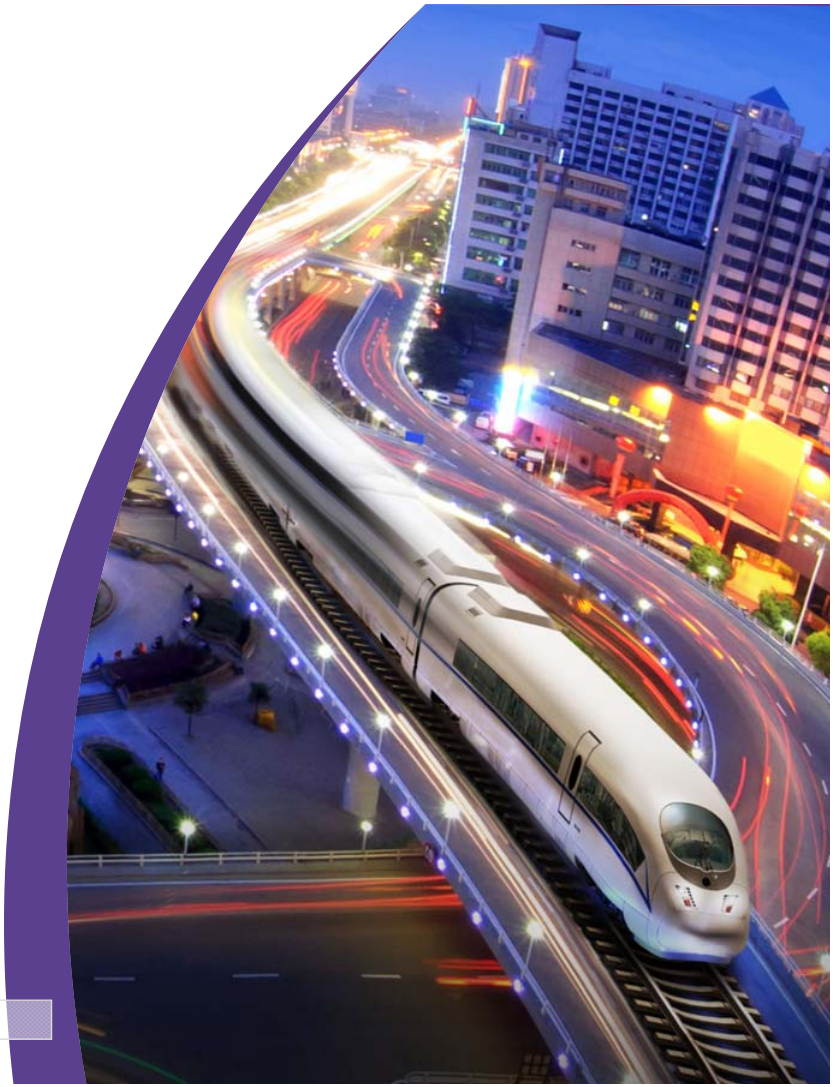
THALES

Mixed-Criticality Systems – a Journey « Embedded » in Space and Time

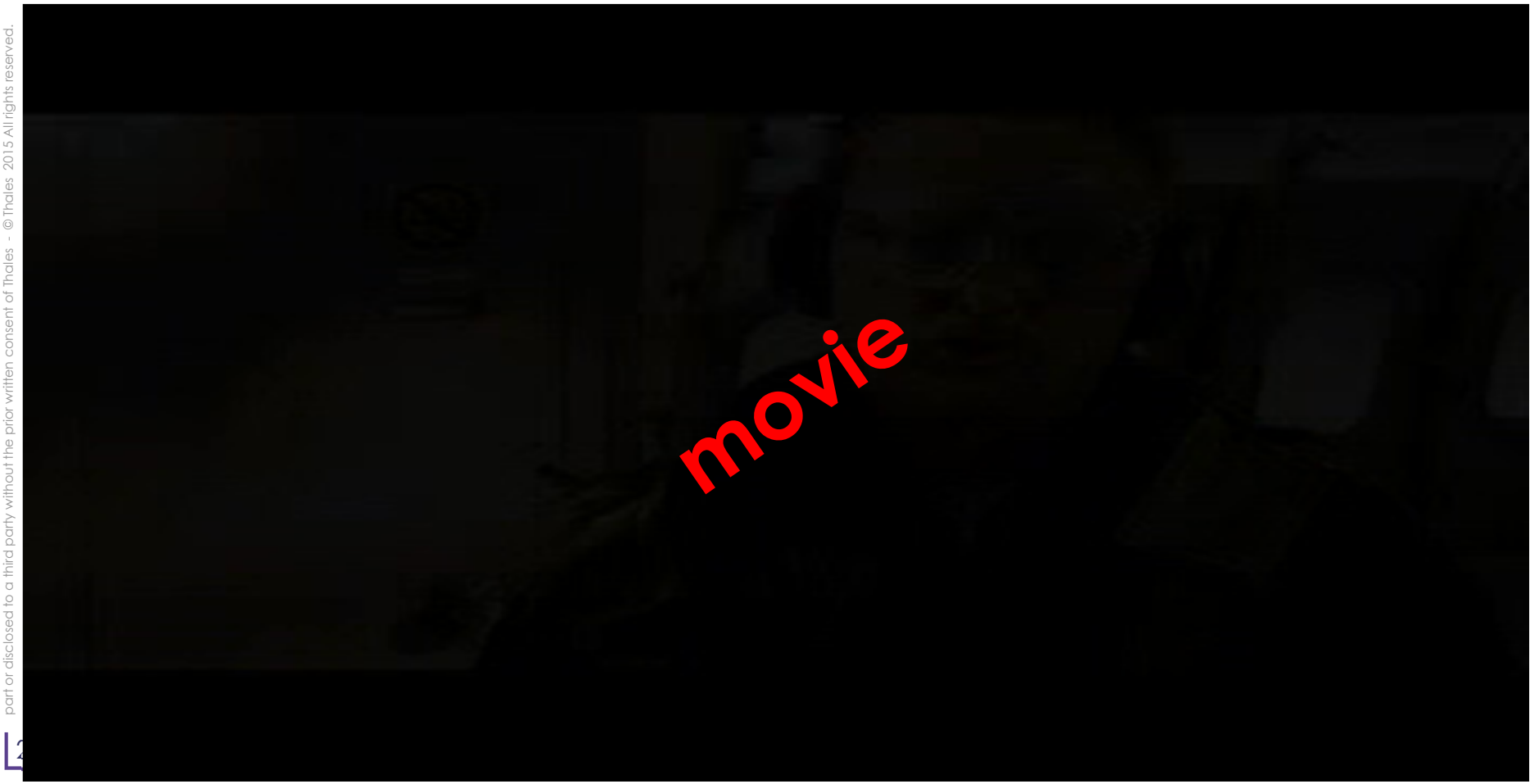
MICHAEL PAULITSCH

ECRTS KEYNOTE TALK
JULY 2015
LUND, SWEDEN



SUPPORTED BY EMC2 GRANT AGREEMENT NO. 621429 AND BY THE AUSTRIAN RESEARCH PROMOTION AGENCY (FFG) PROJECT NO. 842568

Working in Aerospace




movie

THALES

Trends

www.thalesgroup.com

OPEN

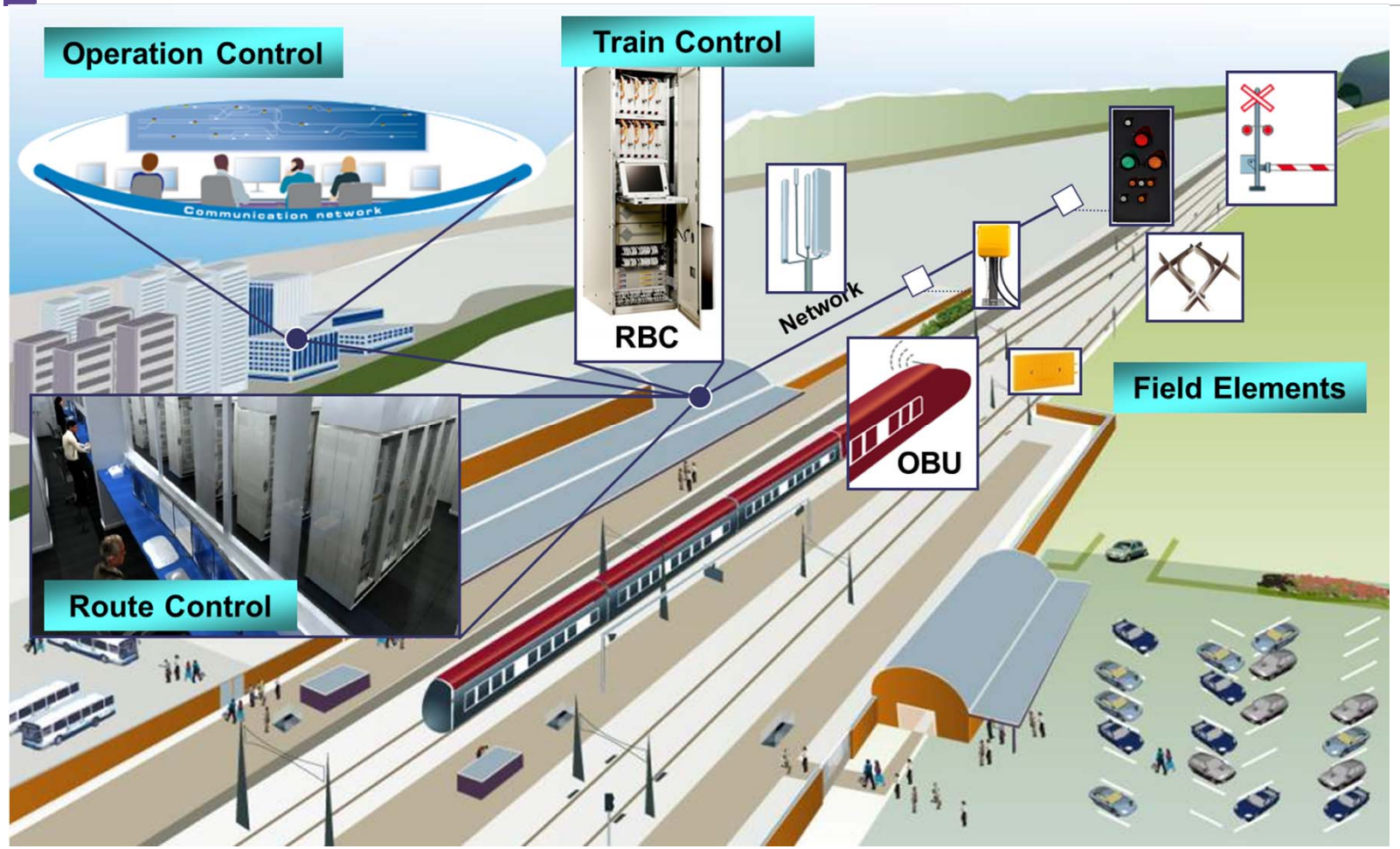
Trends in Aerospace

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

Trends in Railway – Signal Control

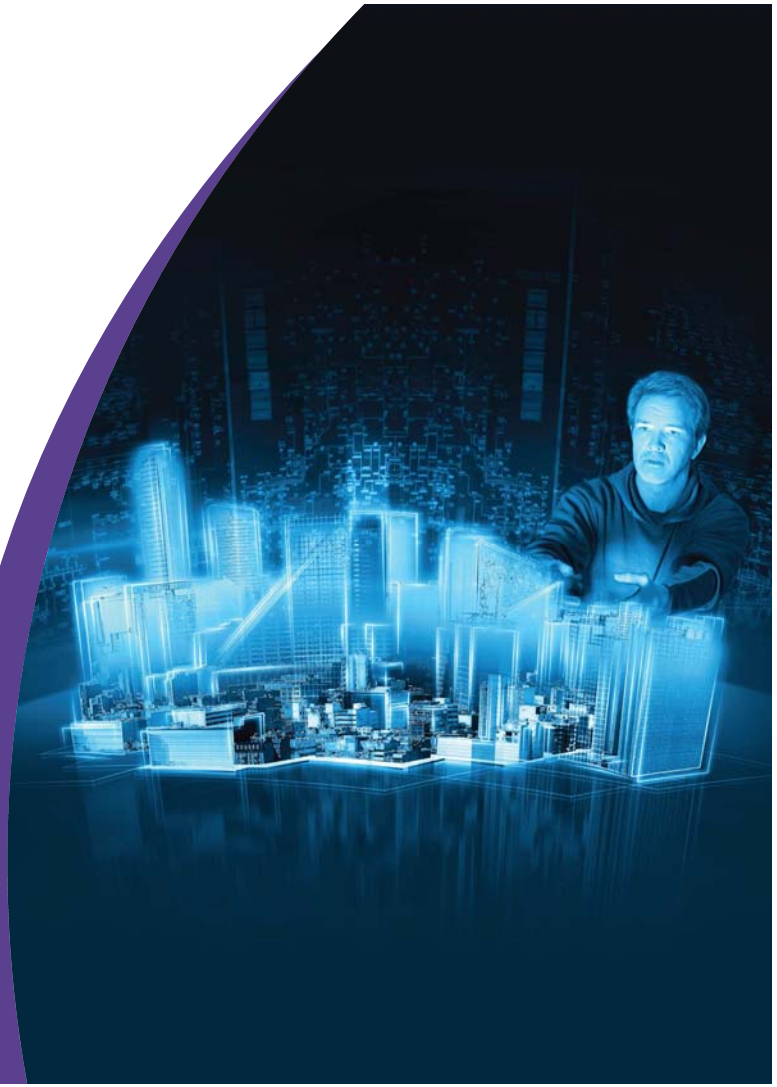


- ETCRS – removal of signals
- Remote moving authority
- Central operation centers

THALES

Definitions

Partitioning, Mixed-Criticality, ….



Mixed-Criticality System in Industry – What's it?

Multiple safety criticalities (residing) on same platform

- Key requirement for platform: Platform needs to fulfill safety requirements at minimum of highest safety requirement of application.
- Criticalities are assigned by safety process and don't change
- Chosen independence between applications to minimize interaction between otherwise independent "safety chapters" (system level safety analysis extremely complicated w/o this requirement).

What it is NOT

- A system where system approach sacrifices lower criticality applications for whatever purpose (directional partitioning property)

"Real-life" aspects:

- Deployed for many years (B777, B787, A380, A350, E170/175, E190/195, …) under the name Integrated Modular Avionic (IMA) systems
- Wish to deploy modern computing platforms like multicore or even

Mixed-Criticality in Academic Literature (especially Scheduling)

Originated in 2007 with Vestal paper

- Higher criticality applications have "higher " priority in case of "issues"
- Requires adaptation of safety process (mentioned in 2007 paper)
- "The thinking behind this multi-criticality bounds on execution time is that the software and hardware are fixed, all that was varying are the verification methods" St. Vestal (2015)
