PROJECT FINAL REPORT

Embedded multi-core systems for mixed criticality applications in dynamic and changeable real-time environments

Project Acronym:

EMC²

Grant agreement no: 621429

Deliverable no. and title	**D13.7 – Final Report** **PartA - Publishable Summary**	
Work package	WP13	Project Management
Task / Use Case		
Lead contractor	Infineon Technologies AG Dr. Werner Weber, mailto:werner.weber@infineon.com	
Deliverable responsible	Infineon Technologies AG Werner Weber, email werner.weber@infineon.com Alfred Hoess, email alfred.hoess@web.de	
Version number	v1.1	
Date	15/08/2017	
Status	Final	
Dissemination level	Public (PU)	

Authors

Partici-pant no.	Part. short name	Author name
01A	IFAG	Werner Weber in collaboration with representatives of all project partners

Document History

Version	Date	Author name	Reason
v0.1	14/05/2017	Alfred Hoess	Initial template and first version
v0.2	15/05/2017	Werner Weber	Project internal review
v1.0	16/05/2017	Alfred Hoess	PMT feedback included, final editing and formatting, deliverable submission
v1.1	15/08/2017	Alfred Hoess	WP updates included, final editing and formatting

Table of contents

List of figures

List of tables

1. Introduction

1.1 Motivation and overall objective

The embedded systems segment is undergoing a disruptive innovation process where different kinds of systems are connected to each other, boundaries of application domains are alleviated and interoperability plays an increasing role. Formerly closed systems are forced to be opened up. Multi-core and Many-core processors become available whose exploitation for critical and real-time applications was too slow, inefficient and expensive at project start.

The overall **objective** of the EMC^2 (*Embedded multi-core systems for mixed criticality applications in dynamic and changeable real-time environments*) project therefore was to foster this change of embedded systems through an innovative and sustainable service-oriented architecture approach for mixed criticality applications in dynamic and changeable real-time environments.

The EMC^2 project was built on the results of previous Artemis and further European projects and strived for making the big step from basic research to industrial applications. The **following challenges** were addressed and solutions developed to overcome the challenges:

- Dynamic Adaptability in Open Systems
- Utilization of expensive system features only as Service-on-Demand in order to reduce the overall system cost
- Handling of mixed criticality applications under real-time conditions
- Scalability and utmost flexibility
- Full scale deployment and management of integrated tool chains through the entire lifecycle

This embedded system approach forces the breakthrough and deployment of Multi-Core technology in almost all application domains – Space, Transport, Health Care, Energy, and Industry – where real-time and mixed-criticality are an issue. Hence, the EMC^2 project was organized in a **matrix structure** with horizontal and vertical activities closely linked to each other:

- **Horizontal activities** with so-called **technological work packages (WP1-WP6)** to develop dedicated technologies required for the development of embedded, mixed-criticality multi-core systems.
- **Vertical activities** with so-called **Living Labs (WP7-WP12)** including several demonstrators for mixed-criticality embedded systems aiming at the specific application domains. Technologies developed in WP1-WP6 were applied and evaluated in the dedicated use cases of the living labs.

1.2 Strategic project objectives

The main strategic objectives of the EMC^2 project are summarized in the table below. More than 60 Key Performance Indicators (KPIs) were defined to evaluate the achievement of these strategic goals.

Strategic project objectives	**ARTEMIS Strategic targets [compared with 2012 levels] taken for the AWP 2013**	**EMC² contribution**	**Measurable Key Performance Indicator**
SPO1	*reduce the cost of the system design by 15%*	The EMC² Multi-core architecture with its development ecosystem of improved programmability, dynamic runtime-environment, and tool support eases design and analysis.	The use cases in the Living Labs estimated the system design effort based on currently deployed methodology and compared it with the monitored effort spent using the EMC² development ecosystem.

SPO2	*reduce the effort and time required for revalidation and recertification of systems after making changes by 15%*	The architectural support for mixed-critical applications, the early consideration of non-functional properties and the holistic integration of development and validation / certification activities in the EMC² interoperability framework.	The use cases in the Living Labs range from organically improved to completely new types of embedded applications. The EMC² partners compared the expected number of redesigns (based on their experience) for each individual use case with the one in the EMC² demonstration use cases.
SPO3	*manage a complexity increase of 25% with 10% effort reduction*	The EMC² multi-core architecture and the development ecosystem help to reduce software complexity and leverage the benefits of module consolidation.	The complexity of a mixed-critical application must be assessed by adding the complexity and the communication / synchronization effort of isolated application components. This KPI was applied to the consolidated application considering improved features and performance figures on a case-by-case basis.
SPO4	*achieve cross-sectorial reusability of Embedded Systems devices and architecture platforms (for example interoperable components (hardware and software) for automotive, railways, aerospace and manufacturing) that will be developed using the ARTEMIS JU results.*	EMC² develops a cross-sectorial embedded hardware architecture including a dynamic runtime environment.	The use cases in the six EMC² Living Labs cover many industrial domains. An integral part of the individual evaluation in each use case was to evaluate whether the EMC² approach is applicable. The partners combined the individual results into a general picture giving a clear indication to what extent EMC² was able to develop a truly cross-sectorial platform.

Table 1: Strategic objectives of the EMC^2 project

1.3 Consortium overview

The achievement of the challenging project objectives required **huge experience in different fields** related with EMC^2 technologies and applications. In consequence, there was a demand for a huge consortium providing all required skills and experiences.

The EMC² project consortium is composed of leading European RTOs and university institutes supporting the R&D work with specific expertise. The consortium includes a substantial number of specialized SMEs in various fields providing skills for the execution of the project, major semiconductor houses with international reputation, specialized hardware and software providers, and system houses from various application fields, which lead the exploitation of project results in the living labs by supplying demonstrators for verification and validation of results.

All partners are leading in their specific area. They cover all planned activities and provide sufficient experience, capacities and skills for the execution of the project objectives. Key contributions of all project partners reveal the **complementarities** in their capacities and experiences. The set-up of the consortium ensured that a **broad value chain** is covered by the project partners - from research via hardware and software development to end-users in major application fields.

Due to the size of the EMC² consortium, partners are just listed in a table. For more details regarding the partners, please visit our public website: http://www.artemis-emc2.eu/partners_authorities/

Partici-pant no.	Part. short name	Participant organization name	Country	ARTEMIS Member State (Y/N)	Other EU Member or State/Ass. country (Y/N)	National eligibility checked by applicant (Y/N)
1	IFAG	Infineon Technologies AG	D	Y	N	Y
2	AICAS	Aicas GmbH	D	Y	N	Y
3	SFR	AVL Software and Functions GmbH	D	Y	N	Y
4	BMW	BMW AG	D	Y	N	Y
5	CAS	Airbus Defence and Space GmbH – CAS	D	Y	N	Y
6	DENSO	DENSO AUTOMOTIVE Deutschland GmbH	D	Y	N	Y
5	EADS	Airbus Defence and Space GmbH – EADS	D	Y	N	Y
7	EB	Elektrobit	D	Y	N	Y
8	eVision	eVision Systems GmbH	D	Y	N	Y
9	FhG	Fraunhofer IESE	D	Y	N	Y
10	NXPGE	NXP Germany	D	Y	N	Y
11	OFFIS	OFFIS e.V.	D	Y	N	Y
12	Siemens	Siemens AG	D	Y	N	Y
13	TUBS	Technical University Braunschweig (C3E, IDA)	D	Y	N	Y
14	TUDO	Technical University Dortmund	D	Y	N	Y
15	TUKL	Technical University Kaiserslautern	D	Y	N	Y
16	SysgoAG	SYSGO AG	D	Y	N	Y
17	AVL	AVL List GmbH	A	Y	N	Y
18	IFAT	Infineon Technologies Austria	A	Y	N	Y
19	TTT	TTTech Computertechnik AG	A	Y	N	Y
20	TUW	Technische Universität Wien	A	Y	N	Y
21	VIF	Virtual Vehicle –Kompetenzzentrum - Das virtuelle Fahrzeug, Forschungsgesellschaft mbH	A	Y	N	Y
22	AIT	Austrian Institute of Technology	A	Y	N	Y
23	Frequentis	Frequentis AG	A	Y	N	Y
24	TAT	Thales Austria	A	Y	N	Y
25	BlueICe	BlueICe	B	Y	N	Y
26	BUT	Vysoke uceni technicke v Brne	CZ	Y	N	Y
27	Freescale	NXP Semiconductor Czech Republic	CZ	Y	N	Y
28	IMA	Institute of Microelectronic Applications s.r.o.	CZ	Y	N	Y
29	UTIA	USTAV TEORIE INFORMACE a AUTOMATIZACE AV CR v.v.i.	CZ	Y	N	Y
30	SysgoSRO	SYSGO s.r.o.	CZ	Y	N	Y
31	Danfoss	Danfoss Power Electronics A/S	DK	Y	N	Y
32	DTU	Technical University of Denmark	DK	Y	N	Y
33	CEA	CEA	F	Y	N	Y
34	INRIA	Inria	F	Y	N	Y
35	Magillem	Magillem	F	Y	N	Y
36	Rockwell	ROCKWELL COLLINS, INC.	F	Y	N	Y
37	Silkan	SILKAN	F	Y	N	Y
38	Thales	Thales Research and Technology	F	Y	N	Y
39	TAF	Thales Avionics	F	Y	N	Y
40	TCS	Thales Communications and Security	F	Y	N	Y
41	Clusters[1]	Pôle Systematic	F	Y	N	Y

[1] Pôle Systematic is French “competitivity cluster”; we adopt “Clusters” as an acronym

Partici-pant no.	Part. short name	Participant organization name	Country	ARTEMIS Member State (Y/N)	Other EU Member or State/Ass. country (Y/N)	National eligibility checked by applicant (Y/N)
42	HUA	Harokopion University of Athens	GR	Y	N	Y
43	UTRC	United Technologies Research Center	IRL	Y	N	Y
44	LERO	The Irish software engineering research Centre	IRL	Y	N	Y
45	Alenia	Alenia Aermacchi	I	Y	N	Y
46	CRF	Centro Ricerche FIAT	I	Y	N	Y
47	SELEX	SELEX ES	I	Y	N	Y
48	CINI	Consorzio Interuniversitario Nazionale Informatica	I	Y	N	Y
49	DITEN	University of Genova	I	Y	N	Y
50	TASI	Thales Alenia Space Italy	I	Y	N	Y
51	POLITO	Politecnico Torino	I	Y	N	Y
52	UNIVAQ	University of L'Aquila	I	Y	N	Y
99	MBDA	MBDA Italia S.p.A., Systems & Missile Design	I	Y	N	Y
100	LEONARDO	Leonardo – Finmeccanica SPA	I	Y	N	Y
53	Fornebu	Fornebu Consulting	N	Y	N	Y
54	UiO	University of Oslo	N	Y	N	Y
55	WG	WesternGeco AS	N	Y	N	Y
56	SRL	Simula Research Lab	N	Y	N	Y
57	ISEP	Instituto Superior de Engenharia do porto	P	Y	N	Y
58	CSoft	Critical Software	P	Y	N	Y
50	INESC-ID	Instituto de Engenharia de Sistemas e Computadores, Investigação e Desenvolvimento em Lisboa	P	Y	N	Y
60	AMBAR	AMBAR Telecomunicaciones S.L.	E	Y	N	Y
61	Quobis	Quobis Networks SLU	E	Y	N	Y
62	Tecnalia	Fundación Tecnalia Research & Innovation	E	Y	N	Y
63	TASE	THALES ALENIA SPACE ESPAÑA	E	Y	N	Y
64	Integrasys	Integrasys SA	E	Y	N	Y
65	HIB	HI-Iberia	E	Y	N	Y
66	IXION	IXION Industry & Aerospace	E	Y	N	Y
67	Visure	Visure Solutions	E	Y	N	Y
68	SevenS	Seven Solutions	E	Y	N	Y
69	Telvent SCHN	Telvent Energia SA; now: Schneider Electric Espana SA	E	Y	N	Y
70	ITI	Instituto Tecnologico de Informatica	E	Y	N	Y
71	Chalmers	Chalmers University of Technology	S	Y	N	Y
72	Ericsson	Ericsson	S	Y	N	Y
73	KTH	KTH Royal Institute of Technology / Information and Communication Technology	S	Y	N	Y
74	LTU	Lulea University of Technology	S	Y	N	Y
75	SICS	SICS Swedish ICT AB	S	Y	N	Y
76	Volvo	Volvo AB	S	Y	N	Y
77	Arcticus	Arcticus AB	S	Y	N	Y
78	ABB	ABB	S	Y	N	Y
79	ArcCore	ArcCore	S	Y	N	Y
80	ALTEN	ALTEN SVERIGE AKTIEBOLAG	S	Y	N	Y
81	Systemite	Systemite AB	S	Y	N	Y
82	imec-NL	Stichting Imec Nederland	NL	Y	N	Y
83	NXPNL	NXP Semiconductors	NL	Y	N	Y
84	Philips	Philips Healthcare	NL	Y	N	Y
85	TUE	Eindhoven University of Technology	NL	Y	N	Y

Partici-pant no.	Part. short name	Participant organization name	Country	ARTEMIS Member State (Y/N)	Other EU Member or State/Ass. country (Y/N)	National eligibility checked by applicant (Y/N)
86	TNO	TNO Innovation for life	NL	Y	N	Y
87	TUDelft	The Delft University of Technology	NL	Y	N	Y
88	Vector	Vector Fabrics B.V.	NL	Y	N	Y
89	Techno	Technolution B.V.	NL	Y	N	Y
90	TomTom	TomTom International BV	NL	Y	N	Y
101	TomTomGC	TomTom Global Content BV	NL	Y	N	Y
91	IFXUK	Infineon Technologies UK	UK	Y	N	Y
92	Sundance	Sundance Multiprocessor Technology Ltd.	UK	Y	N	Y
93	UoMAN	The University of Manchester	UK	Y	N	Y
94	UoBR	The University of Bristol	UK	Y	N	Y
95	Systonomy	Systonomy	UK	Y	N	Y
96	EnSilica	EnSilica	UK	Y	N	Y
97	TVS	Test and Verification Solutions	UK	Y	N	Y
98	RTU	Riga Technical University	LV	Y	N	Y
	Technion	Technion, Israel Institute for Technology[2]	IL			

[2] Involved into EMC² as subcontractor of Infineon Technologies AG; please find details in section 5.4.

2. Project concept

A cornerstone for a very large project like EMC² was the technical coordination – focusing on efficient communication and knowledge transfer between technology and application innovation oriented work packages. A correct trade-off has to be identified in order to generate the correct innovation requested by the end-users and finally to perform exploitation of the proposed technologies. For that purpose, the following three axes were proposed:

a) Project structure organized as a matrix to closely link related activities

Due to the size of the EMC^2 project work packages were treated as more or less individual projects (meaning with own sections on objectives, task descriptions, deliverables, milestones, timeplan etc.). The collaboration principle between technology work packages (W1-WP6) and living labs (WP7-WP12) is illustrated in Figure 1.

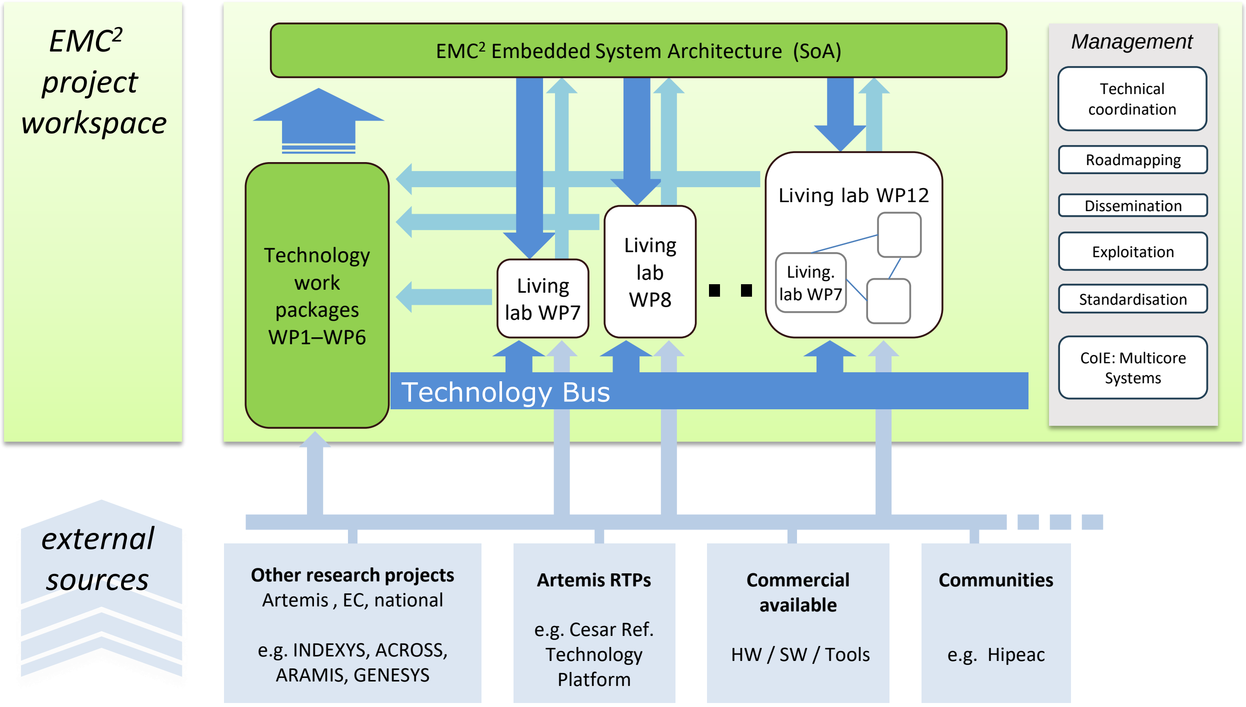


Figure 1: Project Overview

As indicated by this figure, the EMC² project consists of six technology oriented work packages, WP1-WP6, six application oriented work packages, the so called living labs, WP7-WP12 and an administrative management work package WP13.

In WP1 existing **System Architectures** were combined with new explorations of scalability and system compatibility. For a dynamic multi-core and multi-criticality software platform a system was provided where multiple applications can share the same virtual machine (VM), and hardware virtualization will be extended to multicore systems.

In WP2 **Executable Application Models and Design Tools for Mixed-Critical, Multi-Core Embedded Systems** were developed by building on the well-established, rigorous system design and automated flows and extending them towards dynamic, heterogeneous, compute-intensive, and mixed-critical systems.

WP3 covered **Dynamic Runtime Environments and Services**. Existing knowledge in mechanisms and architectures for run-time environments were enhanced to support mixed-critical systems, security techniques, safety and real-time properties.

In WP4, **Multi-core Hardware Architectures and Concepts** were developed with partial reconfiguration for application specific acceleration opening up a new era of time multiplexed hardware co-processors reducing power and improving efficiency in computational intensive applications. Their functional properties were delivered as a service through reconfigurable multi-core processors.

Key innovation areas of WP5 - **System Design Platform, Tools, Models and Interoperability** - were the generalization of HW/SW co-engineering, operational implementation of several value transversal services bridging the gap between business and engineering by providing a non-monolithic integration framework which generalized the concept of "Internet of things" for tools and adaptors.

In WP6 - **System Qualification and Certification** – main innovations were software-based fault-tolerant algorithms and architectures for multi-cores, safety and security assurance methodologies for a holistic approach to system dependability, and scalable verification solutions increasing the level of abstraction for both functional verification and safety qualification.

The application related innovation aspects were treated in the six **Living Labs** (LLs) which leverage the technological advances developed in work packages WP1-WP6. WP7 covered **Automotive** applications with related innovations – among others - in highly automated driving, SW defined radio, EV and HEV energy recuperation. In applications for **Avionics** (WP8), innovations were complete interoperability, implementation of SOA beyond DDS, new hybrid approaches consisting of statically configured high-criticality computers, and safety-critical execution elements in real-time environment. Another area is **Space Application** (WP9), targeting to proof the validity of different Multi-Processor based system architectures and related development methodologies and tool chains, opening new application domains to the use of multi-cores. In **Industrial Manufacturing and Logistics** (WP10), single multi-core designs with potentially variable number of cores can reduce the need for custom PCB layouts. The main technological innovations involved in WP11 - **Internet of Things** - were web real-time communications, ultra-low power architecture for sensor networks and synchronized low-latency deterministic networks. WP12 - **Cross Domain Applications** - took results from previous research projects and from the EMC^2 technology work packages and used them in innovative ways to prepare the ground for future multi-core applications.

b) Project organization with 9 main milestones to enable synchronization between the different activities over the entire project

The project organization relied on three main innovation cycles synchronized by 9 overall milestones (6 for each technology work package WP1-WP6, and 6 for each living lab WP7-WP12). Each innovation cycle included a development phase, followed by an integration and finally evaluation phase. The evaluation phase from the previous cycle served as input for the next development phase for possible adjustment of the technology. Following innovation cycles were foreseen:

- Requirements / concepts: purpose of the first phase was to build up state of the art – state of practice based on existing technology at the start of the project. No (major) innovation was planned during this phase.
- 1st innovation cycle: first integration and evaluation of proposed technical innovation
- 2nd innovation cycle: second (and final) integration and evaluation of proposed technical innovation

c) Know-how transfer implemented by common partner participation to both technology and application innovation oriented work packages

A major aspect in this large project was the know-how transfer between the different work packages. Hence, a strong interaction was required in order to ensure (i) correct understanding of the needs by the work packages and therefore development of the correct innovation, and (ii) understanding, hand-over and finally domain-specific tailoring of the developed technology for exploitation of the project outcomes. This know-how transfer is strongly supported by common partners participating in the different work packages.

3. Key results achieved

This section presents several highlights of the achievements made in the 12 technical work packages. In order to not overload this report, for specific details, readers are kindly referred to the public deliverables that either are (or will be subsequent to the acceptance in the final review) published via the EMC² website (see http://www.artemis-emc2.eu/publications/public_deliverables/).

With respect to technology productivity (thus hardware) according to Moore's Law the microprocessor transistor count doubles every two years. However, with respect to software, things are completely different. Michael Keating[3] (Synopsis Fellow) explains: "We live in a world where function (hardware and software) is described in code. But code does not scale. Individual coders cannot code more lines of code (RTL or C++) than they could decades ago. And as projects get bigger, productivity actually decreases. One engineer can code about 10,000 lines of (debugged, production-ready) code in a year. But 100 engineers cannot code 1,000,000 lines of (debugged, production-ready) code in a year – the problems of coordinating work and debugging complex problems degrade productivity".

EMC² was built on the very fast technological advances of micro-electronics in past decades and made use of the amazing capabilities at lowered cost levels. Systems can be quickly put together since the next technology generation is already waiting around the corner. Today, new quick methodologies are primarily exploited in consumer-oriented products, where errors may be tolerated and a new execution attempt started. This way and similar way(s) of handling errors (by bug fixing) are acceptable for consumer products.

In professional areas, such iterative approach is not feasible. In industrial areas, e.g. automotive, avionics, space, industry, health care and infrastructure, much higher levels of operational reliability are needed. In all professional domains higher HW/SW complexity is observed. Moreover, in many cases systems need to fulfill real-time safety requirements and provide the capability of dynamic reconfiguration during runtime. Prime task of EMC² therefore was to bring the two worlds together: The consumer world, which is already using advanced µC systems and the professional world demanding higher reliability, complexity and real-time capabilities.

The technological innovation focuses on three key aspects: Mixed criticality requires the handling of applications with different priorities. Dynamic reconfiguration shall cover the full range of dynamic changes on application level. In terms of hardware complexity the number of control units may change at runtime.

Based on these technological innovations, EMC² (Embedded Multi-core Systems for Mixed-Criticality Applications in Dynamic and Changeable Real-Time Environments) enables application innovations in a series of industrial domains. Within the project, automotive, avionics, space, industry, health care and infrastructure use cases were addressed. Compared to state of the art at project start, performance and energy efficiency were improved, while at the same time costs were lowered.

[3] Michael Keating, „The Simple Art of SoC Design", Closing the Gap between RTL and ESL

3.1 Technology achievements

3.1.1 Service oriented architecture on multicore embedded systems

Embedded systems are getting continuously more powerful and now include multi-core processors (Multi Processors System on Chip, MPSoC). These MPSoCs are controlling systems that are increasingly complex. The cost of development and maintenance of software for those complex systems, both in time and money, is becoming excessive. There are also other issues like reliability, dependability, and cyber-security when parts of these systems are safety critical. The EMC^2 project's aims were to address these topics in different forms. Work Package 1 (WP1) considers the use of Service Oriented Architecture (SOA) to address the above issues and highlighted the limitations of the SOA concepts for these applications.

EMC² (WP1) has proposed a reference service oriented architecture, which extends the results of another ECSEL JU project: the open source Arrowhead Framework. This framework relies on proven technology, existing international standards, and has extensive documentations to promote the integration and migration to the use of SOA on MPSoCs in applications with mixed-criticality within dynamic and changeable real-time environments.

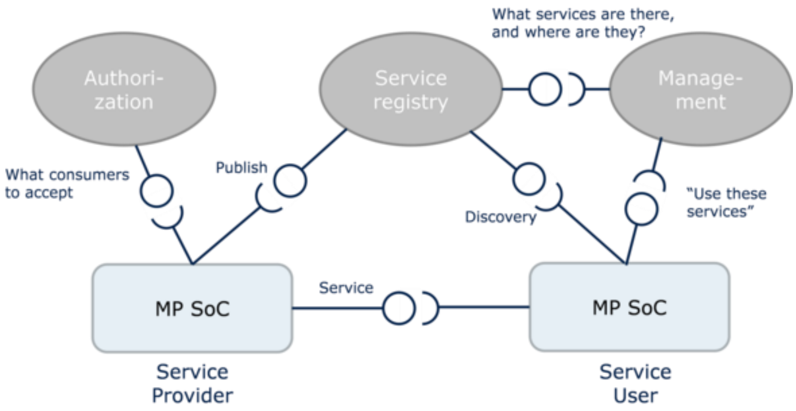


Figure 2: Service oriented architecture concept

Through technology transfer to the Living Labs (WP7-12), several demonstrators using SOA were developed within EMC^2. They include video analytics (WP12), accuracy, fault tolerance and reliable timing system distribution (WP11), and several automotive use cases (WP7). One of the latter demonstrators illustrated fault tolerance with recovery systems while another exemplified dynamic changes with wireless sensors and actuators introduced at runtime. The services of these inserted providers were registered properly. Yet another demonstrator displayed interoperability and migration from an existing architecture to SOA in a vehicle with drive by wire, traction control and ABS (i.e. an application with mixed-criticality), Figure 3.

