A safety concept for a wind power mixed-criticality embedded system based on multicore partitioning

Jon Perez, David Gonzalez, Salvador Trujillo
Embedded Systems Group
Ik4-IKERLAN Technology Research Centre
Mondragon, Spain
jmperez,dgonzalez,strujillo@ikerlan.es

Ton Trapman, Jose Miguel Garate
Software and Performance
Alstom Renewables
Barcelona, Spain
anton-aart.trapman,jose-miguel.garate@power.alstom.com

***Abstract*—The development of mixed-criticality systems that integrate applications of different criticality levels (safety, security, real-time and non real-time) can provide multiple benefits such as product cost-size-weight reduction, reliability increase and scalability. However, the integration of applications of different levels of criticality leads to several challenges with respect to safety certification standards.**

This paper defines a safety certification strategy for IEC-61508 compliant industrial mixed-criticality systems based on multicore partitioning. The final objective is the certification of a wind-turbine mixed-criticality control system according to IEC-61508 and ISO-13849 industrial safety standards. This approach is illustrated with a simplification of the safety concept currently under detailed review by a certification body.

***Index Terms*—mixed-criticality ; safety; IEC-61508; certification; multicore; partition**

I. Introduction

Conventional embedded system architectures in multiple domains follow a federated architecture paradigm, in which the system is composed of interconnected embedded subsystems where each of them provides a well defined functionality. The ever increasing demand for additional functionalities leads to a considerable complexity growth [1] that in some cases limits the scalability of the federated approach. For example, a modern off-shore wind turbine dependable control system manages up to three thousand inputs / outputs, several hundreds of functions are distributed over several hundred nodes grouped into eight subsystems interconnected with a fieldbus and the distributed software contains several hundred thousand lines of code.

The integration of additional functionalities also leads to an increase in the number of subsystems, connectors and wires increasing the overall cost-size-weight and reducing the overall reliability of the system. For example, in the automotive domain, field data has shown that between 30-60% of electrical failures are attributed to connector problems [2].

The integration of applications of different criticality (safety, security, real-time and non-real time) in a single embedded system is referred as mixed-criticality system. This integrated approach can improve scalability, increase reliability reducing the amount of systems-wires-connectors and reduce the overall cost-size-weight factor. However, safety certification according to industrial standards becomes a challenge because sufficient evidence must be provided to demonstrate that the resulting system is safe for its purpose. Higher safety integrity functions must be interference free with respect to lower safety integrity functions.

This paper contributes with the definition of a safety certification strategy for IEC-61508 compliant industrial mixed-criticality systems based on multicore partitioning, and illustrates it with a safety concept for a wind-turbine mixed-criticality control system. Both the strategy and the example safety concept consider the usage of Commercial off-the-shelf (COTS) multicore processors.

The paper is organized as follows. Section II introduces basic concepts and Section III analyses related work. Section IV describes the proposed safety certification strategy and Section V briefly describes the safety concept. Finally, Section VI draws the overall conclusion and future work.

II. Background

A. Certification standards

IEC-61508 [3], [4], [5] is an international standard for electrical, electronic and programmable electronic safety related systems. IEC-61508 is a generic safety standard from which different domain specific standards have been derived for industrial and transportation domains, e.g. machinery, industry process, automotive, railway, etc.

Safety Integrity Level (SIL) is a discrete level corresponding to a range of safety integrity values where 4 is the highest level an 1 is the lowest. As a rule of thumb, the highest the SIL the highest the certification cost.

B. Fail-safe and fail-operational

Safety systems can be classified as either fail-safe or fail-operational. A system is fail-safe if there is a safe state in the environment that can be reached in case of a system failure either by the safety function or diagnostics, e.g., a process plant can be safely stopped, a train can be stopped, a lift can be stopped, etc. A system is fail operational if no safe state can be reached in case of a system failure, e.g., a flight control system aboard an airplane, drive by wire in a car, etc.