- Programming approaches for higher criticality are different
  - Simpler control flow; different verification and validation techniques
  - Even safety margin can be smaller for higher criticality code
  - Example: High MC/DC (Modified Code / Decision Coverage) coverage for high criticalities requires high

From mainstream industrial viewpoint: Research focuses more on "performability" and not (safety) criticality

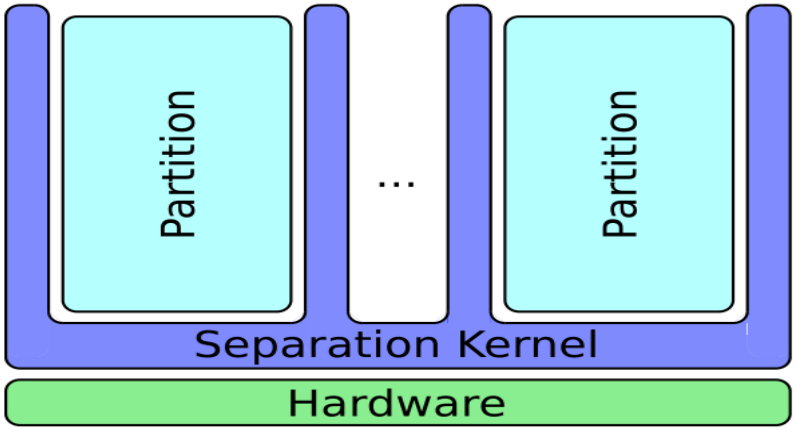
Some techniques may, however, be applicable to systems where safety requires only attribute integrity and "less" availability

Research on flexibility of scheduling however is generally valuable

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

- Partition/process: independent segregated environment
- Separation kernel / Memory Management Unit: control instance
- Temporal partitioning does not need to be implemented in time slots ! E.g. P. Binns with slack scheduling on Primus EPIC

THALES

Application Needs – Fail-Safe Versus Fail-Operational

Partitioning for safety does not always require safety attribute availability

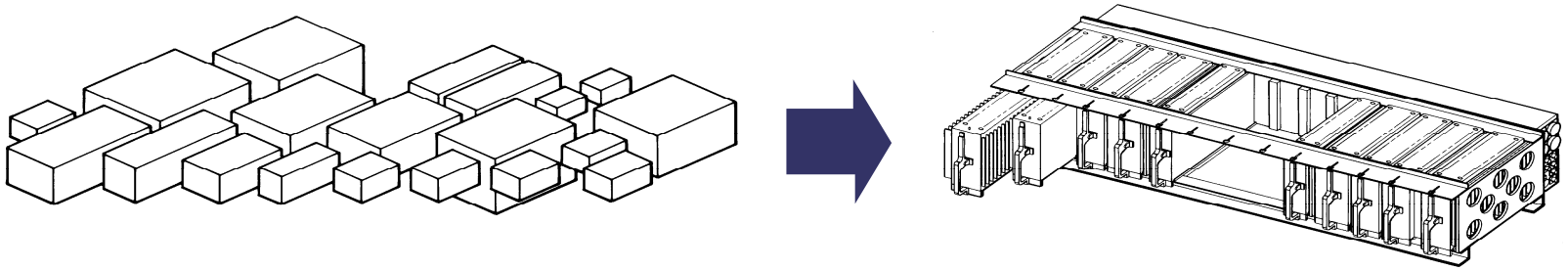
- Fail-operational: Attribute integrity and availability
  - Aerospace: plane cannot "stop" in air
- Fail-safe Attribute integrity
  - Railway signaling: train can stop
  - Minimum level of availability is required to avoid or minimize manual operation
- Be careful: security aspects

THALES

Examples of Real Mixed-Criticality Systems

Boeing 777 – First Airplane with Integrated Modular Avionics (IMA)

Boeing 777 Integration



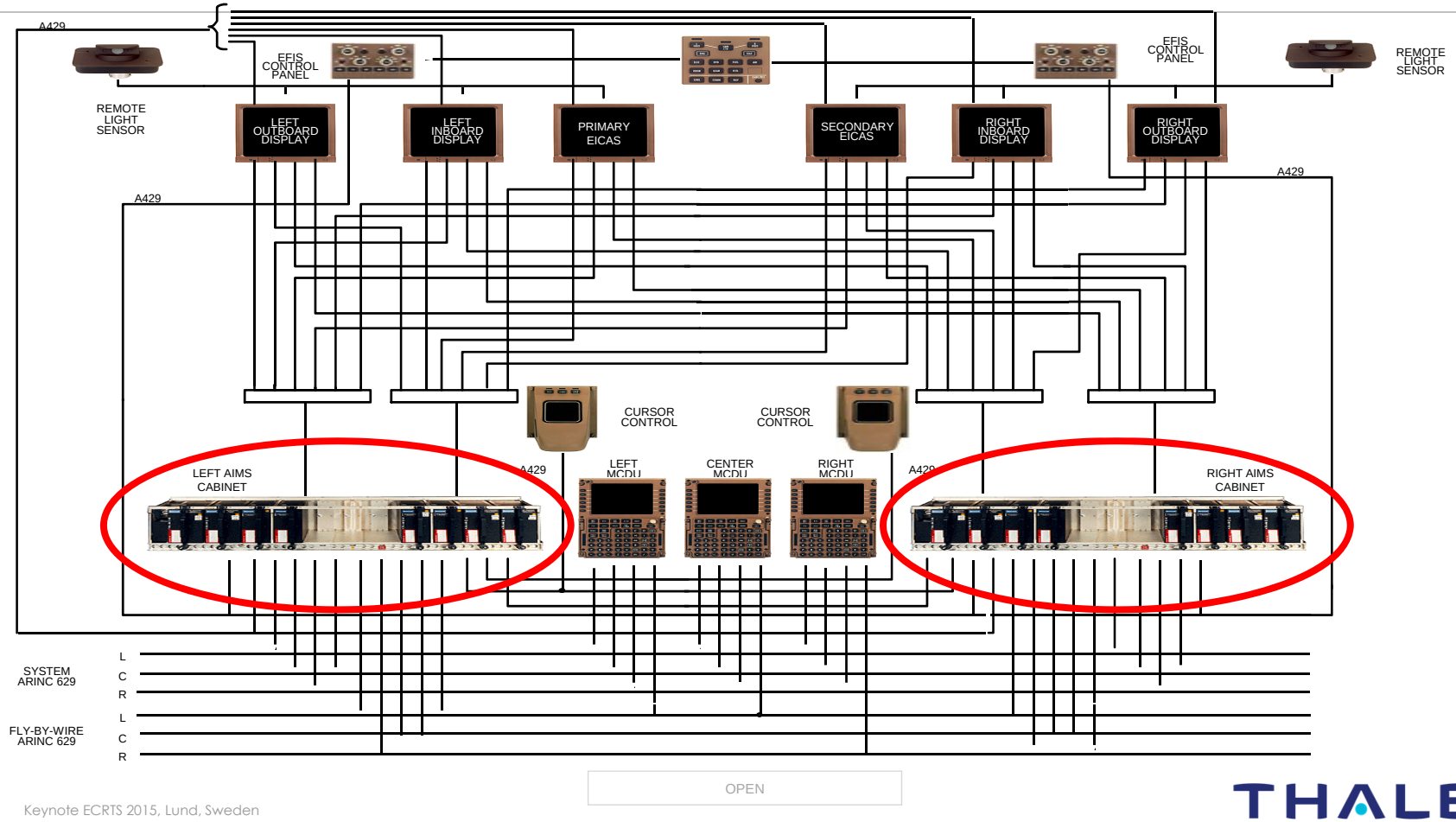
Dual integrated cabinets provide all processing and I/O resources for:

- Displays (incl. Graphics Generation)
- Flight Management (incl. Autothrottle)
- Central Maintenance
- Communication Management (incl. Flight Deck Communication)
- Airplane Condition Monitoring
- Flight Data Recorder
- Data Conversion Gateway
- Quick Access Recorder