This last demonstrator used the proposed reference architecture to provide services at runtime to different stakeholders, who can use different communication protocols (i.e. promoting interoperability in a transparent fashion). One example is a remote computer consuming information services from the core running critical systems to validate vehicle dynamics simulations. The demonstrator additionally illustrates a migration path from CAN to Ethernet in a redundant distributed active safety system enabled by SOA.

The concept of SOA on MPSoC lowers the complexity and cost of software development and maintenance. This is in part because the architecture allows the software modules to be loosely coupled and lately bound. That is, the software is not fixed at design time. At runtime, the software modules have to register the services they provide. These services are then matched to service consumer applications by an Orchestration service. The reference architecture promotes interoperability, dependability, and (cyber and IPR) security.

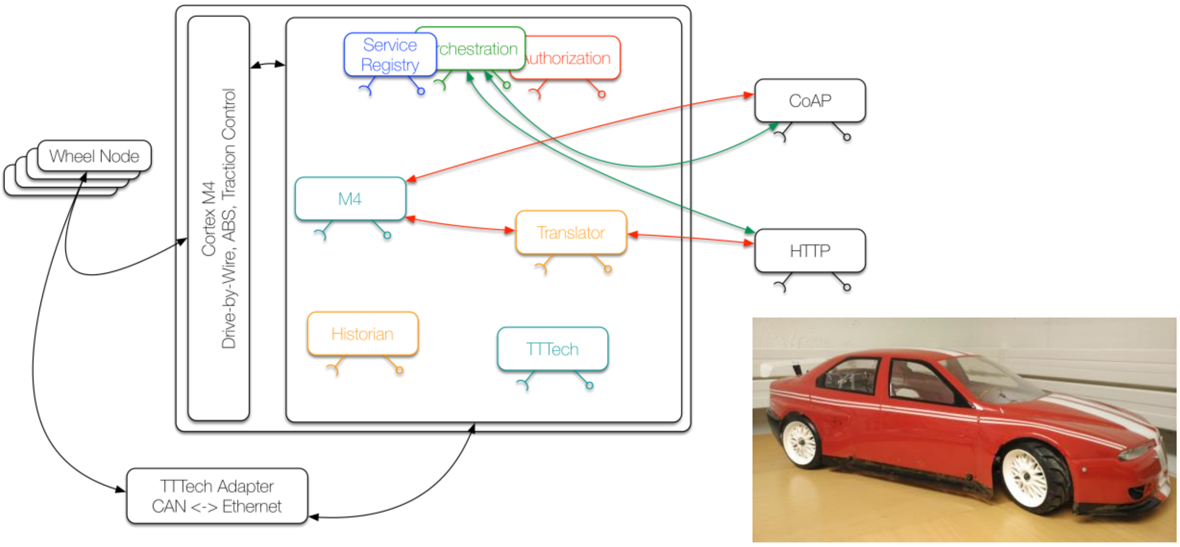


Figure 3: SOA application in automotive domain

The proposed architecture offers the integration of and migration to SOA for existing systems. Work performed in EMC² clearly states that SOA can not depend on its own guaranty hard real time performances and must rely on the other technology work packages (WP2, 3 & 4) within EMC² to achieve that when and if required. Tools to determine system predictability when utilizing SOA have been developed within the project.

3.1.2 Model based design for mixed-criticality systems

The second work package focused on the model based design for mixed-criticality systems. Overall goal here was the optimization of quality of service (QoS) in mixed critical applications. A first use case is represented by an avionic control and payload platform for multi-rotor systems (quadcopter). A safety-critical part with multiple hard real-time constraints and a computation intensive mission critical part is integrated on the same execution platform sharing a processor core.

The safety critical part concerns the flight system: Three parallel flight control tasks need to be executed in 2 ms, six sensor channels require a reaction in a 2-30 ms period, and three sensor-computing tasks need to be executed in 2 ms. These timing requirements are strict, even small violations will accumulate and lead to a crash. The quadcopter video application calls for high throughput: The quadcopter's mission of fast object detection (e.g. the red, moving ball shown in Figure 4) calls for a minimum of six frames per second and demands high data throughput.

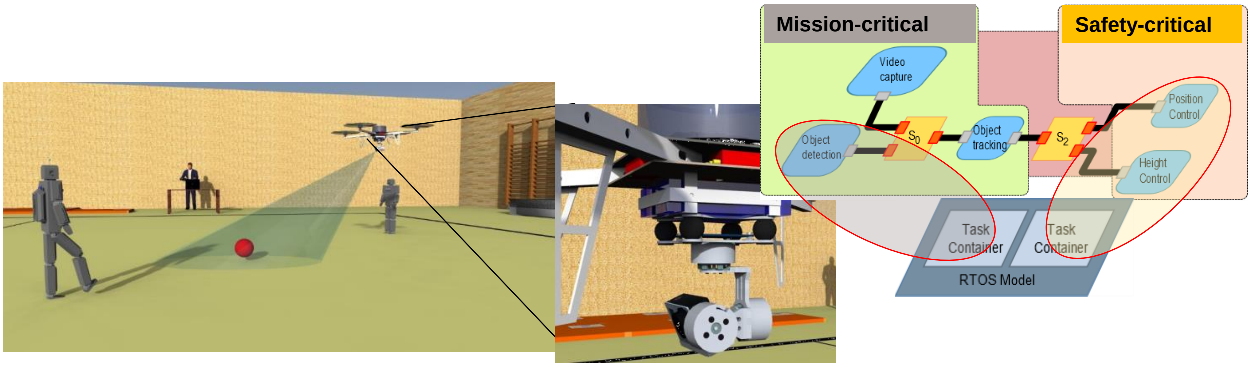
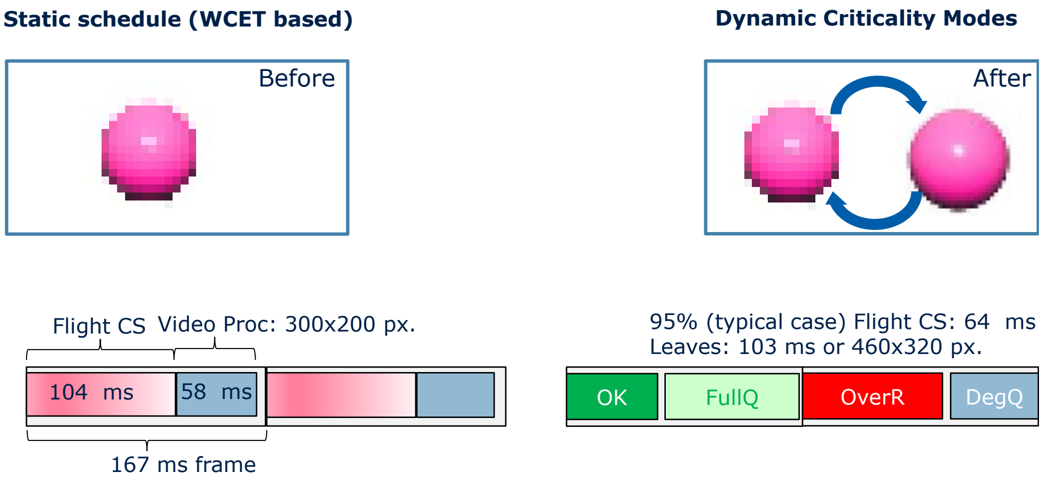


Figure 4: Use case combined avionic control and payload platform

In the frame of a classical hard real-time approach, we would schedule the tasks according to their worst case execution time (WCET). Each frame needs 167 ms corresponding to six frames per second (see above-mentioned requirement to achieve reliable detection with moving objects). The flight control system (FCS) takes up to 104 ms. This leaves 63 ms for the video mission. In order to suit this short time, the only choice is to reduce the video resolution to 300 x 200 pixels. With this resolution, video processing takes about 58 ms. Therefore, the timing requirements are fulfilled, but with degraded quality of the video.


Static schedule (WCET based)
Before
Dynamic Criticality Modes
After
Flight CS
Video Proc: 300x200 px.
104 ms
58 ms
167 ms frame
95% (typical case) Flight CS: 64 ms
Leaves: 103 ms or 460x320 px.
OK
FullQ
OverR
DegQ

Criticality Policy	# Degraded	# Full Quality	Av. Throughput
Static	30	0	1055 Kib/sec
Dynamic	13 (±3)	17(±3)	1923 Kib/sec (182%)

Figure 5: Optimized QoS in Mixed-Critical Applications with Dynamic Criticalities

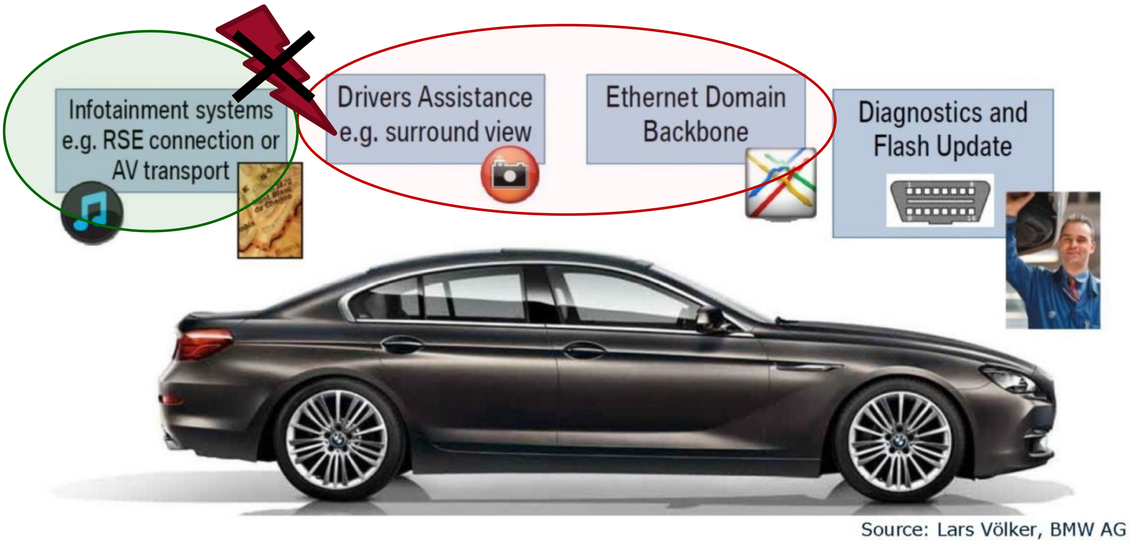
WCET (worst case execution time) analysis is often very over-pessimistic. 95% of use cases were below 64 ms. An example is given by the flight control system: When 64 ms are reached, the system is switched to high criticality, i.e. by degrading video quality as mentioned above. However, when the flight control tasks are finished before reaching 64 ms, it is switched to low criticality, i.e. by using high quality video images. The evaluation shows that in 30 runs, 17 were in high quality mode, while the other 13 runs were executed in degraded mode. The average video throughput was 82 % higher compared to using only WCET. There is a high degree of flexibility in the parameters. Timing in the simulation was calculated under high level and pessimistic assumptions. In real cases, even better results are assumed.

3.1.3 Dynamic runtime environments and services

Roughly 30 mechanisms for the isolation between safety critical parts and user domains were developed in EMC² work package 3. These were transferred to several automotive use cases, among others for safe and efficient driving, intelligent cars and green driving (Figure 6). The three most important mechanisms are aiming at:

- improving security (hypervisors for protecting bare metal virtual machines lead to 2.400 lines of code instead of 10.000 for an SOA)
- remote processor messaging lightweight inter-processor communication protocol (compared to SOA, flash use could be reduced by about 37% and at the same time RAM use was reduced by about 87%)
- global control-layer for networks on chip (NoC) to guarantee contention free and safe transmissions (compared to SOE, up to 80% improvement for safety guarantees has been achieved)

The figure below shows several selected examples which give a good overview on the conducted work. Please refer to the deliverables for a comprehensive list of all solutions.


Infotainment systems e.g. RSE connection or AV transport
Drivers Assistance e.g. surround view
Ethernet Domain Backbone
Diagnostics and Flash Update

Source: Lars Völker, BMW AG

Figure 6: Dynamic runtime environments and services in automotive use cases

3.1.3.1 Embedded Remote Procedure call and Blind Hypervision

NXP/Freescale provides the cross-core Remote Procedure Call implementation as the base enablement software for many SoC platforms, including the selected SoCs. The Embedded Remote Procedure Call (eRPC) library has been designed and implemented. The RPC is a mechanism used to invoke a software routine on a remote system via a simple local function call. When a remote function is called by the client, the function's parameters and an identifier for the called routine are marshalled (or serialized) into a stream of bytes. This byte stream is transported to the server through a communication channel. The server then unmarshalls the parameters, determines which function was invoked, and calls it. If the function returns a value, it is marshalled and sent back to the client.

RPC implementations typically use a combination of a tool (eRPC generator) and an IDL (interface definition language) file to generate source code to handle the details of marshalling a function's parameters and building the data stream. The tool also generates code for a server side shim that knows how to unmarshall a request and call the appropriate function. An IDL file is used to tell the generator tool about data types and RPC services.

Main eRPC features are:

- Lightweight but scalable, targeted to embedded systems
- Small generated code size to allow the usage on multicore parts with small memory size
- Abstracted transport interface
- Abstracted and Replaceable serialization layer
- Small size of serialized data
- Design to work well with C, but also flexible enough to support object-oriented languages
- Asynchronous notifications from server to client
- Multithreading of servers when built with an RTOS
- Unique specification of a function to be called

CEA introduced the Blind Hypervision to protect Virtual Machines privacy. The mechanism intends to guarantee both confidentiality and integrity of the code and data of virtual machines (VM) from software attacks from other VMs and from the hypervisor itself.

The untrusted hypervisor manages the VMs without being able to access their code and data by relying on two trusted components:

- The Secure Memory Management Unit (S-MMU) enables to securely partition and isolate memory areas dedicated to the hypervisor and each Virtual Machine.
- The Trusted Loader (with cryptographic functions) enables the hypervisor to load / update / migrate Virtual Machines without accessing their code and data in clear.

A demonstration with a lightweight hypervisor blindly scheduling 2 bare metal VMs (a malicious one and a target victim) validates the implemented security properties. The small size of the Trusting Computing Base (the Secure Kernel size is 2400 LoC) makes its formal verification feasible at a relatively low cost. Compared to classical hypervision implementations (with size of the TCB ranging from 10.000 to several 100.000), this solution provides a high-level security assurance.

3.1.3.2 Safe and dynamic networking

Another use case in the automotive domain is directed towards distributed congestion control for safety and comfort purposes (Figure 7). This use case faces a couple of challenges: Communication bottlenecks need to be avoided. In dense traffic, more than 300 vehicles per kilometer highway can be present, each communicating. Nevertheless, the total channel load needs to be kept below 70 % in order to ensure reserve bandwidth for emergency services. Finally, reliability needs to be higher than 99 % for receiving two messages per second in 125 m range.

Figure 7: Distributed congestion control for safety and comfort

WP3 partners, including NXP, proposed a demonstrator considering the safety-critical application of wireless communication based on WiFi-p (IEEE 802.11p) in the platooning of commercial vehicles, see **Fehler! Verweisquelle konnte nicht gefunden werden.** – off-chip networking technology. usinesswise, platooning in the logistics market is forecasted to be an attractive segment in the adoption of automated driving and WiFi-p, whilst providing additional benefits in traffic efficiency, safety and fuel-economy/CO2 reduction. WP3 partners realized a laterally automated platoon with a headway distance of 0.5 sec (11 m) at 80 km/h, on the foundation of the Cohda Wireless MK5 On Board Unit empowered by NXP’s RoadLINK dual-tuner chipset. The custom-development on the MK5-based subsystem provides redundant and low latency platooning-control-information, bi-directional audio between the trucks and video see-through from 1st to 2nd truck, using multiple service channels in the licensed ITS band at 5.9 GHz. In anticipation of possible future congestion in this band and a number of decentralized congestion control mechanisms as proposed in ETSI standards were investigated, simulated and prototyped.

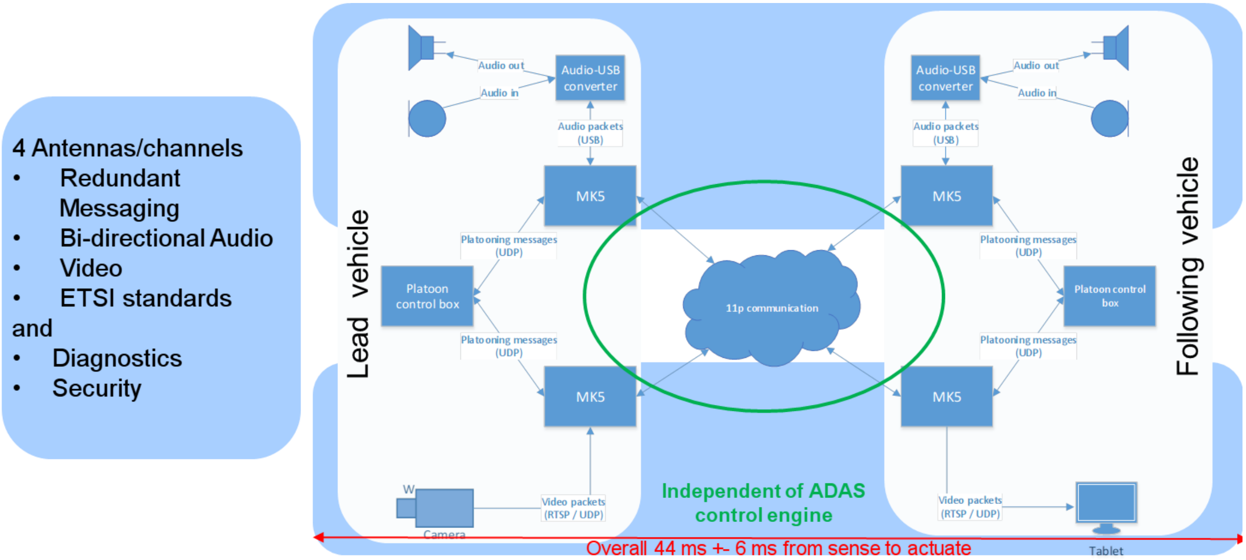


Figure 8: WiFi-p subsystem with mixed criticality data-streams

Additionally, TUBS proposed an alternative approach for providing efficient service guarantees in NoCs for mixed-critical real-time system. The mechanism combines the global scheduling for the end-to-end guarantees, with the local arbitration performed in routers shown in **Fehler! Verweisquelle konnte nicht efunden werden.**. It introduces scheduling modules, the so-called resource managers (RMs), with which applications have to negotiate their accesses to interconnect. Synchronization is achieved by using control messages and a dedicated protocol. RMs conduct a global, priority based scheduling and grant transmissions access to the NoC for a predefined amount of time. This allows exclusive access to the NoC, hence reducing blocking and decreasing the size of necessary buffers in routers. Moreover, it supports sharing of the same VC between transmissions with different criticality levels while preserving service guarantees. This can be used to decrease the number of required VCs in a system or to increase the number of hosted applications. The efficiency of the solution derives from work-conserving priority aware scheduling which directly addresses requirements of transmissions with different criticality levels.

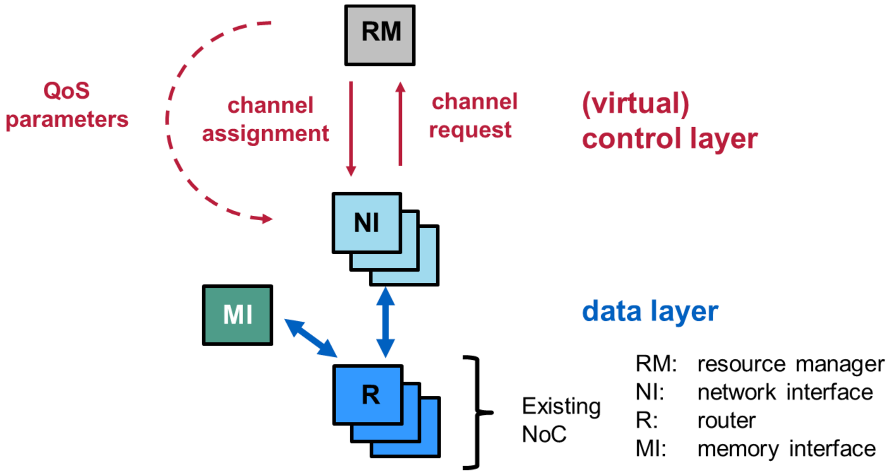


Figure 9: Structure of the SoC system with the global and dynamic arbitration

The average utilization is additionally improved through dynamic budgeting – up to 80 % improvement in comparison to TDM-based systems for realistic use cases. This allows to drastically reduce the average latencies (i.e. temporal over-provisioning) compared to other time-triggered architectures. Therefore, RMs allow to overcome the drawbacks of previously described approaches through reducing hardware overhead compared to non-blocking routers as well as temporal overhead compared to TDM. The introduced solution does not require a modification of routers and therefore can be used together with any architecture utilizing non-blocking routers. Safety of the mechanism must be confirmed with tools for the formal verification of the worst-case behavior, e.g. end-to-end latency. The introduced efficiency drives new platform specific solutions, which can be accompanied by domain specific tools for mapping, testing and deployment.

In WP3, Infineon considered an AURIX Microcontroller equipped with Ethernet connectivity. The main goal was to enable AURIX as a safe and efficient platform for mixed criticality communication. The work concentrates mainly on the network layer form **Fehler! Verweisquelle konnte nicht gefunden werden.**.

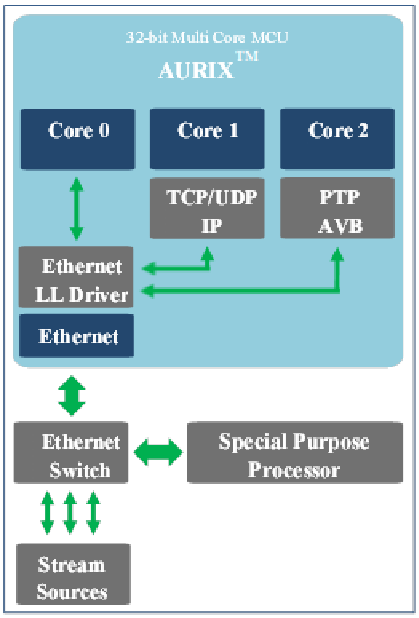


Figure 10: Infineon setup for the safe handling of mixed-critical communication.

The first period of work considered the evaluation of the AURIX platform, also in the scope of WP4. The system shown in **Fehler! Verweisquelle konnte nicht gefunden werden.** offers a high computing hroughput and scalable performance, because it can process several tasks in parallel on different independent cores. Furthermore, AURIX offers a system to protect memory resources from accesses, which are unintended or malicious. As result, the system can execute protected communication over several parallel "channels". These channels shall be separated due to the requirement that they can transfer information of different contents and of different behavior profiles. Therefore, Infineon introduced a solution using methods, which support the freedom from interference between communication channels. Such a solution was presented in a demonstrator.

Based on the analysis, the following extensions and mechanisms were implemented for RTEs:

- Isolation at application level
- Separation of message flows
- Isolation at stack level to enable deterministic behavior

Isolation at application level is achieved by putting the code images and the related data into different memory areas. Accesses to these areas are controlled and protected by AURIX HW. Separation as a method is used in the demonstrator to build a secure environment for the applications in the multicore AURIX. The demonstrator supports the isolation at two levels: Separation based on destination / source of the message flow and separation on the type of messages.

Consequently, the presented demonstrator introduces a setup, which is able to isolate the message flows, using the HW protection mechanisms of the AURIX HW. The units build messages, which are identical with the frames provided by the Ethernet protocol. The separation of the incoming message flow is the base for all isolation activities between messages. This is achieved by using two decision units, which must decide which type of message has been received. In the demonstrator, two different targets can be addressed: TCP/UDP or AVB/PTP packages.

3.1.4 Multi-core hardware architectures and concepts

In total, WP4 developed 25 hardware technologies. These are represented in 10 EMC² use cases of living labs across the entire application spectrum. Some of the highlights are:

- *Analog Mixed Signal Power System*: A novel efficient mechanism to regulate the power on a chip was developed and implemented on a test chip prototype. The key achievement consists in the efficient on-chip power management. It is one of the highlights in the project with regard to the technology readiness level and the expected time to market. The technology will be part of the next generation SoC chips.

- *Zynq based development platform EMC^2 Development Platform (EMC2 DP)*. The main achievement was a fully functional and commercially available hardware platform based on Xilinx Zynq. This result will be used as the hardware platform in the H2020 Tulipp project. The platform can already be acquired as a product and has already plans for the next generation.

• Now – Zynq SoC SoM

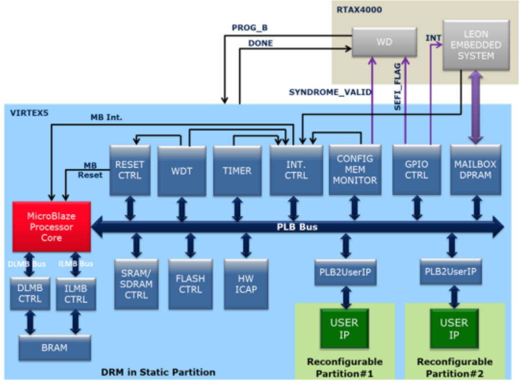
40mm x 50 mm – Z7030

• Soon– Zynq MPSoC SoM

40mm x 50 mm - ZU5EV

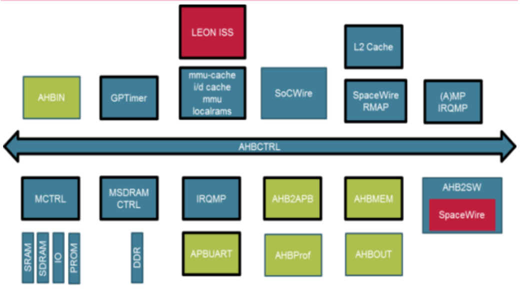
- *Reliable and Self-Healing Dynamic Reconfiguration Manager (DRM) for space applications*. A mechanism for the dynamic partial reconfiguration on Xilinx Virtex V devices was developed. It is especially designed for space applications. The DRM was transferred into an EMC² WP9 use case.