III. Related work

Multiple analyses [6], [7], [8], [9], [10], [11] and research publications [12], [13], [14], [15], [16] indicate that is likely to be a significant increase in the use of multicore devices over the next years replacing applications that have traditionally used single core processors. Multicore and virtualization technology can support the development of mixed-criticality systems by means of software partition, or partition for short. Partitions provide functional separation of the applications and fault containment, to prevent any partitioned application from causing a failure in another partitioned application.

However, the development of safety critical embedded systems based on multicore and virtualization technology is a challenge [17], [18], [19], [20], [21], [22]. Providing sufficient evidence of isolation, separation and independence among safety and non-safety related functions distributed in a multicore processor is not a trivial task [21], [22].

IEC-61508 safety standard does not directly support nor restrict the certification of mixed-criticality systems. Whenever a system integrates safety functions of different criticality, sufficient independence of implementation must be shown among these functions [3], [4]. If there is not sufficient evidence, all integrated functions will need to meet the highest integrity level. Sufficient independence of implementation is established showing that the probability of a dependent failure between the higher and lower integrity parts is sufficiently low in comparison with the highest safety integrity level [4].

Therefore, spatial and temporal isolation are key requirements in mixed-criticality systems because otherwise low criticality applications could interfere with those of high criticality. While spatial isolation can be commonly achieved using state of the art solutions (e.g., MMU), temporal isolation at application level depends on the time guarantees provided by the underlying multicore processor. The usage of time deterministic architectures and processors [19] could simplify the collection of evidences for a certification process because determinism is a sufficient precondition for logical reasoning required for time behaviour analysis [1]. However, most of the existing COTS multicore processors were not designed with a focus on hard-real time applications but towards the maximal average performance. This is the source for multiple temporal isolation challenges [21], [22].

The avionics industry has widely adopted the Integrated Modular Avionics (IMA) [23] architecture, which allows integrating several applications on a single processing element. Applications are encapsulated into partitions that are temporally and spatially isolated from one another, enforcing fault containment [24].

However, the migration of an existing set of pre-certified single-core avionics IMA systems into a multi-IMA multicore system is not a trivial task. The fundamental challenge is to ensure that the temporal and spatial isolation of the partitions will be maintained without incurring huge recertification costs [8], [9], [16], [25], [26], [27], [28], [29].

IV. Safety Certification Strategy

This section describes an IEC-61508 compliant safety certification strategy for mixed-criticality systems based on multicore partitioning, based on the following assumptions:

- The IEC-61508 standard mainly targets fail-safe systems.
- The fault hypothesis defines overall safety assumptions
- An hypervisor ported to a given platform is provided as a certified compliant item
- The hypervisor supports a static cyclic scheduling algorithm with guaranteed time slots defined at design time
- A system level diagnosis strategy is defined

As multicore partitioning based solutions are still not common practice in industry, the strategy shown in Figure 1 considers a three step safety concept transformation from a federated architecture to a multicore integrated architecture.

- Transform federated to multiprocessor: Transform the safety concept of a federated architecture to a multiprocessor safety concept using well known techniques that are common practice in industry
- Transform multiprocessor to multicore: Transform previous safety concept to multicore safety concept still abstracted from detailed analysis of shared-resources. Analyse and select the platform with regard to isolation
- Analyse multicore shared resources: Define, analyse and asses in detail shared resources and their effect

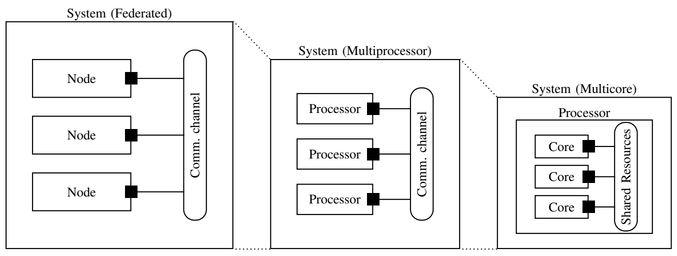

Fig. 1. Safety concept transformation strategy in consecutive steps.

