Multicore Debugging Challenges for the Automotive Domain

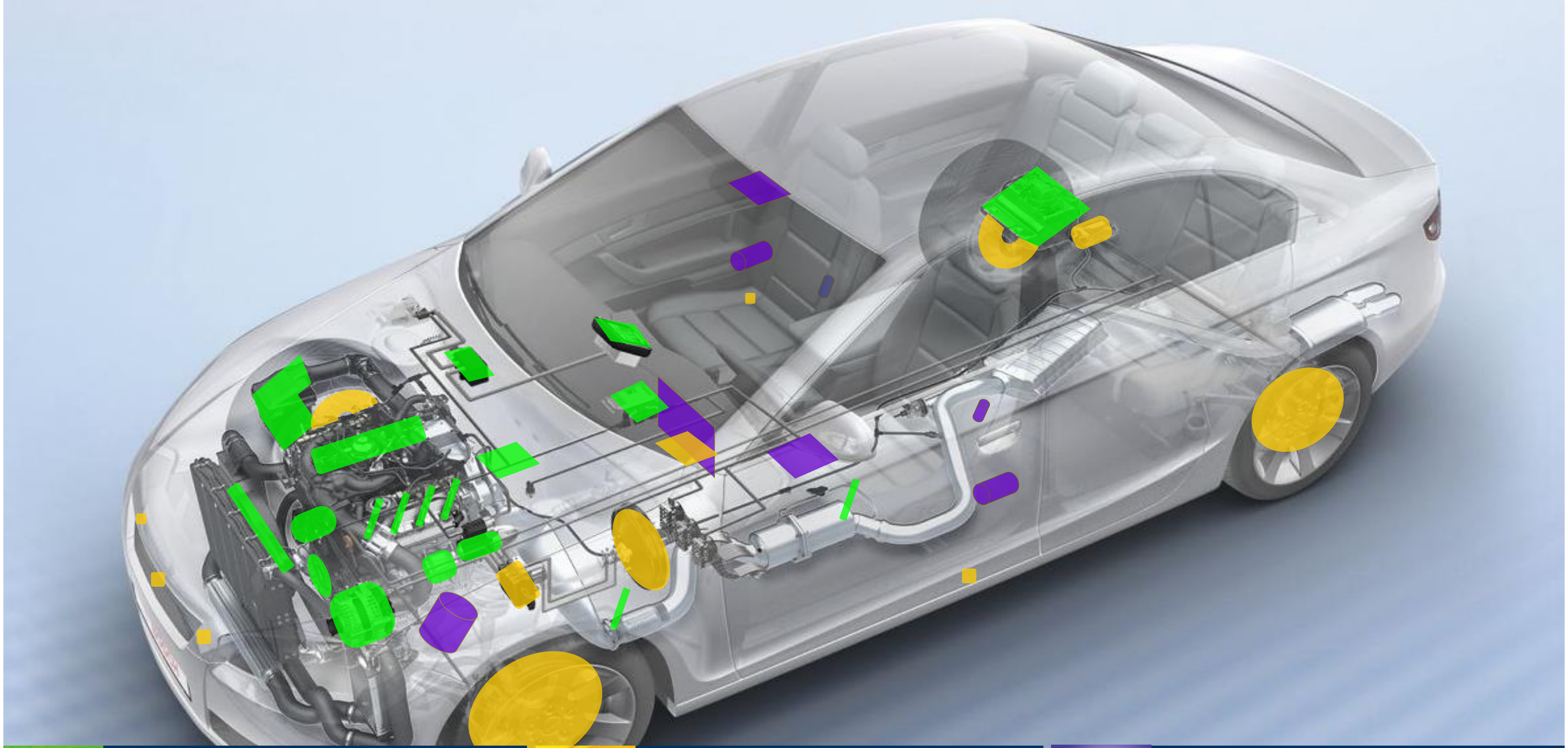
1. International Workshop on
Multicore Application Debugging (MAD 2013)