Dynamic Reconfiguration Manager Architecture

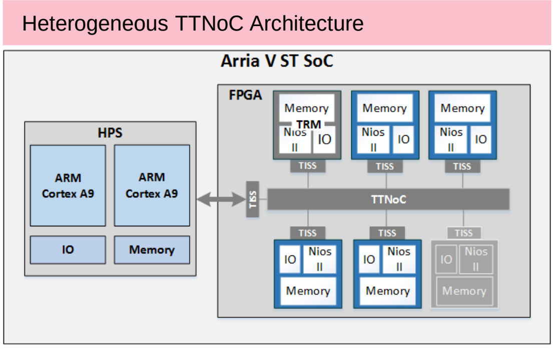


- *SocRocket architecture extensions*. The key achievement is the virtual hardware platform for fast prototyping. Significant results were achieved in deterministic cache memories and with effective isolation mechanisms for mixed-criticality applications.

SoCRocket Virtual Architecture



- *Heterogenous TTNoC (Time Triggered Network on Chip) Many-Core Architecture*: A hardware platform was achieved which provides full temporal and spatial isolation of cores. It is designed for mixed-criticality systems.



- *3D Time of Flight Architecture*. Novel imaging concepts of multi-core processors for mixed-criticality applications have been achieved. The demonstration is built using commercially available products from the partners like 3D ToF camera shown in Figure 13.

- *High Accuracy Time Synchronization System.* A long term research innovation initiative to achieve scalable nanosecond synchronization for industrial applications. The focus in the project was to improve services and integrate them in the new range of products, e.g. White Rabbit ZEN and White Rabbit LEN.

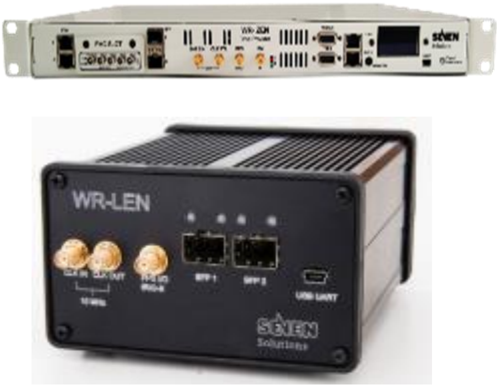


Figure 11: White Rabbit ZEN, LEN

In order to not overload the report, just two of the 6 highlights shall be presented in more detail, namely the EMC2-DP and Asymmetric Multiprocessing as well as 3D Time of Flight Architecture.

Use case Asymmetric Multiprocessing

EMC2-DP is a hardware platform developed by partner Sundance. It enables the usage of System-on-Module-Components with Zynq. EMC2-DP is compatible with the Xilinx Software Defined System-on-Chip (SDSoC) 2015.4. Asymmetric Multiprocessing on the EMC2-DP platform with MicroBlaze and the 8xSIMD EdkDSP vector floating point accelerator achieved the following key parameters:

- Constant FIR filter: 1227 MFLOP/s.
- Adaptive LMS filter: 776 MFLOP/s

The comparison of these figures with those of a 666 MHz ARM Cortex A9 (with a NEON vector processing unit) shows a 2,3 times faster processing time and a 78 times faster processing time than a MicroBlaze processor. The figures below present the EMC2-DP edge detection in Full HD.

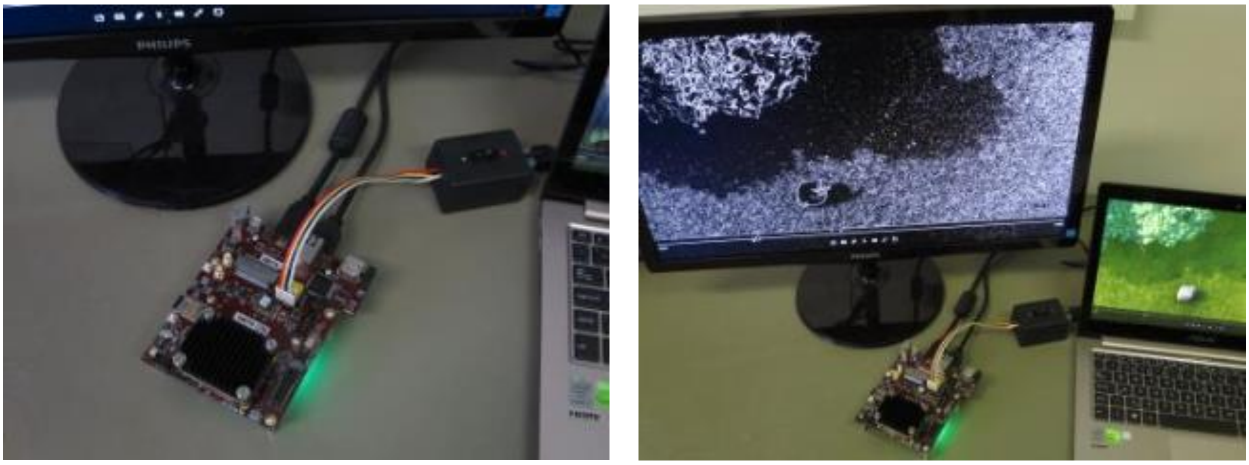


Figure 12: EMC2-DP edge detection in Full HD

Use case: Time-of-Flight 3D Imaging

Time-of-Flight (ToF) 3D imaging concepts are targeting multi-cores and mixed-criticality applications. The key achievements from EMC² WP4 are twofold. First, the achievements concern the (low-resolution) Time-of-Flight and (high-resolution) RGB sensor fusion. For the first time, a high-performance sensor fusion solution for embedded systems was achieved. It provides upscaled resolution, increased sharpness, less noise, less motion artefacts and high frames per second. Second, key result is the hardware accelerated Time-of-Flight processing relying on the novel Zynq-based system solution for mixed critical applications.

Figure 13: 3D Time-of-Flight camera

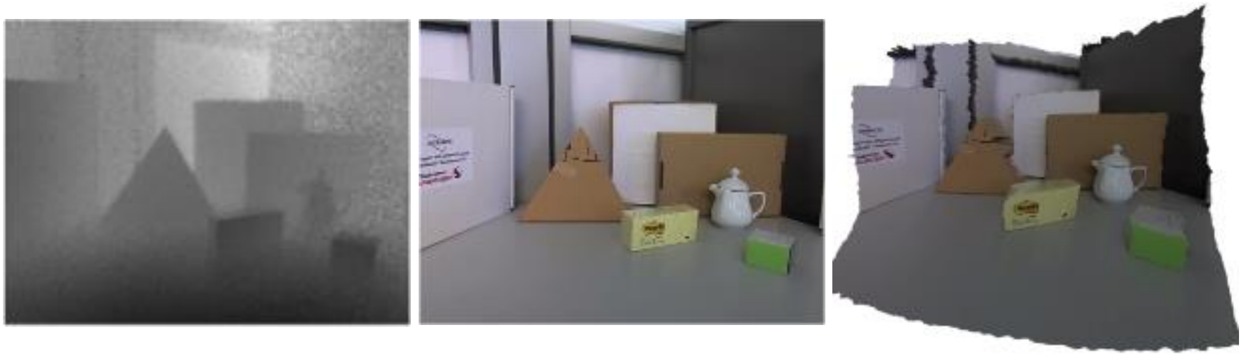


Figure 14: Time-of-Flight 3D imaging (left: low resolution ToF image; center: high resolution RBG image; right: fusion result of low resolution ToF and high resolution RBG image)

3.1.5 Make the invisible visible for complex embedded systems

Many challenges of software engineering are based on the underlying conflict of adding new functionality versus increasing or even preserving the architecture quality of the system under development and maintenance. Customers enjoy the new features and managers are satisfied with the additional sales revenues.

Only software architects and developers appreciate the benefits of a high-quality software architecture dependency structure. However, any large-scale code base is full of unwarranted complexity. Methods are urgently needed to decide which work on architectural qualities has high priority by aligning maintenance work with the realization of important features, identifying whether these have been realized in a maintainable way, and improve their realizations in case of not.

The proposed solution by work package 5 “Feature-based Quality Analytics” uses the history evolution of the software by analyzing the software code versions of the software repository. In principle, it is adding the expression of ***time*** to software analytics. It goes beyond the current mechanisms of repository systems, traces and tracks Lines of Code to several feature implementations up to the present. It – **first -** identifies which parts of the code have survived all changes across all feature realizations and can still be attributed to each specific feature. The **second** step is to measure Feature Coupling of two features by resolving the extent of changes on overlapping modules touched by both respective feature realizations. Statistical methods are used to compute a distance of features. The lower the distance of a feature to other features the higher its coupling.

“Feature-based Quality Analytics” recommends focusing on the features being important **and** highly coupled. There is evidence based on the investigation of open-source projects that these feature realizations make it hard to add new features and lead to higher bug-ratios of the affected overlapping modules.

> „… **large-scale codebases** are full of unwarranted complexity. It‘s unreasonable to address them all at once … to get the most out of your redesign, you should **improve the part of the code you will most likely work with again in the future**.“[3]

Furthermore WP5 “Feature-based Quality Analytics” provides the integration of ***programmable hardware*** (via the language VHDL) into the tool chain. Together with the inclusion of the expression ***time***, an effective method has been established for managing unwarranted complexity even in large-scale code bases for embedded systems.

For high performance embedded systems (e.g. devices for the Internet Of Things, IoT) the inclusion of programmable hardware into software analytics is essential as classical sequential programming is now combined with the parallel programming paradigm of programmable hardware (FPGAs).

The Iceberg – Features versus Architecture Quality

Therefore, methods are urgently needed to decide which work on architectural qualities has high priority by aligning maintenance work with the realization of important features, identifying whether these features have been realized in a maintainable way, and improve their realizations in case of not.

The resolution of important features certainly depends on the domain and application context of the software. We used the Kano requirements model to get a general notion of important features.

The Kano model distinguishes between exciting, performance and basic requirements. Normally, the exciting features represent the Unique Selling Points of a product, whereas the performance features are the ones to compete with other vendors or providers. The basic ones are deemed mandatory to sell the product at all.

The exciting features are the ones to expect more enhancements to come. Some of them are most probably the ones with ongoing requests to come. The realization code of these features is the one most likely to be worked with again in the future.

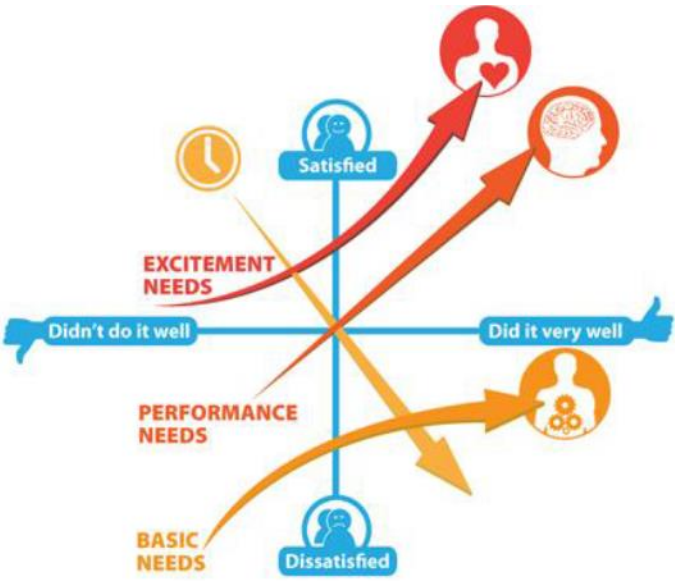


Figure 15: Kano Requirements Model

Furthermore, a significant amount of legacy code-bases needs to be made "multi-core" ready and even supporting parallel programming paradigms, where in those cases even basic performance features might be critical to the respective product. It does not necessarily imply safety properties of the software, but such a feature might be critical in the sense of being an absolute selling point of a product.

Consequently, the realizations of critical features have to be absolutely maintainable. Every defect affecting this part of the code has to be removed in a quick and definite way. As for the exciting features mentioned above, the question arises how easy it is to add enhancements of them to the software.

Bearing all this in mind, the objective on work package WP5 was the analysis of code histories to directly link "visible" functionality to "un-visible" quality problems and prioritize maintenance budget.

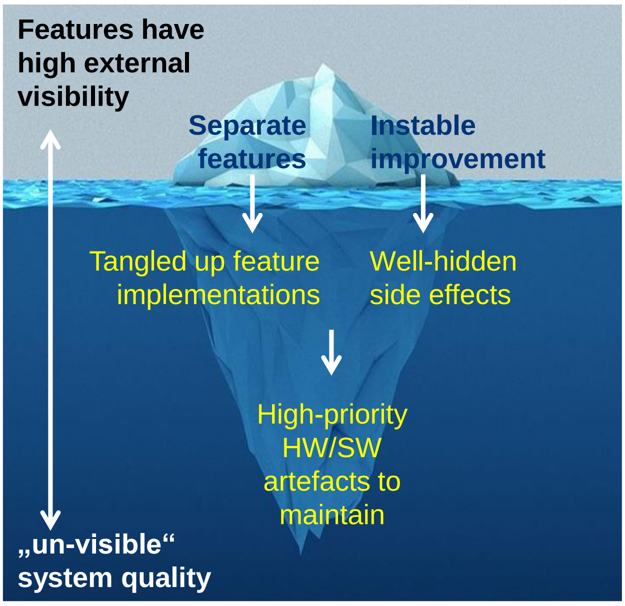

Features have high external visibility
Separate features
Instable improvement
Tangled up feature implementations
Well-hidden side effects
High-priority HW/SW artefacts to maintain
„un-visible" system quality

Figure 16: Make the invisible system quality visible by suitable features

The key achievement of WP5 is an analysis tool chain with overview dashboards to make the "un-visible" visible. The dashboard means a web portal presenting all evident states of a development department automatically and highly actual. Subsequent to checking in a new Code, within a few minutes the information and selected user features shall be presented on the dashboard.

Considering a company developing video conference equipment, for instance. Synchronizing the movement of lips with the audio is an essential feature for this application. Such parameters shall also be measured and visualized automatically after each software modification. A development engineer receives feedback regarding performance improvements or decrease and can initiate modifications very early in the development process. The dashboard also provides information on change analytics.

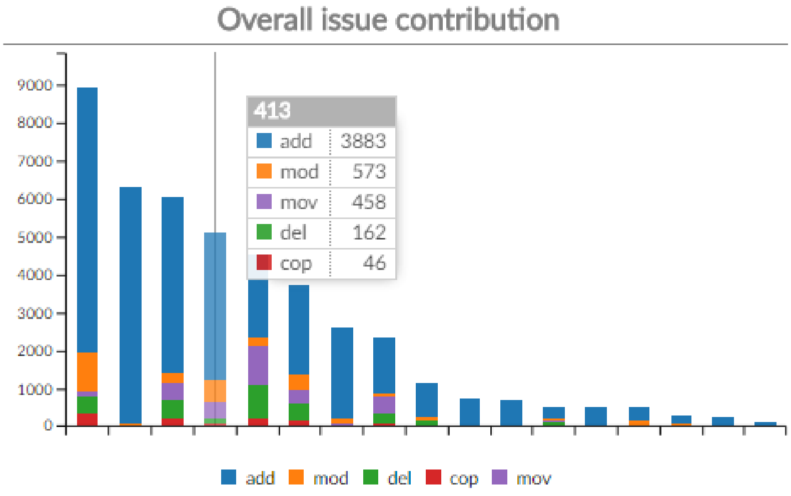


Figure 17: Overall issue contribution

The tool chain performs high correlations of software / programmable hardware artefacts with high feature tangles and bugs as well as important results of performance measurements. The tool chain was validated in 10 Siemens industrial grade projects. Five years of code history were used for the validation of the corresponding up to 1.7 million lines of code (MLoC) in VHDL, C, C++, Java and C#.

„The method has provided us a new level of transparency to find the right tradeoff between new features, risk and technical debt reduction."[4]

Thanks to the Artemis project iFEST, EMC² and long term support from Siemens, the developed tool chain "Feature-based Quality Analytics" has now reached a maturity level, which initiated the discussion about founding a start-up.

3.1.6 Safety and security of mixed-criticality embedded systems

Future (EMC²) systems are open and adaptive. These characteristics imply new levels of complexity and uncertainty, which, in turn, lead to significant challenges regarding the assurance of safety [1][2]. Without adequate safety assurance, however, the full potential of EMC² systems cannot be unlocked. In the EMC² project, we consequently tackled two of the main challenges related to safety assurance in next generation systems.

On the one hand, there is the need to take security into equation as a potential cause for safety incidents. Thus, we developed comprehensive tools and techniques for safety-security co-engineering. On the other hand, there is the problem of dynamic integration and reconfiguration of systems, which makes it impossible to completely analyze systems at development time already. Moreover, while safety and related certificates demand (almost) no change to a certified system, security requires almost continuous updates (i.e. security patches) to react on newly discovered vulnerabilities [3]. Thus, we developed an approach for conditional runtime certification of EMC² systems, enabling systems to autonomously and dynamically evaluate their safety in the context of dynamic changes (such as dynamic integration or dynamic updates).

As regards the co-engineering of safety and security, comprehensive work has been carried out in the EMC² project and a range of interesting results has been achieved:

- Development of a domain independent co-engineering framework, based on IEC 61508 (AIT)
- Development of a pattern-based co-engineering process (IESE, VIF, AIT, AVL, TU Graz; Safecomp paper [4]) which defines an iterative process of optimizing safety and security of a system and coordinates between engineering disciplines and eases the identification of optimal trade-offs
- Development of a co-analysis approach for the smart grid domain, based on IEC 61508 and IEEE 1686 (Telvent, Use Case 11.5))
- Development of an extended STPA-Sec version (AIT & AVL & ViF)
- Development of enhanced analysis techniques: FMEVA and SeCFTs

[4] Matthias Herbort, Evosoft GmbH, Certified Siemens Senior Software Architect

Investigation of potential combination of Failure Mode, Vulnerabilities and Effects Analysis (FMVEA) and Security-enhanced Component Fault Trees (SeCFTs) (AIT & TUKL). Both approaches extend established safety analysis methods with security and are able to cooperate and be used in unison during the engineering process. There are some specialties regarding the approach to likelihood (safety is able to calculate hard probabilities for hardware elements while security has challenges with regards to the probability of the materialization of a threat) which are resolved by the combination of the engineering approaches.

- Progress wrt secure dynamic reconfiguration (AICAS)
- RoViM: Rotating Virtual Machines for Security and Fault-Tolerance (AIT, applied in Use Case 11.2), multiple redundant virtual machines, rotating between the states active, cleaning and standby
- Fault injection methodology; leveraging and extending VeTeSS results (IFXUK):
  - Extended Guidelines on the Test Plan generation
  - Fault injection result correlation at different levels
  - Verification and Validation automation
  - Tool qualification methodology and guidelines as well as tool qualification for Fault Injection

During the project, an overview of the current status of approaches towards safety and security in automotive and other domain standards was developed [5]. Based on this overview EMC^2 partners contributed to the development of an interaction approach between safety and security engineering and contributed this to the reworking of the automotive functional safety standard and the newly developed automotive cybersecurity standard.

Since 2016, the IEC TC65, Industrial Process Measurement, Control and Automation, Ad-Hoc-Group AHG1 became a new Working Group WG20 “Framework for Functional Safety and (Cyber-) Security”, elaborating a Technical report TR 63069 with this title. EMC² partner AIT contributed to this report the EMC² concept of interaction between the expert teams from the safety and the security domain as depicted in Figure 18. This will become part of the CD (Committee Draft) of IEC TR 63069, the first version of the report, which will be officially distributed to the National Committees of the member states for comments at the end of July 2017. This is relevant for the whole set of generic basic safety and security standards and the derived domain-specific standards of IEC in the long-term future.

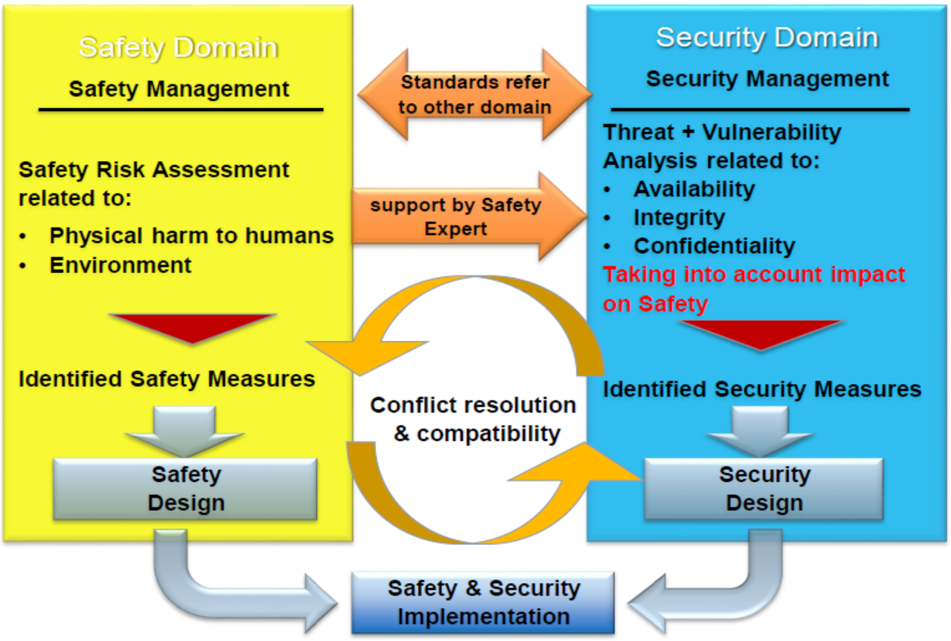

Safety Domain
Safety Management
Safety Risk Assessment related to:
• Physical harm to humans
• Environment
Identified Safety Measures
Safety Design
Standards refer to other domain
support by Safety Expert
Conflict resolution & compatibility
Security Domain
Security Management
Threat + Vulnerability Analysis related to:
• Availability
• Integrity
• Confidentiality
Taking into account impact on Safety
Identified Security Measures
Security Design
Safety & Security Implementation

Figure 18: Interaction between Safety and Security Domain, proposal for IEC TR 63069, “Framework for Functional safety and (Cyber-) Security”

Another major result in this context is the cooperative safety and security process framework as part of the V-Modell XT for the automotive domain (detailed in [6]). The described framework aims in harmonization of the specific outcomes and analysis activities of the safety, security and quality related activities.

This approach provides an overview of industrial established practices in relation to comprehensive dependability engineering (combined safety, security and quality engineering) and a mapping to the respective lifecycle phases (concept phase, system development, hardware and software development).

The tight integration of security considerations together within the safety and reliability lifecycle is required to ensure consistent system definition, implementation and validation. The consideration of the entire product lifecycle is important in order to obtain a clear view of the entire system and to tailor the dependability development accordingly. This targets the deployment of necessary and sufficient methods, which at the same time are not over-engineered (and therefore competitive for the market from a pure cost point of view).

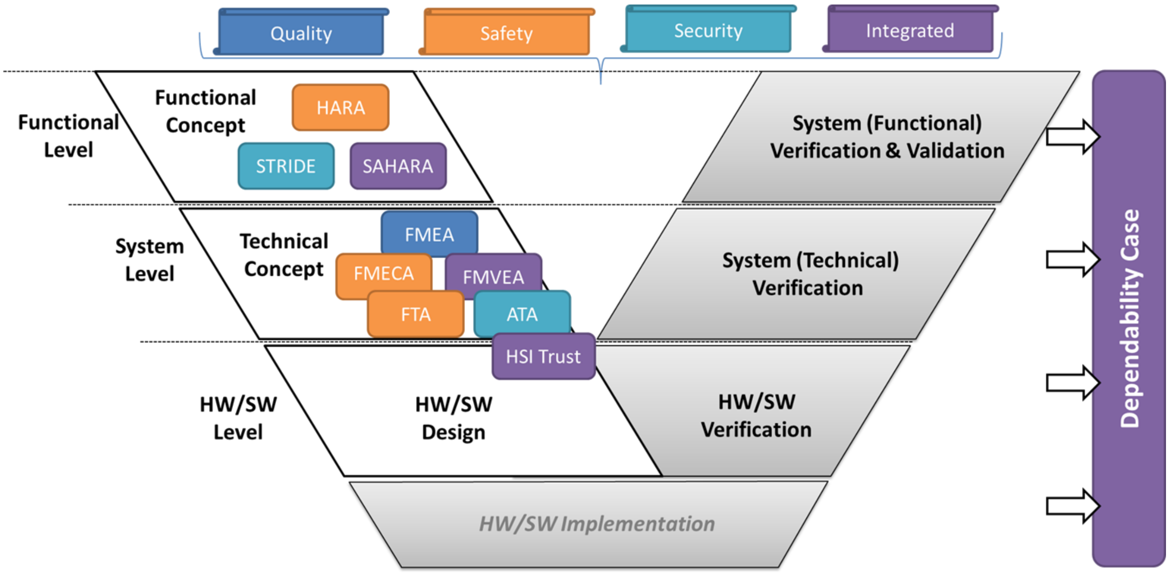


Figure 19: Mapping of Methods to Specific Development Phases.

As regards the conditional runtime certification approach, the overall concept is based on Conditional Safety Certificates (ConSerts) [7]. ConSerts are defined based on a sound safety argument or safety case, where context dependencies that cannot be resolved already at development time (e.g. because a service of another system is used dynamically or the component under development is to be deployed as a dynamic update) are formalized as context-demands to be checked at runtime. Depending on the fulfillment of such demands, corresponding guarantees can be determined dynamically and ultimately, and dynamic safety assessments are enabled in a system of systems context. Based on the resulting guarantees of the overall system composition, it is possible to dynamically parameterize systems, unlock or constrain behavior. In EMC², the conditional safety certification approach has been further advanced, specifically with respect to the mixed criticality aspect, which is one of the central challenges of EMC² [8] [9]. Now, the negotiation of conditional safety certificates does not only take place between different systems “horizontally”, but also between (mixed-critical) applications (i.e. software) and the platform (i.e. hardware) “vertically” (see Figure 20), e.g. with respect to the parallel use of platform resources and potential unwanted interferences based thereon. As a result, the systems themselves are now enabled to perform necessary safety checks – between different systems, but also between different applications and a platform – dynamically at the time of integration. The results have been validated based on use case transfers within the project. In particular, the approach has been implemented in Use Case 7.3 and the resulting demonstrator has been shown at the final review.