A. IEC-61508 and fail-safe systems

IEC-61508 based safety-critical embedded systems must be developed with a safety life-cycle that aims to reduce the probability of systematic errors and ensure that sufficient fault avoidance and fault control techniques are implemented. Regarding temporal isolation, this means that isolation needs to be systematically guaranteed (or give safe worst case bounds) and diagnosis techniques must be used to detect temporal isolation violations (e.g., watchdog, logic execution, etc.). If this unexpected violation occurs, diagnosis should lead the system to safe-state (e.g., reset). Therefore, the lack of complete temporal isolation would reduce the availability of the system but should not jeopardize safety.

B. Fault hypothesis

The fault-hypothesis [30] of this strategy consists of the following assumptions:

- FSM: All safety relevant systems are developed with an IEC-61508 Functional Safety Management (FSM)
- Node: The node computer forms a single Fault-Containment Region (FCR) that can fail in an arbitrary failure mode. The permanent failure rate is assumed to be in the order of 10-100 FIT (i.e., about one thousand year) and the transient failure rate is assumed to be in the order of 100.000 FIT (i.e., about one year)
- Processor: The multicore processor might not provide complete temporal isolation (or not sufficient evidence for certification), but bounded temporal interference can be estimated and validated with measurements
- Hypervisor: The hypervisor provides interference freeness among partitions (bounded time and spatial isolation), it is certified and fails in an arbitrary failure mode when it is affected by a fault
- Partition: A partition can fail in an arbitrary failure mode, both in the temporal as well as the spatial domain

C. Compliant item: Hypervisor and platform

Hypervisor is a layer of software (or a combination of software / hardware) that allows running several independent execution environments in a single computer platform. Hypervisor solutions such as XtratuM [31] have to introduce a very low overhead compared with other kind of virtualizations (e.g., Java virtual machine); the throughput of the virtual machines has to be very close to that of the native hardware.

The strategy assumes that a hypervisor and platform are provided as a single certified compliant item according to IEC-61508. The safety manual should state that the compliant item provides the following techniques and properties:

- Startup, configuration and initialization: The hypervisor must start up, configure and initialize in a known, repeatable and correct state within a bounded time (e.g., internal data structures, virtualized resource initialization, etc.). Configuration data is static and defined at design stage.
- Virtualization of resources: Provide a virtual environment in a safe, transparent and efficient way (e.g., CPU, memory and Input / Output (I/O) devices)
- Isolation, diagnosis and integrity:
  - Spatial isolation: To prevent one partition from overwriting data in another partition, or a memory address not explicitly assigned to this partition
  - Temporal isolation: To ensure that a partition has sufficient processing time to complete its execution, ensuring that partition cyclic schedule and time slots are assigned as statically configured
  - Health monitoring: To control random and systematic failures at hypervisor or partitions level. Actions to handle these errors are statically defined.
  - Exclusive access to peripherals: Protect access to peripherals used by a safety partition
  - Hypervisor Execution Integrity: The hypervisor execution should be in privileged mode, isolated and protected against external software faults.
- Communication and synchronization:
  - Inter-partition communication: The hypervisor must support mechanisms that allow safe data exchange between two or more partitions
  - Time Synchronization: Fault-tolerant time synchronization that provides a global notion of time to the hypervisor partition scheduler

D. Scheduling

The scheduling of partitions should follow a static cyclic scheduling algorithm with pre-assigned guaranteed time slots defined at design time. The scheduling of partitions among cores should be synchronized based on the global notion of time provided by the hypervisor.