Nov 14 -15, 2013
Technische Universität München, Germany

Overview

1. The automotive context
2. Failure Modes and Debugging Needs

Automotive Technology: an overview



🟩 clean & economical — Powertrain systems

🟨 safe — Safety systems

🟪 comfortable — Comfort systems

Engine Control Domain

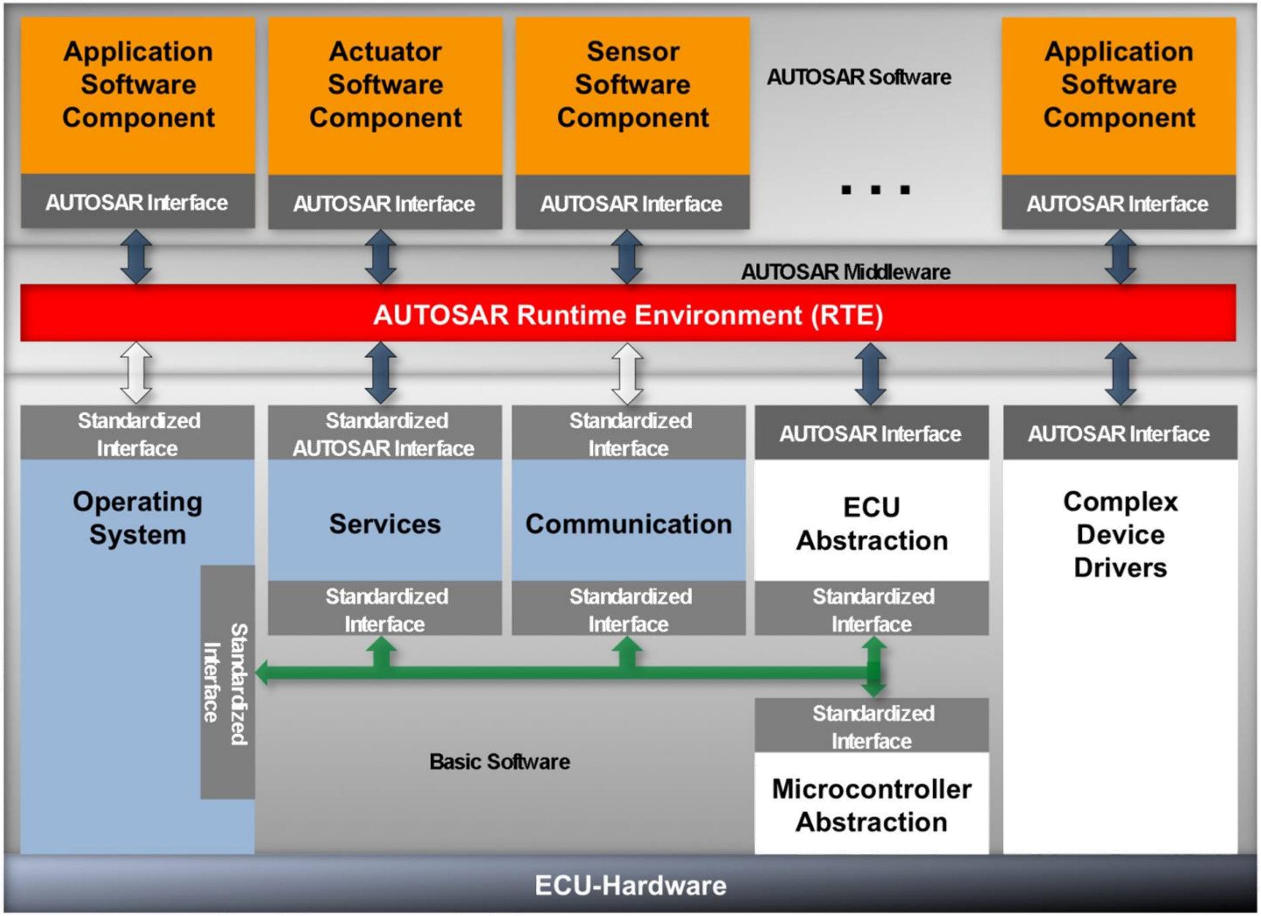
- Engine Control Unit for Diesel and Gasoline Engines
- **Challenges**: Real-Time computation of several injections for up to 12 cylinders @ 8,000 RPM, reading and processing of up to 50 sensor signals, ...
- **Hardware**: typically 1-2 MB internal flash (optional 2-4 MB external flash), 64-144 kByte internal data-RAM, CAN, TT-CAN, Flexray, up to 48 A/D converters
- **Software**: > 500,000 LOC,
  > 1,000 components,
  > 2000 variant points