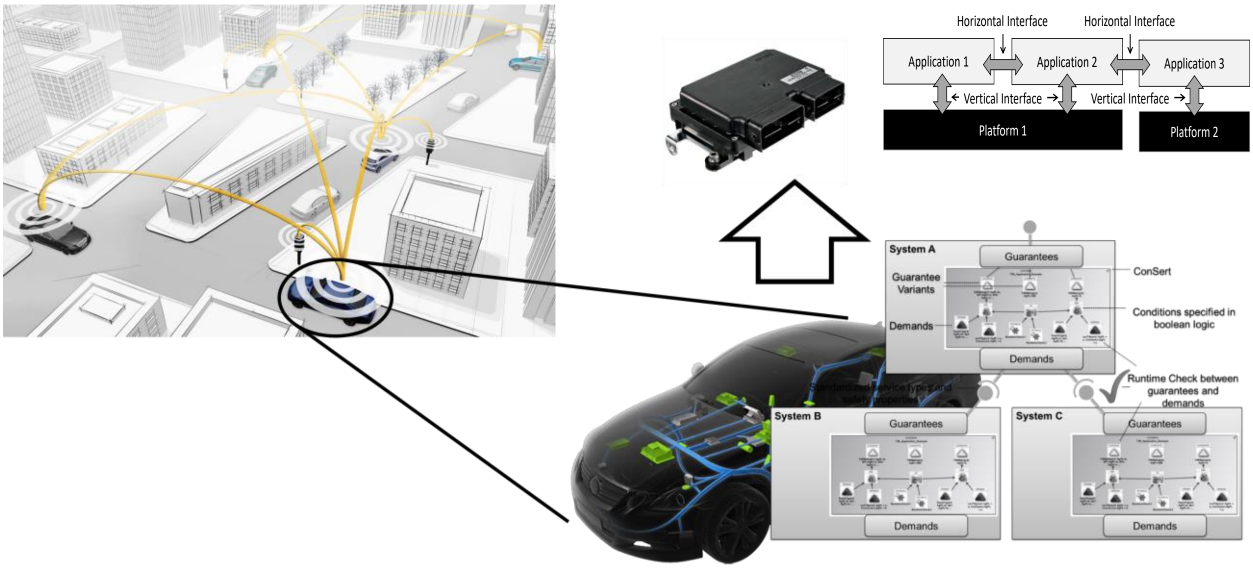


Figure 20: Conditional runtime certification

In summary, the work which has been carried out in WP6 leads to two essential achievements: First, the integration of safety and security co-engineering to handle the impact of security on safety. And second, the conditional runtime certification for assuring safety of EMC² open and dynamic embedded systems. The concept has been successfully applied to automotive (vehicle-to-vehicle and vehicle-to-infra-structure) use cases.

3.2 Application achievements

The technologies summarized in sections 3.1.1 to 3.1.6 were applied in a series of industrial domains and use cases, i.e. automotive, aviation, space, industrial manufacturing, logistics, Internet of Things, health care, railway and seismic surveying.

Figure 21: Application areas considered in EMC²

3.2.1 Automotive applications

Reduce the number of Automotive Control Units to save cost and increase performance

In modern vehicles, up to 100 heterogeneous single-core systems (Electronic Control Units) exist. Each system (e.g. radio, air conditioning, engine management) has its own ECU, specialized for its individual criticality level. EMC^2 aims at reducing this large number to a few homogeneous multi-core systems for mixed criticality levels.

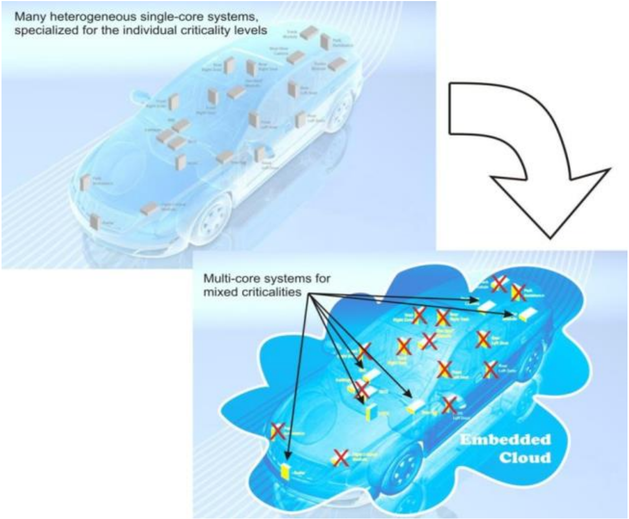

Many heterogeneous single-core systems, specialized for the individual criticality levels
Multi-core systems for mixed criticalities
Embedded Cloud

Figure 22: Vision – reduce number of ECUs in vehicles

The tasks now running on separate ECU's will run in the future on multi-cores in order to reduce the number of ECUs. Therefore, systems could work as services (e.g. in an embedded cloud or an external cloud outside the vehicle) and no longer as independent systems.

Our assumption is that by introducing SoA, the effort needed for development, integration, verification, upgrade with new functionality at design time and after design time would be less than with today's code. SoA principles are somewhat independent of whether it is single-core or multi-core; it is the implementation that is different in these two cases. Another aspect related to the implementation of SoA is the topology of the architecture, i.e. if it is centralized, distributed or domain oriented.

Safe and dynamic updates

State of the art is that car manufacturers or vendors of the electronic control unit (ECU) need to pack one image, which will be flashed as a whole onto the ECU (Figure 23, left – static configuration of monolithic software). This image contains the complete SW for the ECU. In future, more applications (also from more different vendors) need to be integrated and due to the increased complexity of the SW (e.g. for autonomous driving) more frequent updates are expected. Always creating one static image will not be feasible in the future (e.g. because of the coordination of different vendors, testing and packaging efforts or the huge size of the image).

A potential alternative could consist in software download like in PCs or smartphones (Figure 23, center figure – dynamic resource allocation): During runtime, resources are dynamically allocated. This may lead to interference and the unavailability of required resources, which is not acceptable for safety critical applications in the automotive domain.

A suitable solution for mixed critical systems was developed in EMC² (Figure 23, right – dynamic resource allocation within predefined boundaries): A software package consisting of software and manifest is provided. The manifest describes the required resources of the SW. With the help of the information in the manifest, it can be checked whether the resource requirements can **always** be fulfilled for the safety critical applications. Safety uncritical applications can be executed as long as sufficient resources are available and get stopped if the resources are needed by safety critical applications.

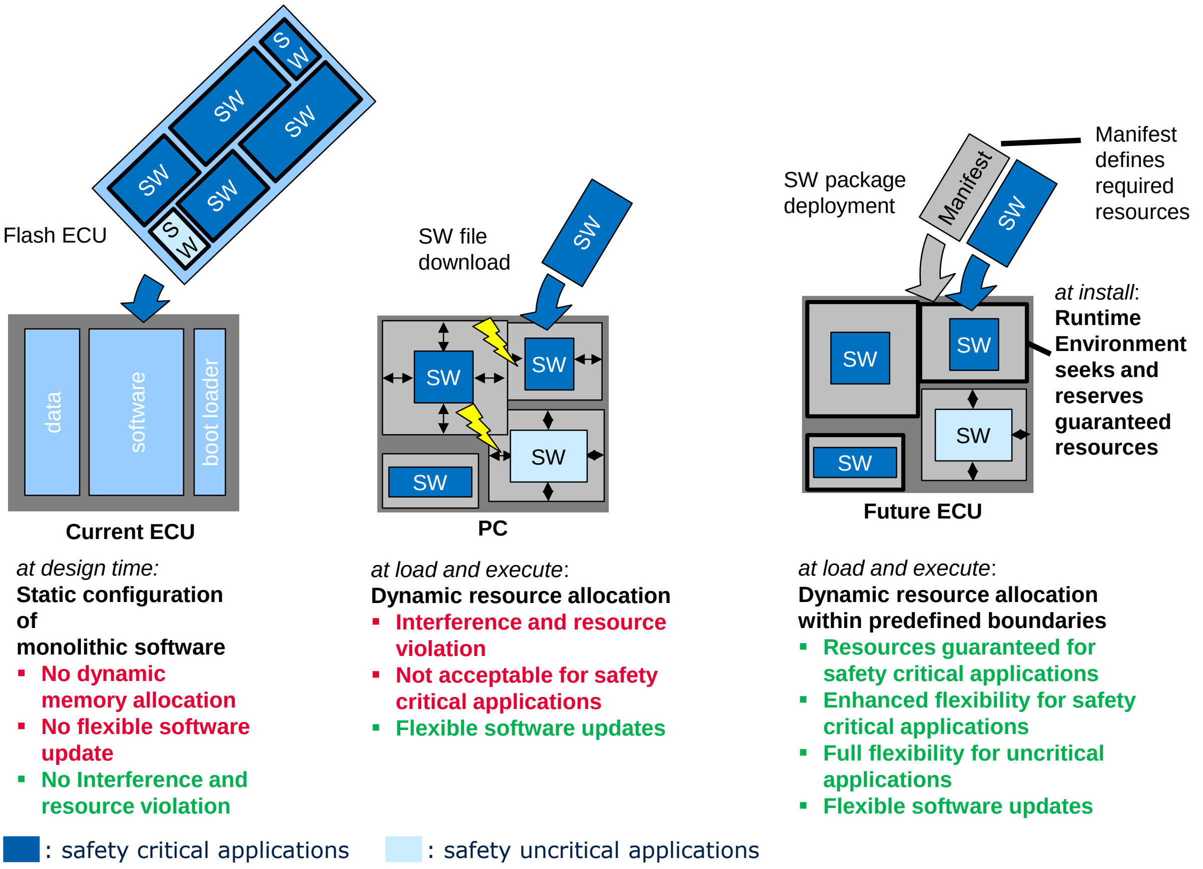
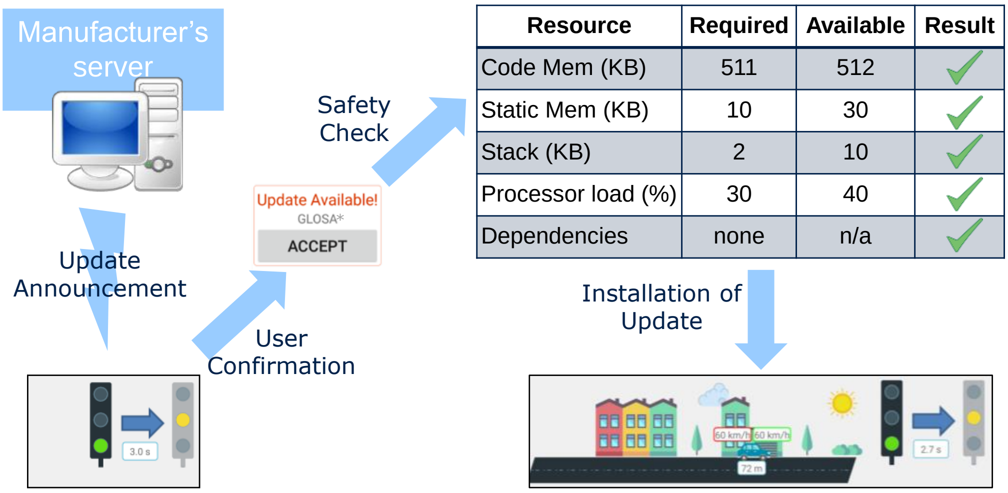

Flash ECU
SW
SW
SW
SW
SW
SW
SW
data
software
boot loader
Current ECU
at design time:
Static configuration of monolithic software
▪ No dynamic memory allocation
▪ No flexible software update
▪ No Interference and resource violation
SW file download
SW
SW
SW
SW
SW
PC
at load and execute:
Dynamic resource allocation
▪ Interference and resource violation
▪ Not acceptable for safety critical applications
▪ Flexible software updates
SW package deployment
Manifest
SW
Manifest defines required resources
at install:
Runtime Environment seeks and reserves guaranteed resources
SW
SW
SW
SW
Future ECU
at load and execute:
Dynamic resource allocation within predefined boundaries
▪ Resources guaranteed for safety critical applications
▪ Enhanced flexibility for safety critical applications
▪ Full flexibility for uncritical applications
▪ Flexible software updates
: safety critical applications
: safety uncritical applications

Figure 23: Safe and dynamic software updates of embedded mixed-criticality automotive systems (red: disadvantages / undesired properties, green: advantages / desired properties of the three approaches)

As an example, this principle was applied to the new advanced driver assistance system GLOSA (Green Light Optimized Speed Advisory). The intention is to make improved usage of traffic light information.

Assuming that the application on the car just receives (wirelessly) information about the phases of a traffic light, this information is presented to the driver and can be used by the driver to optimize his speed (e.g. because of the information about the next phase of the traffic light, see Figure 24, left bottom part). A server at the manufacturer announces the availability of a software update. In this case, it is assumed that the user is asked if he wants to receive the update (e.g. concerning extended functionality resulting in automated adaptation of vehicle speed). For safety critical updates, the updates may also be forced. Using the manifest in the SW package, a safety check is performed. In the right top of Figure 24 some parameters are exemplarily shown, which will be checked. If all checks are successfully passed, the installation of the update can be started. Subsequent to the update installation for Green Light Optimized Speed Advisory, based on the information about the phases of the traffic light, the optimal speed is calculated to pass the traffic light and the speed of the car is controlled accordingly.



Resource	Required	Available	Result
Code Mem (KB)	511	512	✓
Static Mem (KB)	10	30	✓
Stack (KB)	2	10	✓
Processor load (%)	30	40	✓
Dependencies	none	n/a	✓

Application on car shows information about next phase of traffic light.

Application on car calculates optimal speed to pass traffic light and controls speed of vehicle.

Figure 24: Illustration of the new software update concept in GLOSA application

3.2.2 Avionics applications

Multicore processors for avionics

Certification authorities for aviation applications have concerns on mixed-critical applications on multi-core processors. Therefore, one goal of EMC² was to implement a safety net for the multicore processors mitigating unforeseen or undesirable multicore processor operation using a software hypervisor that monitors the multicore processor. The monitor will decide if the multicore processor operates in normal conditions. In case of abnormal behavior of the multicore processor, the monitor can warn the pilot that the system is no longer operational.

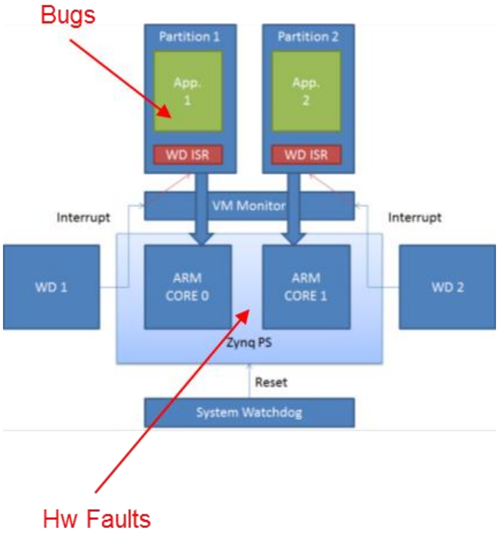


Figure 25: Monitoring multicore processors using software hypervisor

Thanks to the implementation of the hypervisor developed in EMC², WP3, the following results were achieved: There is no functional intrusion between the different partitions (Figure 25). The bounded temporal interference of one faulty partition on the other fault-free partitions is below 4 %. This means that the faulty partition has shown that after a fault, the time interference against the fault-free partition is less than 4 % of the execution time of the fault-free one. Short temporal interference is guaranteed from the hardware features around (or external to) the multicore processor and necessary in order to guarantee the quality of service.

New architecture features for multicore processors for avionics were implemented: First, the multicore avionic watchdog that is based on combined hardware and software (virtualization) has been implemented. Second, the software implementation for the above described timing interference detection has been done. The experimental setup is shown in Figure 26. The demonstrator consolidates four representative avionic applications (implementing some important parts of complete applications) with different design assurance levels (DALs) into different cores of a single multicore CPU.

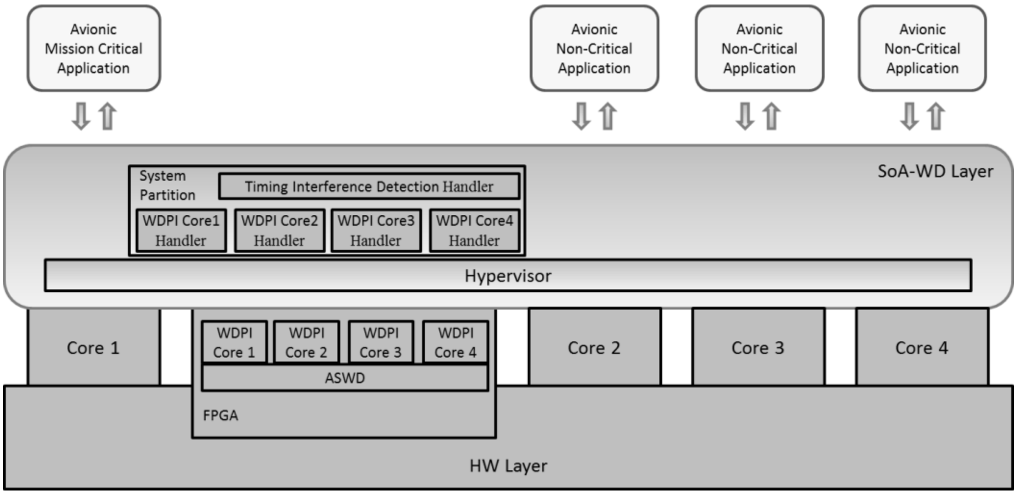


Figure 26: Final Quad-Core Demonstrator architecture

The four different applications (one mission-critical and three non-critical applications) with different DALs are running on different cores inside the multicore. We used a hypervisor to create four partitions in order to grant application isolation. Isolation consists in allocating each application to a dedicated set of resources (memory, I/O, processor core), which it uses exclusively and never shares with other applications. Each application runs in its own partition and has its own virtual memory space and its own channel to access peripherals. The hypervisor guarantees the isolation. It detects and stops any attempt of one application running in its own partition to access resources belonging to another partition.

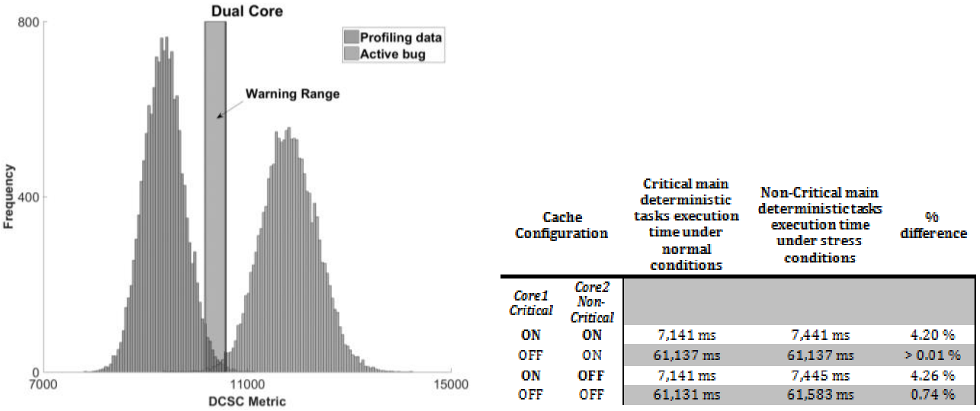
Moreover, the hypervisor manages the complex avionic watchdog developed in FPGA for mixed-criticality applications and the timing interference detection facility. The scope of the avionic watchdog is to reset the equipment when an unpredictable situation in the software leads to an incontrollable behavior. The simplest watchdog extension to the multicore is to provide separate single watchdogs - one for each core. This solution is not applicable for a mixed critical application because if the non-critical application continues to fail, the equipment keeps resetting, which causes a loss of the function of the critical application for a problem in a non-critical application.

The timing interference detection is a software implementation introduced into the SOA-WD Layer. It provides functions to cope with interference detection and interference recovery. These functions are related to the detection of the timing interference among the application placed on different cores. The scope is to preserve the safety critical equipment function from reducing the non-critical applications whenever the deterministic behavior of the critical application is likely to be highly disturbed by the not critical applications. The interference recovery software can change the scheduler to stop applications on the non-critical cores. This allows to manage a soft degradation of the equipment function (provided by the non-critical applications) keeping the functionality on the main core, which is executing the critical application.

The test campaign performed a verification of the hard real-time behavior of the critical application when a bug occurs in the other applications. There are three non-critical applications and at least one was modified to simulate a bug causing an infinite loop with continuous accesses to the shared memory hierarchy. The bug in the non-critical application was realized using the fault benchmark application.

Results on dual-core architecture

The first experiments were performed on the dual core architecture with a Xilinx Zynq-7000 device. In these experiments, two applications were used, one for each core (one critical application and one non-critical). The bug in the non-critical application was inoculated using the fault benchmark application. The graph reported in Figure 27 shows the measurements taken during these experiments, together with measurements taken during the profiling phase on the same device. The profiling phase is performed with the critical application alone.



Cache Configuration		Critical main deterministic tasks execution time under normal conditions	Non-Critical main deterministic tasks execution time under stress conditions	% difference
Core1 Critical	*Core2 Non-Critical*			
ON	ON	7,141 ms	7,441 ms	4.20 %
OFF	ON	61,137 ms	61,137 ms	> 0.01 %
ON	OFF	7,141 ms	7,445 ms	4.26 %
OFF	OFF	61,131 ms	61,583 ms	0.74 %

Figure 27: Experimental results on the dual core Xilinx Zynq-7000.

misbehavior detections	*Warning detections*	*Tolerated*
14859 (~99.06%)	2 (~0.01%)	139 (~0.93%)

Table 2: Detection results on the dual-core Xilinx Zynq-7000 architecture.

The graph shows the profiling data together with 15,000 measurements taken on the critical application alone and while the other application was affected by a bug using the fault benchmark application. The test result (Table 2) shows that the malfunction is detected in 99 % of the cases while a small number triggered detection by the warning rule. The third column of Table 2 counts the number of executions that did not trigger any of the two rules. These executions, labelled as tolerated, did not cause the metric value to change in a significant way. Thus, the temporal behavior of the critical application was not changed enough to require a recovery action.

Results on quad-core architecture

The system architecture was extended to a quad-core configuration with a watchdog entry for each core (Figure 28, left part). The same test is performed on the final demonstrator. Since there are three non-critical applications in the final demonstrator, there are three possible scenarios:

- Scenario 1: One of the non-critical applications is replaced by the fault benchmark application.
- Scenario 2: Two of the non-critical applications are replaced by the fault benchmark application.
- Scenario 3: All non-critical applications are replaced by the fault benchmark application.

The effects on the system change significantly in the different scenarios, as reported in Table 3 and in Figure 28 (right). The online timing interference detection mechanism provides the proper recovery actions on the system according to the design assurance level of the applications placed on different cores.

	misbehavior detections	*Warning detections*	*Tolerated*
Scenario 1	1668 (11.12%)	1114 (~7.43%)	12218 (~81.45%)
Scenario 2	11348 (~75.65%)	528 (3.52%)	3124 (~20.83%)
Scenario 3	14990 (~99.93%)	0	10 (~0.07%)

Table 3: Detection results on the Final Demonstrator (based on a NXP i.MX6Quad-core).

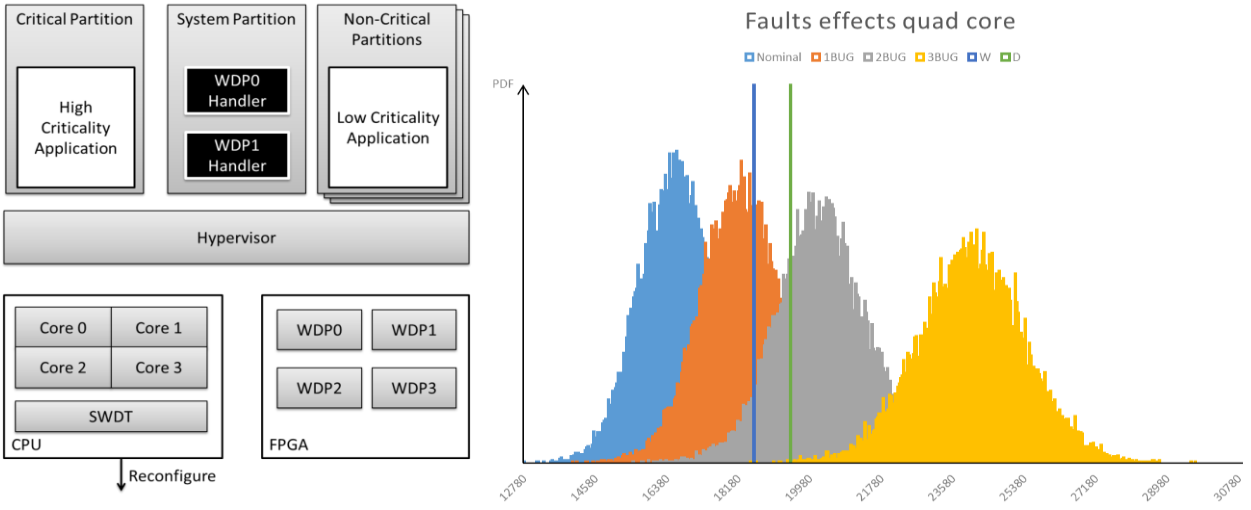


Figure 28: Experimental results on the Final Demonstrator (based on a NXP i.MX6Quad-core).

The graph shows the profiling data together with 15,000 measurements taken on the critical application alone and while other applications were affected by a bug. Figure 28 shows an overlap between the measurements gathered in the profiling phase and the measurements obtained in scenario 1. This is a hint at how the quad-core architecture is less sensitive to misbehavior in one core, due to different architectural choices performed in the quad-core processor with respect to the dual-core processor. Such architectural choices are protected by the intellectual properties (IP) of the MPSoC vendor, whose details are not open to the public.

Helicopter Terrain Awareness and Warning System (HTAWS)

The purpose of the EMC² demonstrator is to enable the use of the second core in avionics applications. The second core can run a low criticality application. The monitor processor will observe the activities of both processor cores. In case of a misbehavior of the low criticality core or reduced performance of the high criticality core, suitable countermeasures to guarantee correct system behavior will be performed. Depending on the observed misbehavior, the following countermeasures are supposable:

1. A short interruption of the low criticality application. This action can be a reaction on increased resource utilization of the low criticality application such that the high criticality application is affected (in terms of timing and execution performance). Halting the second core enables the first one to consume more shared resources.
2. A restart of the low criticality application. In case the low criticality application misbehaves seriously, the high criticality application must be protected from any kind of harm. A complete restart of the second application is a suitable way in this case.