E. Diagnosis strategy

In order to manage the complexity management [1] arising from the safe integration of multiple mixed-criticality partitions, a diagnosis strategy is defined taking into consideration the following assumptions:

- Partitions are developed abstracted from the platform
- The hardware platform provides autonomous hardware diagnosis an diagnosis to be commanded by software
- The execution platform (hardware and hypervisor) is abstracted from the partitions to be executed. The hypervisor provides health monitoring that might be complemented with additional system diagnosis partition(s)
- The system architect is responsible for the architectural design, safety integration and must take care of:
  - Analysing safety manuals of integrated safety partitions and compliant items
  - Selection of partitions and diagnosis partitions
  - Defining the design time static configuration, e.g., scheduling and allocation of resources

Based on this assumptions, the recommended diagnosis strategy is described below:

- The partition should be self contained and should provide safety life-cycle related techniques (e.g., IEC-61508-3 Table A.4 defensive programming) and platform independent diagnosis (e.g., IEC-61508-2 Table A.7 input comparison voting) abstracted from the details of the underlying platform
- The hardware provides autonomous diagnosis (e.g., IEC-61508-2 Table A.9 Power Failure Monitor (PFM)) and diagnosis components to be commanded by software (e.g., IEC-61508-2 Table A.10 watchdog)
- The hypervisor and associated diagnosis partitions should support platform related diagnosis (e.g., IEC-61508-2 Table A.5 signature of a double word)
- The system architect specifies and integrates additional diagnosis partitions required to develop a safe product taking into consideration all safety manuals

V. Case Study

This section briefly describes a case-study where previously defined safety certification strategy (Section IV) is applied for the definition of a safety concept for a mixed-criticality wind power control based on multicore partitioning.

A wind park is composed of interconnected wind turbines and a centralized wind park control center as shown in Figure 2. As previously explained current wind turbine control unit follows a federated architectural approach and provides three major functionalities:

- 'Supervision': Wind turbine real-time control and supervision.
- 'SCADA': Non real-time Human Machine Interface (HMI) and communication with SCADA system
- 'Safety Protection': Safety functions that ensure that design limits of the wind turbine are not exceeded

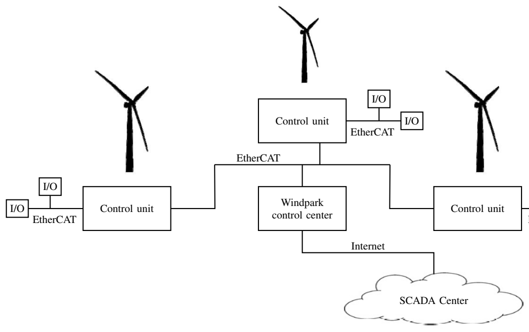

Fig. 2. Simplified wind park diagram.

The safety protection system must ensure that design limits of the wind turbine are not exceeded (e.g., over speed) and if exceeded output safety-relays connected to the safety-chain must be opened. As shown in Figure 3, there is a safety-chain composed of safety-relays in serial that activates the 'pitch control' safety function whenever the chain is opened. The 'pitch control' safety function leads the wind turbine to a safe-state within a Process Safety Time (PST). The safety protection system must meet 'PLd' level of ISO-13849 [32] and IEC-61508 SIL2/3.

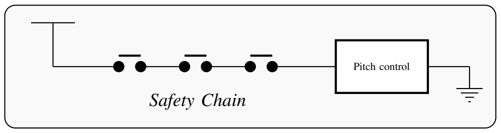

Fig. 3. Wind turbine safety chain.

A. Safety concept

This section describes the safety concept of a mixed-criticality wind power system based on multicore and virtualization partitioning.

1) Transformation (Federated to multiprocessor): The first step is to transform a subset of the current federated architecture into an integrated architecture based on two or more processors. The safety concept behind the architecture shown in Figure 4 is common practice in industry: $1oo2(D)$ dual-channel architecture based on two independent processors, two shared diverse input sources (rotation speed) and two output-relays connected in serial to the safety-chain.