THALES

Cockpit Boeing 777

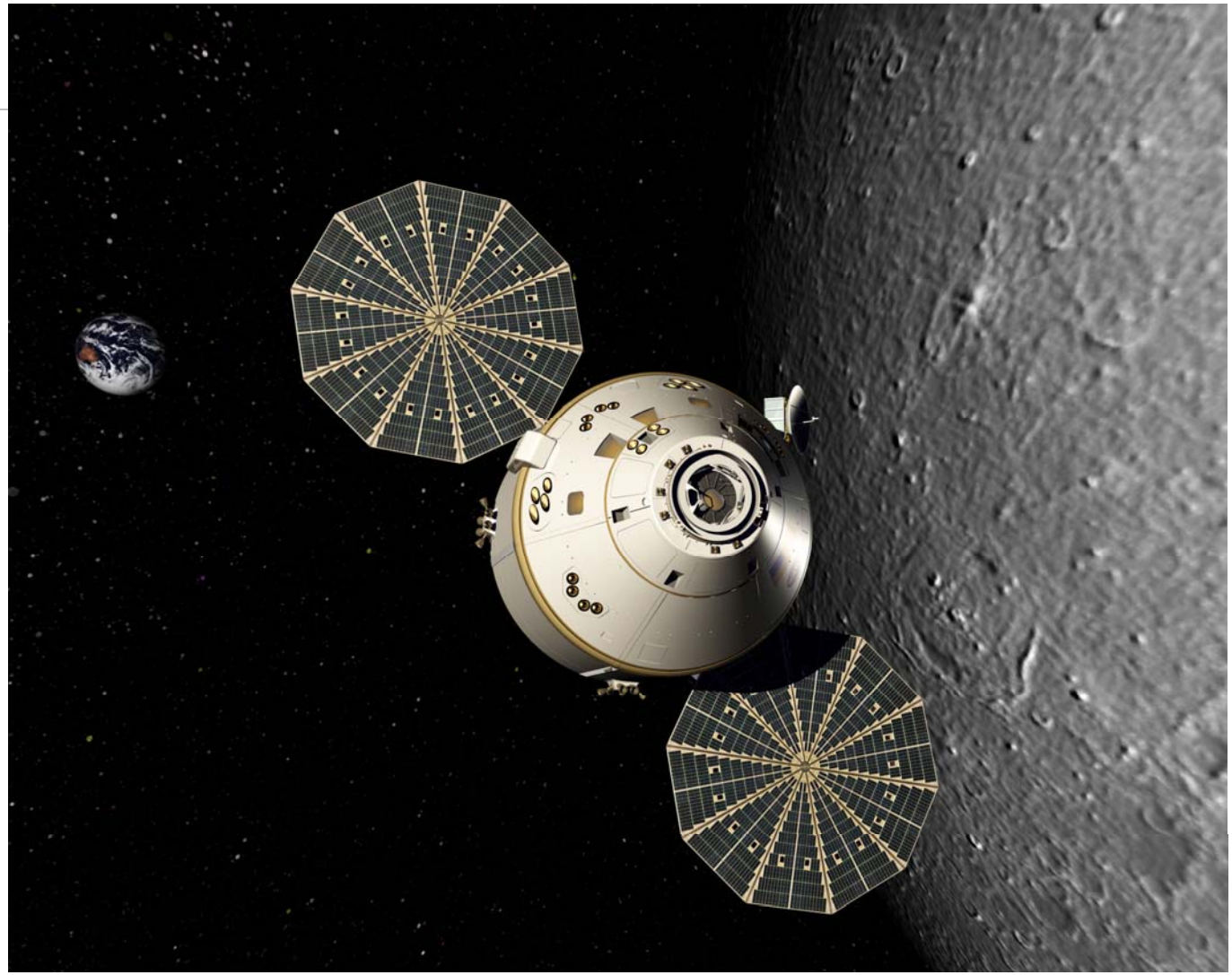
Boeing 777 Avionics Architecture



THALES

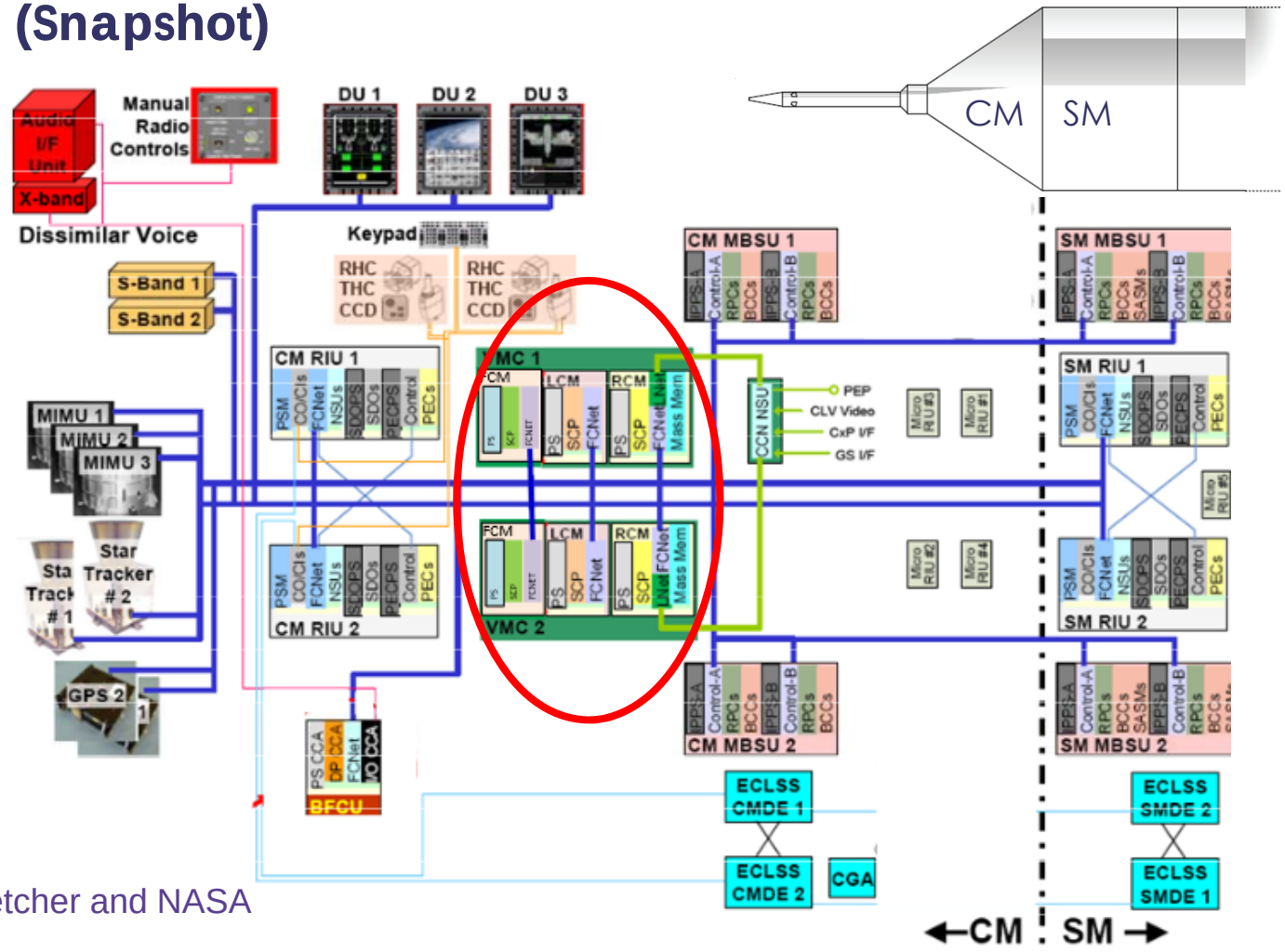
Orion

Next generation U.S. spacecraft



Inside View – Cockpit Orion

Avionics (Snapshot)



THALES

Safety Process

www.thalesgroup.com

Aerospace – A Long Tradition of Safety Civil Certification Standards

Part 21: Certification of Aircraft & Related Products, Parts & Appliances
CS 25: Certification Specifications for Large Aeroplanes
CS 25.1309: Equipment, Systems & Installations
AMC 25.1309: System Design & Analysis

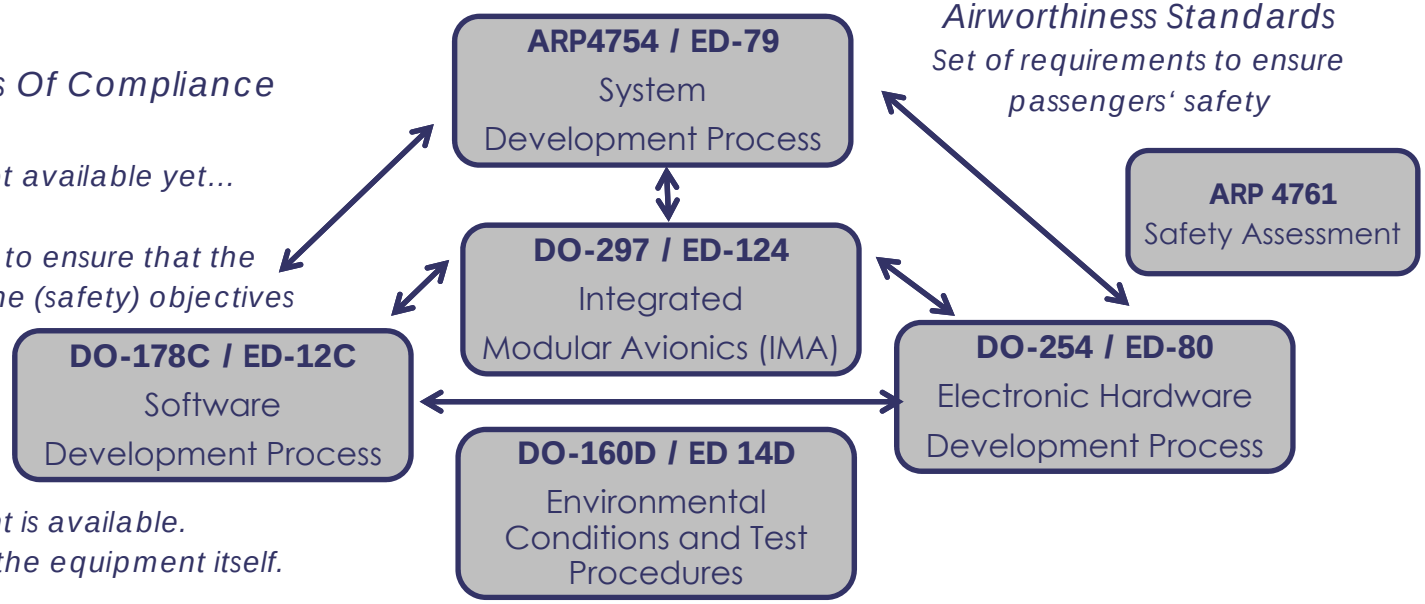
Legislation

Airworthiness Standards
Set of requirements to ensure passengers' safety

Acceptable Means Of Compliance

The equipment is not available yet…

Structured approach to ensure that the equipment WILL meet the (safety) objectives