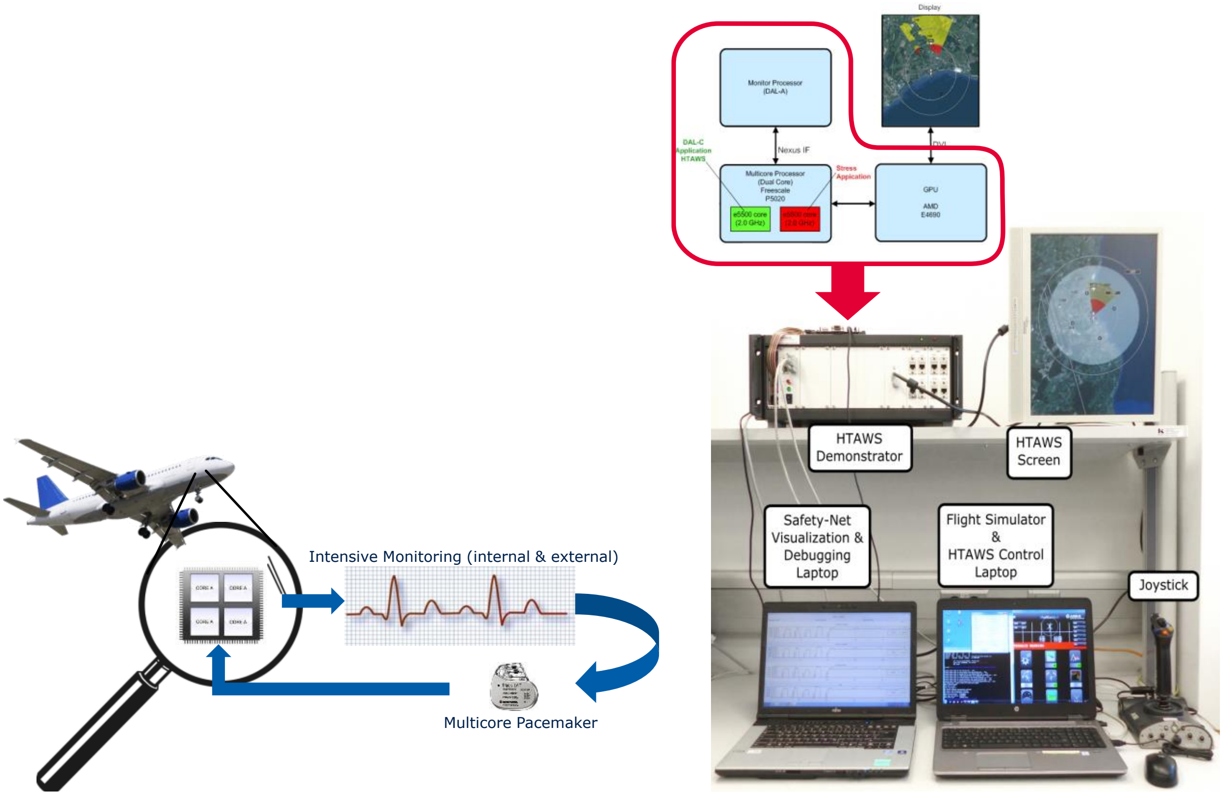


Figure 29: Multicore demonstrator for HTAWS use case (left: idea of processor monitoring; right: demonstrator setup)

Two monitoring and pacemaker approaches were implemented:

- External monitoring and control by a separate Safety-Net hardware for highly critical avionics systems identifies misbehavior of multicore software.
- Internal monitoring and control by enhanced system software support saves the extra hardware (costs, weight and space).

While the internal monitoring approaches are evaluated by using benchmark applications, the external monitoring is implemented together with a real avionic application as a demonstrator system. The application is a Helicopter Terrain Awareness and Warning System (HTAWS) available as single-core optional system for helicopters. It is rated as DAL-C, i.e. it must fulfill several requirements regarding certification. For the integration of such an existing software into a multicore system, a technology is necessary that can guarantee a certain core performance for one core without any modification of the already certified software. Our proposed *Fingerprint* technology - implemented within an external Safety-Net processor - provides exactly this feature. The basic idea and some evaluation results are presented in the following.

The HTAWS demonstrator computer (see Figure 29, right part) comprises a dual-core NXP T5020 computing platform, an FPGA-based implementation of the EMC² Safety-Net system, a GPU subsystem for graphics computing, and several other input/output devices (storage, networking). The original HTAWS application has been ported to the new platform and runs on one core of the NXP T5020 while the other one is available for concurrent applications. The Safety-Net monitors the HTAWS application's performance and adjusts or limits the concurrent memory accesses from the other core if necessary.

The used Freescale P5020 processor provides integrated performance and event counters. These counters can count for example the number of cache misses, the number of loads, the number of memory accesses and many other events. The idea of the application monitoring is based on these counters: At development time, a fingerprint of at least one application is taken, possibly from several counters. At runtime, the values of the same counters are compared to the fingerprint. In case the deviation is above a given threshold, an abnormal behavior is detected.

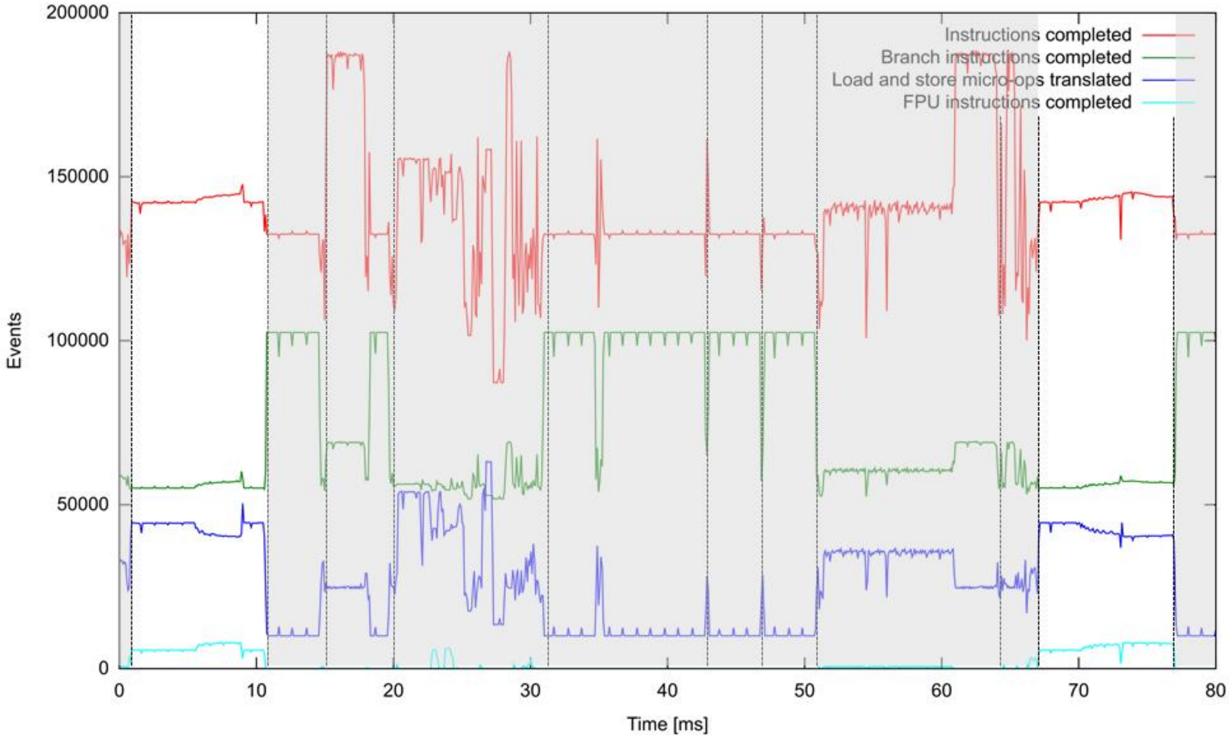


Figure 30: Measured event counter values of the HTAWS demonstrator application (white parts).

Example counter values used for generating and comparing the fingerprint are shown in Figure 30. These measurements originate from the HTWAS application that is implemented as an Integrated Modular Avionic (IMA) system. The white parts belong to the HTAWS application of interest. Evaluations show that it is possible to reliably detect a slow-down of an application starting from a 2.5 % as shown in Figure 31. If the application is thwarted less, it can be detected only in some specific cases.

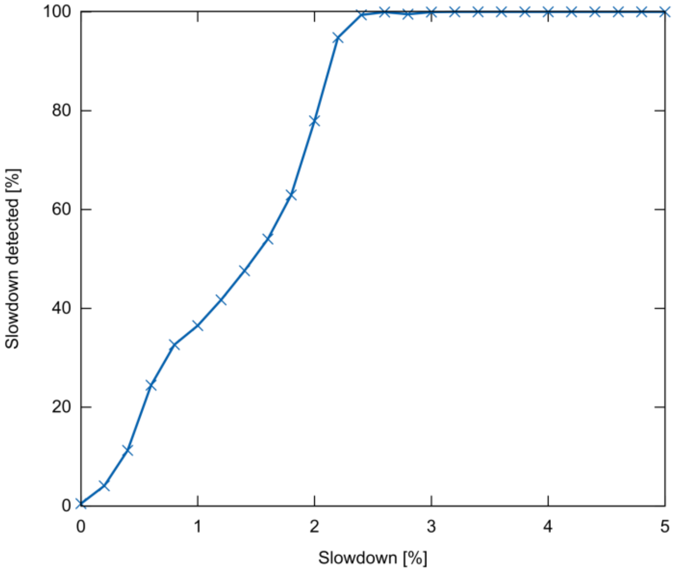


Figure 31: Slow-down detection rate

Figure 32 shows the average time required to detect a certain delay. It shows that a delay of 1 % needs on average about 44 ms to be detected (if it is detected at all according to Figure 31). The time required to detect a delay is improving significantly at about 2.5 % slow-down until it reaches the optimum value of 3 ms at a slow-down of 8.5 %. This 3 ms delay is currently the minimum possible value since this period is required for determining the execution path of the fingerprint, i.e. it is the delay of the fingerprint detection algorithm.

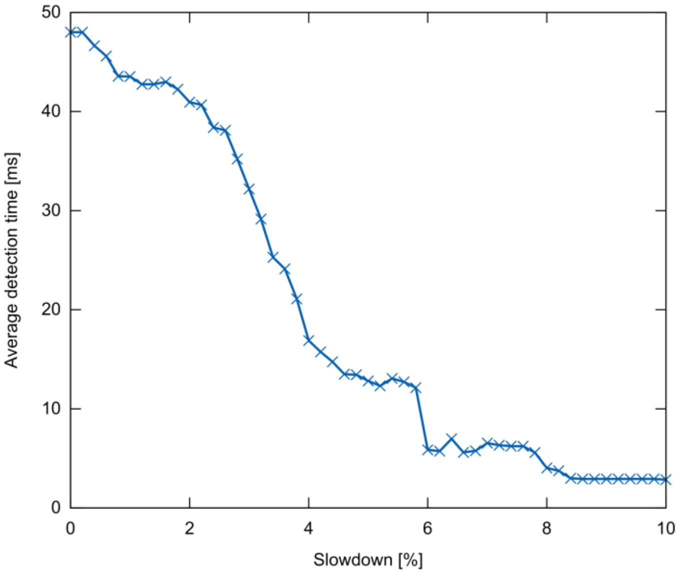


Figure 32: Slow-down detection delay

A self-assessment according to the latest position statement of Certification Authorities (CAST-32a) published in November 2016 has been performed. It turned out that the Safety-Net approach can cover up to 21 issues (out of 37 in the topic *The Planning and Setting of Multicore Resources*, *Interference Channels and Resource Usage*) directly, 8 need additional functionality and 8 cannot be detected at all.

3.2.3 **Space applications**

MPSoC hardware for space

A heterogeneous time-triggered architecture was implemented on a hybrid MPSoC (Multiprocessor System on Chip) platform. One focus in the last project period laid on the evaluation of the integration capabilities of the heterogeneous many-core architecture implemented on top of a hybrid SoC platform. In particular, for the purposes of the project an Arria V SoC development board (see Figure 33) was used. The three basic components of a hybrid SoC platform are a hard processing subsystem, FPGA and interconnection. We identified a number of ways how to utilize advantages of such an architecture in mixed-criticality environments.

An arbitrary version of the architecture includes five many-core components designated as µComponents, four of them are based on Nios 2 processing units and a single ARM component that occupies both HPS and FPGA. Figure 34 provides an overview of the current layout and its integral parts. Each component was built using a generic template, which can be configured according to the needs of an application.

A major challenge for such an architecture and its usability is the tool integration and the ability to automate the design process. Thus, one of the goals in the project was to explore this topic and identify the possibilities for automated tools that integrate the whole process from hardware design to the application. This includes:

- A component design with implementation of the processing unit and its properties (e.g. type of the core, frequency, interrupts, memory management etc.), memory hierarchy and peripherals
- TTNoC configuration, routing tables, global time settings and external synchronization

- Application mapping and communication schedules

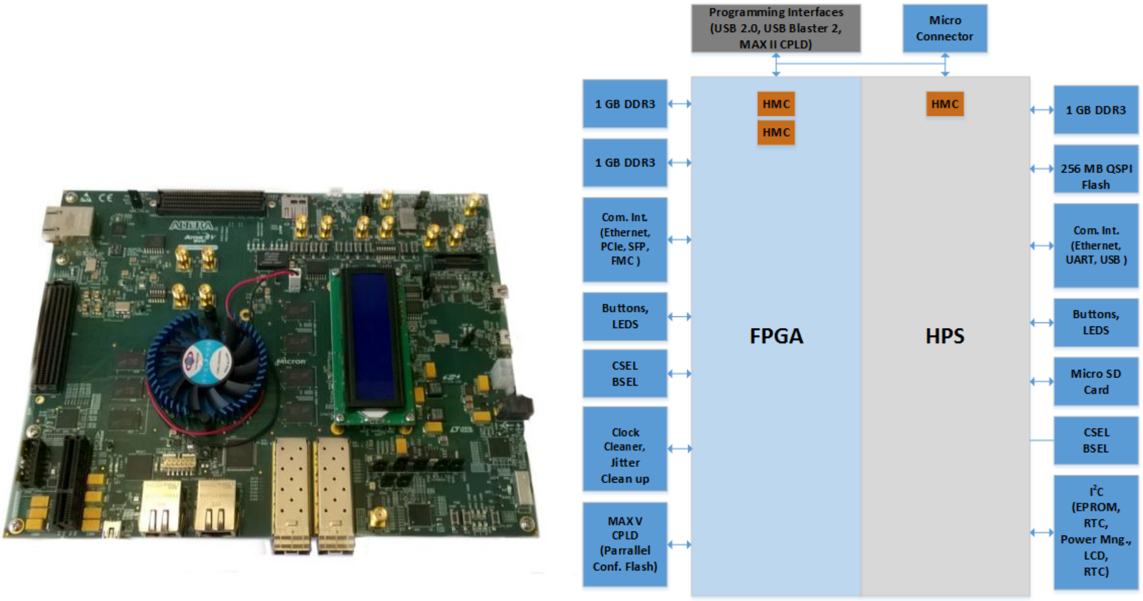

Programming Interfaces (USB 2.0, USB Blaster 2, MAX II CPLD)
Micro Connector
1 GB DDR3
1 GB DDR3
Com. Int. (Ethernet, PCIe, SFP, FMC)
Buttons, LEDS
CSEL BSEL
Clock Cleaner, Jitter Clean up
MAX V CPLD (Parallel Conf. Flash)
HMC
HMC
HMC
FPGA
HPS
1 GB DDR3
256 MB QSPI Flash
Com. Int. (Ethernet, UART, USB)
Buttons, LEDS
Micro SD Card
CSEL BSEL
I²C (EPROM, RTC, Power Mng., LCD, RTC)

Figure 33: Arria V SoC Development Board and Block Diagram of the Architecture

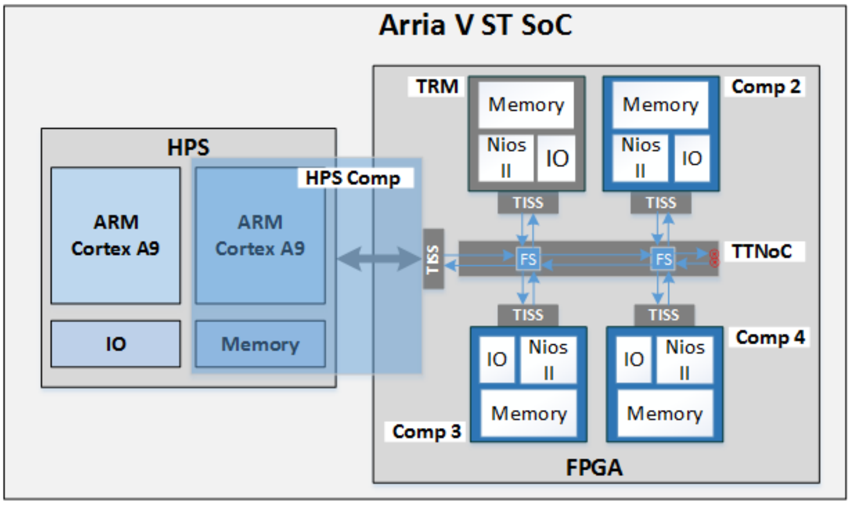


Figure 34: Heterogeneous TT MPSoC architecture

Other important aspects considered the fault tolerance capabilities of the platform and the synergy with the TTNoC architecture. Some of the important aspects are the reconfiguration capabilities on the platform level, e.g. the ability to reprogram or reconfigure HPS and FPGA in case of fault or an error occurrence. The simple architecture presented above uses very few resources on the FPGA. The implementation presented in Figure 34 uses about 12 % of the FPGA logic elements, 22 % of the block memory bits and 12 % of the pins. This platform is capable of hosting a much larger architecture, which can be implemented on a cheaper lower end platform as well.

Key achievements of this work are given by the defined hardware architecture, in particular the successful use of Time-Triggered Network-on-Chip (TTNoC), the trusted interface subsystem and the trusted resource manager. The above-mentioned properties could be highly beneficial for space applications.

Payload applications for space

The objective of this use case was to evaluate the use of the MPSoC image processor based on CCSDS (Consultative Committee for Space Data Systems) 122 & 352 regulations. The work performed focused on the evaluation of the impact that different SW optimizations preformed in sequential C code will have in a certain HW platform, especially when transforming sequential code into concurrent code capable of taking advantage of multi-core platforms. During these experiments, the target platform modelled has been a mono- and dual-core LEON3-based platform running on a XILINX ML506 FGPA-based platform.

The fact that an FPGA-based platform has been selected could be weird since one of the main goals of the tool is to model the behavior of platforms before its physical availability. However, this board has been used in order to enable the comparison between the performance results obtained by the simulator and the results obtained from real executions. Considering that the number of clock cycles required by the HW does not typically change when selecting a different technology, once the simulator has been validated against an FPGA-based implementation, the extrapolation of the results to other kinds of implementation of the same VHDL code is simple. It can be easily done using the simulator, just by adapting the clock frequency, and other numbers, such as the delay of the external SDRAM chips.

To check the tool, two SW applications have been selected: a CCSDS 122 coder, and an AES coder (Figure 35). Both applications have been used to evaluate how the SW parallelization behaves in a multi-core platform. For such purpose, they have been first evaluated running the initial, sequential version on a mono-core platform, under a Linux build OS. Then, the simulator has evaluated the impact of the modifications done in the SW code to enable concurrency, and the effect of using a dual-core platform. The expected behavior of other configurations with more cores has also been simulated, but not verified in the board, since its FGPA does not enable the implementation of more cores.

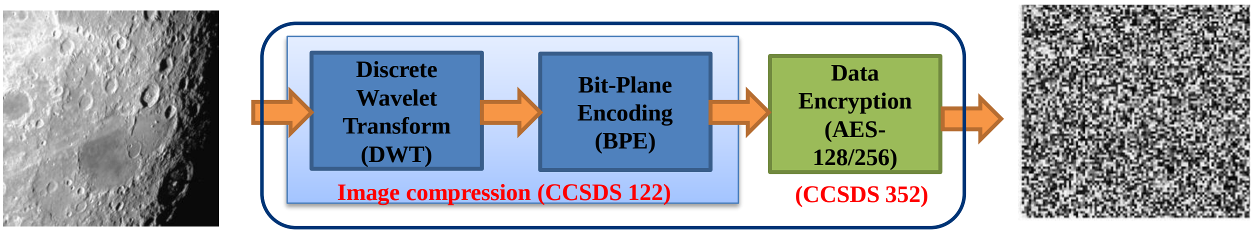

Discrete Wavelet Transform (DWT)
Bit-Plane Encoding (BPE)
Data Encryption (AES-128/256)
Image compression (CCSDS 122)
(CCSDS 352)

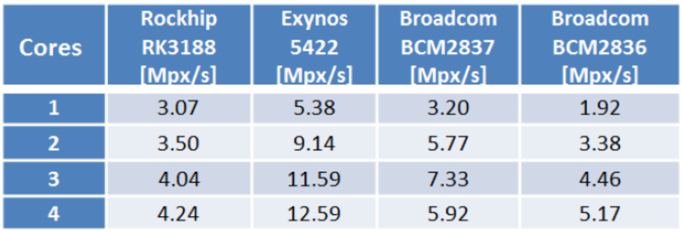
Figure 35: Payload application for space

The implementation of the CCSDS 352 standard for data encryption was based on the OpenMP[5] paradigm. The partial implementation of the CCSDS 122 standard for image compression was based on the OpenMP paradigm Discrete Wavelet Transform (DWT). Validation took place on a multicore architecture (ARM quad-core processor).

In order to test the parallel version of CCSDS 122.0-B-1 and CCSDS352-B-1, a set of ARM multicore hardware platforms has been used. Three platforms use a quad-core processor (RK3188, BCM2837 & BCM2836) and the other one uses an octa-core Exynos-5422. However, the octa-core is based on a Heterogeneous Multi-Processing (HMP) solution. Therefore, only the most powerful processors (quad-core Cortex-A15 @2GHz) were used in order to compare it with the previous processors.

There are minor differences when the speedup is computed with three image sizes (1000x1000, 2048x2048 and 4000x4000). The throughput improvement is summarized in the next table for an image size of 4000x4000 pixels.

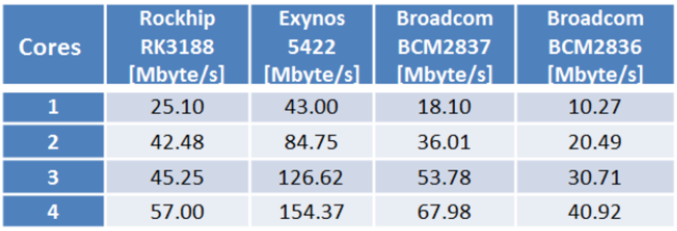
[5] Open Multi-Processing, an (API) that supports multi-platform shared memory multiprocessing programming



Cores	Rockhip RK3188 [Mpx/s]	Exynos 5422 [Mpx/s]	Broadcom BCM2837 [Mpx/s]	Broadcom BCM2836 [Mpx/s]
1	3.07	5.38	3.20	1.92
2	3.50	9.14	5.77	3.38
3	4.04	11.59	7.33	4.46
4	4.24	12.59	5.92	5.17

Table 4: Discrete wavelet transform (DWT) and bit plane encoding (BPE): Throughput (Mpixel/s) with a 4000x4000 image (8bit)

In a similar way, the speed improvements of data encryption according to AES CTR mode has been computed with a 64 Mbyte data chunk. The AES is a well-suited algorithm for parallelization as can be seen in next figure where the scalability is close to be linear. The scalability variation between processors for AES (linear) and DWT+BPE (nonlinear) reveals the dependency between the algorithm and the hardware architecture.



Cores	Rockhip RK3188 [Mbyte/s]	Exynos 5422 [Mbyte/s]	Broadcom BCM2837 [Mbyte/s]	Broadcom BCM2836 [Mbyte/s]
1	25.10	43.00	18.10	10.27
2	42.48	84.75	36.01	20.49
3	45.25	126.62	53.78	30.71
4	57.00	154.37	67.98	40.92

Table 5: Data encryption (AES CTR): Throughput (Mbyte/s) with a 64Mbyte data chunk

Platform application

Multi-core architectures will be adopted in the next generations of avionics and aerospace systems. With the advent of Integrated Modular Avionics (IMA), these systems progressively became mixed-criticality systems, and spatial and temporal isolation is thus a key requirement. In such a context, virtualization appears to be a promising technique to implement robust software architectures in multi-core avionics platforms. Hence, this work explores the possibility of using virtualization in the definition of a multi-core platform for the aerospace domain. This is a first attempt to exploit para-virtualization on Cobham-Gaisler LEON4, the European Space Agency next generation microprocessor.

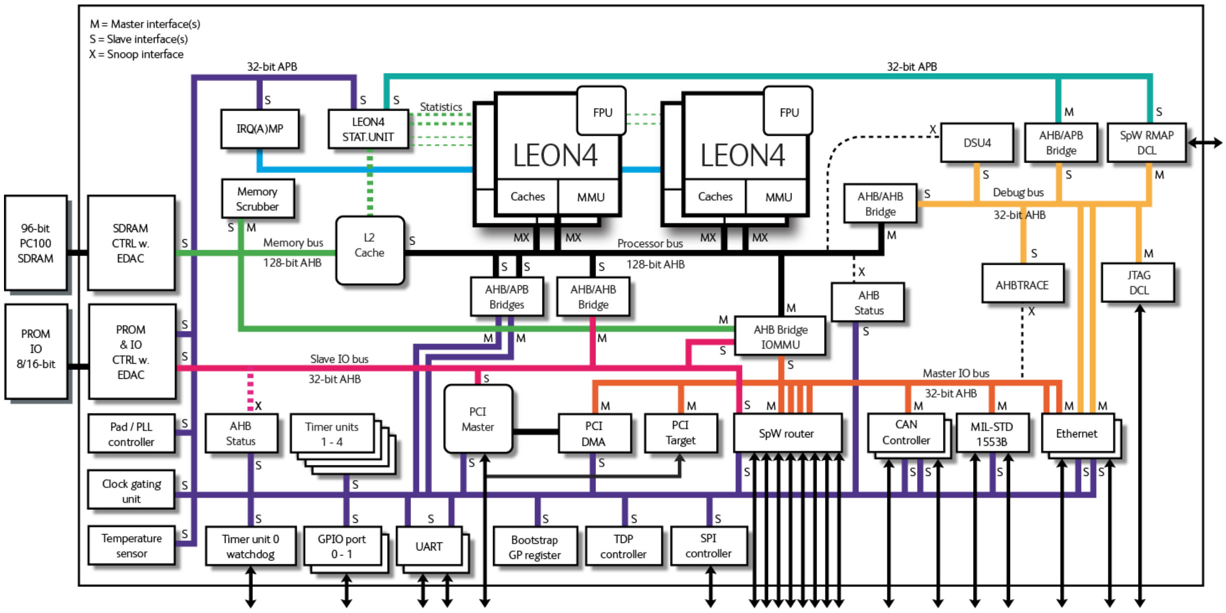


Figure 36: Next generation European Space Agency microprocessor (Cobham-Gaisler LEON 4)

Multicore processors provide better cost and power dissipation than single-core solutions of comparable computing power. Moreover, most of the commercially available solutions are multi-core, pushing a niche area as Space Industry to adopt such applications. Drawbacks of multi-core solution in hard real time applications include programming difficulty (e.g. algorithm parallelization) and minor predictability and determinism (due to contention of shared resources). The task has provided valuable evidence that multicore drawbacks in hard real time applications may be effectively tackled providing also in this area the benefit of multicore solutions.