Generic View: The System Setup

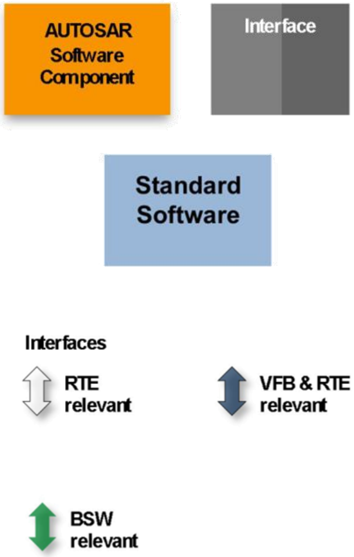
- Time-Triggered & Event-Triggered tasks
- High degree of internal and external communication
- Time-Critical Cause-Effect Chains
- Safety & Security
- Number of variants
- Number of developers
- SW sharing
- Increasing complexity of systems
- Calibration

Multicore Application Debugging

AUTomotive Open System ARchitecture



Legend

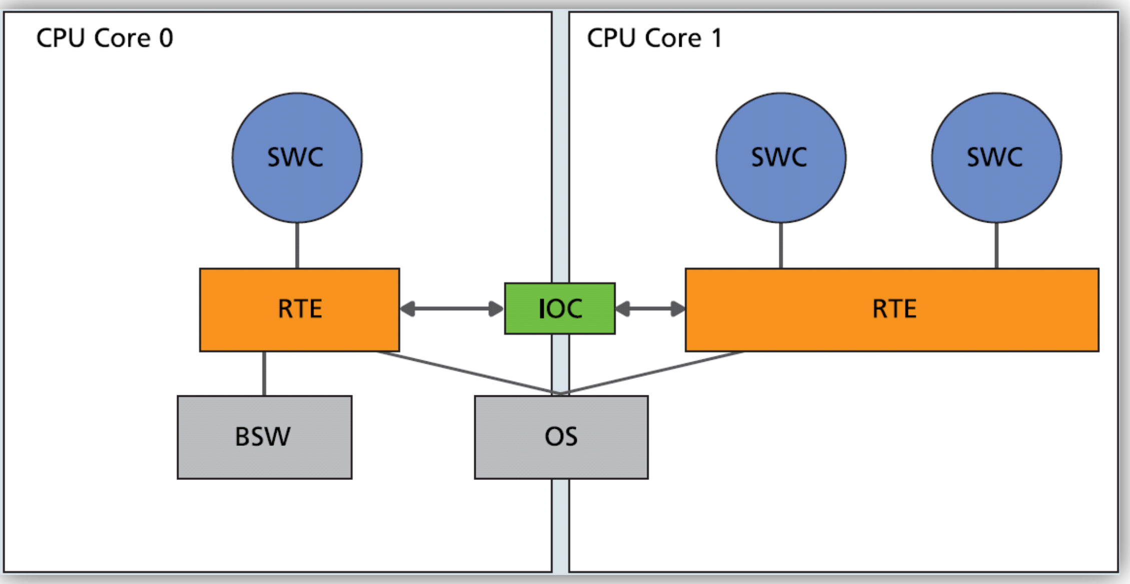


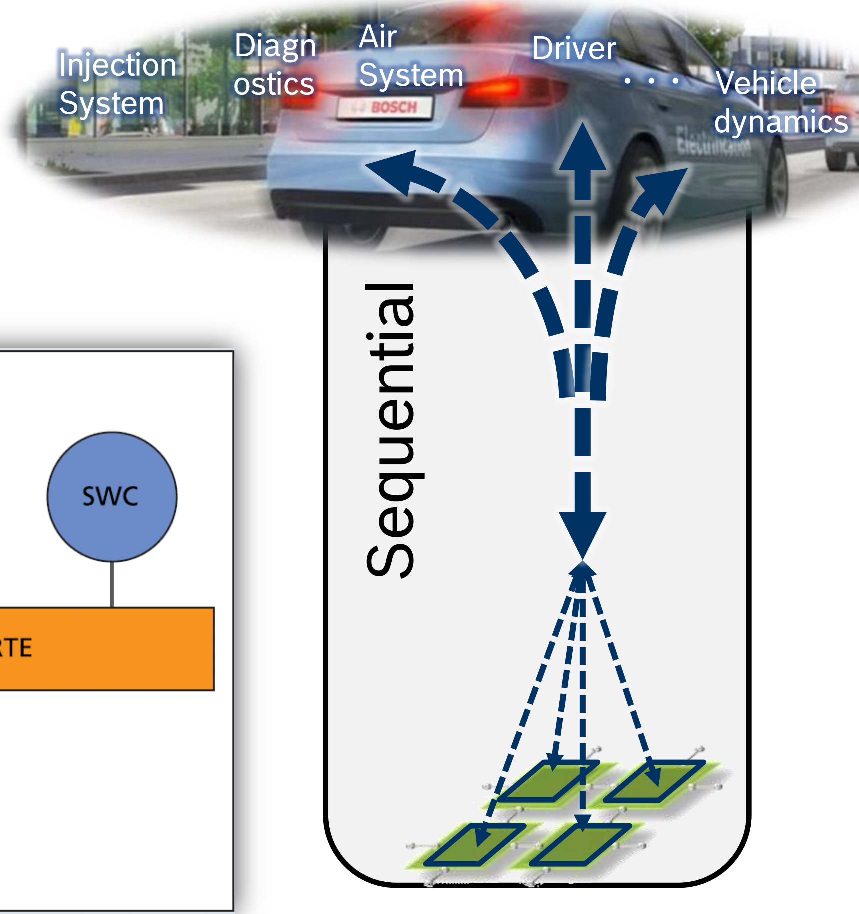
Source
http://www.autosar.org

AUTOSAR 4.0: From Single-Core to Multi-Core

Underlying Assumptions

- Shared Memory
- Unrestricted accesses
- Static Distribution





Overview

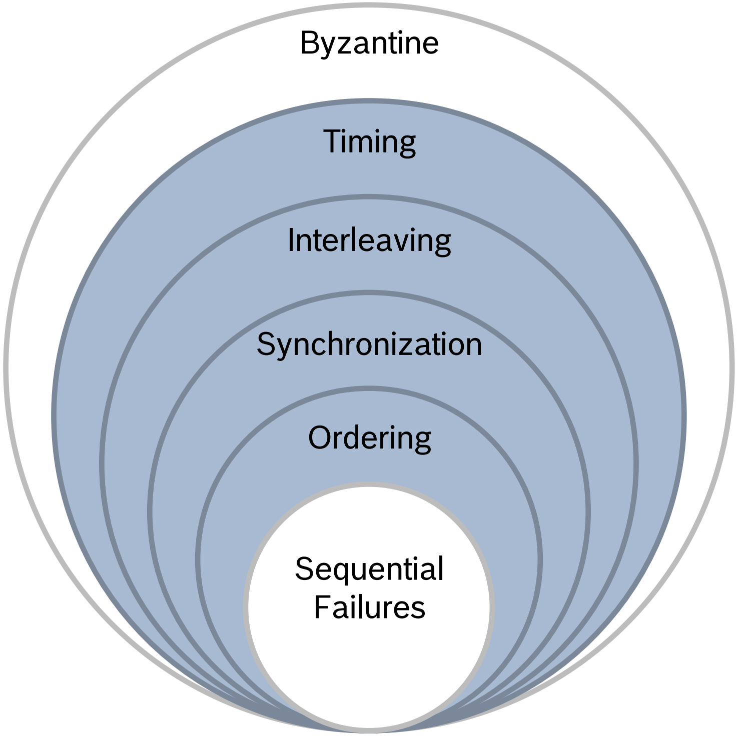
1. The automotive context

2. **Failure Modes and Debugging Needs**

The Different Failure Modes

- ➔ Sequential failure modes
  - Remain same for parallel systems
  - Established methods are sufficient

- ➔ Concurrency-related failure modes
