Program-level Soft Error Derating in a Brake-by-Wire System

Daniel Skarin and Johan Karlsson

Department of Computer Science and Engineering
Chalmers University of Technology
Göteborg, Sweden

Martin Sanfridson

Volvo Technology
Göteborg, Sweden

Motivation

Current automotive electronic systems are used to assist the driver

- Anti-lock braking system (ABS)
- Electronic stability program (ESP)
- Adaptive cruise controller (ACC)

Safe shutdown is a viable approach to handling failures in these systems

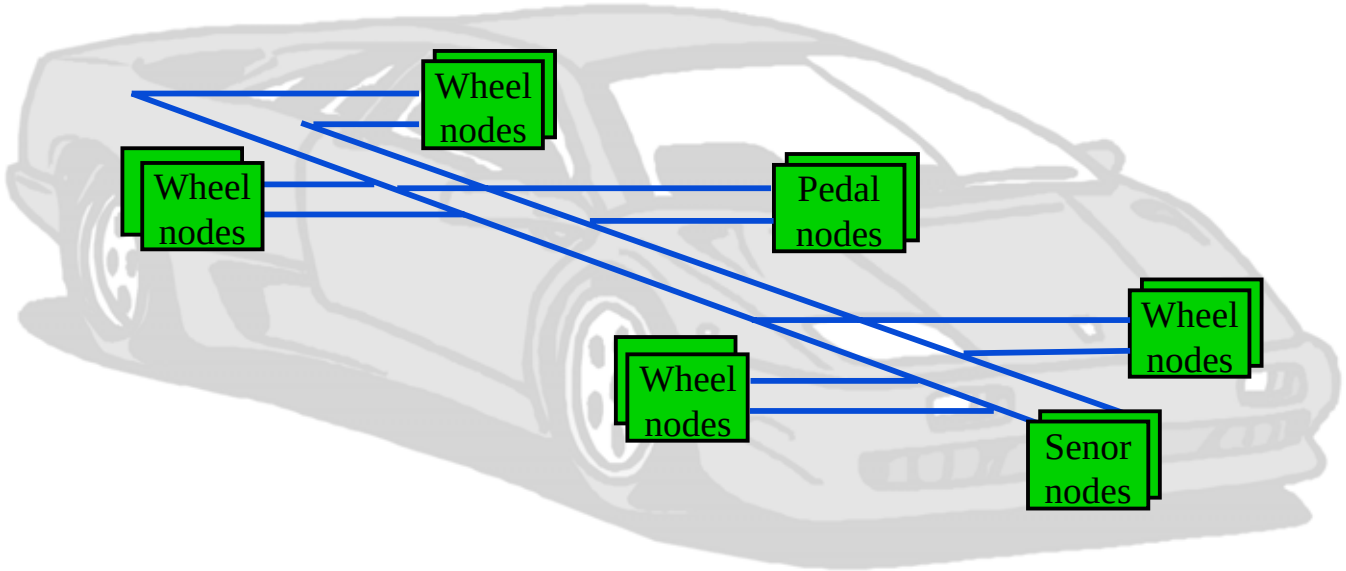
Future electronic systems will include

- Advanced active safety system (e.g. collision mitigation)
- Brake-by-wire
- Steer-by-wire

Reliability and safety requirements of automotive electronic systems will become strict because

- Advanced active safety systems take full control of vehicle
  => Failures may have more severe consequence than in today's systems
- Brake-by-wire and steer-by-wire cannot be shut down while driving.

Brake-by-wire and Collision Mitigation System



- Full authority system - Takes control of the vehicle in emergencies
- False activations are potentially very dangerous
- Main challenge - Systems must be low cost and extremely reliable

Architectural Trade-Offs

Node replication

- Single nodes – cost-effective, but may not achieve adequate partitioning coverage
- Double nodes – provide effective physical partitioning, but costly
- Triple nodes – high degree of fault tolerance, but may be too costly