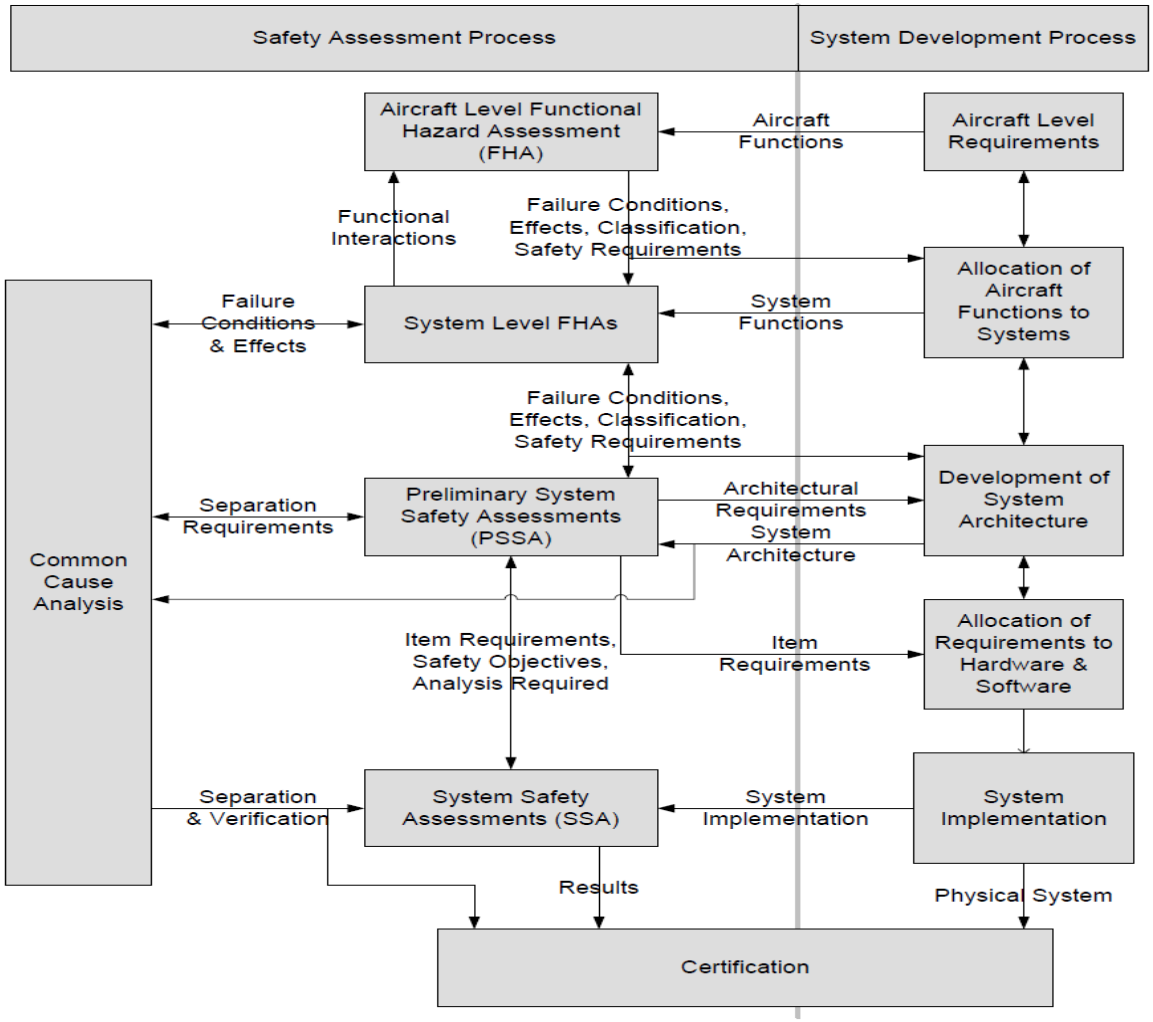
The equipment is available.
Tests are applied on the equipment itself.

OPEN

THALES

Safety in Aerospace – System Development

Example ARP4754: System Development Process with strong safety focus



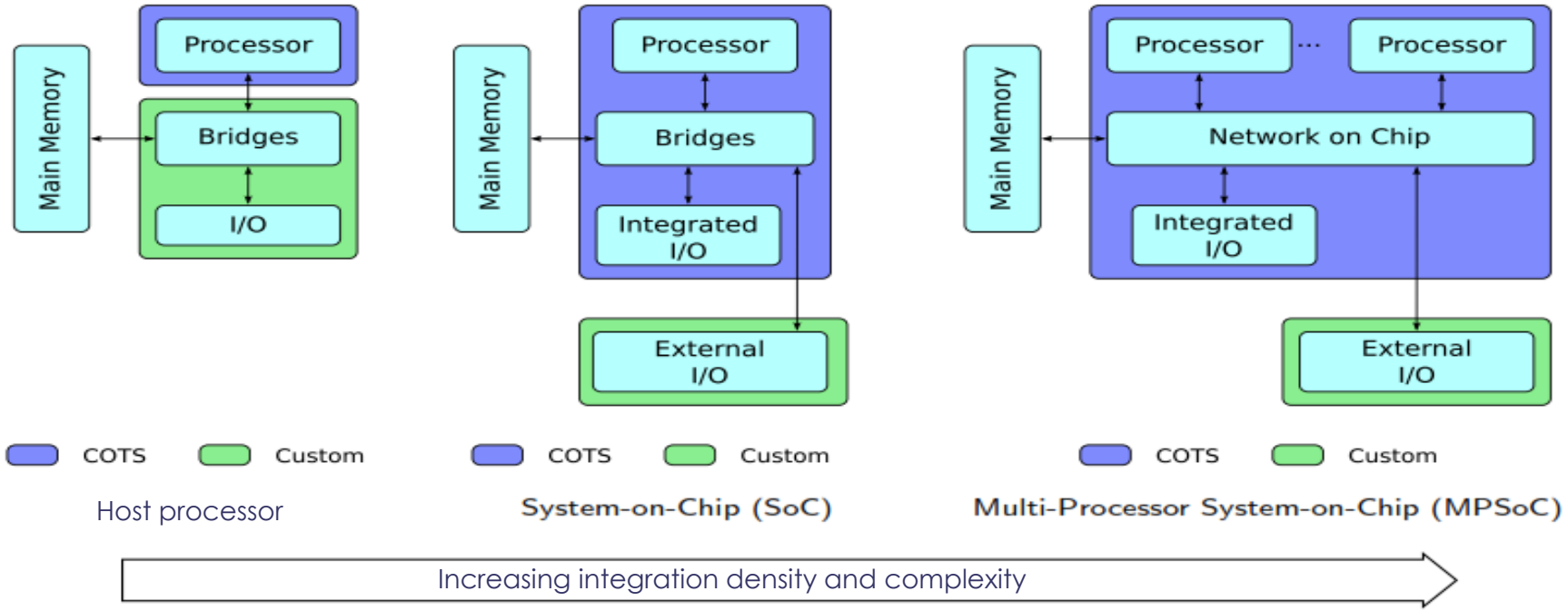
THALES

Multi-core

www.thalesgroup.com

OPEN

Chip Evolution



THALES

View of Aerospace Multi-Core Certification Body Related to Timing

- **Only selective view of publicly available FAA CAST paper 32**
- **(Functional) interference channels of multicore 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, …

Multicore: 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

- O. Kotaba, J. Nowotsch, M. Paulitsch, S. Petters, H. Theiling. Multicore In Real-Time Systems - Temporal Isolation Challenges Due To Shared Resources. WICERT workshop as part of DATE 2013.
- D. Dasari, B. Akesson, V. Nelis, M.A. Awan, S.M. Petters. Identifying the Sources of Unpredictability in COTS-based Multicore Systems. SIES conf. 2013.

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

Assessment of Multi-Core Worst-Case Execution Behavior Overview

Motivation:

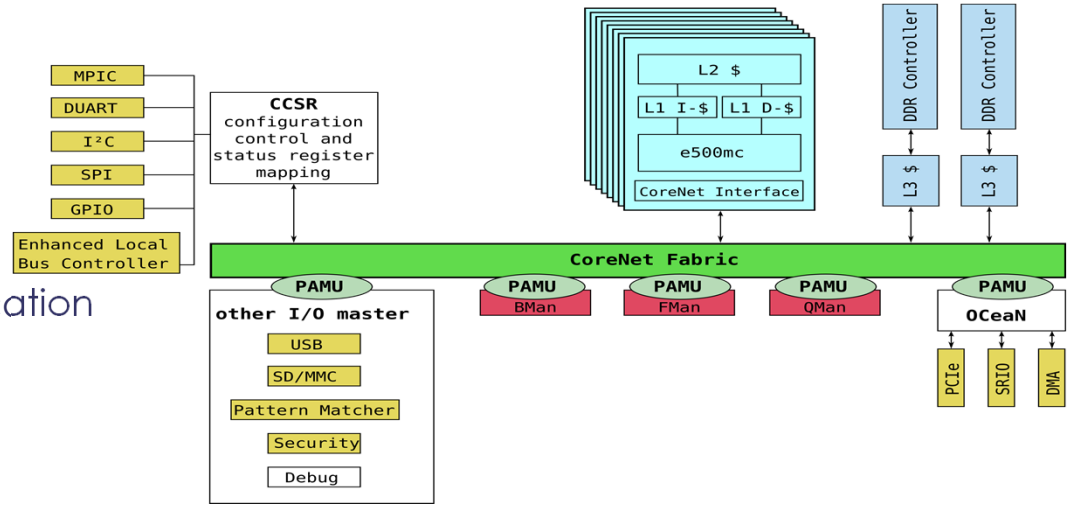
- Integration leads to common use of shared resources. Partitioning impact needs to be evaluated for safety-critical applications, such as IMA

Goal:

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

Approach

- memory-intensive tests



Freescale P4080

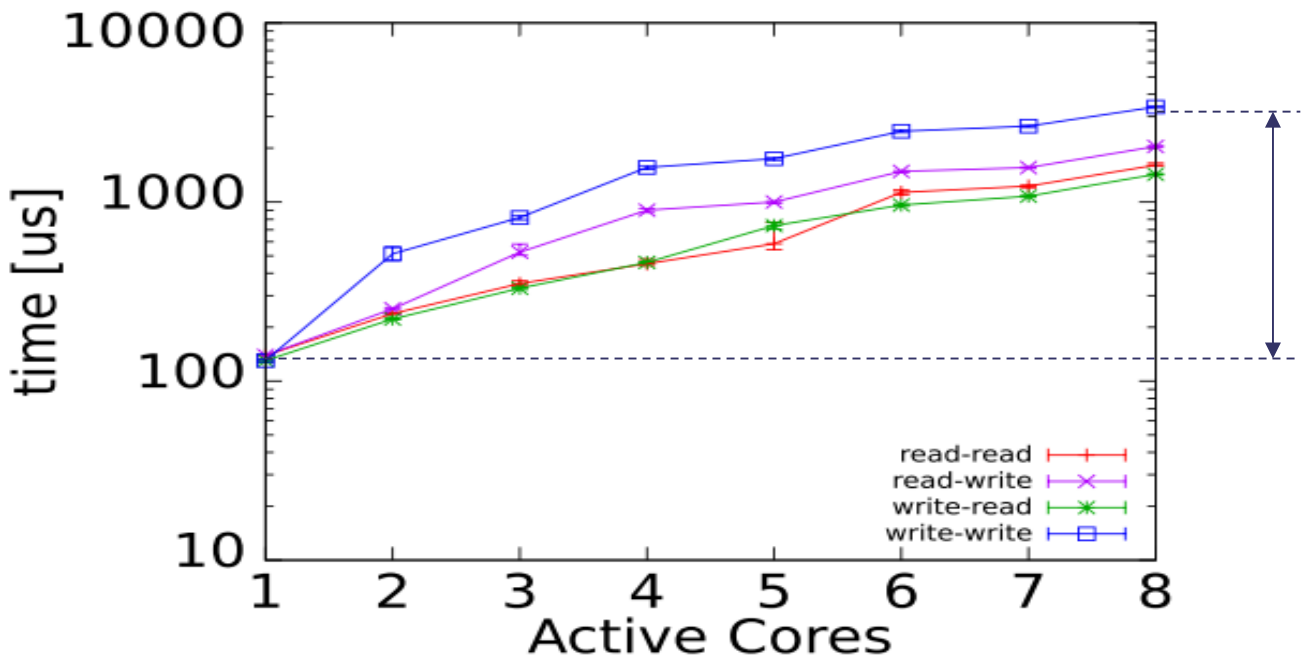
Focus of work:

- Network on Chip (not much data available); some memory access performance tests

  Details of work published at EDCC2012 (J. Nowotsch, M. Paulitsch)

THALES

Assessment of Multi-Core WCET
Memory (DDR) Accesses (8 Cores)



1/30 about 4 times slower overall (8*1/32)

Worst-case access time increases over-proportionally with more cores.