Based on LEON4 platform system (e.g. LEON4-RASTA) a prototypal demonstrator representative of a simplified satellite platform has been implemented in order to evaluate the performance of the proposed approach.

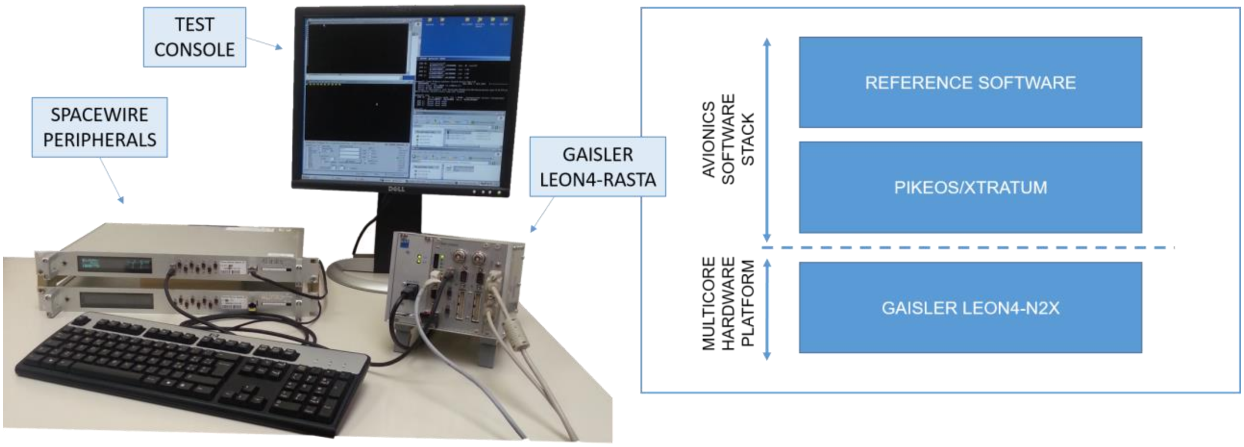


Figure 37: Simplified satellite platform

The designed system aims to model the satellite’s telecommand and telemetry function, the management of peripheral devices, the management of large file transfers as well as the execution of legacy code related to typical spacecraft device (e.g. Star Tracker). The platform management presents specific requirements of reliability, robustness, as well as various levels of criticality and dependability both at software and hardware level. The use of hypervisors and para-virtualization techniques is considered as a feasible way to deal with these requirements.

Two different hypervisors, SYSGO PikeOS and FentISS XtratuM, have been considered for the implementation of the reference application on selected multicore platform. Both hypervisors are interesting solutions for the aerospace domain and ensure time and space partitioning on multicore systems with mixed criticality applications. SYSGO PikeOS provides many features useful to perform the multithreading programming. Moreover, it ensures good software stability. FentISS XtratuM provides native primitives having very good performances. Both data exchange and device access are very fast and memory access as well.

3.2.4 **Industrial Manufacturing and Logistics**

Quality control by 3D inspection

The inspection system reconstructs the 3D shape of a captured object as well as its corresponding texture. The aim of this use case was to take advantage of multicore platforms and the tools provided within the EMC^2 project (“art2kitekt”) to obtain a highly parallel and scalable version of the inspection system. For that purpose, sequential and parallel models for the task of 3D object reconstruction have been compared. Object reconstruction was used to distinguish different objects and to find surface defects based on texture comparison (see Figure 38).

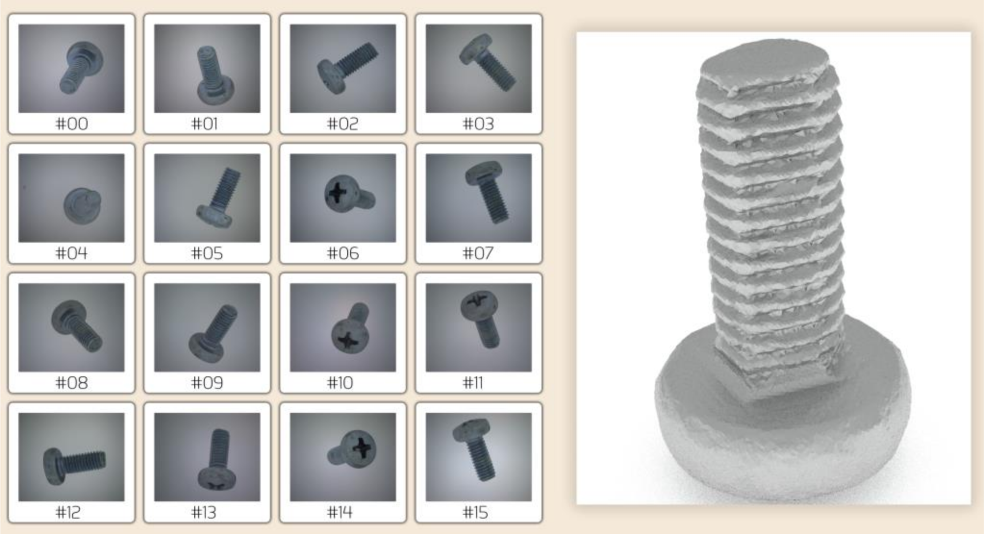


Figure 38: Quality control by 3D inspection

A battery of experiments was run in order to confirm how throughput linearly increases with a growing number of available computing nodes as it is expected. At continuation, a table with a summary of the devised experiments is shown. Computing nodes have 4 cores. The available hardware has the following characteristics and provides 32 cores:

- Processors: 2
- Cores (at each processor): 8
- Multithreading capabilities: yes

Number of nodes	Throughput
1	6 objects/minute
2	11 objects/minute
4	22 objects/minute
8	45 objects/minute

Table 6: Increase of inspection throughput with number of computing nodes

As our experiments show, for one computing node a throughput of almost 6 objects per minute has been achieved. The system took around 293 seconds to perform 28 3D reconstructions. The reconstruction process includes loading cameras number and calibration, reading images from disk, remove lens-object distortion from images, segment silhouette, octree computing, surface marching cubes computing as well as centroid and alignment.

The objective was to exploit coarse-grained parallelism using multicores to implement manufacturing quality control using machine vision. The key achievement of the 3D inspection at project end is given by an increased overall inspection performance by 300 %. It was demonstrated that it is possible to enable algorithm parallelism for the 3D reconstruction software. OpenMP was applied to some functions of the software and an evaluation of the performance for intensive computation tasks on a variety of execution platforms was presented and compared. With OpenMP parallelization and an execution platform composed of 2 processors, 16 cores and multithreading capabilities, a reduction of computation time from 24.563 milliseconds to 7.996 milliseconds was achieved by exploiting coarse parallelism and thus decreasing latency. Furthermore, inter-node parallelism was achieved by fairly dispatching full captures (32 images + 3D calibration) to different computing nodes. Each computing node performs a 3D reconstruction and some 3D inspections indicating if the analyzed object is correct or not. In this way, throughput of the inspection system raises.

3.2.5 Internet of Things

Web-based multimedia communications

This use case addressed a large-scale application of Unified Communication Services using HTML5 based Web Browsers on Embedded Systems. A device of the family of Android TVs was selected to act as one of the web points with the possibility to add multimedia peripherals like camera and microphone. The application is a peer-to-peer videoconferencing solution between two or more devices.

The result of this use case is a demonstrator of unified communication services where media sources have been distributed in different processes obtaining some improvements in terms of priority management, bandwidth use and timing. It is worth mentioning that some problems have been found for some resources due to the browser capabilities. The resolution of these incidents is not in the scope of this project, but its resolution in the future would enable the architecture implemented in this task for a wide set of use cases.

Open Deterministic Networks - Networked Smart Vision System

Networked smart vision systems are becoming part of future cyber-physical systems of surveillance, airport facilitation or automated driving applications. The objective of this use case was the reliable exchange of meta-data (important information) to solve complex vision tasks (e.g. passenger tracking) by utilizing deterministic network capabilities.

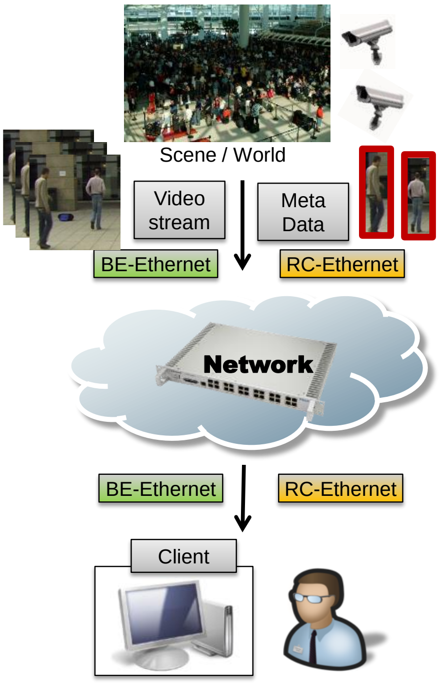


Figure 39: Networked smart vision system (RC: rate constrained; BE: best effort)

A demonstrator was developed and built to overcome the following problems: Video surveillance applications (as well as applications from other domains) face the challenge of limited bandwidth so that important meta-data extracted by computer vision tasks can be lost in case of a network overload. Moreover, information sent via network may not arrive on time. For this purpose, the event data and video streams have their own communication links, which provide certain guarantees to the associated applications. In this way, the surveillance application can provide service guarantee behavior depending on the particular communication link. The communication links are composed of the open deterministic (TT-ES, Time Triggered Ethernet) and the best effort network traffic (QoS-ES, Quality of Service). The latter link allows controlling the video stream by quality of service (QoS). Hence, if the available bandwidth decreases, the quality of the video also decreases, e.g. the frame rate drops. At the same time, information of high importance or real-time constraints, e.g. informing other cameras and clients about a person, can be sent over the TT-ES link. This link ensures that the messages arrive even if the network is in overload situations.

The experimental set-up consists of multiple visual sensors and at least one client. TT-Ethernet Switches are used to connect the sensors as well as one client. One sensor system, however, is only connected via a QoS-ES link to the network. In order to simulate network overload situations, a noise generator is connected to the network. In overload situations it can be demonstrated that messages over TT-ES are still reliably delivered (packet loss is prevented) whereas the video streams and messages via the QoS-ES link may have data losses, so that events may never reach the client. With TT-ES all packets arrive on time, fulfilling nearly real-time constraints.

Synchronized low-latency deterministic networks

Objective of this use case was to provide synchronization capabilities to distributed multi-core systems using dependable low-latency networks with high-accuracy time synchronization utilizing standard protocols (IEEE 1588) in a smart grid. The demonstrator setup is shown in the figure below.

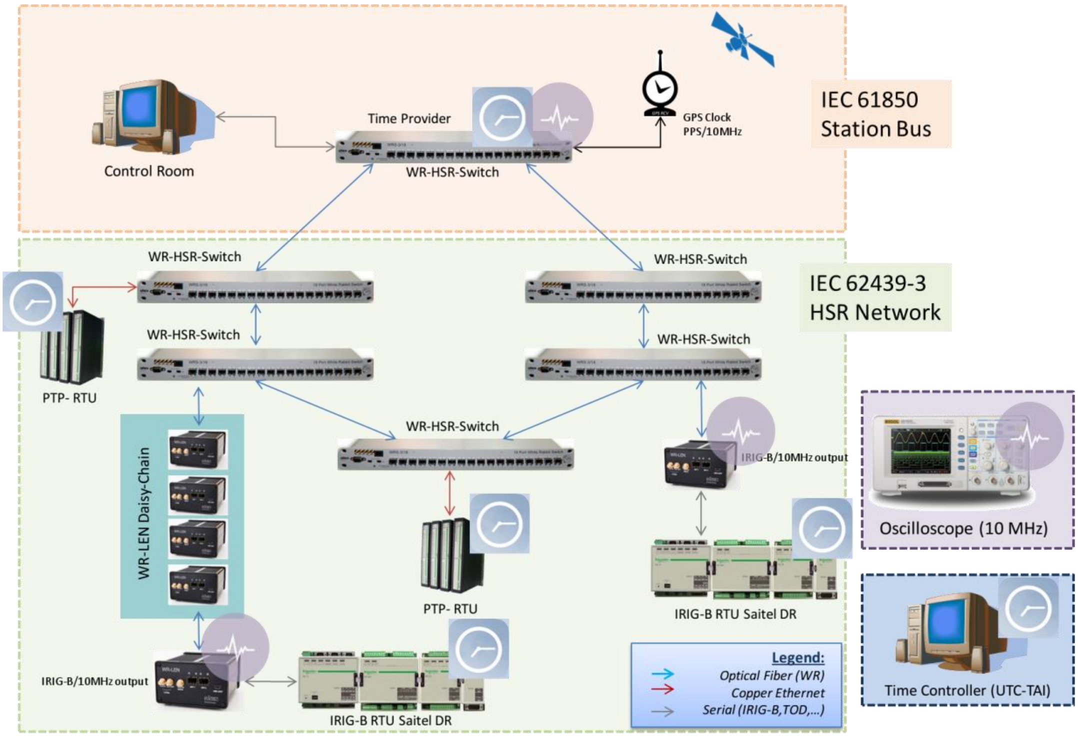


Figure 40: Low latency deterministic networks – demonstrator setup

Based on this demonstrator the following results were achieved:

The synchronization of devices is feasible with an accuracy of less than 1 nanosecond. This is possible thanks to the technology used in the devices, which is called White Rabbit. Once the devices are synchronized, their oscillators and internal clocks and thus, time, differs by less than one nanosecond. In addition to this, it is possible to synchronize up to 14 nodes in a cascade (daisy chain) configuration maintaining the sub-nanosecond accuracy in all nodes. Timing devices can now work as a boundary clock, transparent clock and hybrid clock.

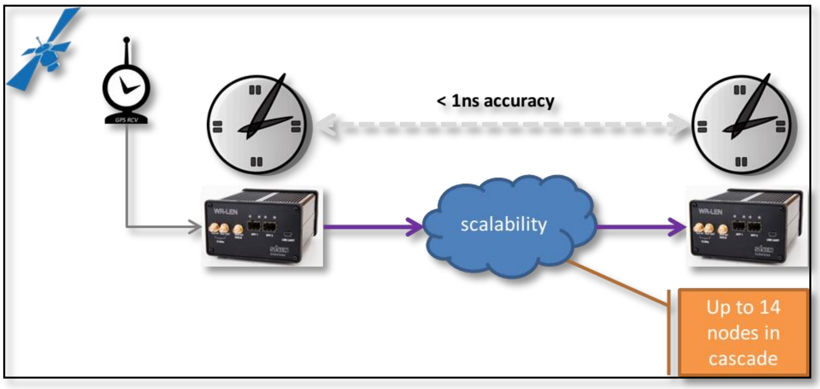


Figure 41: Low latency deterministic networks – synchronization accuracy of less than 1 nanosecond

A boundary clock is a device that is slave on at least one of its ports from which it takes the timing reference, and master in the rest of the ports, giving time to the nodes connected to its master ports. A transparent clock is a device that is neither master nor slave, it only forwards the timing frames from the master of the network to the rest of the nodes. This is commonly used in the industrial domain (Smart Grid). It is important to remark that a transparent clock does not synchronize to the master. A hybrid clock is a device that, by the same way transparent clocks do, forwards the timing frames from the master to the rest of the network, but contrary to transparent clocks, hybrid clocks do synchronize also to the master reference.

The benefits of using transparent and hybrid clocks are threefold:

- Best recovery of the network in case of failure. The timing network recovers faster than using boundary clocks because they are not topology dependent.
- Common use in the industrial domain. Thus, the technology is compatible with that domain.
- Less jittery and more stable synchronization.

The devices implement three different kinds of timing protocols to improve interoperability for Smart Grid. This means that we are able to disseminate time through a network using different mechanisms at the same time, which is also important for the industrial domain, since not all devices implement standard PTP (they use IRIG-B). The devices implement:

- White Rabbit (1 nanosecond accuracy)
- PTPv2 IEEE 1588 (standard timing reference)
- IRIG-B (less accurate, but industrial)

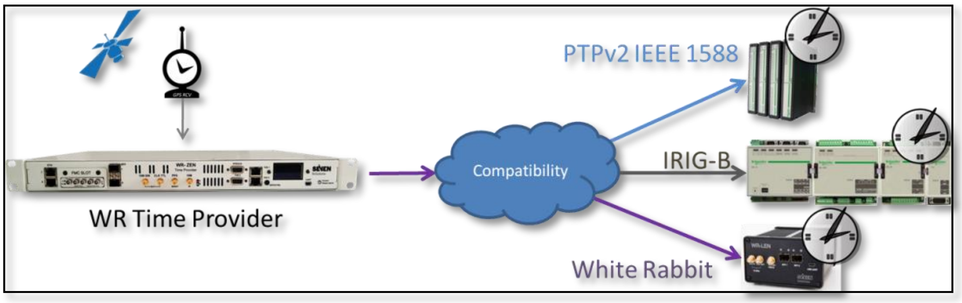


Figure 42: Low latency deterministic networks – 3 protocols can be used in parallel

The demonstrator includes a couple of redundancy features and single point of failure avoidance for both timing and data:

- A protocol called HSR (High-seamless Redundancy Protocol) has been developed which is used to guarantee single point of failure avoidance by duplicating all the frames and using a ring network topology.
- Frames are duplicated and sent on the two ports connected to the ring, so that if one node fails (it means that one path is down), the data will be received on the other path of the ring.

- This has been done for both timing and data frames. Each node of the ring receives the same time reference on both ports, so first it synchronizes to one reference while the second one is a backup. In case of failure, it is possible to switch to the backup timing reference, and remain synchronized in ZERO-time.
- The oscilloscope presents 4 Pulse Per Second (PPS) signals with a resolution of 5 ns. This shows how the PPSs are completely aligned even after the primary time reference is lost.

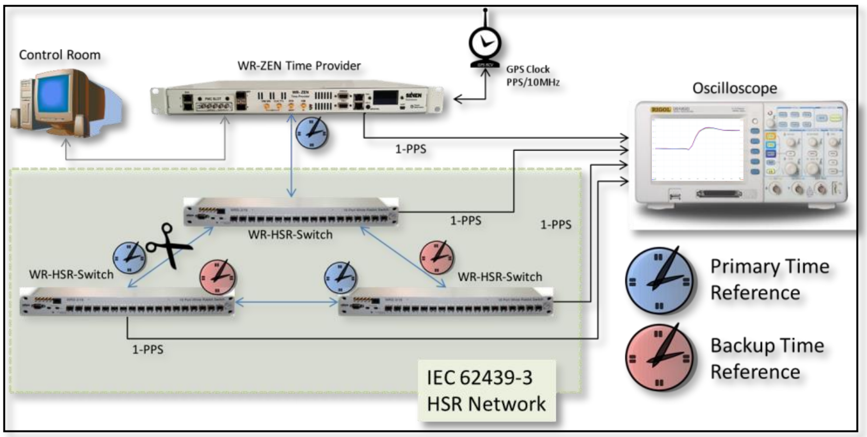


Figure 43: Low latency deterministic networks – redundancy features for single point of failure avoidance

Use case Ultra low power high data rate communication

The goal of this ultra-low power high data rate project is to realize a high data rate wireless SoC (System on Silicon) which combines re-configurability with very ambitious performance goals. The demonstrator realizes both a BTLE (Bluetooth Low Energy) and 802.15.4 functionality demonstrating that the architecture is able to support multiple standards by modifying the software. The architecture will also allow the evolution into a WLAN (Wireless Local Area Network) SoC. The demonstrator exhibits breakthrough performance in terms of power consumption and link budget. Link budget defines the distance over which a radio link can communicate. In the EMC^2 project, the solution is demonstrated on an FPGA based platform.

Summary of the work done and the achieved results:

(1) An architecture has been developed which was demonstrated to result in a breakthrough link budget for a BTLE and 802.15.4 demodulator. During the last year, BlueICe used this architecture for other wireless packet oriented communication standards, e.g. DVBS2X satellite communication. Also here the architecture confirmed its breakthrough link budget.

(2) The second goal was to achieve breakthrough power consumption. Also this goal has been achieved, even if typical processor based architectures tend to suffer on this solution parameter. Some highlights are:

- A power consumption of less than 300 µA has been shown for the PHY part of the platform (the demodulator). This is referenced to the 55 nm TSMC technology.
- Sleep power of less than 0.5 µA has been demonstrated. This is possible by the fact that in SLEEP mode only a small fraction of the platform SRAM needs to be supplied. Power supply of all other parts of the system can be removed.
- Active processor power of the BLUSP core is ~50 % of comparable existing architectures. Moreover, the performance of the processor allows it to operate at a very low frequency.

3.2.6 Cross domain applications

Seismic processing

The image below indicates the configuration of a seismic towing configuration. Typically, 8-14 streamers are towed behind the ship. Streamer lengths vary between 10 km and 14 km. Each streamer includes about 200.000 sensors. To acquire and process the sensor data, each streamer requires 100 to 200 computers. Roughly 300 Mbit per second of data per streamer need to be processed in real-time on sea and stored in the seismic volume. Further seismic processing is done on the ship, and subsequently on land.

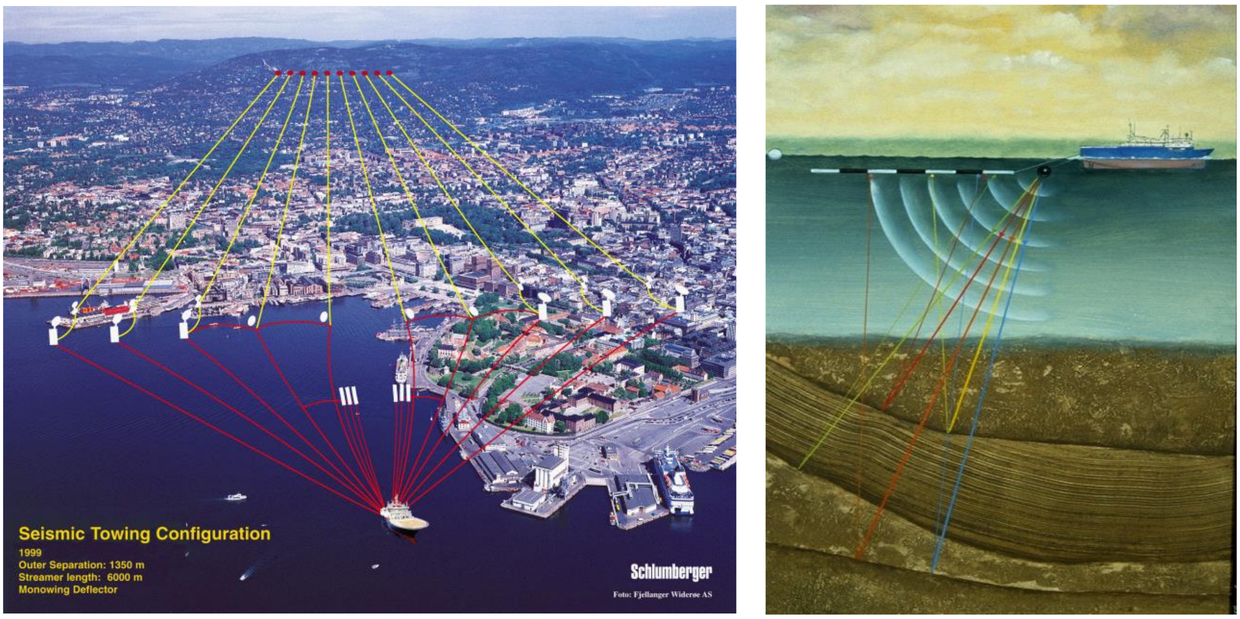


Figure 44: Seismic towing configuration and processing requirements

WesternGeco/Schlumberger researchers that develop new algorithms most often use MATLAB for their work. Transforming the results of the researchers work into industrialized and commercial products is a major effort executed by our engineering centers. The main goal of our EMC^2 activities was to simplify and accelerate this process by automating the translation process from MATLAB to C++ and at the same time generate code that is multi-core aware. This gives the additional benefit that the code runs much faster. Goal in EMC² was the generation of C++ code that runs five times faster than Matlab code.

Since performance is very important, it is handled as a white-box approach, i.e. a need for further manual optimization after the initial code generation is assumed. Therefore, it was decided to use the Armadillo C++ template library as a platform for the generated code. Since Armadillo syntax is quite close to MATLAB, it will ease the understanding of the generated code, and therefore make it more easy to continue manual optimization.

Work performed in the project leads to two key achievements: First, the targeted reduced execution time has been achieved: Automatically generated multi-core C++ code runs 2 to 60 times faster than serial MATLAB code. A prototype example, with a couple of hours of additional hand optimization, did run 250 times as fast. Reduced execution time translates into reduced costs for seismic processing. Second, engineering time could be reduced as well: New algorithms exploiting multi-cores can be implemented much faster. Automatically transforming MATLAB code into C++ code is estimated to take less than 10 % of the efforts previously used.

Video surveillance for critical infrastructure

Video applications are entering into more and more markets such as surveillance, medical applications, automated driving, quality control in production, automatic access control, etc. The objective of this use case consisted in the acceleration of an object (i.e. face) detection algorithm by using multi-core or FPGA architectures.

Consumer applications are developed in huge quantities and manufacturers of those systems can use technologies like Application Specific Circuits that are not affordable for markets, which require highly specialized solutions. These solutions cannot afford specific chip developments and have to build systems using standard components like processors or FPGA devices.

