On safety and real-time in embedded operating systems

using modern processor architectures in different safety-critical applications

OSPERT WS - Keynote - 2018-07-03

Michael Paulitsch
Legal Notices and Disclaimers

This presentation contains the general insights and opinions of Intel Corporation (“Intel”). The information in this presentation is provided for information only and is not to be relied upon for any other purpose than educational. Use at your own risk! Intel makes no representations or warranties regarding the accuracy or completeness of the information in this presentation. Intel accepts no duty to update this presentation based on more current information. Intel is not liable for any damages, direct or indirect, consequential or otherwise, that may arise, directly or indirectly, from the use or misuse of the information in this presentation.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel and the Intel logo are trademarks of Intel Corporation in the United States and other countries.

*Other names and brands may be claimed as the property of others.

© 2018 Intel Corporation
AT INTEL

WE’RE POWERING THE FUTURE OF COMPUTING AND COMMUNICATIONS, DELIVERING EXPERIENCES ONCE THOUGHT TO BE IMPOSSIBLE.
What is Dependability & Security?

Dependability an integrating concept that encompasses the following attributes:

- **Availability** - readiness for correct service
- **Reliability** - continuity of correct service
- **Safety** - absence of catastrophic consequences on the user(s) and the environment
- **Integrity** - absence of improper system alteration
- **Maintainability** - ability for a process to undergo modifications and repairs

**Security**: composite of the attributes of confidentiality, integrity, and availability, requiring the concurrent existence of 1) availability for authorized actions only, 2) confidentiality, and 3) integrity with “improper” meaning “unauthorized”

Laprie et al 2004:
Safety Assurance Levels in Aerospace and Railway (e.g. DO-178C/ED-12C, EN 50129, …)

Software/hardware whose anomalous behaviour would cause or contribute to a failure of system function resulting in a failure condition for the aircraft / railway system that is:

- **Level A - Catastrophic**  
  \(10^{-9}\) failures/hour

- **Level B - Hazardous/Severe-Major**

- **Level C - Major**

- **Level D - Minor**

- **Design Assurance Level E - No Effect**

- **Safety Integrity Level - SIL 0 (non-SIL)**

- **SIL 4** \(10^{-8}\) failures/hour

- **SIL 3**

- **SIL 2**

- **SIL 1**

- **SIL 0**
Avionics

Electronics in Airplane
Trends in Aerospace

Trend towards new and additional IT-services and denser functional integration:

- Demand for new and additional IT-services on aircraft itself and between aircraft and ground
- Integrate formerly physically separated functions onto one platform
- New failure modes and failures
- New threats and vulnerabilities (security, but affecting safety)
Trend Towards Integrated Modular Avionics (IMA)

Due to weight constraints integration of multiple aircraft functions (of possibly different criticality) onto common platforms is an ongoing architectural trend in aerospace.

A380 IMA components

Source: Airbus © Airbus

Relationship of IMA applications and HW/SW Modules

Source: ARINC297 © ARINC
Mixed-Criticality System in Industry – What’s it?

Multiple criticalities (residing) on same platform

- Key requirement for platform: Platform needs to fulfill safety requirements at minimum of highest safety requirement of application. Security criticality requirements may be derived from safety requirements or from security data separation.

- Criticalities are assigned by safety or security process and typically don’t change during operation

- Safety: Chosen independence between applications to minimize interaction between otherwise independent “safety chapters” (system level safety analysis extremely complicated w/o this requirement).

- Security: co-habitance of different security levels needed for cost reasons or because of inherent security function (gateway, firewall)

- Deployed for many years in aerospace (B777, B787, A380, A350, E170/175, E190/195, …) under the name Integrated Modular Avionic (IMA) systems
Aircraft Cockpit

Legend:
PFD … Primary Flight Display
ND … Navigation Display
MFD … Multi-Function Display
EICAS … Engine Info & Crew Alert System
Boeing 777 Avionics Architecture
Real-Life Mixed Criticality System
Boeing 777 – Avionics – Computer Level
Avionics based on ARINC629 system bus and ARINC659 (SafeBus).