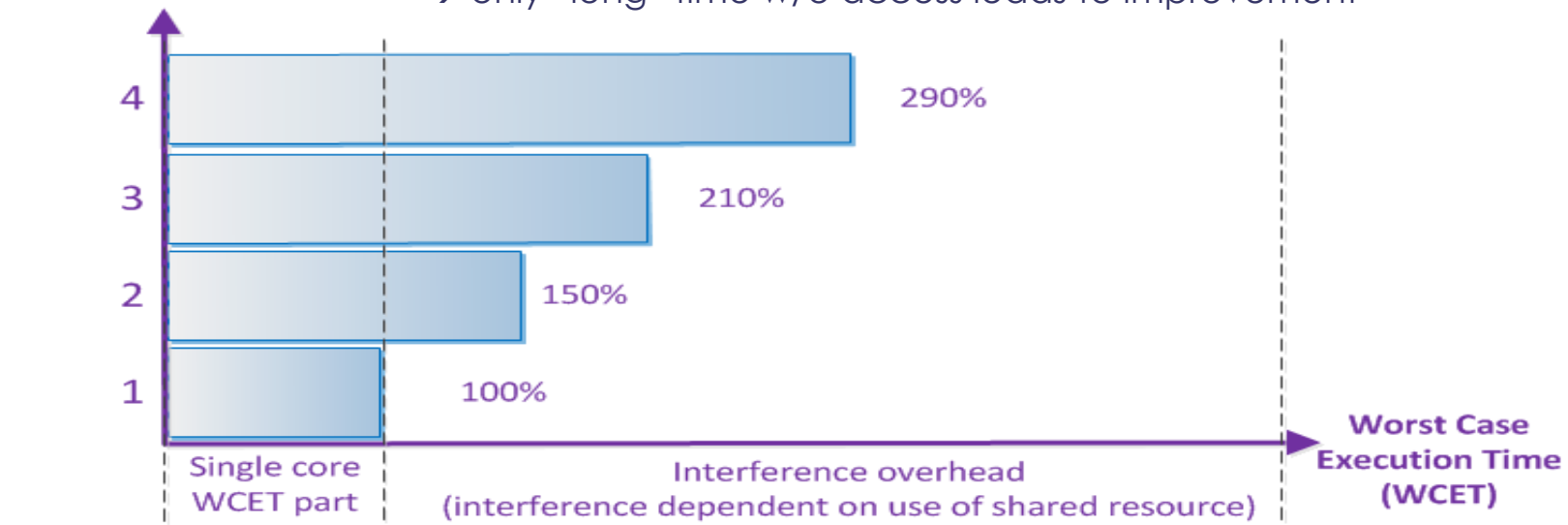
THALES

System Integration Problem Due to Multicore Processors
Interference Depends on Program

Worst-case execution time depends on use of shared resource (like common memory)

- Some idea about access times (Freescale P4080):
  - Access to main memory is 80-90 core cycles for one core
  - Access to main memory with 8 cores running in parallel is ca. 1000 cycles

→ only "long" time w/o access leads to improvement



OPEN

THALES

Some Measured Values for Freescale P4080 Interference Between Single-Core and 8-Core Systems

- **Worst case influence (for 8 core multicore system)**
- **Worst case observed versus worst case analysis → some conclusions can be drawn for average case (slack between average and worst case)**

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

>> 8 times greater

Difference greater for multicore (more "slack")

Context info: EEMBC benchmark; OET … Observed Execution Time; bound … analyzed using AbsInt AiT

THALES

WCET for Multi-Core Computers 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

Average-Case Extension

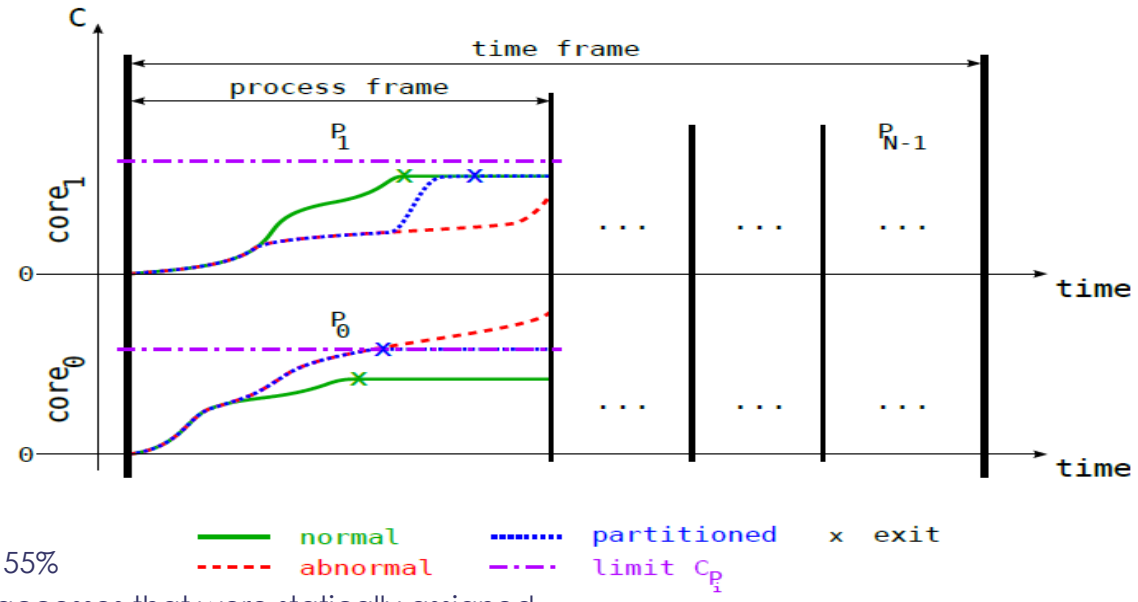
- Applied to bare-metal OS layer

Evaluation

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

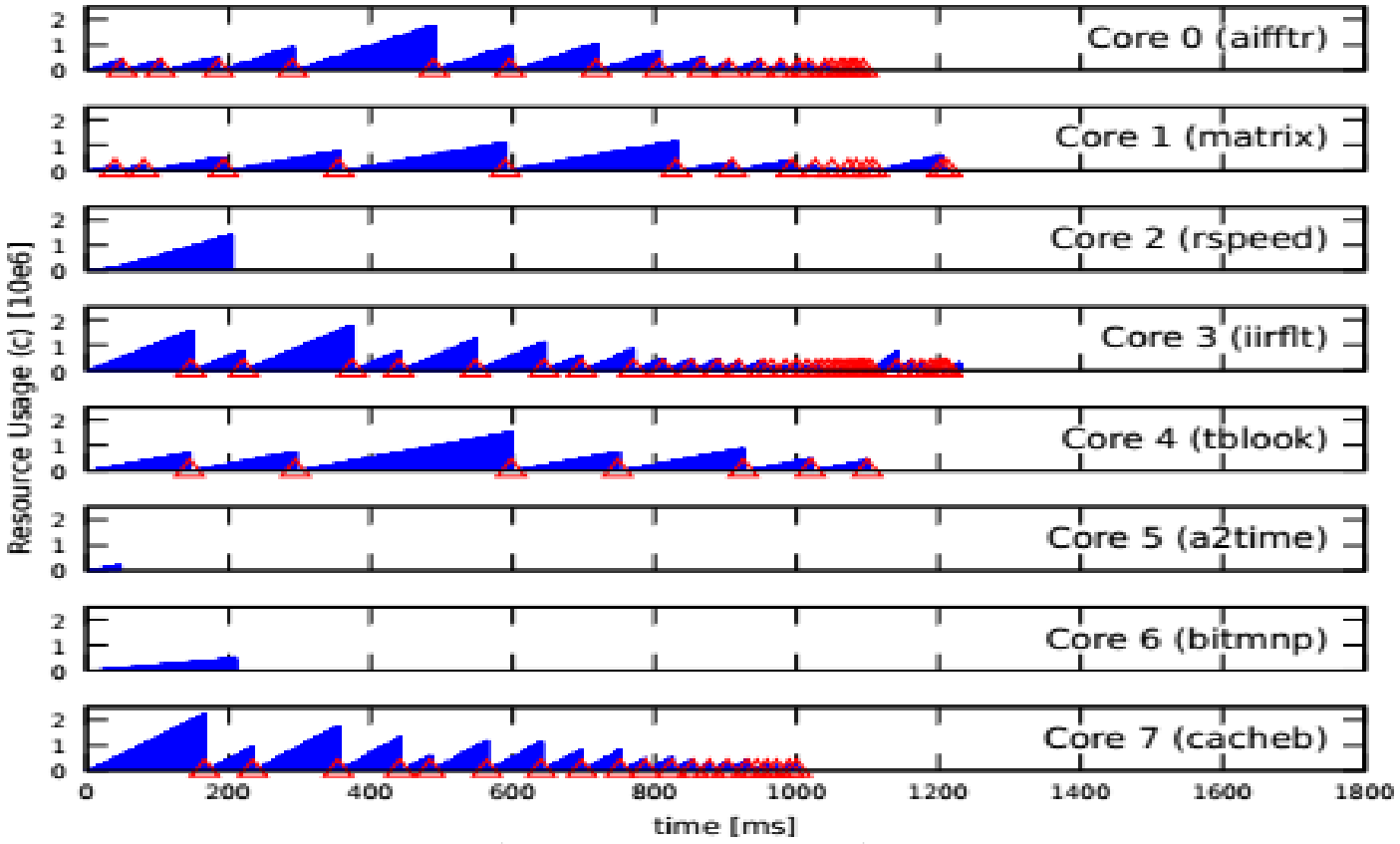
WCET reduction:

- Utilisation increase: core 98.9%, system 55%
- Additional accesses: 2 to 70 times the accesses that were statically assigned