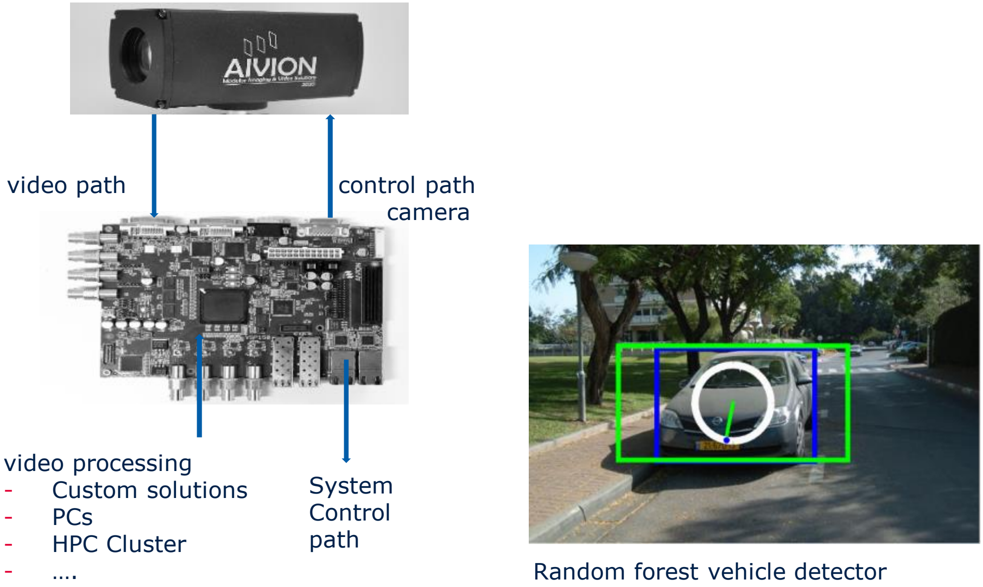


Figure 45: Video surveillance application

Within this use case, object detectors (e.g. face / license plate detection) were implemented in Xilinx Zynq (Figure 45, left part). Experiments with high dynamic range detection of license plates were performed. Further experiments considered random forest vehicles (Figure 45, right part).

Smart camera object recognition targeted the leverage computation performance on small platforms, the reduction of time-to-market for algorithm adaptation and development as well as the reduction of power consumption and therefore of heat emission. To achieve these goals, we developed and applied two particular technologies: High level synthesis with Xilinx Vivado tool chain and Zynq FPGA was implemented on the EMC^2 video board (developed by Sundance in WP4). The detection algorithm is a heavily modified version of the Viola & Jones algorithm now suitable for parallel execution. It is based on “Local Binary Patterns” (LBP) and “Local Rank Differences” (LRD).

With these technological advances, we achieved significant progress:

- In terms of performance, a gain of a factor of 60 was achieved. Computation time to process 1280x720 pixel images on a laptop, i5 2.8 GHz clock requires 3 seconds. In comparison, our solution is almost real-time (computation time ~20 ms).
- The algorithm adaption is now faster by almost a factor of 10.
- The C based algorithm development requires 80 % less effort in comparison to classical HDL (VHDL, etc.) approaches, once the board support package is available.
- Finally, the adaption within the C based algorithm did reduce the latency time from ~10 seconds down to less than a second within the current demonstrator.

Use Case summary:

- Technology has developed extremely fast during the project, and task T12.2 could use tools that did not even exist in the beginning of the project.
- The original goal to provide a demonstrator for smart camera applications has been achieved.
- With the methodology and the development platform that has been used in the project, it is possible to create smart video analytic systems from an abstract C level.
- This allows fast development cycles *and* affordable product architectures.

Medical imaging

In the health care domain, very large images need to be processed. Complex image processing is necessary to keep all image information available, while at the same time increasing the information level of the images. As very complex and configurable algorithms are in use, it is important to be able to develop the applications independent of the underlying hardware configuration, while assuring low latency for parts of the processing pipeline. To prepare for evolution and to address the variability in products, it is necessary that the software can easily be ported to different underlying hardware configurations and that the underlying heterogeneous hardware resources are transparently managed without application developers' intervention. Dynamic analysis tools will automatically detect software defects and a runtime analysis tool will enable high level partitioning decisions.

The use case considered in EMC² aimed at advanced diagnostic magnetic resonance (MR) imaging. Figure 46 shows a typical clinical workflow for creating a magnetic resonance imaging scan for a single patient. Based on the exam request received from the referring physician, the radiologist determines in the morning which scan protocols should be used. At some time during the day, the real MRI examination is planned, which takes 20 to 45 minutes. Only during this time frame the patient is at the MRI scanner. Once the scan is done, the patient is dismissed to either his room or even to home in case of an out-patient. Though some image processing is done during the scan time and a first evaluation is made by the operator, the real review of the examination data by the radiologist only takes place after the patient has left the scanning room. An example of processing and post-processing is given in Figure 47.

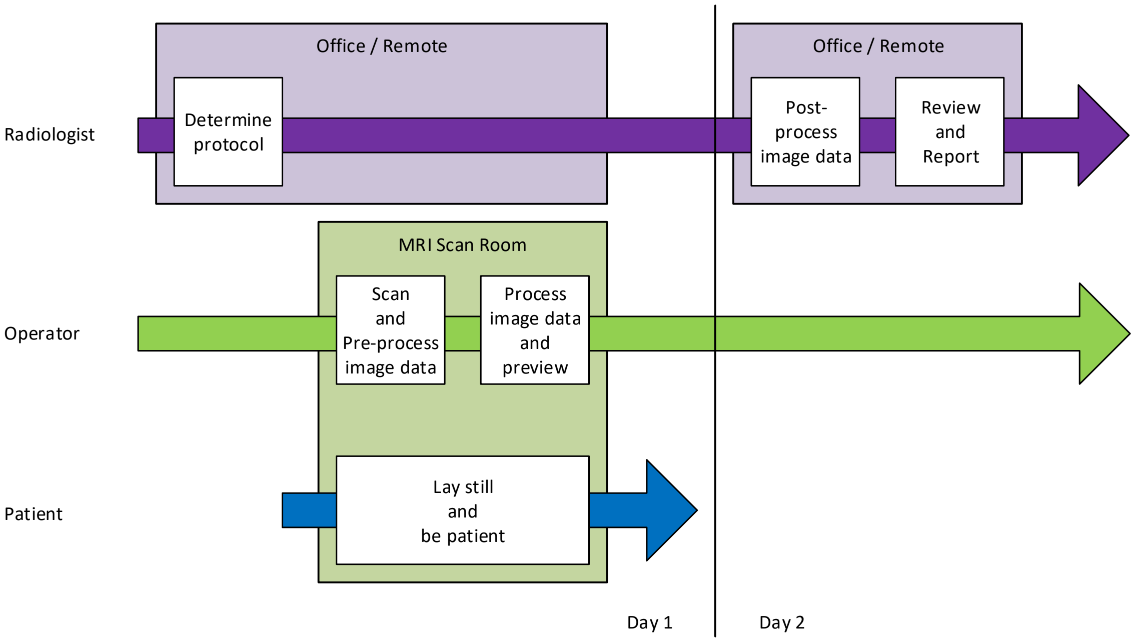


Figure 46: Clinical workflow (typical example)

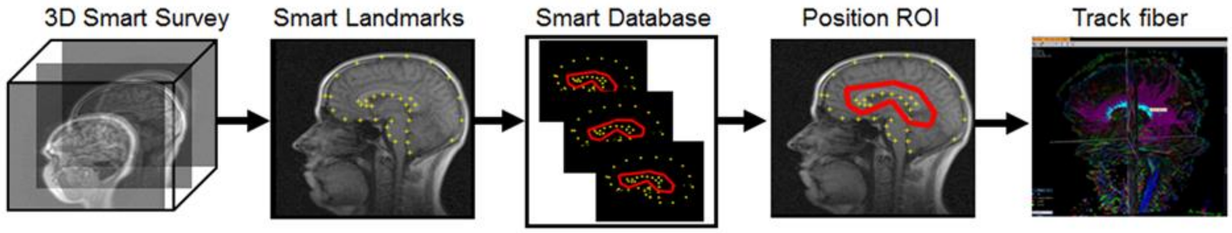


Figure 47: Processing and post-processing example (Smart DTI fiber tracking)

It is only at the moment when the radiologist is analyzing the data that it becomes clear whether or not the data is of sufficient quality to be able to perform a diagnosis on it. Even though the operator is performing a first check on image quality, it still requires the expert eyes of the radiologist to qualify the final image quality. Also in the case of post-processing, which is performed after the examination has been conducted; the quality check is performed after the patient has left the scanner (and likely the hospital). Problems in data acquisition may not be visible to the operator (which does basic checks on quality), yet may prevent the radiologist to draw conclusions from the post-processed image. If the data is of insufficient quality, he has to order a rescan, which is both costly and puts an additional burden on the patient.

The solution to this problem would be to execute post-processing already at the MRI scanner. By performing more advanced checks, which can be performed by the operator using post-processing, the quality check can be performed while the patient is in the scanner. If quality is proven to be insufficient, the scan can be done again. Multi-core computing is a pre-requisite for being able to perform such post-processing on the scanner console, and the mixed criticality of such a system is the main hurdle to be taken with EMC^2 technology:

- Magnetic Resonance Imaging scanners create images based on spin physics. These physics dictate that latencies during scan and image pre-processing are very small (ns to µs).
- The response of the scanner should give a real-time feel to the operator at all times (ms), and image processing should not cause timing problems for the very low latency activities.
- Post-processing currently can be done based on a time/cost-value base. When this process is run on scanner hardware, the requirement becomes similar to processing.

In a state-of-the-art MRI scanner, the “host” computer runs the main process framework. In addition, it runs the user interface and various background tasks for scanning and communication control. During a scan, a real-time Data Acquisition System “DAS” computer takes over and ensures the generation of time aligned raw imaging data. Highly computationally intensive processes are required for the reconstruction of raw data into useful diagnostic imaging data. Therefore, the reconstruction processes are executed on a separate “recon” system. Offline, the images are investigated by a radiologist on a “viewing station”. An overview is given in Figure 48 (left).

The target of the EMC^2 project was to merge multiple hardware systems (host, recon, data acquisition, and post-processing/viewing) on a single hardware system as shown in Figure 48 (right). The first integration step combined “host”, “recon” and “DAS” as shown in Figure 49 (left) on a standard HP computer with a 6-core processor with hyper-threading (12 virtual cores) and 64 GB RAM. This prototype employs a type 1 embedded hypervisor designed for mixed criticality and a very small footprint, minimal latency, and optimizations for maximum performance. On top of the hypervisor the real-time operating system VxWorks (which is identical to the existing operating system of the “scan” computer) and the non-real-time Windows 7.0 are installed.

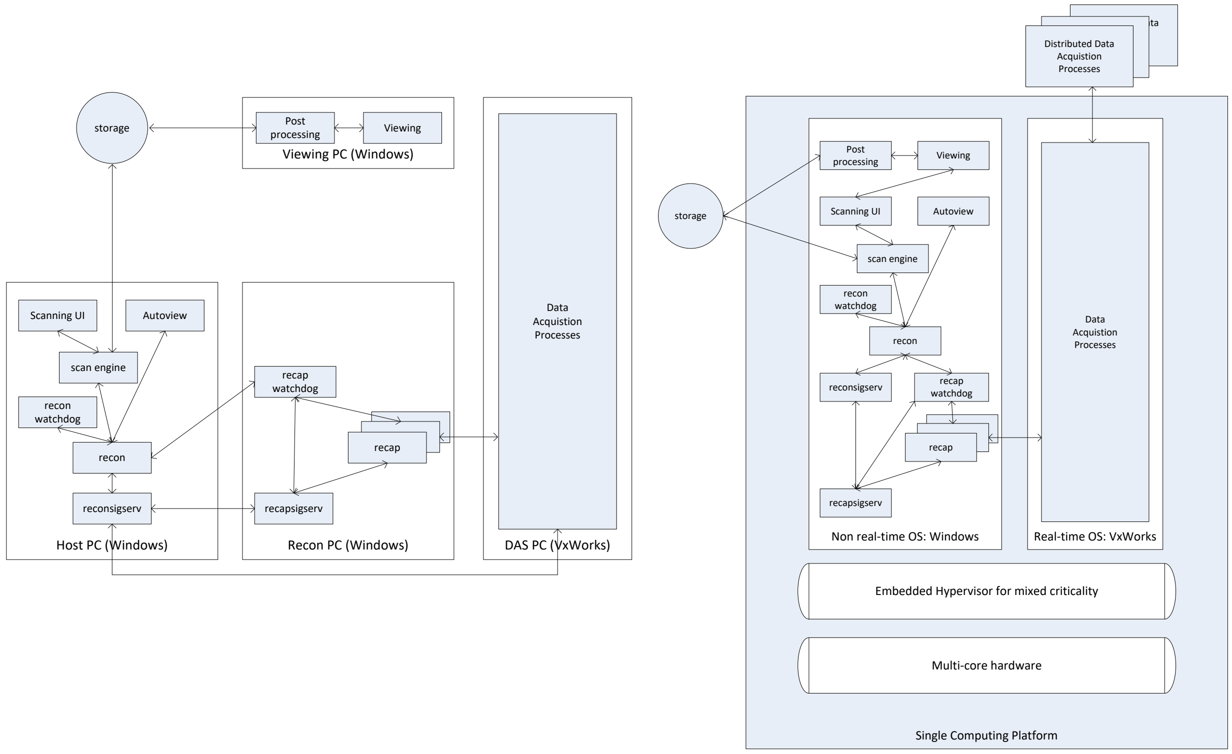


Figure 48: State-of-the art (left) and EMC^2 target (right)

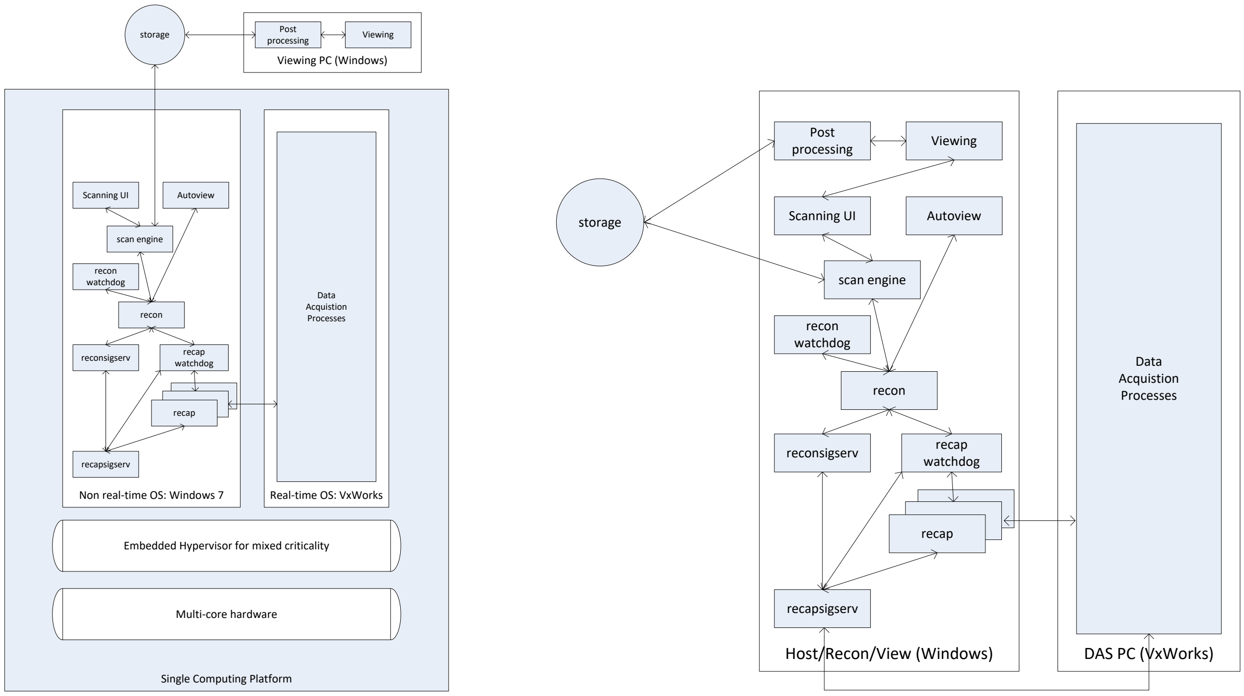


Figure 49: Final prototypes (left: combined host/recon/DAS; right: combined host/recon/view)

Evaluation of the prototypes yielded the following key results:

- For low- and mid-end systems, Philips has productized the first EMC^2 prototype, which integrated host and recon on a single hardware platform. When hardware with more cores becomes available, this will be extended to high end systems. Feasibility of viewing on the console has been proven.
- Integration testing showed that the requirements were not completely met: The target to integrate the data acquisition on the same system could not be met with the new state-of-the-art hypervisor

technology. A second prototype was built on which the “host”, “recon” and “viewing” processes were successfully integrated (Figure 49, right). Scanning is possible without aborts, but 1 out of 1000 samples exceeds the maximum allowed peak latency.

Control applications for critical infrastructure

This use case considers an application covering the business domain of power system control. The main target was a “Cooling System for Transmission Plant” (CS_UC) - Figure 50. The application is a closed loop control system, where a number of sensor elements and actuators are connected by various interfaces. The system performs relevant actions depending on the input signals, the internal system state, the configurable logic, and possibly on operator commands. The system is required to perform a variety of computation intensive operations, with very high real-time requirements, on data coming in concurrent streams.

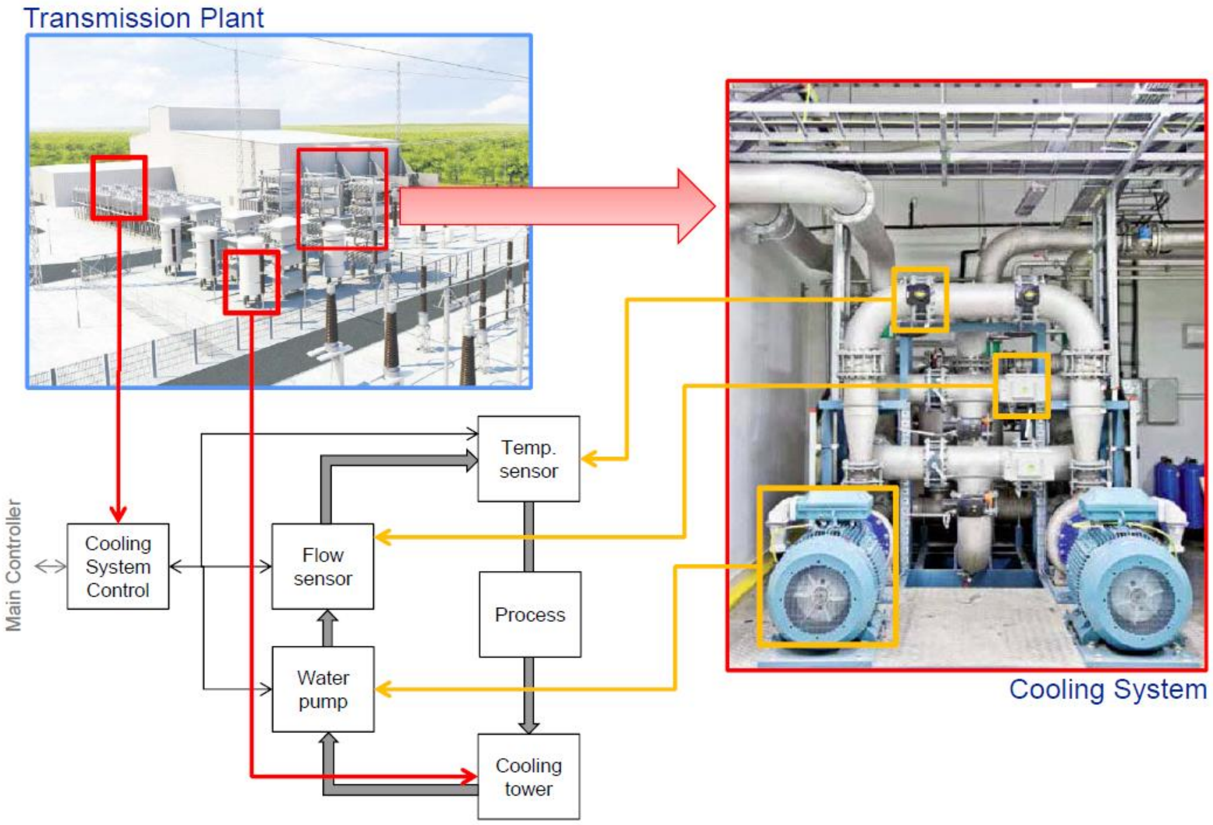


Figure 50: Context for the Cooling System for Transmission Plant Application (CS_UC).

The use case objectives focus on the creation and execution of a tool-chain to support application development, together with multi-core design aspects such as application partitioning and mapping. Thus, in brief, we target to (semi-) automatically provide an (almost) optimal solution for multi-core SW/HW partitioning and mapping, and to (semi-) automatically exchange tools at different stages in the design flow within the used tool-chain.

The realization of an open tool chain for seamless tool integration should be capable of integrating tools to achieve higher engineering efficiency (e.g. reduced design time, increased quality, reduced time to look up information, reduced tool switch time), extend the life-cycle of system engineering tool chains and provide an easy replacement of tools (avoiding vendor “lock-in”), traceability and tool independence.

Challenges occur in particular when there is a heterogeneous set of tools in use, e.g. with different data formats, different semantics, complex interaction scenarios, multiple users, tools not built for integration into a tool chain or tool interfaces not available.

The final prototype was deployed within the present development network, and it contained all the artefacts realized in the project. One user was able to employ in actual context the assessment of the results. An orchestrator (including name translation and notification services) is deployed on a virtual machine accessible through local set-up, while respective tool adaptors (for the design tool HiDraw and the Microsoft TFS modules for requirements and version control) were located on the developer's machine and on the "TFS Server", respectively. An overview of the architectural perspective is shown in Figure 51.

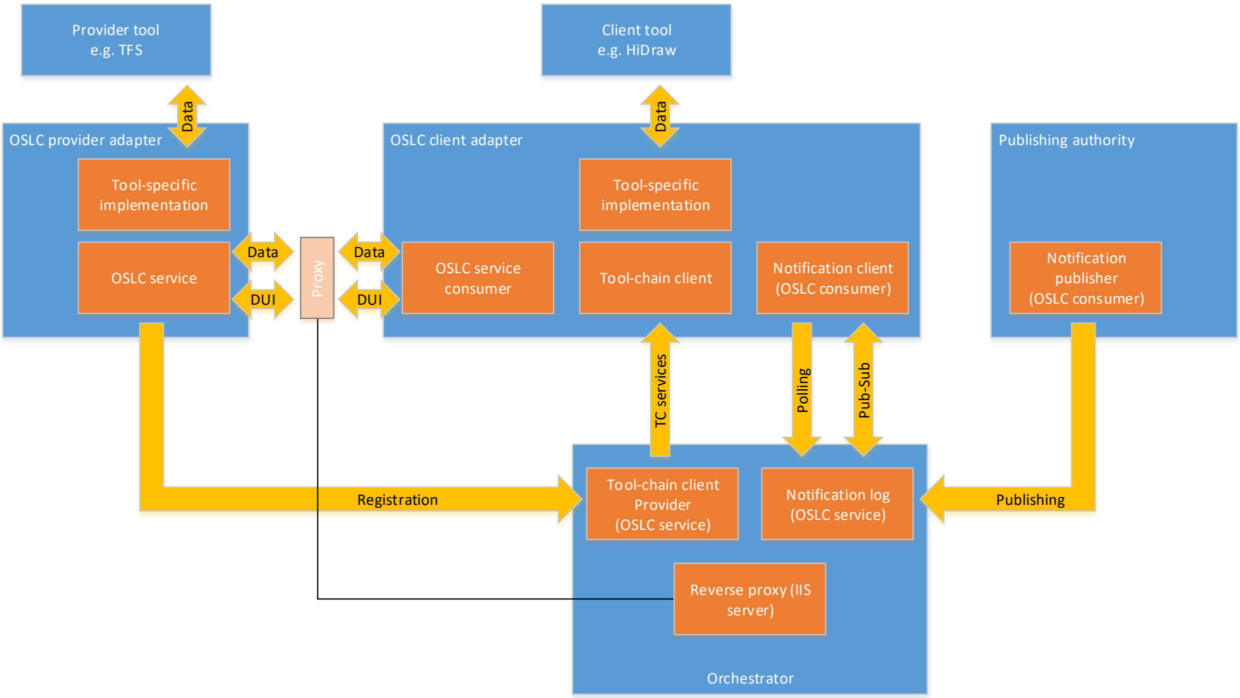
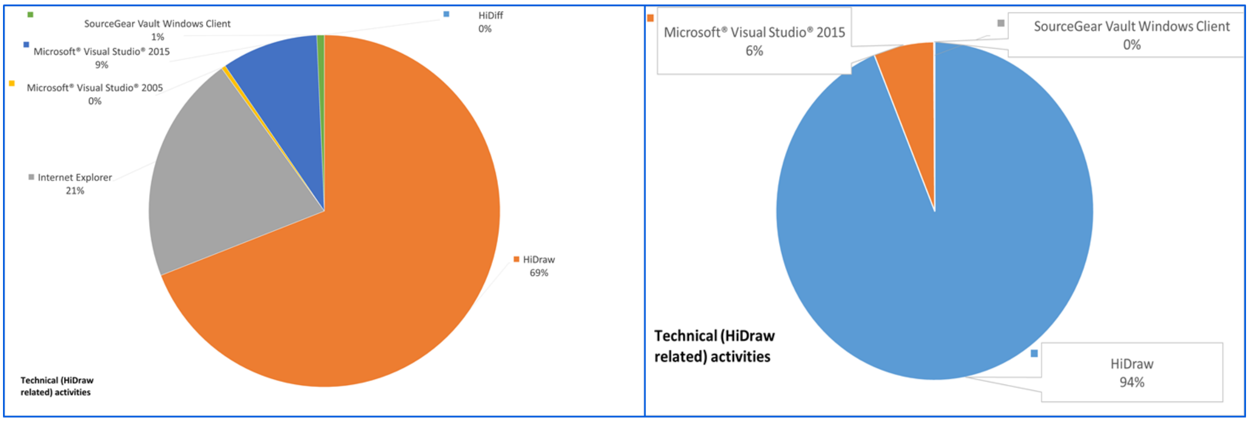


Figure 51: Overall architecture of the implemented system

The final results show an important decrease of actual work time within technical tools, reached due to the integration capabilities. TST and TN final values have defaulted to (approx.) 0 s, respectively 100 %. A graphical representation illustrating cumulated values for IFT and TST and the impact on technical work duration is given in Figure 52.



a) Technical developments without an integrated tool chain

b) Technical developments with the integrated tool chain

Figure 52: Results a) before and b) after the taken approach

Figure 52 shows that the usage of the tool chain increases by a quarter the possible amount of technical work, reduces (by a third) the technical supporting activities (in this case requirements analysis) and eliminates additional tools used today to navigate between tools - as Internet Explorer, with usage data as in Figure 52 a). The results further support either a more efficient and focused development context, or that more efforts can be deployed in the same amount of time within the project. In conclusion, all the goals specified for this use case have been reached and demonstrated, except for the tool replacement.

