

18-642: Single Points of Failure

11/1/2018

These tutorials are a simplified introduction, and are not sufficient on their own to achieve system safety. You are responsible for the safety of your system.



“Put all your eggs in one basket, and then watch that basket!”

– Mark Twain

Carnegie
Mellon
University



Edge
Case
Research

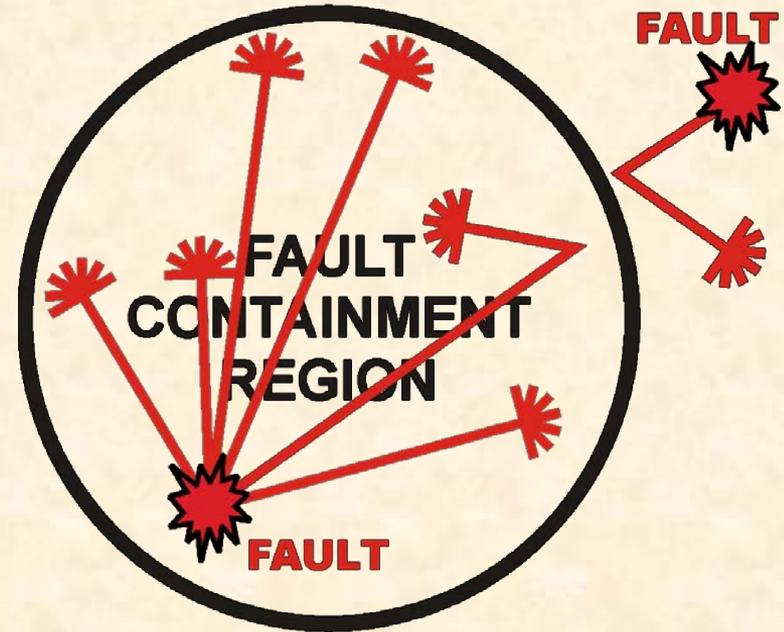
Avoid Single Points of Failure

■ Anti-Patterns for Critical Software:

- Hardware single points of failure
- Correlated, accumulated multi-point failures
- Making assumptions about failures
- Non-diverse, low-SIL software

■ Fault Containment Region (FCR)

- Faults from outside FCR are kept out
 - Faults inside FCR are kept in
- But, within FCR a single fault has arbitrarily bad effects
 - It's like a shotgun blast inside the FCR
 - Applies to both SW faults and HW faults (e.g., single event upsets)



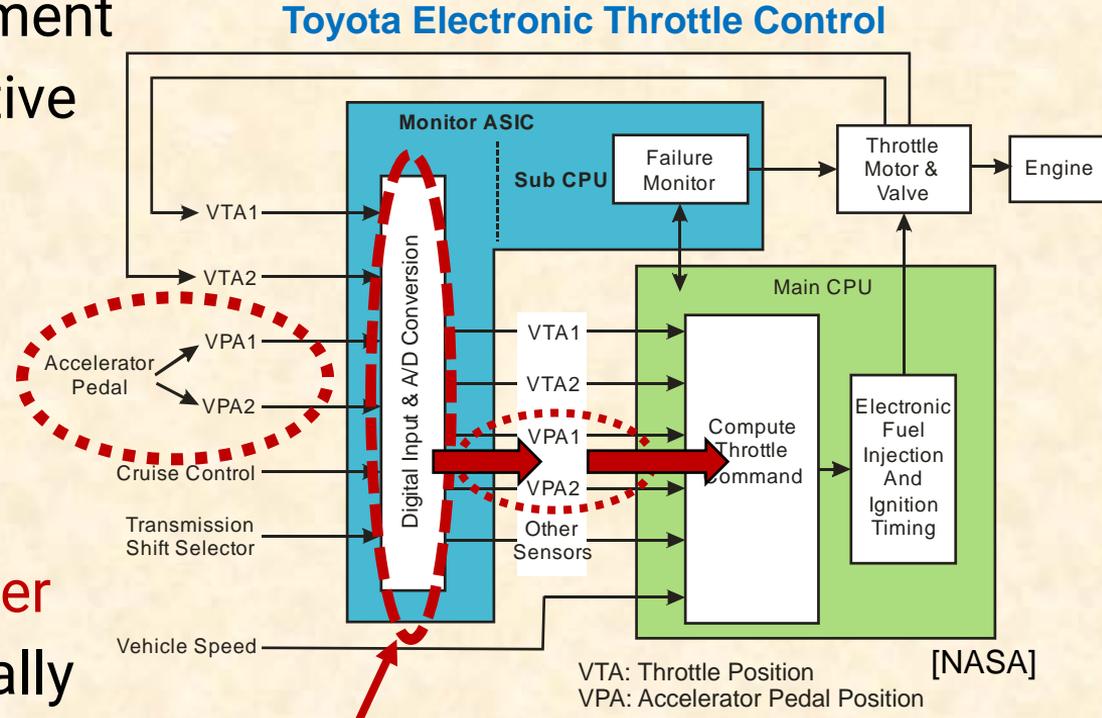
Toyota Unintended Acceleration (UA)

■ Perhaps 89 deaths, hundreds of serious injury lawsuits

- \$1.6B class action settlement
- Jury found system defective
 - Toyota “acted in reckless disregard”
- Many of issues were SW, but also a HW problem:

■ Two accelerator inputs

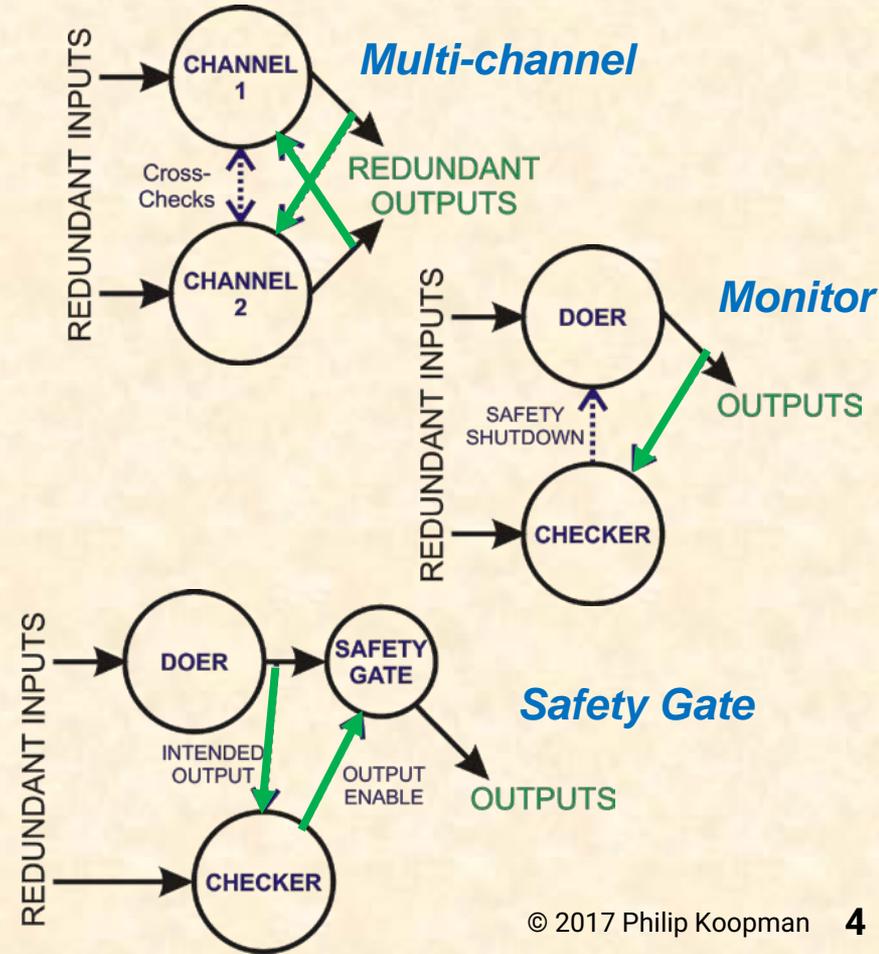
- But – **shared A/D converter**
- Could result in electronically “stuck” accelerator pedal



Single Point of Failure

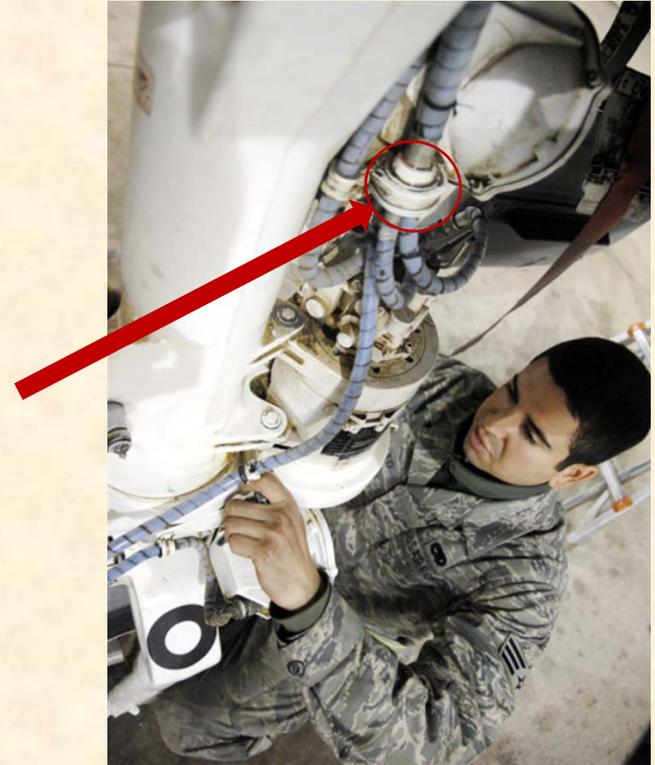
Eliminating Single Points of Failure

- **Multiple FCRs required for life-critical and highly mission-critical systems**
 - This isolates faults in redundant components – no single point of failure
 - Avoid an Achilles' Heel in your system
 - All software on CPU can be a “single point”
- **Multi-channel (e.g., 2 of 2)**
 - Compare identical component outputs
- **Doer/Checker (monitor/actuator pair)**
 - “Checker” makes sure “Doer” is safe
- **Safety gate**
 - Only permits safe outputs to issue



■ Correlated faults if multiple FCRs are likely to fail together

- Common design faults (including software)
- Common manufacturing faults
- Shared infrastructure (e.g., power, clock)
- Physical coupling
 - Shared wiring harness, connectors
 - Shared location (e.g., hot spot)



USAF: <https://goo.gl/df5pdg>

■ Accumulated faults

- Fault not detected
- Fault not repaired before next mission

■ Safety is improved by using multiple FCRs

- Hardware redundancy / HW isolation
 - Typically each FCR should be an independent chip
- Software must be practically “perfect”
- Common patterns: multi-channel, checker, safety gate



■ Pitfalls are numerous and sometimes subtle

- Two copies of same SW fail the same way
- Ensure multi-channel doesn't fail as “always trust one channel”
- Ensure the checker doesn't fail as “always checks OK”
- Look for hidden correlation (HW design defects, shared libraries, shared requirement defects, physical connection, shared clock, shared power, ...)