THALES

Evaluation – Runtime Analysis

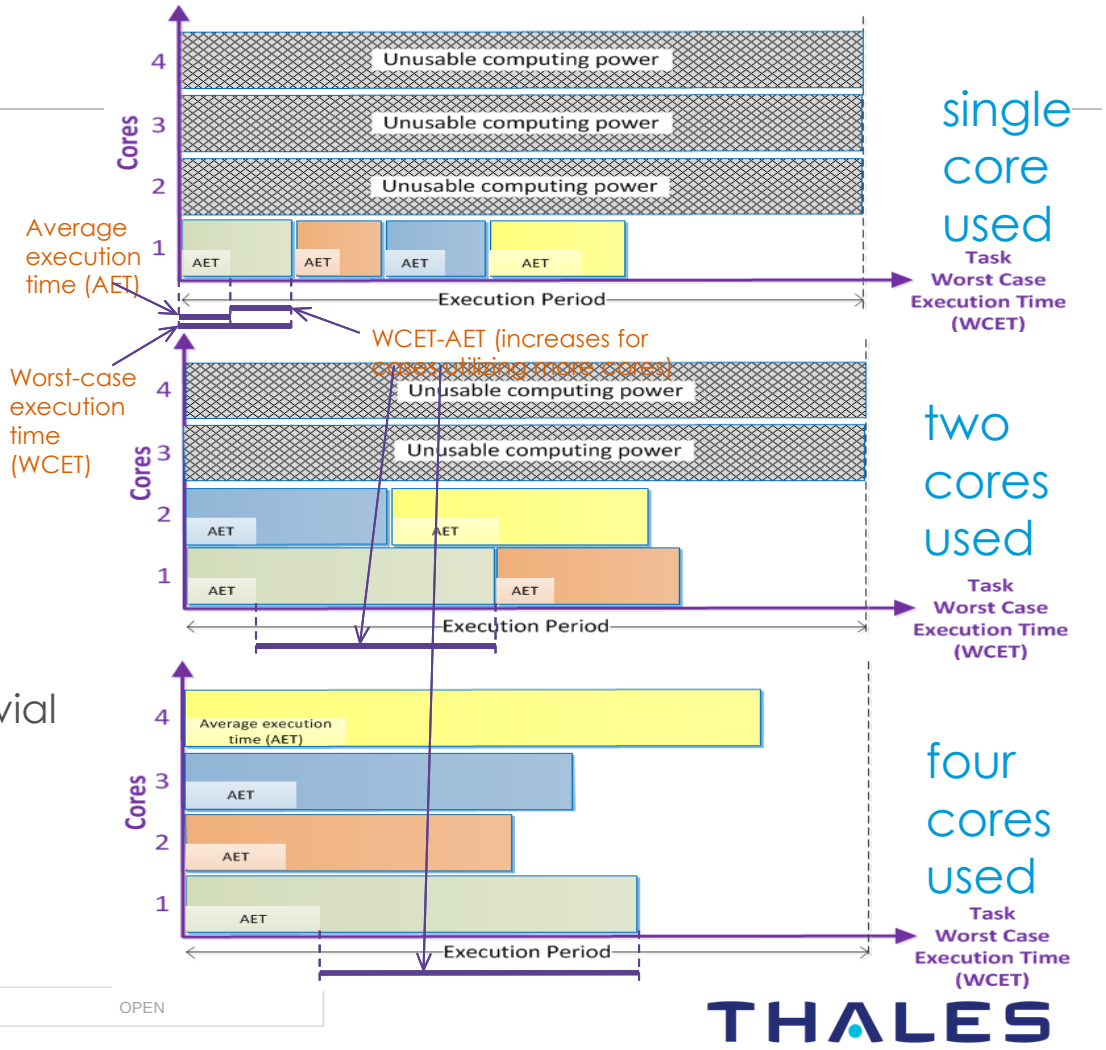


OPEN

THALES

Example Scheduling Approach (3 Variants)

- **Worst-case execution times of tasks very dependent on utilization of cores**
- **Difference between average and worst-case execution likely increases for cases with more cores used**
- **Conclusions:**
  - System optimal solution non-trivial once multiple cores are used
  - Harvesting unused execution time in average case at application level can be distinguishing factor



OPEN

THALES

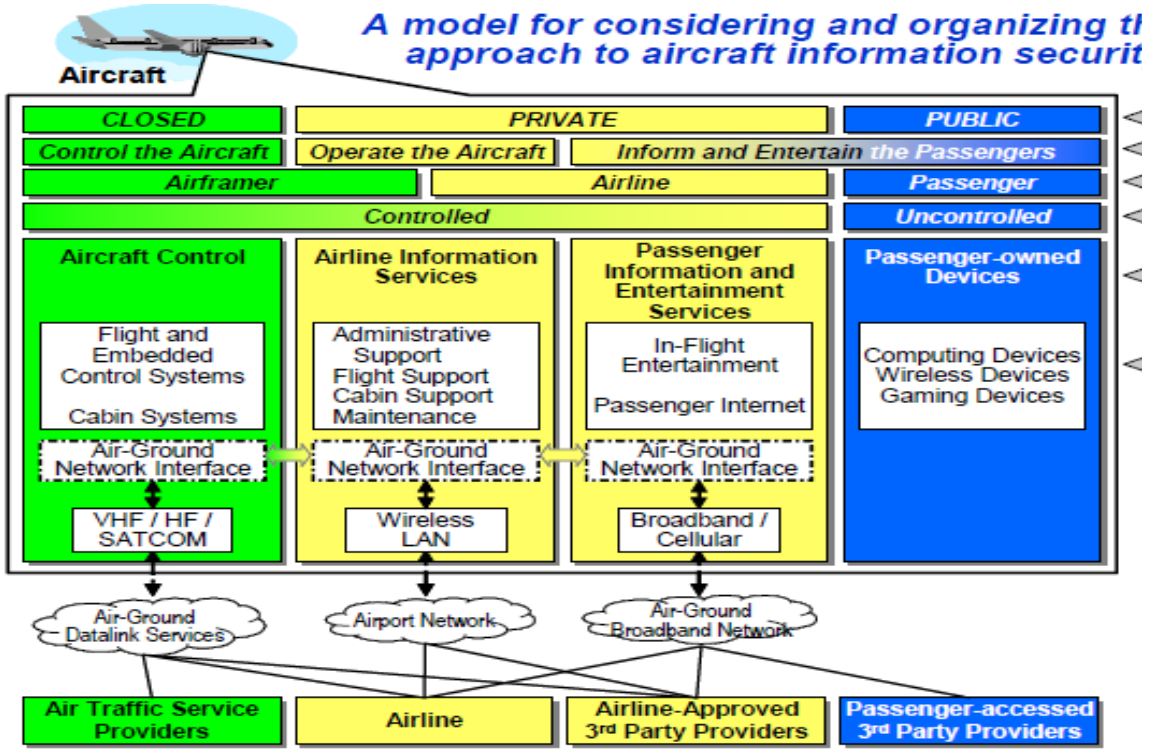
THALES

Security and Partitioning

Mixed Criticality Systems

www.thalesgroup.com

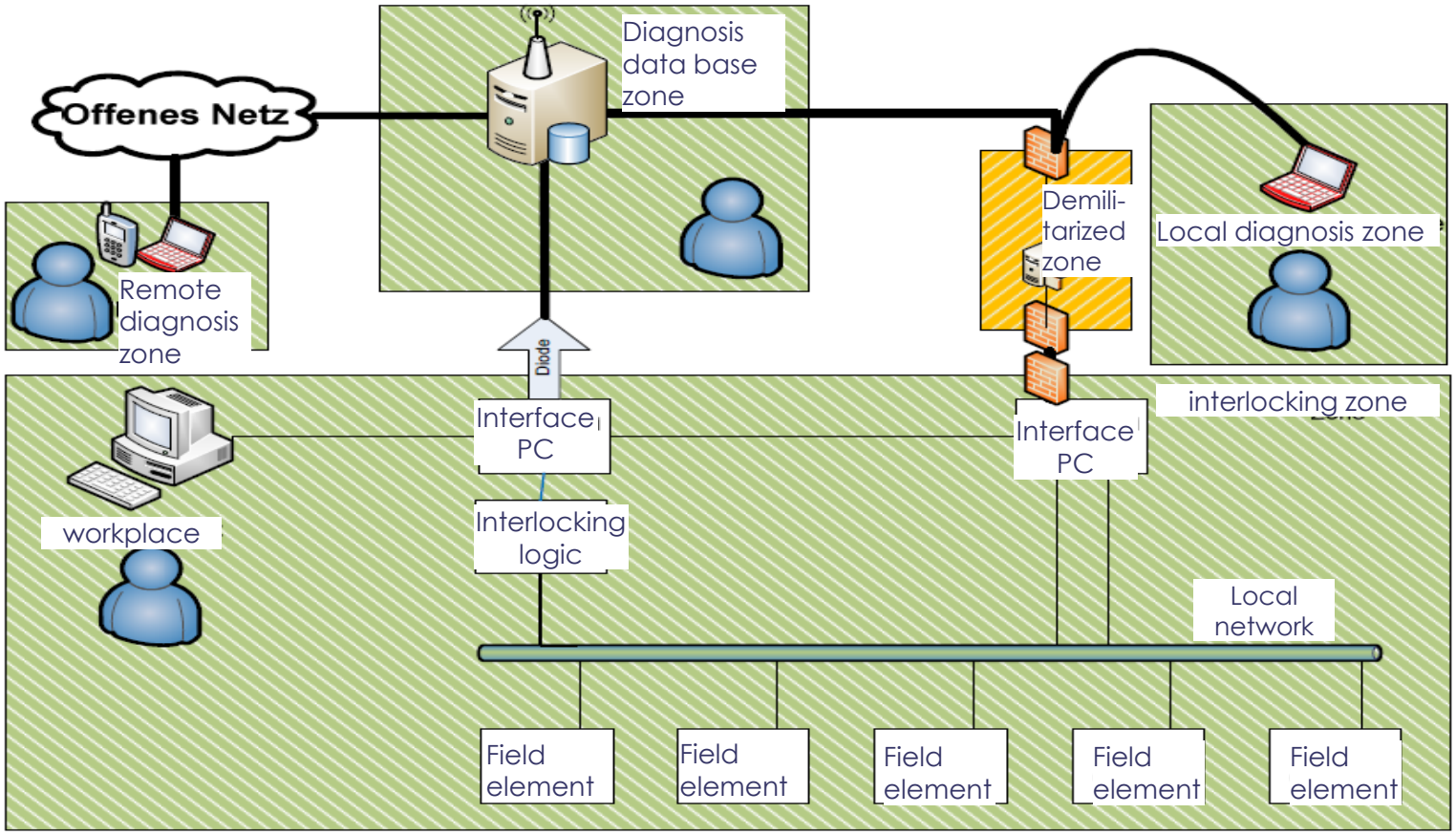
Why Extend Mixed-Criticality Systems to Security?



- Functional integration
- Passenger & Maintenance / Services Integration
- Modern A/C are Systems of Systems (ATM)

THALES

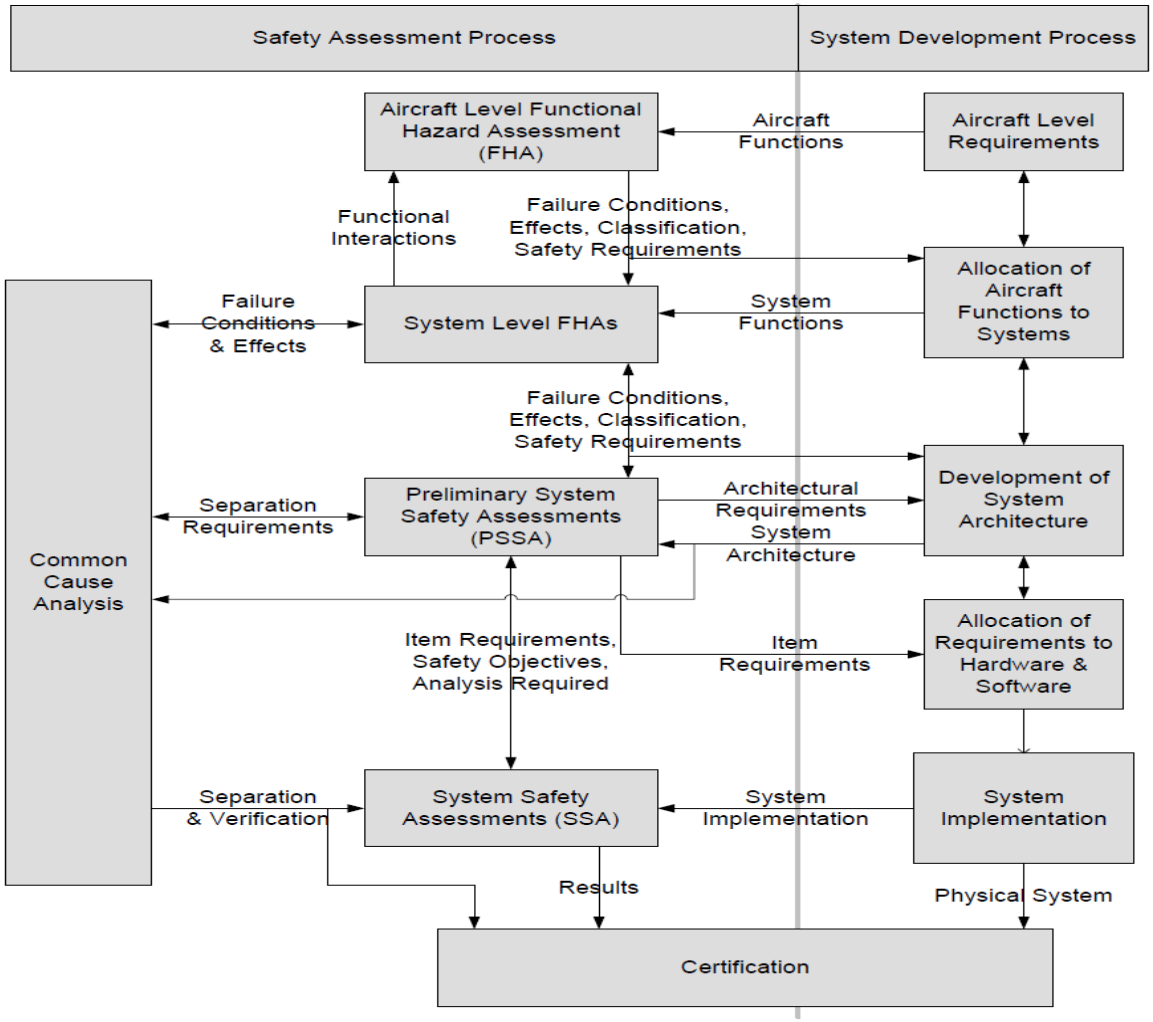
Security Zones – Example Architecture DIN VDE V 0831-104