Deterministic relatively simple compute and network architecture
Partitioning

Is a concept for spatial and temporal separation/segregation of functionally independent components:

- Prevents interference between two components
- Incremental development

Types of partitioning
- Time partitioning: temporal aspect
- Space partitioning: memory aspect
- I/O partitioning: time and space partitioning for I/O

Implementation means
- Partition/process: independent segregated environment
- Separation kernel / Memory Management Unit: control instance
- Temporal partitioning: time slicing; dynamic (fair) scheduling policies
How to Achieve Availability and Integrity in a Mixed-Criticality System?

Correctness of implementation important for safety and availability

Examples of High-Assurance Requirements

- Domains need to fulfill **separation** requirements despite possible integration on same hardware to ensure proper item integrity and availability

- **Controlled information flow**: Communication between domains need to fulfill rules to ensure proper protection of functions – stronger focus on
  - Integrity and availability of functions
  - Authorized flow definition
Orion
Multi-Purpose Crew Vehicle
Next generation U.S. spacecraft
Long mission times (weeks to 6 months)
Avionics (Snapshot)

- Time-triggered network
- High-integrity compute
- System-level redundancy management
Multi-core
Time Partitioning
Chip Evolution

Host processor

- Processor
  - Bridges
  - I/O

Main Memory

- Processor
  - Bridges
  - Integrated I/O
  - External I/O

Main Memory

- Processor
  - Network on Chip
  - Integrated I/O
  - External I/O

Increasing integration density and complexity
View of Aerospace **Multi-Core** Certification Body Related to Timing

Only selective view of publicly available FAA CAST-32 paper

(Functional) interference channels of multi-core processors

- Concerns: there may be software or hardware channels through which the MCP cores or the software hosted on those cores could interfere with each other

Shared resources like Memory / Cache

- Concerns: Memory or cache memory that are shared between the processing cores
- … can lead to problems such as the worst-case execution times (WCETs) of the software applications hosted on cores increasing greatly due to repeated cache accesses by the processes hosted on the other core, leading to repeated cache misses.

Planning and Verification of Resource Usage

- Concern: Interconnect Fabrics / Interconnect Modules as source of non-deterministic behavior, fear of resource capacity violation, …
Multi-core: General Possible Undesired Effects (Temporal)

Other possible undesired effects affecting temporal determinism

- How does current hardware affect mixed criticality and especially interference?
- What can be done about it (analysis, improvement, inclusion in processes) especially in current commercial off the shelf (COTS) architectures.

Details in papers


<table>
<thead>
<tr>
<th>Shared resource</th>
<th>Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>System bus</td>
<td>Contention by multiple cores</td>
</tr>
<tr>
<td></td>
<td>Contention by other device - IO, DMA, etc.</td>
</tr>
<tr>
<td></td>
<td>Contention by coherency mechanism traffic</td>
</tr>
<tr>
<td>Bridges</td>
<td>Contention by other connected busses</td>
</tr>
<tr>
<td>Memory bus and controller</td>
<td>Concurrent access</td>
</tr>
<tr>
<td>Memory (DRAM)</td>
<td>Interleaved access by multiple cores causes address set-up delay</td>
</tr>
<tr>
<td></td>
<td>Delay by memory refresh</td>
</tr>
<tr>
<td>Shared cache</td>
<td>Cache line eviction</td>
</tr>
<tr>
<td></td>
<td>Contention due to concurrent access</td>
</tr>
<tr>
<td></td>
<td>Coherency: Read delayed due to invalidated entry</td>
</tr>
<tr>
<td></td>
<td>Coherency: Delay due to contention by coherency mechanism read requested by lower level cache</td>
</tr>
<tr>
<td></td>
<td>Coherency: Contention by coherency mechanism on this level</td>
</tr>
<tr>
<td>Local cache</td>
<td>Coherency: Read delayed due to invalidated entry</td>
</tr>
<tr>
<td></td>
<td>Coherency: Contention by coherency mechanism read</td>
</tr>
<tr>
<td>TLBs</td>
<td>Coherency overhead</td>
</tr>
<tr>
<td>Addressable devices</td>
<td>Overhead of locking mechanism accessing the memory</td>
</tr>
<tr>
<td></td>
<td>I/O Device state altered by other thread/application</td>
</tr>
<tr>
<td></td>
<td>Interrupt routing overhead</td>
</tr>
<tr>
<td></td>
<td>Contention on the addressable device - e.g. DMA, Interrupt controller, etc.</td>
</tr>
<tr>
<td></td>
<td>Synchronous access of other bus by the addressable device (e.g. DMA)</td>
</tr>
<tr>
<td>Pipeline stages</td>
<td>Contention by parallel hyperthreads</td>
</tr>
<tr>
<td>Logical units</td>
<td>Contention by parallel applications</td>
</tr>
<tr>
<td></td>
<td>Other platform-specific effects, e.g. BIOS Handlers, Automated task migration, Cache stashing, etc.</td>
</tr>
</tbody>
</table>
Assessment of Multi-Core Worst-Case Execution Behavior - Overview