The node has a Hardware Fault Tolerance (HFT) of one ($HFT = 1$)) based on two independent processors. Each processor controls one independent safety-relay that can be de-activated (safe-state) either directly commanded by 'safety protection' or indirectly by 'diagnosis'. If the 'diagnosis' detects a fatal error, it does not refresh the associated watchdog and this leads to a reset of the node. As a summary:

- '$P0$' and '$P1$' are independent single core processors
- '$P0$' processor executes safety related partitions only: 'safety protection' and 'diagnosis'
- '$P1$' processor executes all partitions
- Each processor controls one independent safety-relay
- EtherCAT 'communication stack' is managed in $P1$ and the safety-communication layer in 'safety protection'
- Local and cross-channel 'diagnosis' in each processor
- An independent 'watchdog' monitors each processor
- An IEC-61508 SIL2 system with $HFT = 1$ requires a Safe Failure Fraction (SFF) of $90\% > SFF >= 60\%$

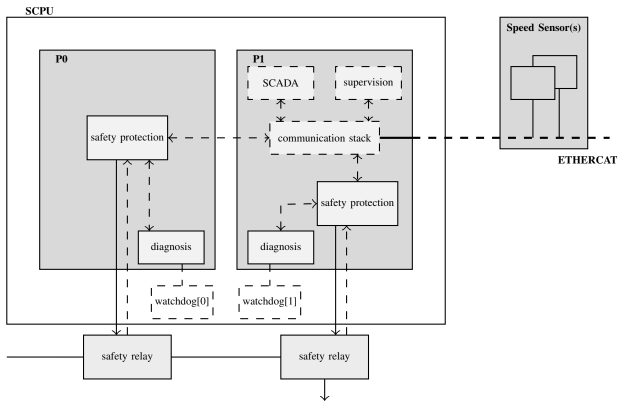

Fig. 4. Safety concept($1oo2$; 2 processors)

The future scalability of this approach is also limited. The number of integrated functionalities will continue to increase, but the usage of fans is not allowed in order to meet reliability and availability requirements. The computation power of the single core processor is limited and if processor '$P1$' does not provide sufficient computation power new processors will be need to be added. Adding new processors and their associated communication buses leads to additional reliability and availability issues (e.g., material reliability, EMC, etc.).

2) Transformation (multiprocessor to multicore): Previous multiprocessor based safety concept shown in Figure 4 is transformed into a multicore architecture shown in Figure 5.

At this abstraction level, different platforms are analysed taking into consideration features such as safety, computation, memory, communication, isolation, etc. The theoretical analysis based on available documentation must be validated with experimental evaluation. The mapping of partitions to cores can also be modified according to platforms specific constraints and properties. The selected platform shown in Figure 5 is an heterogeneous quadcore processor (two 'x86' cores and two 'LEON3 FT' softcores), that meets application requirements, application dependencies with 'x86' architecture and has been positively assessed [33].

In addition to this, the diagnosis strategy defined in the previous transformation needs to be reviewed taking into consideration the details of the new platform. For example, a single processor node requires a processor that meets IEC-61508-2 Annex E in order to claim a $HFT = 1$ and this is not common for COTS processors. If this claim does not hold, a higher SFF is required (a IEC-61508 SIL2 system with $HFT = 0$ requires $99\% > SFF >= 90\%$), which implies additional diagnosis techniques and updates in previously selected ones.

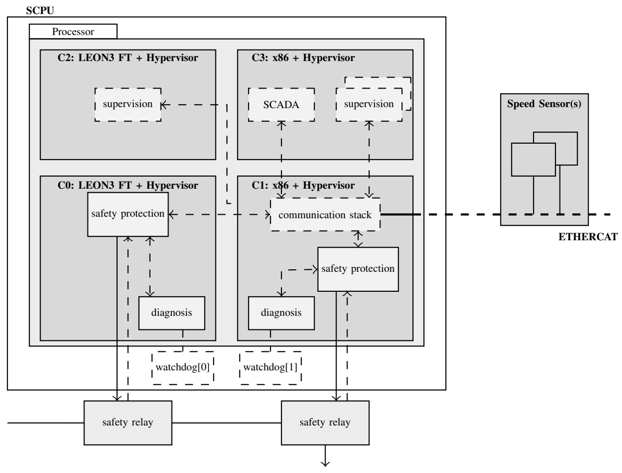

Fig. 5. Simplified safety concept (1oo2), multicore).

