Autonomous systems: surviving the nanosecond blue screen

Eric Verhulst

Altreonic NV – kurt.mobi

Smart Systems Industry Summit 17 Oct 2017

Use-case: KURT e-vehicle

- A steer and drive-by-wire modular and scalable e-vehicle for urban environments
- Roadmap towards autonomous driving
  - Trustworthy and safe over whole life
  - Requires to know reliability of components
- City-KURT:
  - System concept, not just vehicles

Autonomous systems: questions

- Is an AGV autonomous?
- Does autonomous behavior require intelligence?
- Does autonomous driving require certification?
- If not, why do airplanes require it?
- What are Level 3, 4 and 5 of autonomous driving?
- How fast does a failing system need to recover?
- Is a Tesla-S with autopilot certifiable?

Altreonic's aim: Trustworthiness

- Trust is a user's perspective!
- Multiple domains:
  - Safety & Security
  - Ergonomics & Privacy
- How to achieve these goals?
  - System reliability depends on reliability of components, system architecture and engineering process followed
  - Managed engineering process
  - Adequate design/architecture
  - Hazards and Faults are part of the Specifications!

ARRL criterion & Autonomous systems

- Assured Reliability and Resilience Level (0 – 7)
- Normative criterion linking faults with architecture and supporting processes
- Reliability = mainly quality assurance issue
  - Component / material selection dominates (graceful degradation)
- Resilience = mainly dynamic
  - Adaptive to survive
- Software reliability:
  - Logical errors, reflect reliability of software process

Embedded systems have hierarchy (1)

- Software has no bugs
  - State space can be very large
  - Bugs are really errors
  - What about incomplete/erroneous Specifications?
- What about AI software?
  - Data can be fuzzy and noisy
  - Self learning
  - Can it be certified?

Embedded systems have hierarchy (2)

- Software runs on digital hardware:
- Digital hardware is never perfect:
  - Material & Manufacturing deficiencies
  - Aging reduces reliability => latent deficiencies
  - Logical design errors (but rare)
  - Externally induced faults:
    - Bitflips, SEUs, alpha particles, power supply instabilities, …
    - Mechanical: bad contacts, vibration, temperature, humidity, ...
    - 1000's of electrically / mechanically connected parts
  - Digital systems can fail in nanoseconds:
    - blue screen of death!