  - Are present in preemptive systems already
  - Probabilities change in parallelized systems
  - Intrusive debugging mechanisms change real-time behaviors
  - Debugging of several cores extremely difficult



The failure modes

From: Henrik Thane, Monitoring, Testing and Debugging of Distributed Real-Time Systems, Doctoral Thesis
Mechatronics Laboratory, Department of Machine Design Royal Institute of Technology, KTH S-100 44 Stockholm, Sweden

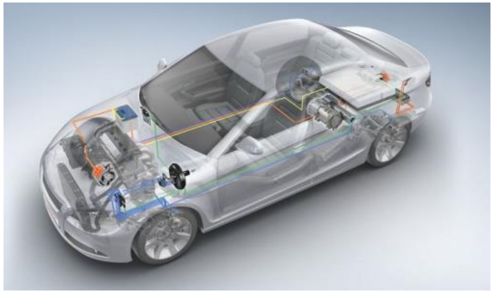
Debugging Needs

Code Level Debugging

- Reasonably usable parallel debugging tooling
- Intrusive-free debugging alternatives
- Dealing with heterogeneous hardware architectures

System Level Debugging

- Capable for usage in real-time operation
- Addressing of singular events and corner cases
- Tool-based support of detecting source of failures
- Debugging of system level assumptions (e.g. deadlines, scheduling orders)



Thank you for your attention.