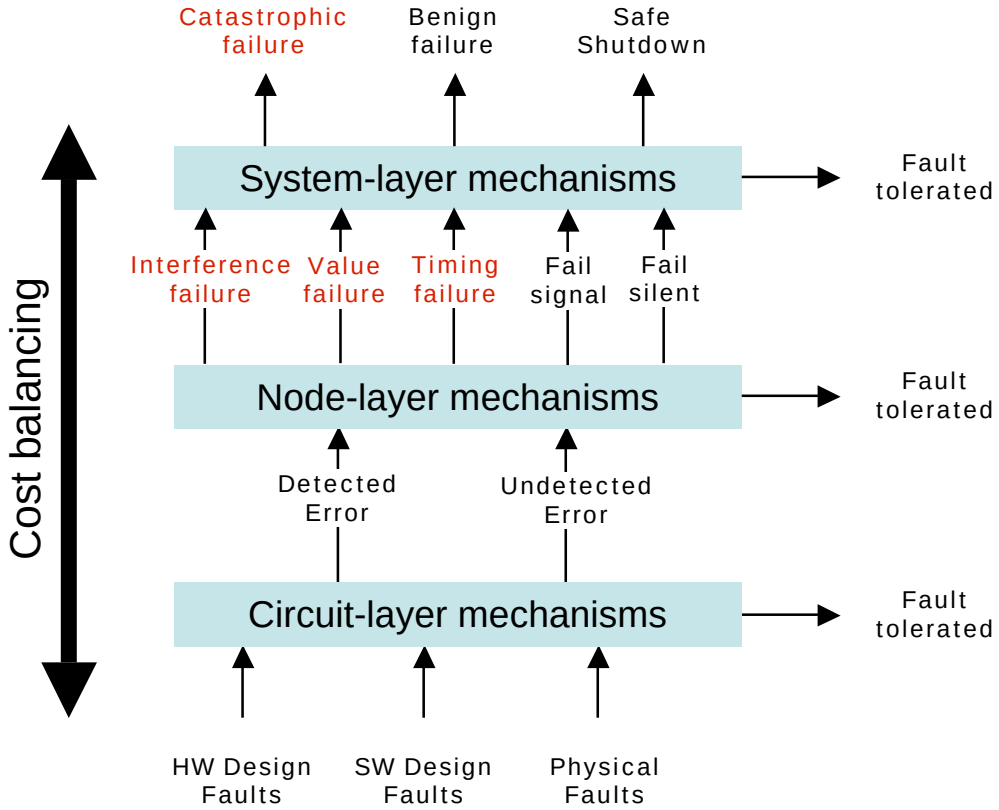
Node design

- Internally fault-tolerant – can become cost-effective with systems-on-chip solutions
- **Self-checking – minimum requirement**
- No error handling (probably not an option)

Network design

- Redundant wired network
  - Bus topology, Star topology, etc.
- Non-redundant wired network with wireless backup

Multi-layer fault-tolerance



Outline

- Objectives, assumptions and research questions
- Causes of soft errors
- Impact of soft errors in the IBM Power6 microprocessor
- Our experimental setup
- Results – impact of soft errors
- Conclusions and ongoing work

Research Objectives

- Investigate the impact of **soft errors** on the wheel control loop of a brake-by-wire system
- Assess the feasibility of using a microcontroller with **non-prefect coverage of soft errors** for the wheel control

Assumptions

- Future microcontrollers will be manufactured in circuit technologies (e.g. 90 nm or 65 nm CMOS) that are **sensitive to cosmic ray induced high energy neutrons**
- These microcontrollers **will be equipped with error detection and error correction mechanisms** that can detect, mask and recover from a majority of the soft errors
- However, these mechanisms **will not have prefect error coverage**
- Hence, some **soft errors will propagate to the architected state** (CPU registers and main memory)

Causes of soft errors

- Terrestrial cosmic rays
  - Primarily neutrons, but also protons and some pions
  - Generated when cosmic particles interact with atomic nuclei in the atmosphere
- Alpha particles
  - Typically emitted from trace amounts of Uranium and Thorium found in production and packing material
- Thermal neutrons (< 0.4 eV) captured by ^{10}B

  $n + {}^{10}B \longrightarrow {}^{7}Li\ (0.84\ MeV) + {}^{4}He\ (1.47\ MeV) + gamma\ (0.48\ MeV)$
- Cross-talk
- Aging faults
  - Negative Bias Temperature Instability (NBTI)
  - ….