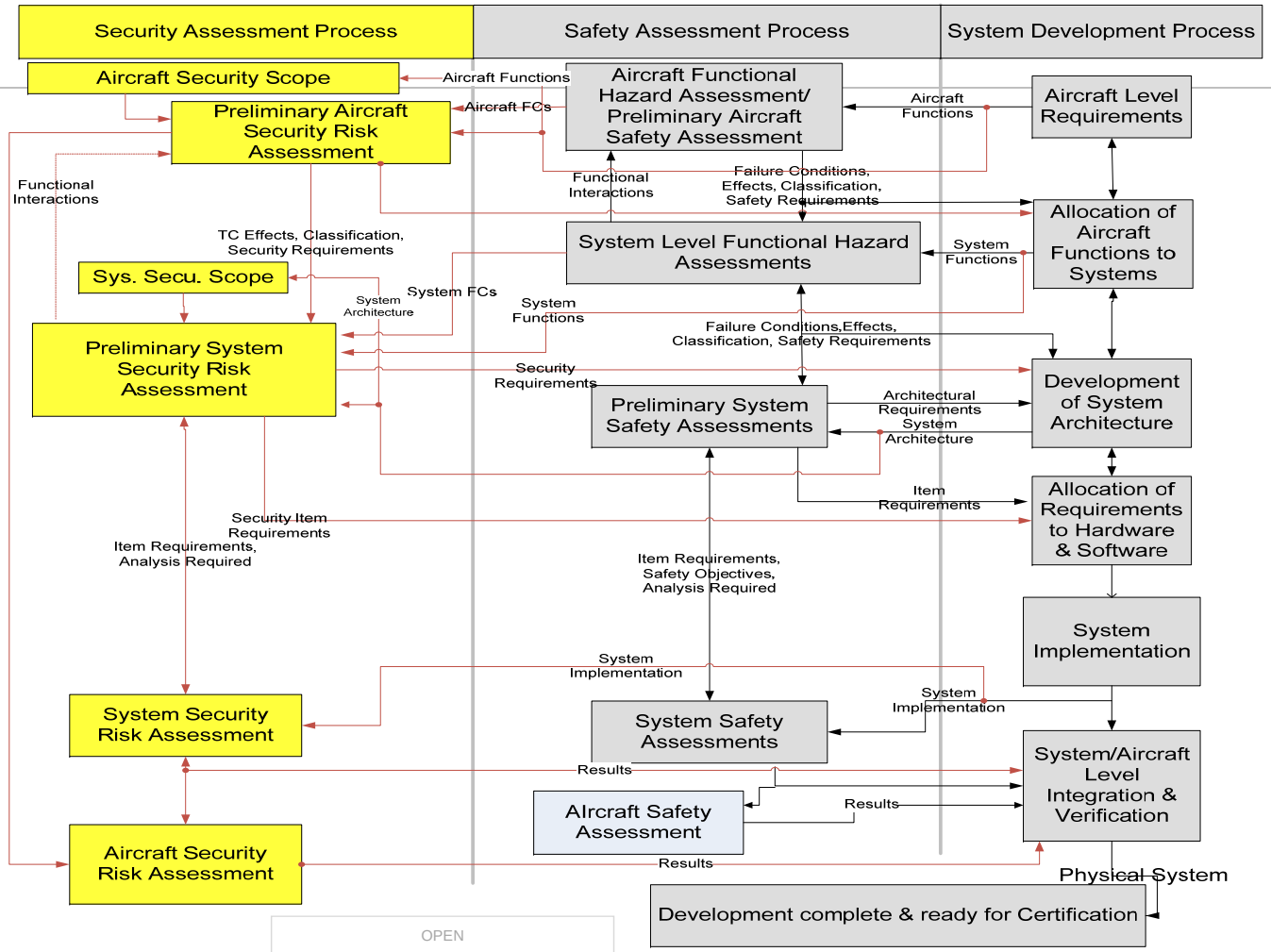
Pic adapted from German standard

Safety in Aerospace – System Development

Example ARP4754: System Development Process with strong safety focus



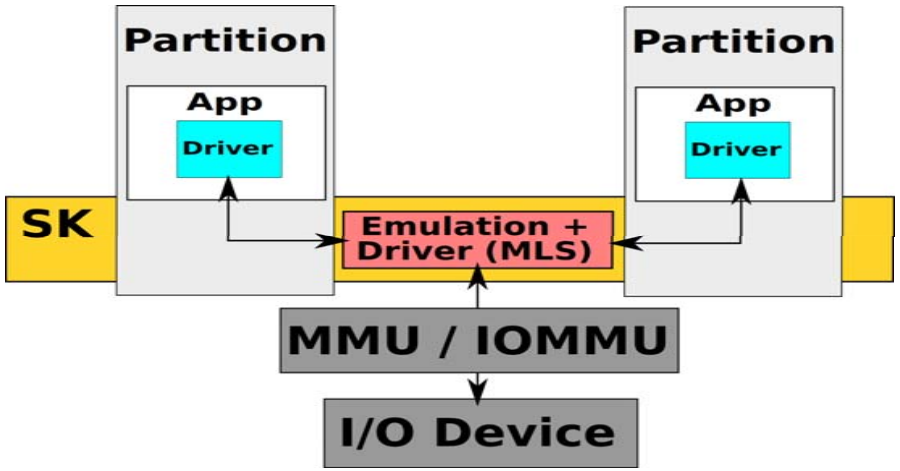
Integration of Security into Safety Process



MILS – Multiple Independent Levels of Security

- **Architecture for a (software) system processing data of different security domains concurrently → Combines applications of different trust within the same system**
- **High-assurance security architecture based on the concepts of separation and controlled information flow**
  - Separation builds on time partitioning and spatial partitioning (e.g. periodic processing, memory protection, **I/O separation**)
  - Controlled information flow: white-list based communication between separate partitions
- **Small analysable components; composability targeted**
- **Certifiable MILS systems are built out of key components (separation kernel, trusted hardware, guards, …)**
  - Have to be **N**on-bypassable, **E**valuable, **A**lways invoked, and **T**amperproof (NEAT)

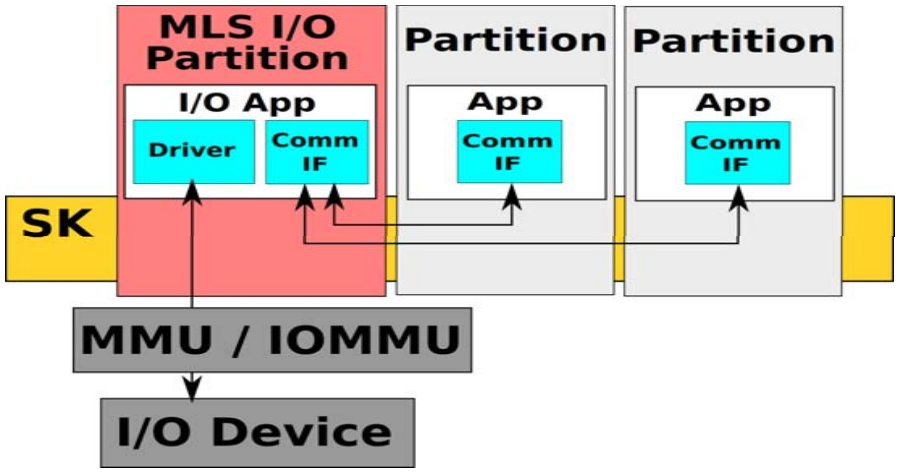
Overview I/O Sharing on Partitioned Systems (I)
→ Software-Based Sharing



- Driver and a possible emulator are located in runtime address space of the Separation Kernel
- Uses Separation Kernel (SK) functions to route data from/to partitions
- Hardware Protection Units support the Separation Kernel for access enforcement

IOMMU – I/O Memory Management Unit
I/O – Input/Output
MMU – Memory Management Unit
MLS – Multi-Level Security

Overview I/O Sharing on Partitioned Systems (II)
→ Central I/O Partition



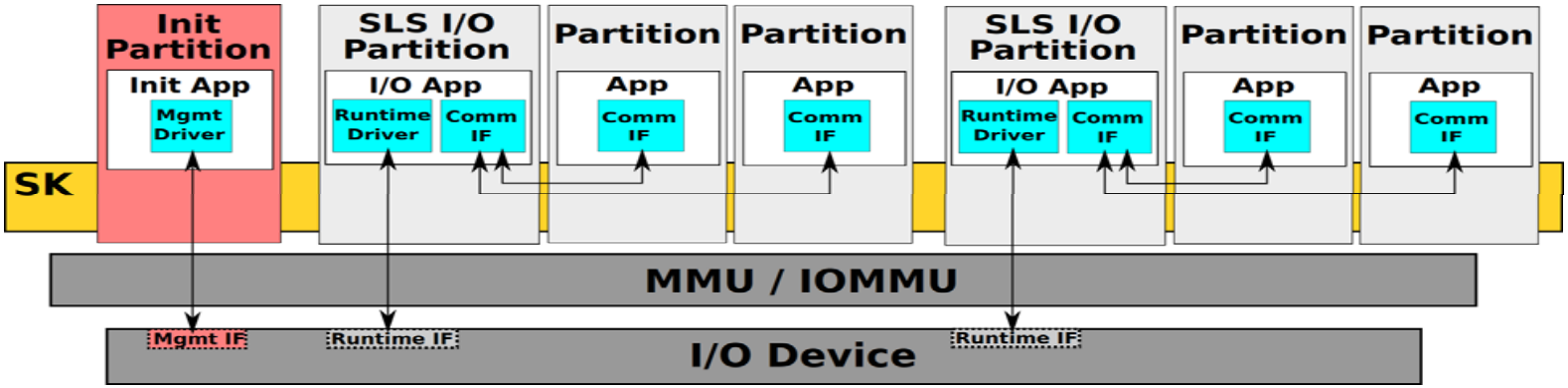
- I/O partition has dedicated access to the hardware
- Runs in an isolated address space
- Implements the driver
- Uses Separation Kernel functions to route data from/to partitions
- Data of different criticality / classification is routed via this partition

MMU – Memory Management Unit
MLS – Multi-Level Security
IOMMU – I/O Memory Management Unit
I/O – Input/Output

Overview I/O Sharing on Partitioned Systems (III)
→ Self-Virtualization of Devices

- Devices provide runtime interfaces to software
- Devices receive configuration via a trusted boot partition / boot driver
- Each level of classification get direct access to a virtual device interface
- I/O partition routes data and performs software steps
- Overall reduction of driver's and I/O partition's code complexity

