# Building robust embedded systems

Amit Kucheria

Pune Kernel Meetup

(07/01/2017)

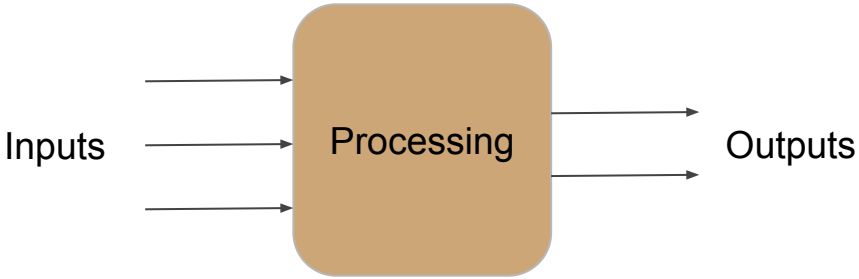
# Examples of embedded systems

- Vending machines
- Medical devices
- Elevators
- Printers
- Cameras
- Airbags
- Routers
- Autonomous Vehicles
- Robots
- Industrial automation
- Voting machines
- Amusement rides
- Railway signalling
- Mobile phones (?)
- TVs (?)
- Airplane Fly-by-wire (?)

# About

- Me
  - Linux user since 1995
  - Linux kernel developer since 2000
  - Worked for Nokia, Canonical (Ubuntu), Linaro
  - Worked on wireless routers, mobile phones, tablets, laptops
- This Talk
  - Touches upon several topics to pay attention to
  - Target is to raise the bar for MVP
  - Not about Linux in particular

# Computer systems



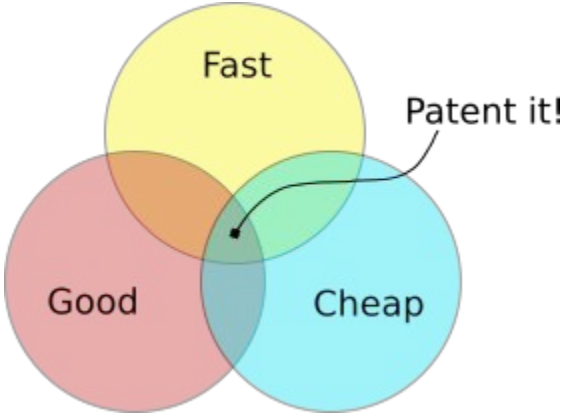
# Design Considerations (Process view)

- Reliability
- Safety
- Availability
- Maintainability
- Verification & Validation

# Design Considerations (Engg view)

- Responsiveness
- Safety-critical?
- Longevity
- Ease of servicing
- Software updates
- System Health Monitoring and Recovery
- Security
- Correctness
- Certification

# Pick two



# A detour

Real-time systems...

...are about speed

...are about predictability

...can be designed by playing with process priority

# What’s your score?

| | |
|---|---|
| ...are about speed | ✘ Speed is a byproduct and not the main goal |
| ...are about predictability | ✔ Bounding the worst case latency is an important goal |
| ...can be designed by playing with process priority | ✘ Study deadline scheduling |

# Responsiveness (capacity planning)

- Compute capacity needed
  - (Size of input * number of inputs per second * processing time per input) + spare capacity
- Worst-case acceptable latency (real-time)
- Graceful degradation?
  - Ratelimiting the number of clients
  - System partitioning so that critical system functionality is unaffected
  - In mission-critical systems, it points to a failure in requirements gathering
- Example of graceful degradation
  - Websites that support that latest W3C standards all the way down to text browsers
  - Apache graceful restart allows old connections to finish before restarting them

# Safety-critical?

- Misbehaviour can result in danger to human lives, damage to equipment or environmental harm
- Examples of bad software:
  - Toyota firmware causing unintended acceleration leading to death
  - Miscalculated Radiation Doses at the National Oncology Institute in Panama
- Extreme solutions for reliable designs (Failure is not an option)
  - Planes usually have 3 independent computer systems processing all data independently and voting. See Byzantine Fault Tolerance
- Resources to write safety-critical software

# Longevity

- Flash memory:
  - Long-term: Will it survive for 15-50 years?
    - *Flash wear-leveling*
    - *Minimise writes (1K-10K erases per block on consumer-grade flash)*
- Temperature
  - Long-term: Extreme temperatures reduces life of chips
  - Short-term: System reboots
- Sourcing of parts
  - Long-term: Can you replace a unit with an identical one in 5 years?

# Ease of servicing

- Remote monitoring
- Remote debugging
- Ability to reproduce problem locally?
- Cost of sending an engineer to a remote site
- Redundancy in HW

# Software updates

- Ability to push software updates safely
  - Regular security fixes
  - Periodic bug fixes
  - Optional feature enhancements
- If you're not planning for this, your product *will* get ignored
- Don't create backdoors for updates, use standard mechanisms
- Good examples:
  - Tesla changes ground clearance via a software update
- Examples of terrible software:
  - TP Link repeaters uses ~700Mb/month to test connectivity

# System Health monitoring and Recovery

- Out of file descriptors
  - Rlimits (http://connect.linaro.org/resource/sfo17/sfo17-114/)
- Out of memory
- Out of flash space
- Network outage (tricky one!)
- Incorrect input
- Incorrect output
- Is a reboot your only recovery plan?

# Security

- Secure boot
  - Only software signed with known keys is allowed to be run
- Network security
  - Unencrypted data sent over network
  - Open ports (webservers)
  - Cleverly crafted packets (ping of death)
- Physical security
  - Most bets are off if someone has physical access to the HW
  - ...but you can make it a bit tedious for the casual hacker
  - Trivia: Hacking deep sea cables

# Correctness

- Correct output in adversity
  - Environmental
    - *E.g. Adjust radio settings (baudrate, channel) in case of jamming*
  - Network outage
    - *E.g. Ad-hoc routing protocols*
  - HW failure
    - *E.g. EDAC RAM, Hard disk S.M.A.R.T.*
- Data conversion
  - Ariane 5 Flight 501 failure due to data conversion error in 64-bit floating point to 16-bit signed integer value
  - Failure of Mars Climate Orbiter due to two different system of units used
- Formal methods to generate proofs

# Certification

- Medical devices
- Hostile environments
  - Temperature, Pressure, Vibration
  - Water and Dust (IP-rating)
- Grades
  - Milspec
  - Commercial-grade
  - Consumer-grade

# Why Linux?

- Architecture support (more than any other OS)
- Royalty-free
- Tons of device drivers
- Source code makes it easy to modify
- Allows graceful degradation by dropping features
- Runs on everything from micro-controllers to mainframes

# Size

- Linux too big for micro-controllers
- Memory footprint < 1Mb
- Flash footprint < 1Mb
- Minimising language runtimes
- uClibc, newlibc, dietlibc
- Busybox
- Openembedded (OE) custom distribution
- cramfs/squashfs: read-only, compressed
- ubifs/jffs2: compressed filesystems for flash, wear-leveling

# Miscellaneous Tips

- Watchdog (SW / HW)
- `$ man setrlimit`
- Stress-ng to probe for weaknesses
- PoE
- Defensive programming
  - Using a subset of the programming language (e.g. avoid dynamic allocation and pointers)
  - Become aware of pitfalls of using gets(), strcpy()

# uControllers

- Zephyr
- FreeRTOS
- Apache Mynewt
- Pick one, doesn’t matter at this point, IMO

# La Fin