Flux of cosmic ray-induced high-energy neutrons

- The neutron flux is influenced by latitude, longitude, altitude, atmospheric pressure, and solar activity
- Reference point: New York City, sea-level, medium solar activity
  - Total flux at NYC is **12.9 cm^{-2} h^{-1}** for neutron energies > 10 MeV
  - Roughly 10 times higher at an altitude of 3000 meters
- The neutron flux at a specific location can be calculated at http://www.seutest.com
- More information can be found in the JEDEC Standard:

  JESD89A - Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices (October, 2006)

Variations in cosmic ray neutron flux at selected locations

Location	Elevation (m)	Atm depth (g/cm2)	Relative neutron flux compared to NYC, sea-level
Bangkok	20	1031	0.52
London	10	1032	0.98
Johannesburg	1770	834	3.13
Stockholm	30	1030	1.04
Los Alamos	2250	786	5.60
South Pole Station	2820	731	9.81

Indicative Figures for the Sensitivity of CMOS circuits

- The ***raw*** soft error rate due to terrestrial high energy neutrons is in the order of 0.001 FIT/latch for sensitive latches in bulk CMOS
- SOI is 2 to 8 times less sensitive than bulk.

Source: Panel presentations at SELSE-2 available at http://www.selse.org

Research Questions

- Will soft errors that reach the architected state (CPU register and main memory) cause catastrophic failures in a brake-by-wire system?
- Can we reduce the probability of such catastrophic failures to a tolerable level by software implemented error detection?

Layout of IBM Power 6 Microprocessor

Please see presentation mentioned below

From presentation at SELSE-3 by Kellington et al., *IBM POWER6 Processor Soft Error Tolerance Analysis Using Proton Radiation,* available at www.selse.org

Overall Derating of BZIP2 running on a POWER6 Processor

Please see presentation mentioned below

From presentation at SELSE-3 by Kellington et al., *IBM POWER6 Processor Soft Error Tolerance Analysis Using Proton Radiation,* available at www.selse.org

Brake-by-wire evaluation

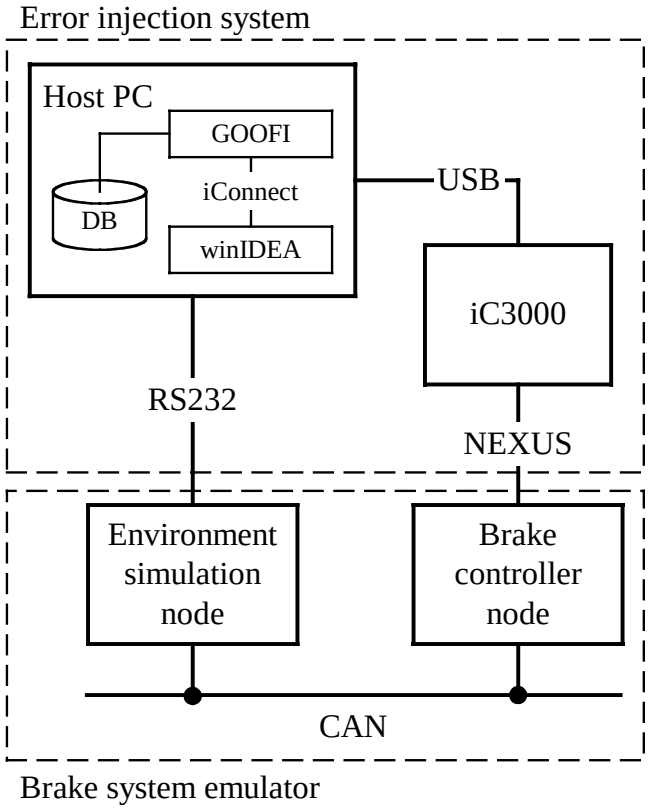
Experimental setup

Brake system emulator

- Two single board computers based on the MPC565 from Freescale
- Brake controller
- Environment simulation model