→ Special Requirements on the hardware have to be fulfilled



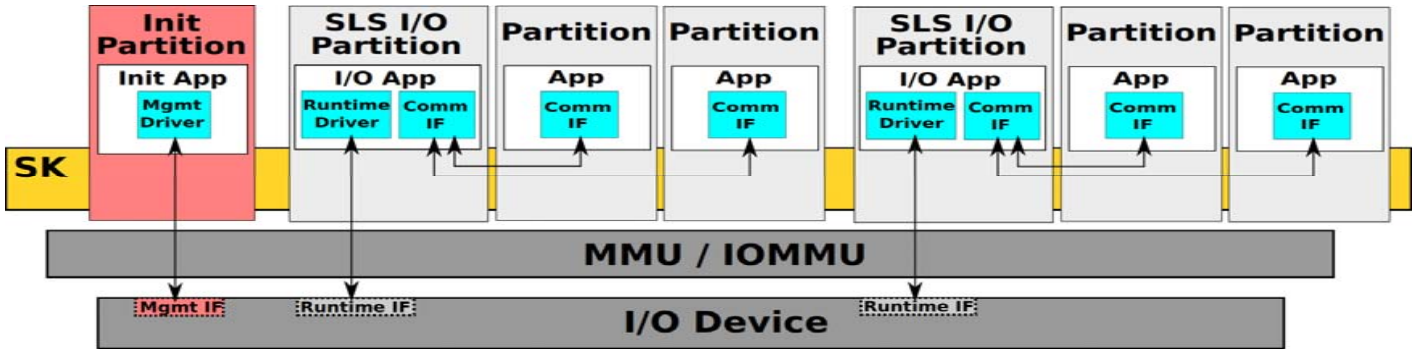
MLS – Multi Level Security SLS – Single Level Security

OPEN

THALES

Hardware Requirements for Self-Virtualizing Hardware

1. Internal Spatial Separation
2. Secure Direct Memory Access (DMA) Transfers
3. Internal Temporal Separation
4. Secure Interrupt Handling / Triggering
5. Secure Initialization

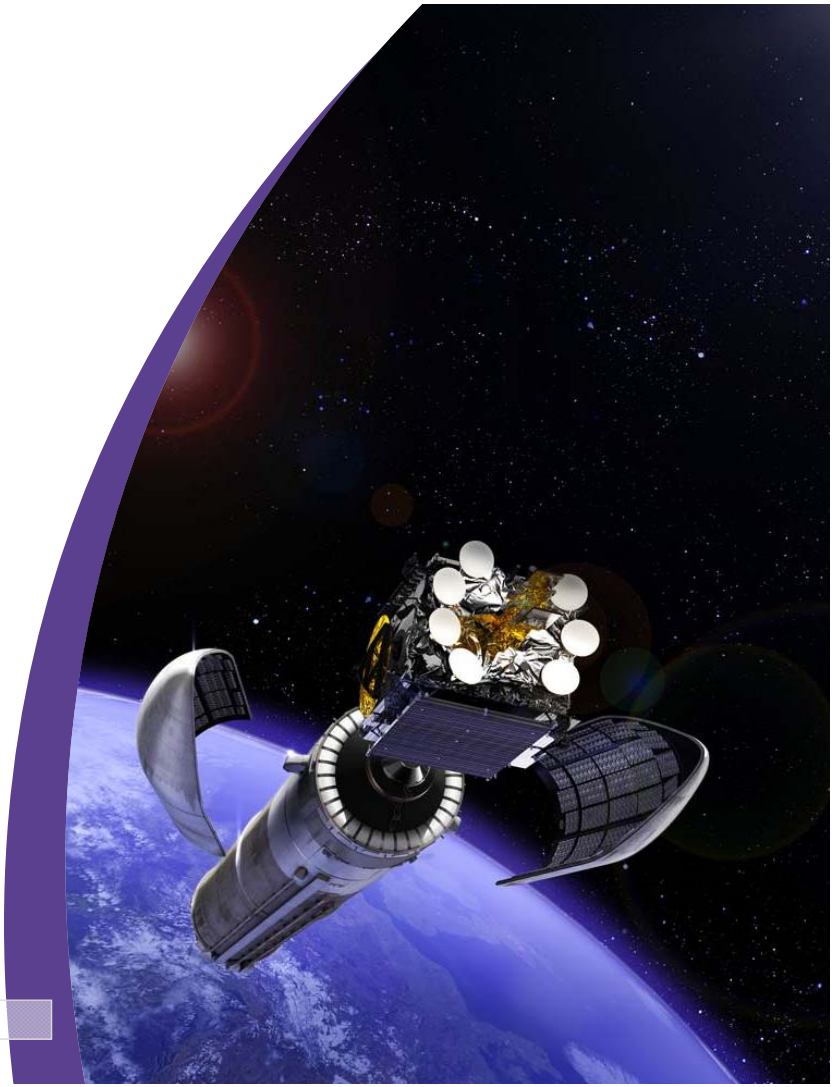


OPEN

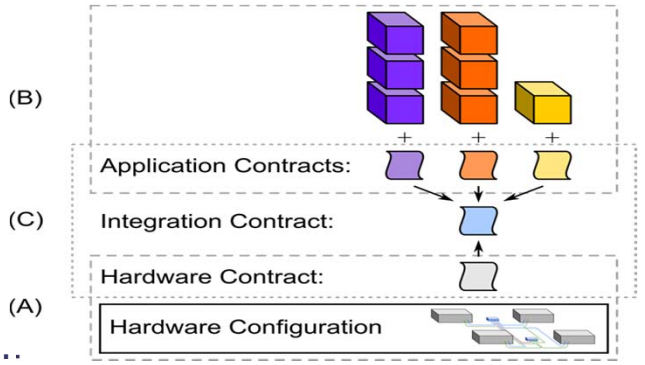
THALES

THALES

Summary



Summary and Review



- **Mixed-criticality and certification applied for decades, processes exist, …**
  - DO-297, CAST-32, DO-330, DO-326/ED202, EN50129, …
- **Criticalities are assigned by safety process and don't change**
- **To be discussed, extended, …**
  - Combination of safety process and performance (or scheduling) approaches interesting
  - Incremental certification is of real value. Partly this has been applied already in avionics, but could provide a real value to other industries as well.
  - Difference between worst-case and best-case approaches increase: is there a potential for new methods and improvements for mixed-criticality architectures
  - Combination of safety development approach and security development approaches

THALES

References related to topic

- Michael Paulitsch, Oscar Medina Duarte, Hassen Karray, Kevin Mueller, Daniel Muench, Jan Nowotsch. Mixed-Criticality Embedded Systems — A Balance Ensuring Partitioning and Performance. In Proc. of Euromicro Conference on Digital Systems Design (DSD) August 2015.
- Daniel Muench, Michael Paulitsch, Andreas Herkersdorf. IOMPU: Spatial Separation for Hardware-Based I/O Virtualization for Mixed-Criticality Embedded Real-Time Systems Using Non-Transparent Bridges. In Proc. of International Conference on Embedded Software and Systems (ICESS2015), August 2015.
- Daniel Muench, Michael Paulitsch, Oliver Hanka, and Andreas Herkersdorf, MPIOV: Scaling Hardware-Based I/O Virtualization for Mixed-Criticality Embedded Real-Time Systems Using Non Transparent Bridges to (Multi-Core) Multi-Processor Systems., Conference on Design, Automation and Test in Europe (DATE), 2015.
- Daniel Muench, Michael Paulitsch, Oliver Hanka, and Andreas Herkersdorf, SgInt: Safeguarding Interrupts for Hardware-Based I/O Virtualization for Mixed-Criticality Embedded Real-Time Systems Using Non Transparent Bridges, International Conference on Architecture of Computing Systems (ARCS), 2015.
- Jan Nowotsch, Michael Paulitsch, Daniel Buehler, Henrik Theiling, Simon Wegener and Michael Schmidt. *Monitoring-Based Shared Resource Separation for Commercial Multi-core System-on-Chip*. 26th Euromicro Conference on Real-Time Systems (ECRTS14). July 2014.
- Kevin Mueller, Georg Sigl, Benoit Triquet, Michael Paulitsch. *On MILS I/O Sharing Targeting Avionics Systems*. Tenth European Dependable Computing Conference (EDCC 2014), Newcastle upon Tyne, UK, May 13-16, 2014.
- Jan Nowotsch, Michael Paulitsch, Arne Henrichsen, Werner Pongratz, and Andreas Schacht. Monitoring and WCET Analysis in COTS Multi-core-SoC-based Mixed-Criticality Systems. Workshop at Design, Automation and Test in Europe (DATE) Conference. 2014.
- Daniel Muench, Michael Paulitsch, and Andreas Herkersdorf. Temporal Separation for Hardware-Based I/O Virtualization for Mixed-Criticality Embedded Real-Time Systems Using PCIe SR-IOV. 10th Workshop on Dependability and Fault Tolerance (VERFE'14) in conjunction with ARCS 2014, Lübeck, Germany, February 25th – 28nd, 2014.
- Daniel Muench, Ole Isfort, Kevin Mueller, Michael Paulitsch, and Andreas Herkersdorf, Hardware-Based I/O Virtualization for Mixed Criticality Real-Time Systems Using PCIe SR-IOV. In Proc. of the IEEE Int. Conf. on Embedded Software and Systems. Sydney, Australia, Dec. 2013.
- Jan Nowotsch and Michael Paulitsch. Quality of Service Capabilities for Hard Real-Time Applications on Multi-core Processors. In Proc. of 21st Int. Conf. on Real-Time Networks and Systems, Oct. 16-18, 2013.
- Michael Paulitsch, Ludwig Girbinger, Jan Nowotsch, and Daniel Muench. Transparent Software Replication and Hardware Monitoring Leveraging Modern System-On-Chip Features. In Proc. of the 19th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), Aug. 19-21, 2013.
- Kevin Mueller, Daniel Muench, Ole Isfort, Michael Paulitsch and Georg Sigl. „Decreasing System Availability on an Avionic Multicore Processor Using Directly Assigned PCI Express Devices". 2013 European Workshop on System Security (EuroSec 2013). April 2013.
- Ondrej Kotaba, Jan Nowotsch, Michael Paulitsch, Stefan M. Petters and Henrik Theiling; "Multicore In Real-Time Systems – Temporal Isolation Challenges Due To Shared Resources,", in Proc. of Workshop on Industry-Driven Approaches for Cost-effective Certification of Safety-Critical, Mixed-Criticality Systems (part of DATE 2013). March 2013.
- Craig Partridge, Robert Walsh, Matthew Gillen, Gregory Lauer, John Lowry, W. Timothy Strayer, Derrick Kong, David Levin, Joseph Loyall, and Michael Paulitsch. 2012. A secure content network in space. In Proceedings of the seventh ACM international workshop on Challenged networks (CHANTS '12). ACM, New York, NY, USA, 43-50. DOI=10.1145/2348616.2348626 http://doi.acm.org/10.1145/2348616.2348626
- Kevin Mueller, Michael Paulitsch, Sergey Tverdyshev, Holger Blasum, Reinhard Schwarz. MILS-Based Information Flow Control in the Avionic Domain: Software Architecture and Verification. In Proc. of the Digital Avionics Systems Conference. 2012.
- Kevin Mueller, Michael Paulitsch, Sergey Tverdyshev, Holger Blasum. MILS-Related Information Flow Control in the Avionic Domain: A View on Security-Enhancing Software Architectures. Workshop on Open Resilient human-aware Cyber-physical Systems. 2012.
- Jan Nowotsch and Michael Paulitsch. "Leveraging Multi-Core Computing Architectures in Avionics". In Proc. of Ninth European Dependable Computing Conference. European Dependable Computing Conference. 2012.