Use Case Summary

- Baseline extraction in real-life operations
- Tool chain tested in real-life operations
- Proven potential for increasing efficiency: 30 %
- Publication accepted: Making Interoperability Visible:
  Data Visualization of Cyber-Physical Systems Development Tool Chains,
  Elsevier Journal of Industrial Information Integration.

Railway applications

The railway use case aimed at a fault tolerant platform that represents a common base for railway applications (both for mainline and urban application). The objectives covered the extension of programming models to better exploit multicore resources, the extension of hardware health monitoring for multicore platforms and the use of the platform in a virtual environment Pike OS hypervisor.

Novel programming models

Deterministic Multi-Threading (DMT) approaches in combination with high-level multithreading frameworks may provide the expected guarantees with respect to replica determinism and desired performance. Integration of DMT into the TAS platform seems to be feasible in the light of the published results in this area.

As a first improvement, the concept of Sparsely Deterministic Memory (SDM) has been introduced. In SDM, the programmer can choose the memory regions (e.g. variables, memory objects) which have to be treated deterministically. Data races occur when more than one thread concurrently access a memory register, and at least one of these is a modifying access (e.g. write). If a data race occurs to a variable, which is declared deterministic, the framework will ensure that the result is deterministic by enforcing a deterministic per-variable ordering of events. Combined with a deterministic mutual exclusion algorithm, SDM yields a deterministic execution scheduling which is sufficient to build replicated applications.

Second, the concept of Asynchronous Programming using Futures has also been explored. A prototype of this programming model was developed in C including various scheduling techniques like work-stealing, thread pools and plain Operating System style threads. This style of parallel programming can be used to create parallel applications by explicitly specifying the flow of data of the program. In addition to that, it is also easy to migrate a sequential application to multi-core using this programming model.

Health monitor

The RAM/Cache test of the health monitor was improved from the last prototype, as it now supports running in parallel on all cores. To that end, a synchronization mechanism was implemented for the cores when entering the critical testing section, since otherwise the test may affect the other core's memory region as this test must run in kernel space.

Also, the test must not last longer than a few µs for each cycle (caused by very short interrupts of the safety critical railway application running on the processor) so the synchronization mechanism had to be quick. The test's functionality is depicted in Figure 53. First, all running applications are blocked and all interrupts are disabled. Then the test is run on both cores (in kernel space) where each core selects a different part in the memory space, which is tested and then restored for the applications to continue their operation. By using two cores instead of one, we noticed a speed gain of a factor of 1.5. The reason this was not doubled lies in the overhead of the synchronization mechanism.

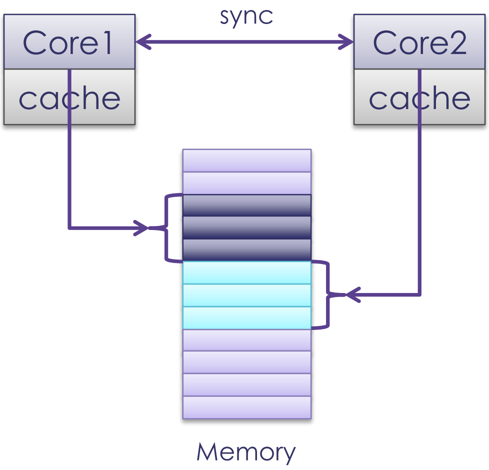


Figure 53: Multi-core synchronization for Safety Health Monitor

Virtualisation on PikeOS

Separation of partitions on top of a shared hardware infrastructure enables the integration of several legacy applications. Apart from the general separation concern, i.e. non-interference between partitions, this separation mechanism must not affect or even prevent the successful execution of the TAS Platform synchronization mechanism. To ensure this property the TAS Platform directs the PikeOS scheduler. Consequently, one of the already identified limitations of this implementation is the synchronization precision compared to a system without hypervisor and time partition. Both of these mechanisms increase communication delays and the jitter in scheduling of the TAS Platform, which in turn only allows more relaxed timing requirements for the safety-critical applications.

The core feature of PikeOS is to provide strict partitioning on top of multi-cores. For sharing the CPUs, PikeOS has a time-sliced scheduler for encapsulation. This scheduling strategy is “as-is” unsuitable for the use of the TAS Platform 2003 system that implements the synchronization mechanism in this setup within a partition of PikeOS, since it would result in unacceptably long reaction times for the safety-critical applications. With the TAS Platform on top of PikeOS prototype, it has been shown that this restriction can be overcome. Figure 54 illustrates the successfully deployed virtualization solution for the TAS platform in order to ease recertification / revalidation.

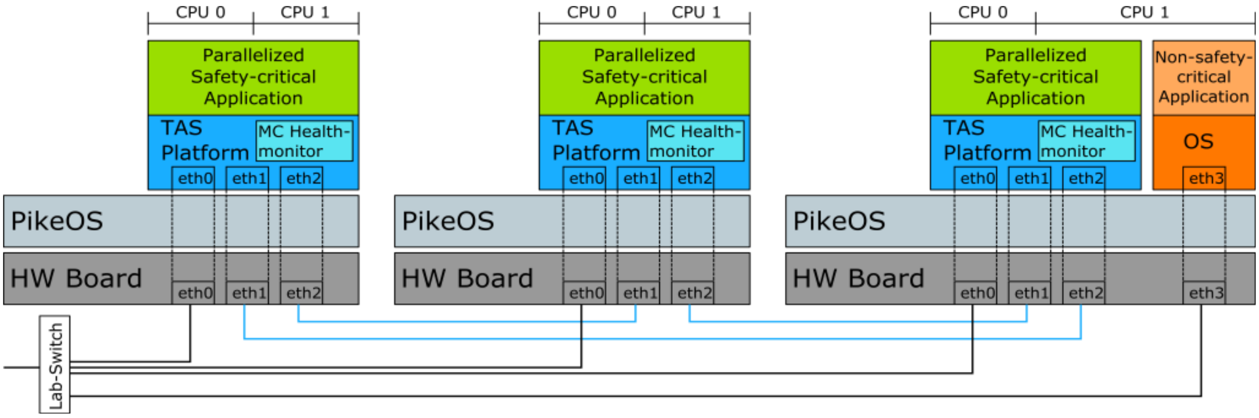


Figure 54: Virtualization solution for the TAS platform

3.3 Highlights from Standardization activities triggered by EMC²

Awareness has risen considerably regarding standardization towards combined approaches to security issues in safety critical systems (IEC, ISO) in many domains. Work in EMC² provided valuable input, particularly WP6, which covers the horizontal activities across Living Labs and domains with respect to the joint consideration of functional safety and cybersecurity and their mutual impact on each other. The second topic covered within EMC² concerned the interoperability Specification – part of the efforts undertaken by the ARTEMIS-IA community towards an interoperability standard for tooling (IOS Specification), which in the beginning was based on OSLC and forwarded to an OSLC group within OASIS (CESAR, MBAT, SafeCer, and particular developed further in CRYSTAL).

The topic of safety & cybersecurity is important from the dynamic and open critical-"systems-of-systems" view of EMC². In IEC and ISO a functional safety standardization was achieved in a considerable progress, with active participation and contribution from EMC² partners.

- In IEC MT 61508-3, the currently very active standardization group for the software part, the task group for "Functional safety and cybersecurity" was established. During the reporting period, a meeting took place in Vienna from Nov. 29th - Dec. 1st 2016, hosted by AIT, where a proposal for this part was discussed taking into account first comments. The intention was to give guidance on how to integrate security engineering in the software safety lifecycle and when to consider security. The next meeting was in Milan, Italy, April 2017, where the topic was discussed with the chairman Ron Bell of MT 61508-1-2, because it is evident that the safety-cybersecurity issue is a system issue, not only a software issue, and additionally has to be taken into hardware as well – as demonstrated in WP 4, Hardware Architecture and Concepts, of EMC². Multi-core and safety/security as a particular topic was proposed for the new Hardware part of IEC 61508 as well. The kick-off meeting for MT 61508-1-2 has taken place at BSI in London from June 13 – 14, 2017. The EMC² - ConSerts approach, as a method to facility security updates without violating safety (i.e. to do it in a certifiable manner) is another action point foreseen to be considered in the new Ed. 3.0.

- In IEC TC65 the former AHG1 – "Framework towards coordinating safety, security", became IEC TC65 WG 20, working now on a TR (Technical Report) IEC TR 63069 "Framework for functional safety and cybersecurity". The approach to cybersecurity & safety co-engineering by defining a workflow and interaction points between safety and cybersecurity activities to develop a safe and secure system is an input derived from EMC² work.

- Partially based on the input from EMC² partners, two new Ad-Hoc Groups continued their work:
  - IEC TC65 AHG2 "Reliability of Automation Devices and Systems". In particular, the system aspect of integration should be considered as well as not reliability calculations of components as done in IEC TC56 and dependability.
  - IEC TC65 AHG3 "Smart Manufacturing Framework and System Architecture", a group of IEC TC65 and, together with ISO TC 184 (Automation systems and integration), cooperating in an ISO/IEC JWG 21 (Smart manufacturing – reference models). Both will focus on cross cutting concerns, considering multiple attributes like safety and cybersecurity.

- Several EMC² partners are involved in the preparation of the next version of ISO26262, planned in 2018. Since the last meeting in JeJu, South Korea, Feb. 6 – 10, 2017, the DIS version (Draft International Standard) now contains requirements and guidelines concerning the cybersecurity impact on functional safety in Part 2 (Safety management), Part 4 (Product development) and Part 6 (Software development).

- The joint ISO/SAE JWG1 "Road vehicles – cybersecurity engineering", ISO 21434, has taken up ideas closely related to the intentions brought forward in EMC², that cybersecurity has to be considered from the functional safety impact as well as from non-safety-related functionalities (e.g. privacy, comfort).

- For the railway use case (WP 12.5), AIT was looking in cooperation with Thales Austria at the impact of cybersecurity in context of the DIN VDE approach (DIN VDE V 0831-104 “Electric signaling systems for railways – Part 104: IT Security Guideline based on IEC 62443”, taken up by CENELEC TC9X SC9XA). Since this analysis was started lately in the project, it is not directly part of the TAT railway demonstrator. TAT will still use the insights from the analysis for ensuring cybersecurity in future versions of their TAS Platform.

A second important activity is related to interoperability specifications. This work started in the ARTEMIS project CESAR, then continued in further European projects such as MBAT, nSafeCer and CRYSTAL and EMC². ARTEMIS-IA and EMC² partners did successfully set up an innovation action in the H2020-ICT-2014-1 call CP-SETIS (Towards Cyber-Physical Systems Engineering Tools Interoperability Standards). CP-SETIS started in March 2015 and finished May 2017. It turned out that the interoperability specification cannot be just one standard or specification – it has to be a collection of standards, guidelines and specifications, which are adopted to become part of the IOS database through a defined process and an ICF – IOS Coordination Forum.

In EMC², this was particularly part of WP5. The goal was mainly to harmonize the different efforts and to create a sustainable structure for further development and maintenance beyond the lifetime of the ARTEMIS/ECSEL projects like EMC². Additionally, the existing ARTEMIS-IA Strategic Standardization Agenda for CPS was updated by the recent achievements and ongoing work in standardization as mentioned above, and delivered as a book[6]. ARTEMIS-IA is hosting and funding the underlying database and their maintenance (commitment of the ARTEMIS-IA Steering Board for the first period).

Standardization work in other areas was ongoing as well, particular examples are:

- Partner AICAS, which is “Specification Leader” in JSR 282 (Java Specification Request 282, for RTSJ V1.1, Real Time Specification for Java). The Real-time and Embedded Specification for Java 2.0 is now complete and will be presented to the Java Community for its final review. In order to bring work on resource management for dynamically loaded code, AICAS has joined the OSGi Alliance and is preparing a proposal to add resource management to and support real-time response in the OSGi framework.

- In the automotive domain, Denso actively participates in the harmonization of AUTOSAR Adaptive, which uses the Yocto project to create a POSIX-based system. The target is a safe and dynamic update process. Moreover, Denso participates in an AUTOSAR working group to improve the classic AUTOSAR with new multicore features, a primarily EMC² driven issue.

[6] E. Schoitsch, J. Niehaus, Strategic Agenda on Standardization for Cyber-Physical Systems, CP-SETIS/ARTEMIS-IA, ISBN 978-90-817213-3-2.

4. Conclusion – evaluation of project achievements

To monitor the project progress with view on the achievement of the overall strategic objectives and in more detail with respect to the individual project objectives, we applied the instrument of technology cockpit charts. These charts provide at a glance the current and the targeted status of any technology developed in the project into certain application(s) of the living labs. Technology cockpit charts were generated for each technology and group of technologies, respectively, developed in the EMC^2 horizontal work packages. Individual technologies were transferred into several use cases. Therefore, a chart was generated for each link of a technology (horizontal) and a use case of a living lab (vertical direction). The transfer progress was tracked over the project lifetime.

Based on this instrument, the technology transfer can be well presented in a matrix organization. With respect to the transfer of technologies, the steps “technology definition”, “technology evaluation”, “technology development”, “technology in transfer to use case” and “transfer to use case complete” were considered. To support the tracking of the technologies and to judge their success during the project duration, EMC² generated dashboards like diagrams (so-called technology cockpit charts). The regular updates present the progress made by illustrating the progress of the needle towards 100 %.

EMC² understands itself as a large platform project, which includes different technologies. It is natural that at project end there will be some technologies being fully integrated and operable in use case demonstrators. However, there are other technologies, which are directed more towards the future and will be exploited in further projects being generated out of the EMC² platform.

In order to support the different readiness levels of the technologies to be developed in EMC² expected at project end, the project defined four different categories of technologies:

- Scope 1: Complete implementation into use cases in the project
- Scope 2: Partial implementation into use cases
- Scope 3: Will be transferred to LL (WP7-12) but not implemented into use cases
- Scope 4: Topic for future applications; technology transfer subsequent to EMC^2

For each technology, the scope at project end was classified. Furthermore, we estimated adequate percentages for the progress levels for each scope:

Scope 1 Configuration		Scope 2 Configuration		Scope 3 Configuration		Scope 4 Configuration	
Definition	12,5	Definition	15	Definition	20	Definition	30
Evaluation	25	Evaluation	30	Evaluation	35	Evaluation	50
Development	25	Development	30	Development	35	Development	20
In Transfer	25	In Transfer	25	In Transfer	10	In Transfer	0
Transfer complete	12,5	Transfer complete	0	Transfer complete	0	Transfer complete	0

The progress levels are used as “background” information presenting the expected result at project end. Thus, at project end, the needle indicating the current progress of the transfer shall point to the right (to indicate that 100 % of the target in the project has been achieved). The following examples present the technology cockpit charts for technologies with different scope levels, each having reached > 95 % of target fulfilment.

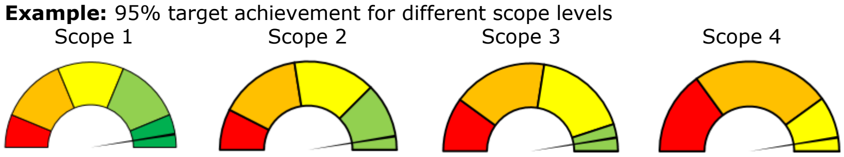


Table 7 summarizes the transfer of different technologies developed in WP1-WP6 into the application use cases (WP7-WP12) of the EMC^2 project. The background color of each technology cockpit chart at the position of the needle determines the background color of the corresponding entry in the collaboration matrix.

Based on the results achieved in the individual work packages, it can be concluded that all strategic objectives were achieved, most of them with a fulfillment level of 100 % and few with a slightly reduced fulfillment level of 80 – 90 %.

			WP7						WP8		WP9					WP10				WP11					WP12				
			UC_ADAS and C2X	UC_Highly automated driving	UC_Design and validation of next generation hybrid powertrain / E-Drive	UC_Modelling and functional safety analysis of an architecture for ACC system	UC_Infotainment and eCall Application Multi-Critical Application	UC_Next Generation Electronic Architecture for Commercial Vehicles	UC_Multi Domain Avionic Architecture	UC_Hybrid Avionics Integrated Architecture	UC_MPSoC Hardware for Space	UC_MPSoC Software and Tools for Space	UC_Optical Payload Applications	UC_Platform Applications	UC_Radar Payload Application	UC_Drives and electric motors in industrial applications	UC_Identification and authentication	UC_Tracking	UC_Manufacturing quality control by 3D inspection	UC_Multimedia communication WEBRTC	UC_Open deterministic networks	UC_Autonomic home networking	UC_Ultralowpower high datarate communication	UC_Synchronized low-latency deterministic networks	UC_Seismic surveying by ship	UC_Video surveillance for critical infrastructure	UC_Medical imaging	UC_Control applications for critical infrastructure	UC_Railway applications
	No.	Technology title	T7.1	T7.2	T7.3	T7.4	T7.5	T7.6	T8.1	T8.2	T9.1	T9.2	T9.3	T9.4	T9.5	T10.1	T10.2	T10.3	T10.4	T11.1	T11.2	T11.3	T11.4	T11.5	T12.1	T12.2	T12.3	T12.4	T12.5
WP1	1.1	System reference architecture of a SoA-based open system of networked multi-core computation units (T1.2, T1.3)		2	2			2				3												2		3			
	1.2	System level convergence of real-time capabilities (T1.4)						3																		4			
	1.3	System and service level security (T1.5)			3			3									4												
	1.4	Safety and fault tolerant concepts for SoA Embedded system architectures (T1.6)			2			3																					
WP2	2.1	DSE: DSE-Ychart: DSE using the Y-chart apporach and DSE for mixed-critical parallel embedded platforms (TNO, UNIVAQ)	3											3															
	2.2	HW/SW support to mixed-critical parallel embedded platforms (UNIVAQ)						3						2															
	2.3	"art2kitekt" - A toolset to model and analyze mixed criticality, multi core real-time systems (ITI)										2							4										
	2.4	Optimal internal resource assignment algorithm for scheduling a task set (TUE)	4																										
	2.5	Formal verification in model driven development of multi-core systems (TUE)	4																										
	2.6	Pareon Verify - dynamic program verification of multicore software + Code quality analysis tools (Vector, CINI)							2																		3		
	2.7	Application Modelling and Implementation Methods (OFFIS, UoMAN)		2						3						2									1				
WP3	3.1	Communication services (T3.2)					1									4	4	3											
	3.2	Virtualization and isolation (T3.3)	4				3									4		3				3							3
	3.3	Security mechanisms and services (T3.4)					3											3			2								
	3.4	Safety platform and real-time mechanisms (T3.5)	1				1	3		2																			
	3.5	Dynamic application and platform control (T3.6)			2		1		1	2																			
	3.6	OS support functions (T3.7)	1				1									4													
WP4	4.1	Heterogenous Multiprocessor SoC architectures (T4.1)	2					2			3		1				4												
	4.2	Dynamic reconfiguration on HW accelerators and reconfigurable logic (T4.2)			1						1			1															
	4.3	Networking (T4.3)	2		1												4					3	3			2			
	4.4	Verification and Validation Techniques (T4.4)			1						3						4												
WP5	5.1	Tools for source code analysis and bugs analysis							1									4				4							
	5.2	Technology toolset (required as a whole)				4														1								2	
	5.3	Tool integration technologies - OSLC, services, platform																										1	
WP6	6.1	WEFACT - Workflow Engine for Analysis, Certification and Test (AIT)			2																								
	6.2	FMVEA - Failure Modes, Vulnerabilitys and Effect Analysis (AIT)			3																3								3
	6.3	Embedded Vision Validation and QA (AIT)																	3										
	6.4	M2C2 - Multidirectional Modular Conditional Safety Certificates (FhG, Tecnalia)			2											4													
	6.5	Safety & Security co-analysis and co-design, contracts for trust (Telvent)				4																		2					
	6.6	AMSPS/PSS concept (IFAT)			2																								
	6.7	PROSSURANCE tool (Tecnalia)			3																								

Table 7: Technology versus use case – cooperation matrix

5. References

[1] Daniel Schneider and Mario Trapp. „Conditional Safety Certification of Open Adaptive Systems." ACM Trans. Auton. Adapt. Syst. 8, 2, Article 8 (July 2013), 20 pages, 2013.

[2] Amorim, Tiago Luiz Buarque de; Schneider, Daniel; Nguyen, Viet Yen; Schmittner, Christoph; Schoitsch, Erwin: Five major reasons why safety and security haven't married (yet), ERCIM News (2015), No.102, pp.16-17, ISSN: 0926-4981

[3] Schoitsch, Erwin, Christoph Schmittner, Zhendong Ma, and Thomas Gruber. "The need for safety and cyber-security co-engineering and standardization for highly automated automotive vehicles." In Advanced Microsystems for Automotive Applications 2015, pp. 251-261. Springer, Cham, 2016

[4] Amorim, T., Martin, H., Ma, Z., Schmittner, Ch., Schneider, D., Macher, G., Winkler, B., Krammer, M., Kreiner, Ch., "Systematic Pattern Approach for Safety and Security Co-Engineering in the Automotive Domain", SAFECOMP 2017, Trento, Italy

[5] Schoitsch, Erwin, Christoph Schmittner, Zhendong Ma, and Thomas Gruber. "The need for safety and cyber-security co-engineering and standardization for highly automated automotive vehicles." In Advanced Microsystems for Automotive Applications 2015, pp. 251-261. Springer, Cham, 2016

[6] G. Macher, E. Armengaud, C. Kreiner, E. Brenner, C. Schmittner, Z. Ma, H. Martin, and M. Krammer, "Integration of Securtiy in the Development Lifecycle of Dependable Automotive CPS", Book Chapter to appear in Handbook of Research on Solutions for Cyber-Physical Systems Ubiquity; http://www.igi-global.com/publish/call-for-papers/call-details/2202, IGI Global, 2017

[7] Daniel Schneider and Mario Trapp. „Conditional Safety Certification of Open Adaptive Systems." ACM Trans. Auton. Adapt. Syst. 8, 2, Article 8 (July 2013), 20 pages, 2013

[8] Amorim, T., Ruiz, A., Dropmann, C., Schneider, D., "Multidirectional Modular Conditional Safety Certificates". SAFECOMP Workshops 2015

[9] Tiago Amorim, Daniel Schneider, Denise Ratasich, Radu Grosu, Georg Macher, Alejandra Ruiz, Mario Driussi: Runtime Safety Assurance for Adaptive Cyber-Physical Systems - ConSerts M and Ontology-based Runtime Reconfiguration Applied to an Automotive Case Study, Book Chapter to appear in Handbook of Research on Solutions for Cyber-Physical Systems Ubiquity; http://www.igi-global.com/publish/call-for-papers/call-details/2202, IGI Global, 2017

[10] EMC² public website www.artemis-emc2.eu

For technical details, readers are kindly referred to the following public deliverables that are available for download from http://www.artemis-emc2.eu/publications/public_deliverables/:

[D1.1] State of the art and SoA architecture requirements report

[D1.5] MW framework specification (publishable summary)

[D1.8] EMC² dependability services

[D1.9] Runtime assessment architecture support concept: design principles and guidelines (publishable summary)

[D3.4] Dynamic Run-time Environments, Implementation, Prototyping, Validation, First Version

[D4.1] Mixed Criticalities Requirements Report

[D4.2] Specification of Demonstrator Platforms

[D4.3] First report on Cores, Memory, Virtualization, Accelerators, Interconnecting Cores, Chips, and Peripheral devices in Mixed-Criticality Systems: Requirements, Concepts and Preliminary Designs

[D4.4] Revised report on Cores, Memory, Virtualization, Accelerators, Interconnecting Cores, Chips, and Peripheral devices in Mixed-Criticality Systems: Results from the First Innovation Cycle

[D4.5] Final report on Cores, Memory, Virtualization, Accelerators, Interconnecting Cores, Chips, and Peripheral devices in Mixed-Criticality Systems: Updates from the Second Innovation Cycle

[D4.6] Specification of preliminary architecture and API respecting the requirements for error handling and redundancy of accelerators

[D4.10] Report describing the evaluation of final implementation, evaluation of innovation results

[D4.11] Verification methods, techniques and processes for mixed-criticality applications on multi-core systems

[D4.14] Recommendations for the certification of mixed-criticality applications on multi-core systems

[D4.17] Final demonstration platforms with revised innovative techniques

[D4.18] AMSPS and Power Management Requirements Report

[D5.26] Requirement document for AMS tools and design flow aspects (first version)

[D6.1] State of the art for System Qualification and Certification, including V&V methods survey

[D6.2] Requirements elicitation

[D6.6] Refined definition of the runtime certificate approach incorporating first concepts regarding the treatment of adaptive behavior

[D6.13] Final definition of the runtime certificate approach incorporating solutions with respect to adaptive behavior

[D6.15] Validation report

[D7.1] Automotive use case descriptions, requirements and evaluation plans

[D7.2] Final automotive use cases implementation and evaluation

[D9.2] Space Applications Concept Report

[D9.5] Space Applications Detailed Description

[D9.6] Space Applications Final Evaluation Report

[D10.3] First prototype application and the description of the conceptual architecture

[D10.5] Completed and final demonstration

[D10.6] Final report

[D11.1] System level requirements

[D11.4] Final demonstrator implementation and evaluation

[D12.1] Requirements, specifications, and evaluation plan

[D12.6] Final evaluation of the innovation results

[D13.13] Project Homepage and dissemination material www.artemis-emc2.eu

[D13.20] EMC² exhibition

[D13.24] Initial Standardization Report and Plan

[D13.25] Intermediate Standardization Report

[D13.26] Final Standardization Report and Outlook