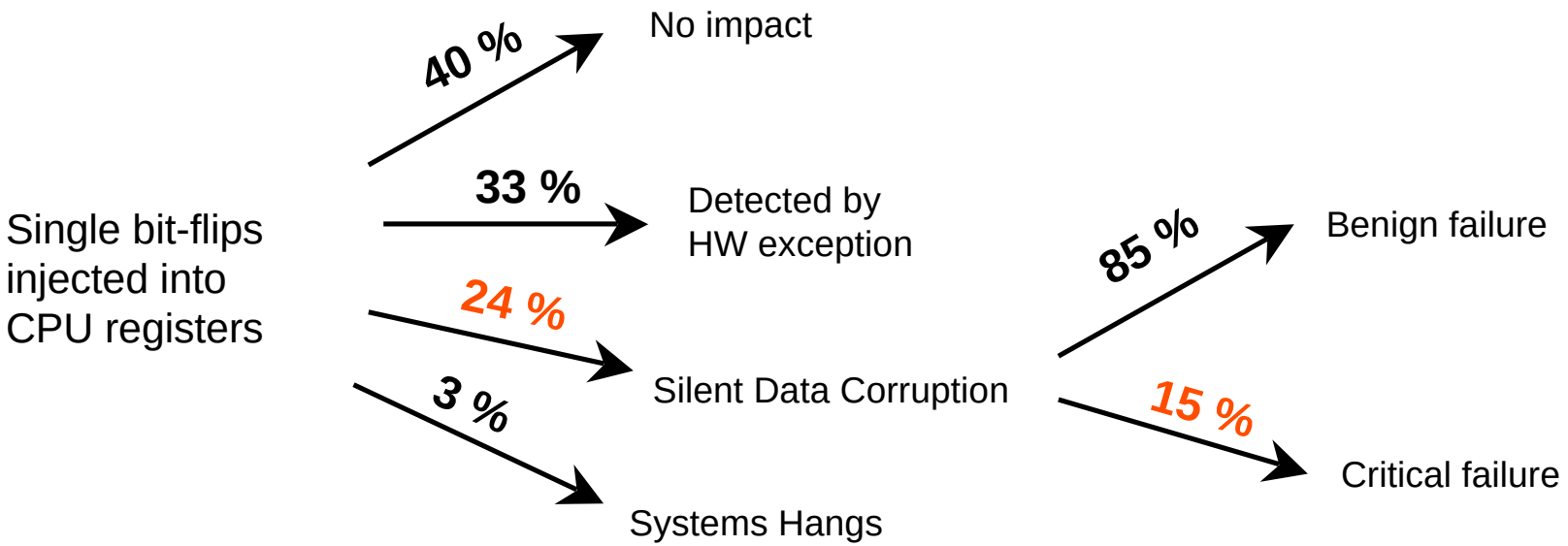
Error injection:

- GOOFI tool
- Pre-injection analysis – injection in live data
- Single bit-flips in registers and data memory



Program derating in brake-by-wire control loop

Maximum deceleration



Critical failures:

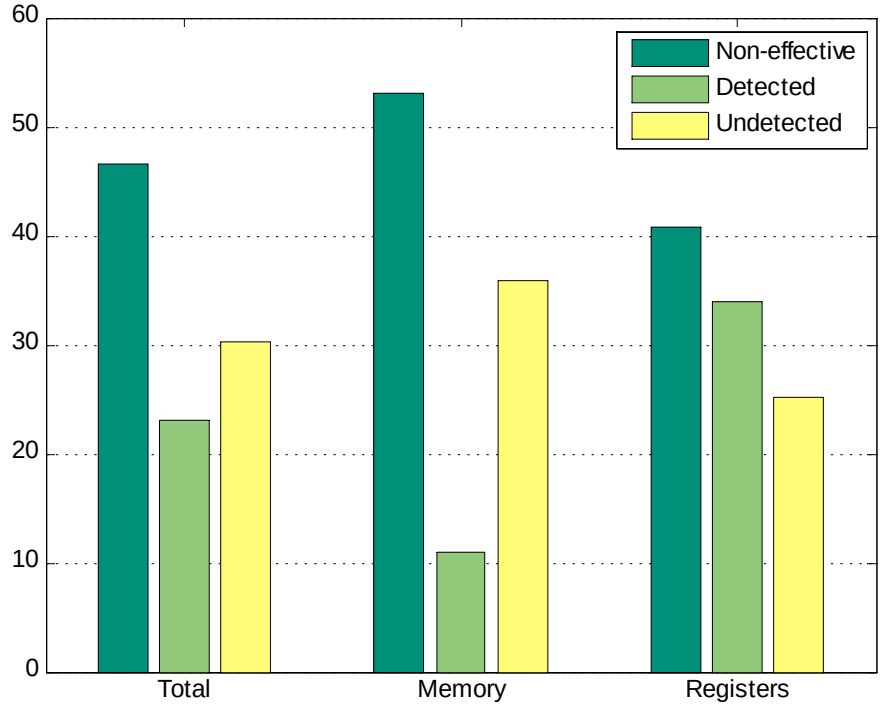
- Wheel locked for more than 0.03 s
- No brake force applied for 0.03s