3) Shared resources: Figure 6 shows the detailed processor diagram taking into consideration major shared-resources. The real platform is composed of two commercial nodes, a dual-core Intel Atom processor connected via PCIe to an FPGA that integrates two 'LEON3 FT' softcores. For the purpose of this analysis, they are considered to be a single silicon rather than two independent silicon. 'LEON3 FT' softcores have associated a local memory for program and data ('LS memory') and use an external shared memory ('external shared memory') for inter-partition communication. 'x86' cores have $L1/L2$ cache and share and external memory ('external shared memory 2'). Communication among partitions allocated in 'x86' and 'LEON3 FT' cores is implemented using an external shared memory accessed by a shared bus (AHB bus - gateway - PCIe). A periodic interrupt common to all cores is used for hypervisor time synchronization purposes.

The extended safety concept includes FMEAs, error reaction

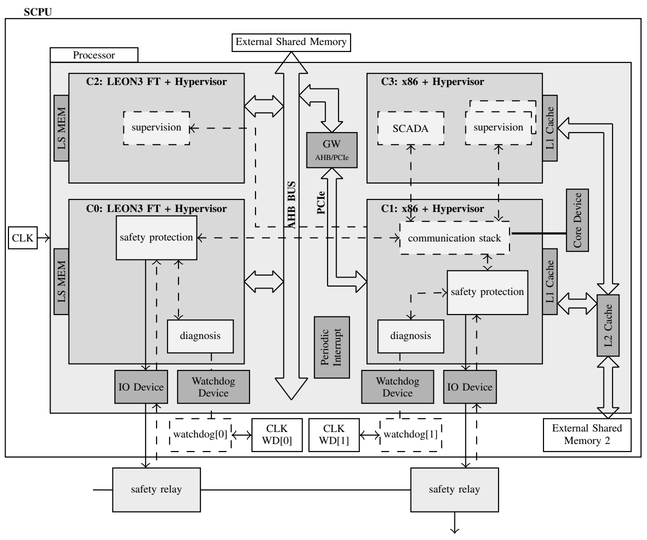

Fig. 6. Safety concept (1oo2), multicore with shared resources).

definitions and it is complemented with a detailed assessment of the platform [33]. Spatial isolation was positively assessed. However, it was concluded that temporal characteristics of partitions could be influenced by different loads scenarios in other partitions due to shared resources. For example:

1) Shared memory: x86' cores use shared-memory and 'LEON3 FT' cores use shared memory for inter-partition communication. Maximum temporal interference suffered by a partition is estimated and measured
2) Shared cache: Atom processor (dual core 'x86') does not support temporal freeness in shared cache, the maximum temporal interference suffered by a partition is measured
3) Interrupts: Some interrupts in the Atom processor can not be rerouted and this can influence the timing behaviour of the hypervisor, the maximum temporal interference suffered by a partition is measured
4) Communication channel: Complete decoupling of sender and receiver partitions connected with a communication channel require temporal isolation

Different solutions are defined in order to avoid and control failures due to previously described temporal interferences:

- Fault avoidance
  - Shared-resources: 'Safety protection' and 'diagnosis' partition Worst Case Execution Time (WCET) are measured for each core type ('x86' and 'LEON3 FT'). Both partitions are scheduled at the beginning of each periodic cycle with a pre-assigned time-slot bigger than the maximum estimated execution time, which considers both the WCET and maximum estimated time interference due to shared resources
  - Interrupts: All unused interrupts are routed to 'diagnosis' or health monitoring
  - Communication channel: The communication among 'safety protection' and 'diagnosis' partitions in different cores is delayed one execution cycle, which

it is considered sufficient to diminish temporal interferences due to shared resources.