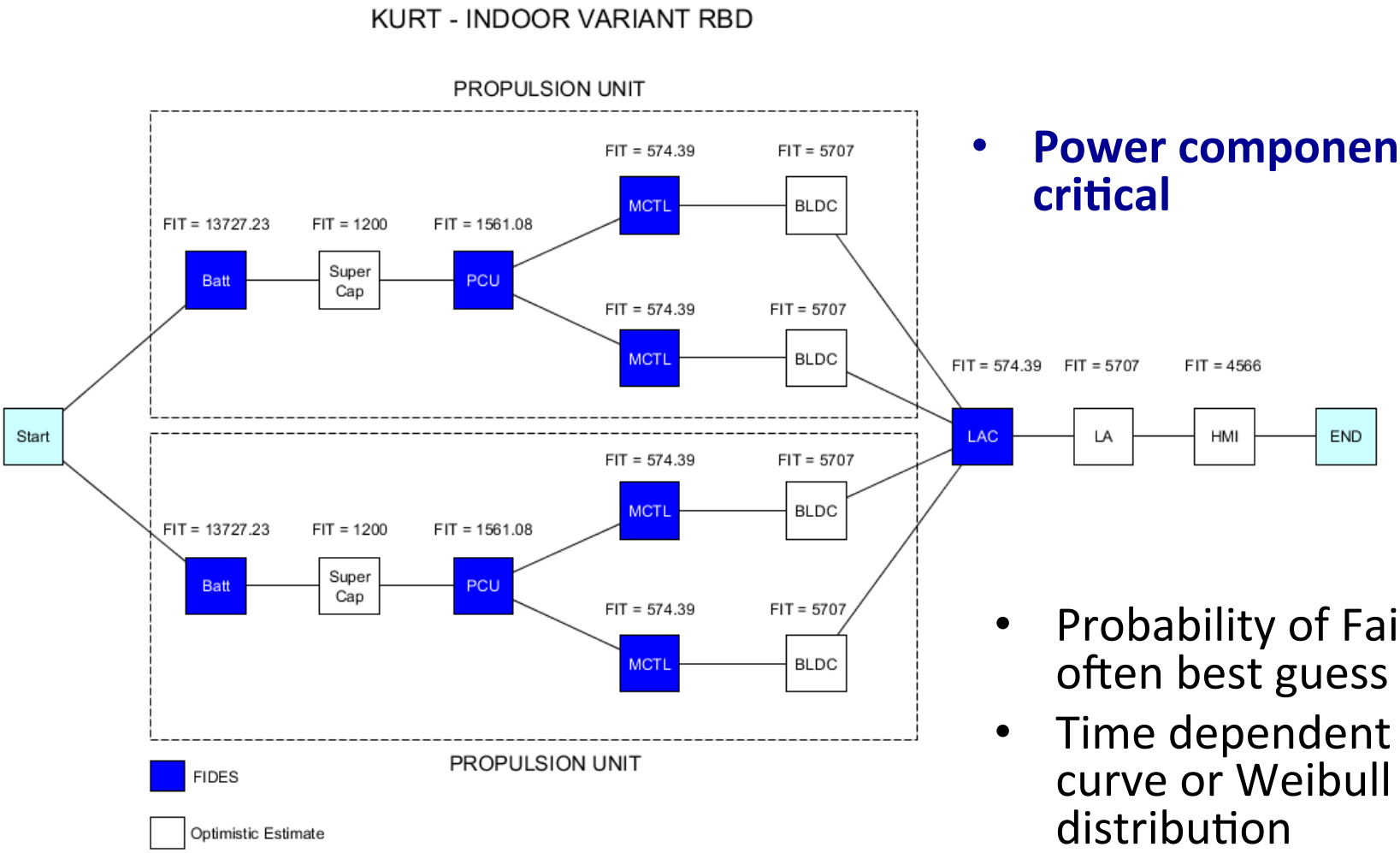
Component selection

- Specifications derived from mission profile:
  - Temperature: - 40°C to + 70 °C
  - Humidity
  - Current, Voltage, static discharge
  - Vibrations
  - Packaging & PCB
  - Connectors!
  - Expected lifetime
- Mostly a Quality Control issue
- Electronics are generally very reliable

Hardware failures over time

- What affects reliability?
  - Small feature sizes: atoms move!
  - High current & temperature: heat kills!
  - Chemical attack
- Critical components require FMEA
  - Connectors
  - Capacitors, especially electrolytes
  - Super capacitors
  - Batteries
  - Maintenance replacement to be designed in!

