Segregation of Subsystems with Different Criticalities on Networked Multi-Core Chips in the DREAMS Architecture

University of Siegen

Roman Obermaisser
Overview

- Mixed-Criticality Systems
- Modular Certification
- Fault Containment
- Integration Levels of Mixed-Criticality Systems
- Temporal and Spatial Partitioning of Time-Triggered Systems
- European Research Project DREAMS: Mixed-Criticality Systems based on Networked Multi-Core Chips
Mixed-Criticality Systems

- Need for mixed-criticality systems due to pressing requirement to reduce the number of nodes and cables
- Integration of functions with different importance and certification assurance levels on a shared computing platform
- Trend to multi-core platforms due to limited scalability of uniprocessors (Pollack's rule)
- Latest processors in industrial applications with multiple cores, but typically only one core is used when highly-critical tasks are involved
System Architecture for Mixed-Criticality Systems

• **Architectural style** with rules for the design of a mixed-criticality system
  – Structuring the system into components (e.g., component model, interface types)
  – Defining component interactions and linking interfaces (LIFs), e.g., time model, interaction patterns, fault model

• **Platform with architectural services** (e.g., communication, diagnosis, fault-tolerance) as a stable baseline for the implementation and integration of applications
Modular Certification

• Composition of safety arguments from individual subsystems
• Each application subsystem is certified to the respective level of criticality
• Generic arguments of a verified platform
• Fault containment is a fundamental requirement for modular certification

"...all the hardware and software shall be treated as safety-related unless it can be shown that the implementation of the safety and non-safety functions is sufficiently independent."
[IEC 61508-2, Ed. 2, p.20]
Fault Containment and Encapsulation

- Establishment of **Fault Containment Regions (FCRs)** as delimiter of the immediate impact of a fault
  - Depends on fault assumptions (e.g., physical faults vs. design faults)
  - Determines containment coverage of fault hypothesis
- *Prevent or bound interference* and unintended side-effects due to integration
- *Temporal Partitioning*: temporal encapsulation of resources (e.g., latency, jitter, duration of availability during a scheduled access)
- *Spatial Partitioning*: prevent FCR from altering code or private data of other partitions
Time-Triggered Control

• Schedule plan provides control signals with respect to a global time base
• Important in many safety-critical systems (e.g., TTP in avionics, FlexRay in automotive)
• Properties
  – Temporal predictability
  – Fault containment
  – Absence of interference
  – Composability
  – Determinism
Time-Triggered Networks at Cluster-Level

- Fault containment and absence/bounding of interference using TDMA scheme enforced by local or central guardians
- Guardians have a priori knowledge about the permitted behavior of components
- Blocking of untimely messages (e.g., babbling idiot)
- Blocking of invalid messages (e.g., syntax, addressing, slightly-off-specification)
- Examples: TTEthernet, TTP, FlexRay, SAFEbus
Time-Triggered Network at Chip-Level

- Time-Triggered Network-on-a-Chip
  - Global time base despite multiple clock domains
  - Source-controlled behavior of network using trusted network interfaces
  - Exclusive slots for components using a TDMA scheme

- Network interface as guardian for a component
- Trusted Resource Manager (TRM) as guardian for configuration
Time-Triggered Operating Systems and Hypervisors

- Temporal partitioning using static cyclic schedule tables based on a time-triggered execution plan
- Spatial partitioning typically based on a Memory Management Unit (MMU)
- Multi-level hierarchic scheduling with guest operating systems in partitions
- Examples:
  - XtratuM
  - PikeOS
  - LynxOS178
  - VxWorks

<table>
<thead>
<tr>
<th>Partition</th>
<th>Partition</th>
<th>Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Application</td>
<td>Hypervisor / OS</td>
</tr>
<tr>
<td>GuestOS e.g., Lithos</td>
<td>GuestOS e.g., Linux</td>
<td>Hardware</td>
</tr>
</tbody>
</table>
Research Challenge: Networked Multi-Core Chips

• Temporal unpredictability of multi-core processors
  – Shared resources are a source of indeterminism in execution time analysis: cache, memory accesses, bus arbitration policy
  – No clear understanding for effects of inter-task interferences on timing analysis
  – Low-criticality tasks must not create unpredictable effects on the execution of critical ones (time analysability)

• Interconnection of multi-core processors
  – Satisfy resource requirements exceeding the resources of a single node computer
  – Higher system reliability than component reliability
  – End-to-end systems engineering addressing significant extra functional properties (e.g., time, energy, reliability, security)
Research Challenge: Heterogeneity

- Mixed-criticality systems typically exhibit heterogeneous platform requirements in addition to different criticality levels.
- Dissimilar requirements in terms of timing (e.g., firm, soft, hard, non real-time).
- Different models of computation (e.g., dataflow, time-triggered messaging, distributed shared memory).
European Research Project DREAMS

• Project full title: Distributed REal-time Architecture for Mixed criticality Systems
• Project duration: October 1, 2013 – Sept. 30, 2017
• Type of project: Integrated Project (IP)
• Budget Total: 15.5 mill. EUR

<table>
<thead>
<tr>
<th>Industry</th>
<th>ONERA</th>
<th>France</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thales SA</td>
<td>Ikerlan</td>
<td>Spain</td>
</tr>
<tr>
<td>Alstom Wind S.L.</td>
<td>SINTEF</td>
<td>Norway</td>
</tr>
<tr>
<td>STMicroelectronics</td>
<td>Fortiss</td>
<td>Germany</td>
</tr>
<tr>
<td>TÜV Rheinland</td>
<td>TÜV Rheinland</td>
<td>Germany</td>
</tr>
<tr>
<td>TTTech</td>
<td>Universität Siegen</td>
<td>Germany</td>
</tr>
<tr>
<td>RealTime-At-Work</td>
<td>TU Kaiserslautern</td>
<td>Germany</td>
</tr>
<tr>
<td>Virtual Open Systems</td>
<td>UPV</td>
<td>Spain</td>
</tr>
<tr>
<td>FENTISS</td>
<td>TEI</td>
<td>Greece</td>
</tr>
</tbody>
</table>
Project Description

Mixed-criticality architecture based on networked multi-core chips

1. Architectural style and modelling methods
2. Virtualization technologies for security, safety, real-time performance, integrity in networked multi-core chips
3. Adaptation strategies for mixed-criticality systems
4. Development methodology and tools based on model-driven engineering
5. Certification and mixed-criticality product lines
6. Feasibility of DREAMS architecture in real-world scenarios
7. Promoting widespread adoption and community building
Main Technical Activities

• Cross-domain architectural style and models for MCS
• Modular certification and mixed-criticality product lines
• Platform with virtualization at chip and network level
• Adaptation strategies for mixed-criticality systems
• Development methodology, variability management and tools
Main Outcome and Results

- Reduced development cost and time-to-market for mixed-criticality applications
- Exploitation of economies of scale through cross-domain components and tools
- Consolidation and integration of virtualization solutions and development methods from previous projects
- Significant advances in virtualization techniques leading to higher reliability, security and safety
- Higher flexibility, adaptability and energy efficiency through integrated resource management
- Leverage multi-core platforms for a system perspective of mixed-criticality applications combining the chip-level and network-level
Conclusion

• Increasing importance of mixed-criticality systems based on multi-core processors
• Temporal and spatial partitioning is the foundation for mixed-criticality integration and modular certification
• Technological challenges in end-to-end virtualization of resources and heterogeneous models of computations