- Fault control:
  - Shared-resources: Safety partitions are executed in two diverse cores ('x86' and 'LEON3 FT') with different hypervisor configuration. Each 'diagnosis' partition refresh an independent watchdog if monitored-time constraints are met.
  - Interrupts: 'Diagnosis' partition traps unused interrupts and decides whether to refresh an independent watchdog based on the severity of the error
  - Communication channel: Safety partitions monitor communication channel time-outs.

VI. Conclusion and Future Work

While mixed-criticality paradigm based on multicore and partitioning provides multiple potential benefits, it is clear that the safety certification of such systems based on COTS multiprocessors not designed for safety is a challenge.

This paper has contributed with a safety-certification strategy for IEC-61508 based safety systems based on COTS multiprocessors that have been illustrated with a safety concept currently under detailed review by a certification body. The assumptions and analysis considered at this stage will be reviewed in the following design stages and validated at the final stage of the case-study within FP7 MultiPARTES project.

Acknowledgement

This work has been supported in part by the European projects FP7 MultiPARTES and FP7 DREAMS under project No. 287702 and No. 610640 respectively and the national INNPACTO project VALMOD under grant number IPT-2011-1149-370000. Any opinions, findings and conclusions expressed in this article are those of the authors and do not necessarily reflect the views of funding agencies. The authors would like to thank Alfons Crespo from UPV, XtratuM team and TU Wien.