Motivation:
- Integration leads to common use of shared resources. Partitioning impact needs to be evaluated for safety-critical applications, such as IMA

Goal:
- Analysis of partitioning features of modern multi-core computer in context of use in IMA
- Impact of integration on worst-case timing (WCET) of application

Approach
- memory-intensive tests

Focus of work:
- Network on Chip (limited data available); some memory access performance tests
  Details of work published at EDCC2012 (J. Nowotsch, M. Paulitsch)
Assessment of Multi-Core WCET Memory (DDR) Accesses (8 Cores)

Worst-case access time increases over-proportionally with more cores.
## Some Measured Values for NXP P4080 Interference Between Single-Core and 8-Core Systems

### Worst-case influence (for 8 core multi-core system)

Worst case observed versus worst-case analysis → some conclusions can be drawn for average case (slack between average and worst case)

<table>
<thead>
<tr>
<th>bmark</th>
<th>max. OET [ms]</th>
<th>upper bound [ms]</th>
<th>bound deviation [%]</th>
<th>max. OET [ms]</th>
<th>upper bound [ms]</th>
<th>bound deviation [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>cacheb</td>
<td>619</td>
<td>705</td>
<td>13.9</td>
<td>1934</td>
<td>9378</td>
<td>384.9</td>
</tr>
<tr>
<td>iirflt</td>
<td>745</td>
<td>951</td>
<td>27.7</td>
<td>2476</td>
<td>12497</td>
<td>404.8</td>
</tr>
<tr>
<td>rspeed</td>
<td>963</td>
<td>1418</td>
<td>47.3</td>
<td>2327</td>
<td>19021</td>
<td>717.3</td>
</tr>
<tr>
<td>a2time</td>
<td>121</td>
<td>251</td>
<td>107.3</td>
<td>334</td>
<td>2971</td>
<td>790.9</td>
</tr>
<tr>
<td>bitmnp</td>
<td>2300</td>
<td>3504</td>
<td>52.4</td>
<td>5781</td>
<td>49170</td>
<td>750.5</td>
</tr>
<tr>
<td>tblook</td>
<td>2699</td>
<td>4556</td>
<td>68.8</td>
<td>7684</td>
<td>61156</td>
<td>695.9</td>
</tr>
<tr>
<td>matrix</td>
<td>464</td>
<td>8075</td>
<td>1642.0</td>
<td>1212</td>
<td>98075</td>
<td>7993.5</td>
</tr>
<tr>
<td>aiffr</td>
<td>188</td>
<td>1217</td>
<td>547.4</td>
<td>489</td>
<td>159313</td>
<td>32513.9</td>
</tr>
</tbody>
</table>

> >> 8 times greater

Difference greater for multi-core (more “slack”)

Context info: EEMBC benchmark; OET ... Observed Execution Time; bound ... analyzed using AbsInt AiT
WCET for Multi-Core Computer Combined with Monitoring

Basic idea to benchmark/analyze hardware and include access interference and monitor memory accesses (RTNS 2013 paper, ECRTS 2014 paper)
- Extension of timing analysis
- Applied to AbsInt’s aiT – commercial static WCET framework (extension memory accesses)
- Runtime Monitoring
- Applied to bare-metal OS layer
- Applied to SYSGO’s PikeOS
- Applied to Windriver VxWorks

Average-Case Extension
- Applied to bare-metal OS layer

Evaluation
- Based on Freescale’s P4080, other processors evaluated
- Benchmarks deduced from EEMBC Autobench benchmark suite

WCET reduction:
- Utilisation increase: core 98.9%, system 55%
- Additional accesses: 2 to 70 times the accesses that were statically assigned (Nowotsch et al, 2014+15)
Evaluation – Runtime Analysis
Complexity Is Increasing …

How would such an approach scale with “more” complex systems?