Program derating 1/(0.24 * 0.15) = 13.9X

Brake-by-wire evaluation

Classification of error impact

- About 30% (1754 of 5802) of the bit-flips caused silent data corruptions
- Memory errors are more likely to cause silent data corruptions

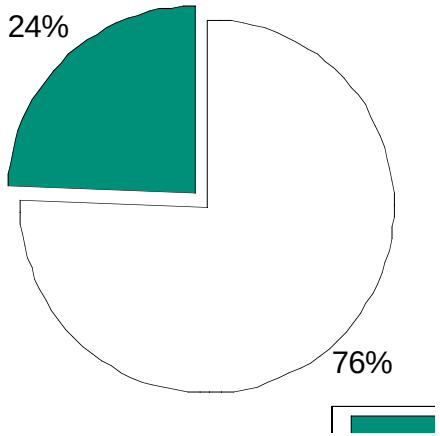


Brake-by-wire evaluation

Critical failures

About 15% (268 of 1754) of the errors that propagated to the output resulted in a critical failure

- Wheel being locked (41% of the critical failures)
- Loss of braking (59% of the critical failures)

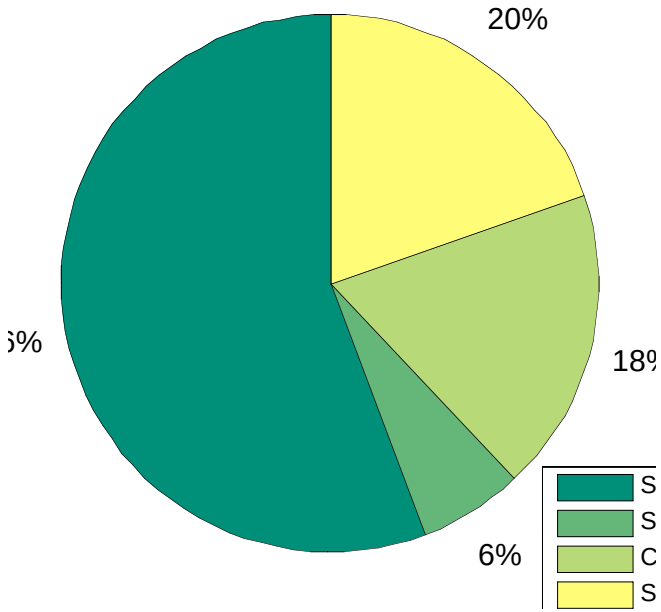


Brake-by-wire evaluation

Critical failures

A majority of the critical failures were caused by

- Errors injected into the stack pointer
- Errors affecting the scheduler
- Errors affecting the brake controller state



Brake-by-wire evaluation

Program-level error masking

- About 47% of the injected errors were non-effective even though errors were injected into live data
- Memory errors were masked more often than register bit-flips
  - 51% of the memory bit-flips were masked
  - 40% of the register bit-flips were masked

Conclusions

- High degree of program-level error masking
  - 47% of the injected errors did not have an effect on the produced brake command even though the bit-flips were injected into live data
- 4.6% of all injected errors resulted in critical failures
- 30% of the injected errors passed undetected
  - About 15% of these errors resulted in critical failures
- Paper available at www.selse.org
- On-going work
  - Implementation of software-based error detection mechanisms
  - Evaluate error coverage (program derating) of these mechanisms

Questions?