ECE 471 – Embedded Systems
Lecture 19

Vince Weaver

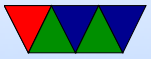
http://www.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

12 November 2013

Announcements

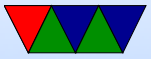
- Project was released. Please take a look.
- Try to let me know project ideas/groups by Friday if at all possible.



Places for More Info

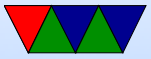
- Embedded projects: `http://hackaday.com`
  They had a recent series on CAN-bus

- Computer Risks and Security Issues: The RISKS digest from comp.risks
  `http://www.risks.org`



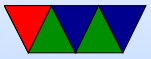
Software Bugs

- Not all bugs are security issues
- Coding bugs can have disastrous effects



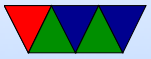
Automotive

- Until recently no standard
- Bugs, Toyota firmware
- `http://www.edn.com/design/automotive/4423428/2/Toyota-s-killer-firmware--Bad-design-and-its-`

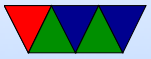


Airplanes

- DO-178B / DO-178C
- Software Considerations in Airborne Systems and Equipment Certification
  - Catastrophic: fatalities, loss of plane
  - Hazardous: negative safety, serious/fatal injuries
  - Major: reduce safety, inconvenience or minor injuries
  - Minor: slightly reduce safety, mild inconvenience
  - No Effect: no safety or workload impact

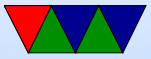


- AA Flight 965. Autopilot to waypoint R. Re-entered it, two starting with R, so it helpfully picked one with highest frequency, did a semi-circle turn to east right into a mountain.

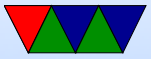


Military

- Patriot missile – clock drift slightly, but when on for hundreds of hours enough to affect missile tracking
- Yorktown smart ship – 1997 – Running Windows NT. Someone entered 0 in a field, divide by 0 error, crashed the ship. Database crash, crashed propulsion system. Rumors that it needed to be towed in, but no, only down for 2.75 hours.
- F-22s computers crashed when crossing 180 degrees



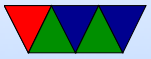
longitude? Lost navigation and communication, had to follow tankers back to Hawaii.



Spacecraft

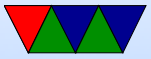
- Mariner 1 (1962) – rocket off course due to mis-transcribed specification into FORTRAN, missing overbar

- Apollo 11 (1969) – landing on moon. Processor normally loaded with 85% load. DELTAH program run which take 10%. But buggy radar device was stealing 13% even though in standby mode. Mini real-time OS with priority killed low-priority tasks so things still worked.

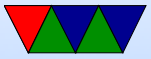


- Ariane 5 Flight 501 (1996) – famous. $370 million. Old code from Ariane 4. Could not trigger on Ariane 4. Horizontal acceleration crashed primary and secondary computers, sending debug messages that the autopilot read as velocity data. Conversion from 64-bit float to 16-bit signed int overflow. Should have had check on it that vertical acceleration, but did not. Not properly simulated in advance. Written in ADA

- NASA Mars Polar Lander (1999) – mistook turbulence vibrations for landing and shut off engine 40m above surface



- NASA Mars Climate Orbiter – ground software using lbf (pound/foot) units, craft expecting Newtons
- NASA Mars Spirit rover (2004) – temporarily disabled due to too many files on flash drive. Constantly rebooting. Radio could understand some commands directly, could reboot with flash disabled. Fixed when deleted some unneeded files. Eventually reformat.
- ESA CryoSat-1 (2005) – lost due to missing shutdown command

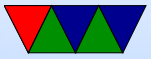


Medical

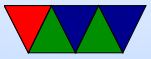
- IEC 62304

- Therac-25 radiation treatment machine. 6 accidents, patients given 100x dose. High power beam activated w/o spreader too. Older machines had hardware interlock, this one in software. Race condition. If 8-bit counter overflow just as entering manual over-ride, it would happen.

  – Software not independently reviewed

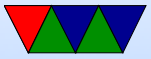


- Something wrong: Printed “MALFUNCTION” and error number 1 to 64 which was not documented in manual. Press P to clear.
- Operators not believe complaints from patients.
- To trigger, had to press X (mistake), up (to correct), E (to set proper) then ”Enter” all within 8 seconds.
- This missed during testing as it took a while for operators to get used to using machines enough to type that fast.
- Used increment rather than move to set flag.
- Written in Assembly Language



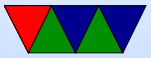
Financial

- Knight Capital. Upgrade 7 of 8 machines, missed last. Re-used a flag definition with new software. Caused massive selloff, $440 million

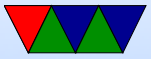


Power

- 2003 Blackout. Race condition, no alarms notify as wires fail, backup of alarms crash server.

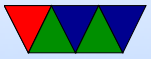


Good Design Practices

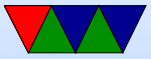


Space Shuttle Design

- HAL/S high-order assembly language (high-level language similar to PL/I)
- PASS software – runs tasks. Too big to fit in memory at once
- BFS – backup flight software. Bare minimum to takeoff, stay in orbit, safely land, fits in memory, monitors pASS during takeoff/landing Written by completely different team.

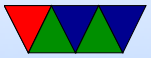


- 28 months to develop new version
- IBM
- originally 424k of core each
- Extensive verification. One internal pass, one external
- 4 computers running PASS, one running BFS
- Single failure mission can continue; still land with two failures
- 4 computers in lock-step, vote, defective one kicked out



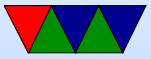
Code Safety Standards

- Avionics: DO-178C (1992 for B)
- Industrial: IEC 61508 (1998)
- Railway: CENELEC EN 50128 (2001)
- Nuclear: IEC 61513 (2001)
- Medical: IEC 62304 (2006)
- Automotive: ISO 26262 (2011)



Writing Good (Embedded) C Code

- Various books. Common one: MISRA: Guidelines for the Use of the C Language in Critical Systems
- Comment your code!
- Strict, common code formatting (indentation)
- More exact variable types (int32_t not int) Size can vary on machine, and on operating system
- Subset to avoid undefined behavior



- Tool that enforces the coding standards
- Good to write safe code even if it isn't meant for a safe application. Why? Good practice. Also who knows who or when your code might be copied into another project.