What about new memory architectures?

Memory accesses are not an optimal measure of progress: are there other metrics achieving better WCET and performance?

What about more DMA channels? I/O?
New Memory Architectures/Properties

Are there different OS structures with different memory properties?

E.g. Optane memory

- Persistence
- Quick access

Can we leverage this for improved guaranteed performance?
Time-Coordinated Computing (TCC) & Time-Sensitive Networks (TSN)

TCC … coordination of peripherals and across SOC

TSN (802.1Q) … Ethernet timing sync, path control and reservation, …

Is there a new system optimum? Are we back in the “old days”?

OS support in critical systems?
In the end …

Simplicity needed for timing guarantees (availability) affecting safety

Integrity is a must

More diverse computing requirements (safety-criticality / real-time ) expected
Differentiate workload

- Tight timing requirements
- Criticality

Possible consequences

- Intelligent load management
- Slicing of computing & networking with guarantees
Virtualization is Key

Current Data Center Hypervisors

- Too large for embedded IoT development
- No safety-critical workload considerations
- Requires too much overhead for embedded development

Current Embedded Hypervisors

- Highly dependent on closed source proprietary solutions
- Expensive
- Makes product longevity difficult
- Hard partition, no ability to share resources

No Open Source Hypervisor solution currently exists that is optimized for embedded IoT development
ACRN™ is a flexible, lightweight reference hypervisor, built with real-time and safety-criticality in mind, optimized to streamline embedded development through an open source platform.

<table>
<thead>
<tr>
<th>Small footprint</th>
<th>Built with Real Time in Mind</th>
<th>Built for Embedded IoT</th>
<th>Safety Criticality</th>
<th>Adaptability</th>
<th>Truly Open Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Optimized for resource constrained devices</td>
<td>• Low latency • Enables faster boot time • Improves overall responsiveness with hardware communication</td>
<td>• Virtualization beyond the “basics” • Virtualization of Embedded IoT dev functions included • Rich set of I/O mediators to share devices across multiple VMs</td>
<td>• Safety critical workloads have priority • Isolates safety critical workloads • Project is built with safety critical workload considerations in mind</td>
<td>• Multi-OS support for guest operating systems like Linux and Android • Applicable across many use cases</td>
<td>• Scalable support • Significant R&amp;D and development cost savings • Code transparency • SW development with industry leaders • Permissive BSD licensing</td>
</tr>
</tbody>
</table>
Railway
Overview Railway – Signal Control

Trends

- Removal of some field elements (signals, …)
- Remote moving authority
- Central operation centers
- Autonomous operation

RBC … remote block center
OBU … on-board unit

© Thales
Thales - TAS Platform

- Vital Hardware & Software Platform, common for all signalling applications in Ground Transportation Systems (GTS)
- Enables hardware independent signalling applications
TAS Control Platform: Supported Redundancy Architectures

TAS Control Platform:
- Supported Redundancy Architectures
  - 2x2002
  - 2002
  - 1001

Diagram showing various architectures for TAS Control Platform with application configurations.
TAS Platform – Safe Computation and Communication Method & Tools

- **Vital Platform**: common for all signalling applications in GTS
- **Enables hardware independent signalling applications**
- **CENELEC EN50129 SIL 4 Certification**
- **A generic product line deployed all over the world**

**TAS Platform – Safe Computation and Communication**

**Method & Tools**

- **PLF Core (OS)**
- **PLF Hardware (Boards)**
- **OCS (Communication)**

**Support & Maintenance Framework**

- **Customer Support**
- **Managed Life Cycle**

**Critical Service Support Functions**

- **Tools for Development Support**
- **Manage Core Software (OS, Safety Layers, Packages)**
- **Managed Computing Boards**
- **Expandable Safe Execution**

**Customer Support**

- **Managed Life Cycle**
- **Customer Support**

**Tools for Development Support**

- **Manage Core Software (OS, Safety Layers, Packages)**
- **Managed Computing Boards**
- **Critical Service Support Functions**

**Expandable Safe Execution**

- **Customer Support**
- **Managed Life Cycle**
- **Expandable Safe Execution**
- **Critical Service Support Functions**

**Post Support Tools**

- **Deliverable Tools**
- **Customer Support**
- **Managed Life Cycle**
- **Critical Service Support Functions**
- **Expandable Safe Execution**
TAS Platform is Based on Linux

In addition to safety layer and functional services (communication)

Integrity of SIL4 is essential!

