.net/Kernel/Index/

Thanks!