FIT calculations for KURT

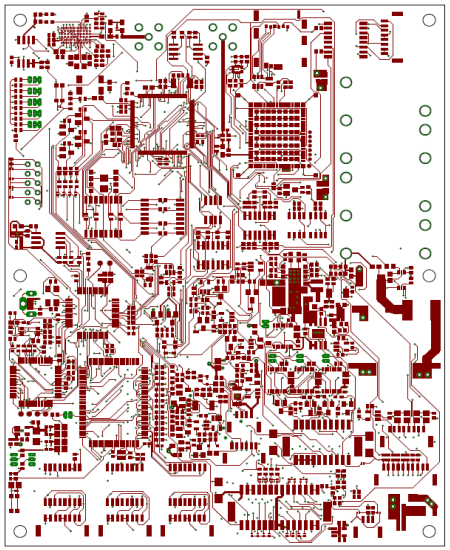


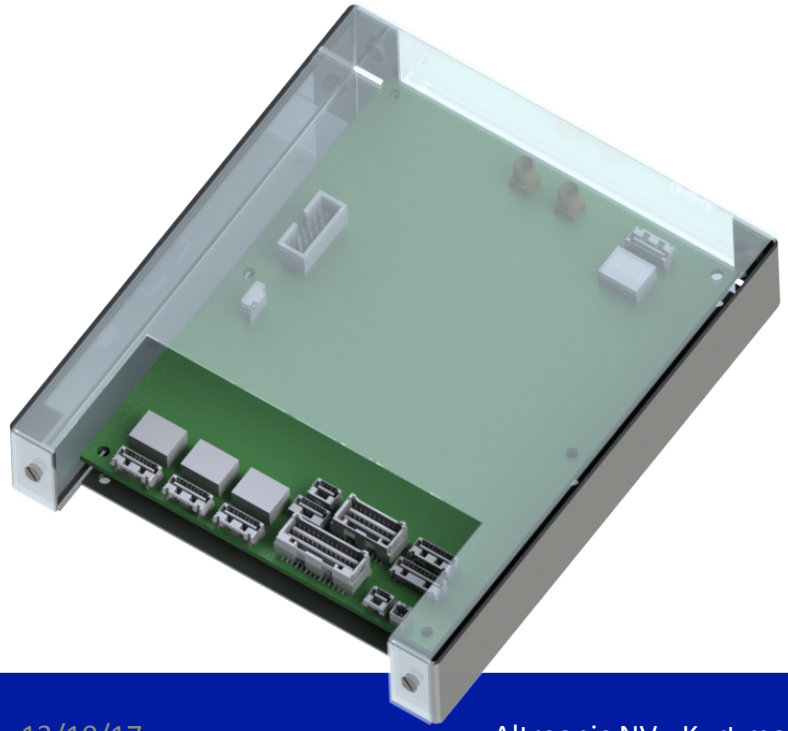
- **Power components most critical**

- Probability of Failure: often best guess
- Time dependent (bathtub curve or Weibull distribution

KURT vehicle controller

- Board has some 800 components
- Many just to keep key components within operating range and filter out disturbances





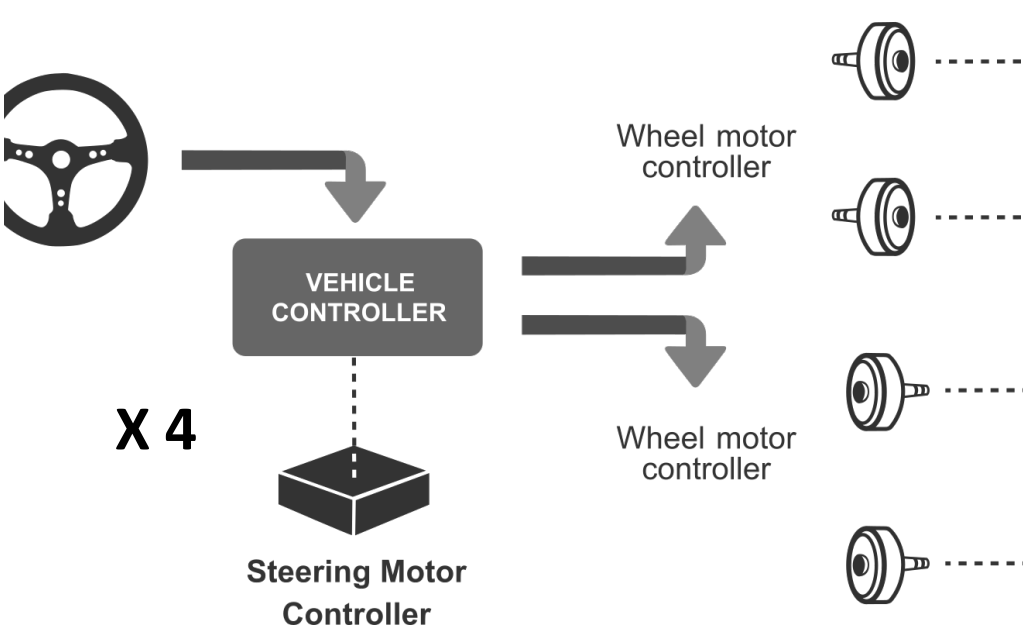
Is this enough?