Supervision of timing

Use existing COTS security packages of Linux possible

Layered safety approach allows integration of security and implement safety functions
Example: TAS Platform in Used in Applications

Interlocking

Onboard System (ETCS)

Exemplary boards

© Thales
### IEC 62443 – An Applicable Security Standard

**Process is Key**

<table>
<thead>
<tr>
<th>ISA-99 / IEC 62443 covers requirements on processes / procedures as well as functional requirements</th>
</tr>
</thead>
</table>

#### IEC 62443 / ISA-99

<table>
<thead>
<tr>
<th>General</th>
<th>Policies and procedures</th>
<th>System</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1 Terminology, concepts and models</td>
<td>2-1 Establishing an IACS security program</td>
<td>3-1 Security technologies for IACS</td>
<td>4-1 Product development requirements</td>
</tr>
<tr>
<td>1-2 Master glossary of terms and abbreviations</td>
<td>2-2 Operating an IACS security program</td>
<td>3-2 Security assurance levels for zones and conduits</td>
<td>4-2 Technical security requirements for IACS products</td>
</tr>
<tr>
<td>1-3 System security compliance metrics</td>
<td>2-3 Patch management in the IACS environment</td>
<td>3-3 System security requirements and security assurance levels</td>
<td></td>
</tr>
<tr>
<td>Definitions Metrics</td>
<td>2-4 Certification of IACS supplier security policies and practices</td>
<td>Requirements to a secure system</td>
<td>Requirements to secure system components</td>
</tr>
</tbody>
</table>

Requirements to the security organization and processes of the plant owner and suppliers
Typical Security Management – Patch Management

Removal of zero-day vulnerabilities following standards: IEC 62443 2-3 for Patch Mgmt

Separate safety and security life-cycles

- Using suitable architectures and processes or physical separation of security and safety functions

NOTE 3 Sometimes it can be necessary to balance between measures against systematic errors and measures against security threats. An example is the need for fast security updates of SW arising from security threats, whereas if such SW is safety related, it needs to be thoroughly developed, tested, validated and approved before any update.

Comment in draft norm (prEN50129: 2016)
TAS Platform Safe Security Approach

Virtualization for security and safety life cycle decoupling

- Integration of Safety and Security

Legend:
KVM ... Kernel-based Virtual Machine
Automotive
Automated / Autonomous Driving

Operational requirements (mission/safety):

- Avionics: safety few hours; operational few hours
- Railway: 24/7 trackside; few hours onboard
- Space: mission and safety: days to months
- Autonomous car: mission time: 1-2 hours?; safety: 1 minute continued operation?
- Automated driving with infrastructure: 24/7?

What does this mean for assurance and temporal supervision/guarantees?

What about integrated compute platforms?
Summary & Conclusions
Diagnosis info and operational management approach key to current and future IoT lead to connectivity needs and potential vulnerabilities

- Affecting safety-critical systems (due to security vulnerability)
- Different workloads and criticalities coexist

Updates will be the norm: Updates for security purposes (removal of zero-day vulnerabilities)

Application-level fault tolerance aspects often driving factor e.g. image processing: degree of correctness

- With learned behavior improvements for safety reasons safety update process changes
- SOTIF (Safety Of Intended Functionality)
  - NEW: updates to improve safety!!
- Leads possibly to “joint goal” of frequent updates due to safety and security improvements
Re-Cap & Future (2)

Safety goal: can be diverse for different criticality
- Real-time guarantees or guaranteed supervision
- Guaranteeing availability will be tough research questions e.g. with correctness of design (integrity is much easier)

Hard challenges:
- Balance between guarantees and performance
- Additional services required from computing platform (complexity)
- Virtualization: Hard challenge is guarantee of safety on top of virtualization (w/o hardware knowledge)
- Long-term guarantees of dependability: 10 to 15 years or more
- Automated safety approaches (automated verification and validation approaches)