References

[1] H. Kopetz, "The complexity challenge in embedded system design," in *11th IEEE International Symposium on Object Oriented Real-Time Distributed Computing (ISORC)*, 2008, pp. 3–12.
[2] J. Swingler and J. W. McBride, "The degradation of road tested automotive connectors," in *Forty-Fifth IEEE Holm Conference on Electrical Contacts*, 1999, pp. 146–152.
[3] IEC, "IEC 61508-1: Functional safety of electrical/electronic/programmable electronic safety-related systems part 1: General requirements," 2010.
[4] ——, "IEC 61508-2: Functional safety of electrical/electronic/programmable electronic safety-related systems part 2: Requirements for electrical / electronic / programmable electronic safety-related systems," 2010.
[5] ——, "IEC 61508-3: Functional safety of electrical/electronic/programmable electronic safety-related systems part 3: Software requirements," 2010.
[6] "Mixed criticality systems," European Comission, Tech. Rep., February 3 2012.
[7] "MULCORS - use of multicore processors in airborne systems (research project EASA.2011/6)," EASA, Tech. Rep., 16th December 2012.
[8] EASA, "Certification memorandum - software aspects of certification - EASA CM SWCEH 002," Tech. Rep., 9th March 2013.
[9] ——, "Development assurance of airborne electronic hardware," 2011.
[10] S. Balacco and C. Rommel, "Next generation embedded hardware architectures: Driving onset of project delays, costs overruns and software development challenges," Klockwork, Inc., Tech. Rep., September 2010.
[11] "2013 - embedded market study," UBM Tech, Tech. Rep., 2013.
[12] M. S. Mollison, J. P. Erickson, J. H. Anderson, S. K. Baruah, and J. A. Scoredos, "Mixed-criticality real-time scheduling for multicore systems," pp. 1864–1871, 2010.
[13] R. Ernst, "Certification of trusted MPSoC platforms," in *MPSoC Forum*, 2010.
[14] H. Kopetz, R. Obermaisser, C. El Salloum, and B. Huber, "Automotive software development for a multi-core system-on-a-chip," in *Fourth International Workshop on Software Engineering for Automotive Systems (ICSE Workshops SEAS)*, 2007, pp. 2–9.
[15] D. Gonzalez, J. M. Garate, A. Trapman, L. Monsalve, and S. Trujillo, "Mixed-criticality in wind power: The MultiPARTES approach," in *ESReDA Conference 2012*, 2012, p. 9.
[16] X. Jean, M. Gatti, G. VBerthon, and M. Fumey, "The use of multicore processors in airborne systems," Thales Avionics, Tech. Rep., 2011.
[17] J. Schneider, M. Bohn, and R. Rbger, "Migration of automotive real-time software to multicore systems: First steps towards an automated solution," in *22nd EUROMICRO Conference on Real-Time Systems*, 2010.
[18] R. Fuchsen, "How to address certification for multi-core based IMA platforms: Current status and potential solutions," in *IEEE/AIAA 29th Digital Avionics Systems Conference (DASC)*, 2010, pp. 5.E.3–1–5.E.3–11.
[19] C. E. Salloum, M. Elshuber, O. Hoftberger, H. Isakovic, and A. Wasicek, "The ACROSS MPSoC – a new generation of multi-core processors designed for safety-critical embedded systems," in *Digital System Design (DSD), 2012 15th Euromicro Conference on*, 2012, pp. 105–113.
[20] J. Abella, F. J. Cazorla, E. Quinones, A. Grasset, S. Yehia, P. Bonnot, D. Gizopoulos, R. Mariani, and G. Bernat, "Towards improved survivability in safety-critical systems," in *IEEE 17th International On-Line Testing Symposium (IOLTS)*, 2011, pp. 240–245.
[21] O. Kotaba, J. Nowotsch, M. Paulitsch, S. M. Petters, and H. Theilingx, "Multicore in real-time systems temporal isolation challenges due to shared resources," in *Workshop on Industry-Driven Approaches for Cost-effective Certification of Safety-Critical, Mixed-Criticality Systems (WICERT)*, 2013.
[22] R. Nevalainen, O. Slotosch, D. Truscan, U. Kremer, and V. Wong, "Impact of multicore platforms in hardware and software certification," in *Workshop on Industry-Driven Approaches for Cost-effective Certification of Safety-Critical, Mixed-Criticality Systems (WICERT)*, 2013.
[23] "RTCA DO-297 integrated modular avionics (IMA) development guidance and certification considerations," 2005.
[24] X. Jean, D. Faura, M. Gatti, L. Pautet, and T. Robert, "Ensuring robust partitioning in multicore platforms for ima systems," in *IEEE/AIAA 31st Digital Avionics Systems Conference (DASC)*, 2012, pp. 7A41–7A49, export Date: 27 June 2013 Source: Scopus Art. No.: 6382408.
[25] J.-E. Kim, M.-K. Yoon, S. Im, R. Bradford, and L. Sha, "Optimized scheduling of multi-IMA partitions with exclusive region for synchronized real-time multi-core systems," pp. 970–975, 2013.
[26] L. M. Kinnan, "Use of multicore processors in avionics and its potential impact on implementation and certification," *SAE Technical Papers*, 2009.
[27] P. Huyck, "Arinc 653 and multi-core microprocessors - considerations and potential impacts," in *IEEE/AIAA 31st Digital Avionics Systems Conference (DASC)*, 2012, pp. 6B41–6B47.
[28] J. Nowotsch and M. Paulitsch, "Leveraging multi-core computing architectures in avionics," in *Dependable Computing Conference (EDCC), 2012 Ninth European*, 2012, pp. 132–143.
[29] S. Fisher, "Certifying applications in a multi-core environment: a new approach gains success," SYSGO AG, Tech. Rep.
[30] H. Kopetz, *On the Fault Hypothesis for a Safety-Critical Real-Time System*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006, vol. 4147, ch. 3, pp. 31–42.
[31] A. Crespo, I. Ripoll, and M. Masmano, "Partitioned embedded architecture based on hypervisor: The XtratuM approach," in *European Dependable Computing Conference (EDCC)*, 2010, pp. 67–72.
[32] IEC, "ISO 13849-1: Safety of machinery - safety-related parts of control systems ," p. 58, 2002.
[33] C. Helpa and H. Isakovic, "D3.5 - assesment of the MultiPARTES platform," TU Wien, Tech. Rep., 2013.