- PoF and Reliability estimates are starting points
- Law of Murphy applies
- Residual and latent faults can still result in catastrophic failures any time
- Hence: to be tackled at higher level.

Fault tolerance for autonomy

- Challenge:
  - Fault occurs while at maximum speed
    - 200 km/hr = 55 m/s
  - Maximum recovery time: 100 milliseconds
  - Fault can happen in 1 nanosecond (@ 1 GHz)
    - Can we detect it?
  - No time for a reboot!
- Vehicle is a component in traffic system
- Vehicle needs to be real-time fault tolerant

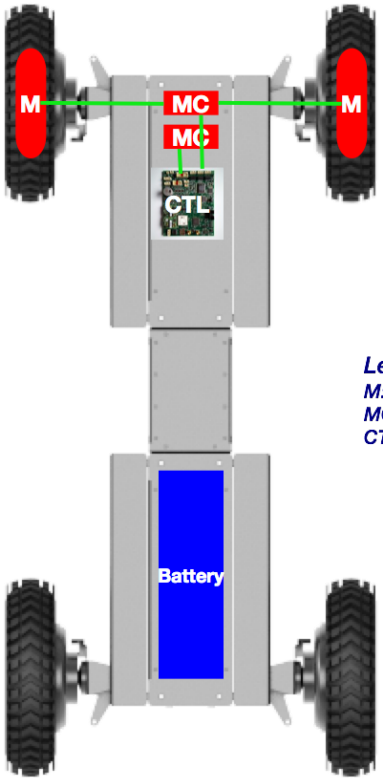
Distributed Software Architecture



- Requires redundancy in hardware
- Avoid Common Mode Failures

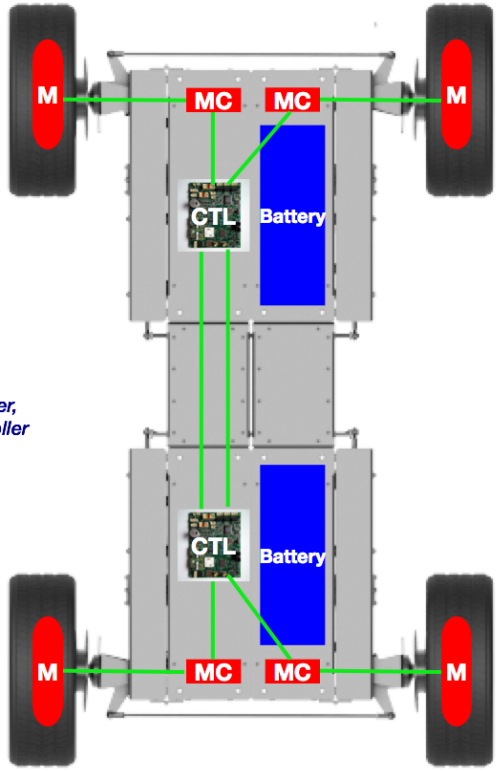
Architectural redundancy

- Graceful degradation for safety

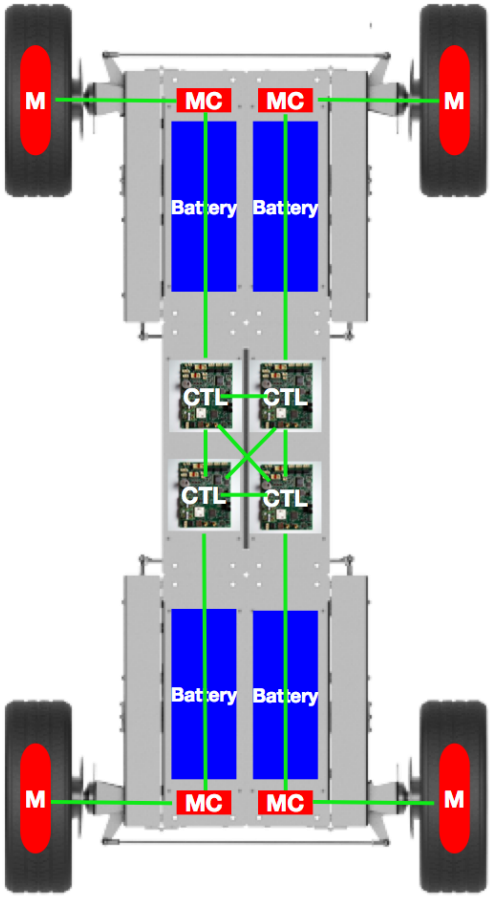


Simple configuration:
Two motors, one controller, one battery

Legend:
M: motor
MC: motor controller,
CTL: Vehicle controller



Dual redundancy configuration:
Four motors, two controllers, two batteries



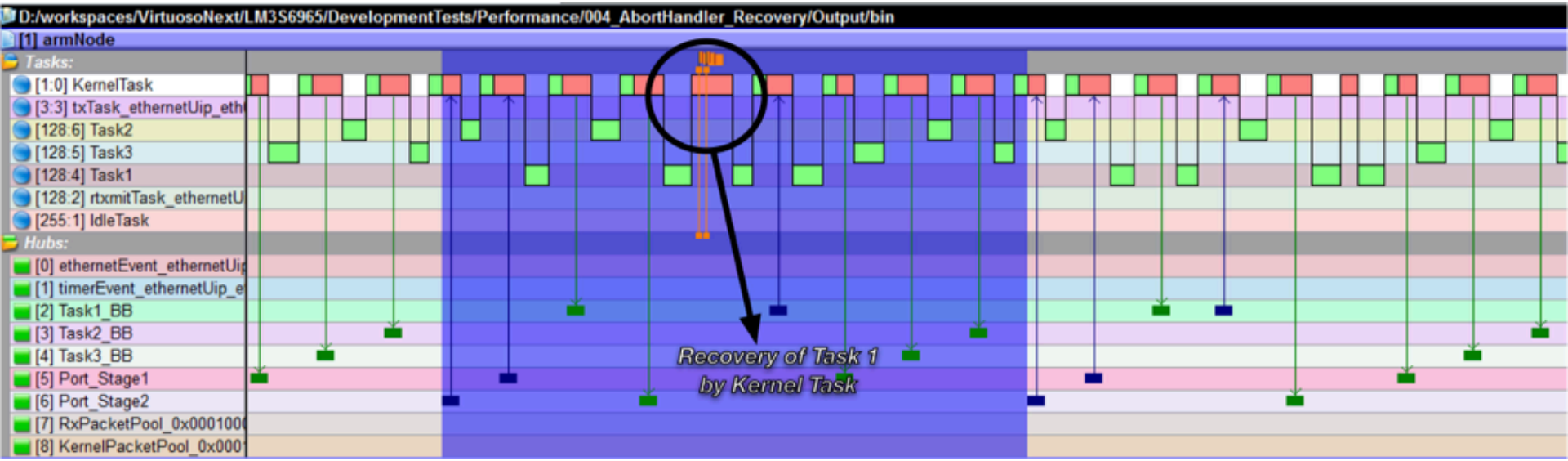
Quadruple redundancy configuration:
Four motors, four controllers, four batteries

Altreonic's VirtuosoNext RTOS: Local fault recovery

- Formally developed, 5th generation
- Distributed ("MP", "SMP") by design
  - Packet switching
- Fine grain space and time partitioning
  - Task level
  - Code and data
- Fine grain fault recovery (microseconds)
  - Recovers from processor exceptions
  - Equivalent to HW redundancy (ASIL-C/D)
  - No full reboot needed
  - Reduces cost of system fault tolerance

Fine grain fault recovery

- With restore of state before: 44 microsecs at 120 MHz
- Reduces need for hardware redundancy: hardware and data is virtually duplicated



Conclusions

- Systems engineering is a multi-domain activity
- State space is extremely large
- Design for Trustworthiness = preventive
  - Procurement: select adequate reliability
  - Architecture:
    - Separation of concerns
    - Respect hierarchy
    - Modular = scalable = redundancy (diversity of needed)
  - Confirmation & Review
    - Certification
    - ARRL-7: the service is the system, not the vehicles