11th International TÜV Rheinland Symposium
„Functional Safety in Industrial Applications“
May 13 – 14, 2014, Cologne, Germany

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 research paper describes a safety concept for a wind turbine mixed-criticality control system based on multicore partitioning that meets IEC-61508 and ISO-13849 industrial safety standards. The safety concept has been reviewed and approved by a certification body.

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

I. Introduction

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 this approach. For example:

- Wind power: 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.
- Automotive: The software component in high-end cars currently totals around 20 million lines of code, deployed on as many as 70 ECUs (Electric Control Unit) that accounts for 30% of overall production costs [2]. The Volkswagen Phaeton has 61 ECUs, 11.136 electrical parts, 2.110 cables and 3.860 meters of cables with a weight of 64 kg [3].
- Railway: The ever increasing request for safety, better performance, energy efficient and cost reduction in modern railway trains have forced the introduction of sophisticated dependable embedded systems [4]. The number of ECUs within a train system is of the order of a few hundred [5, 6].

The integration of additional functionalities 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 [7].

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.

This publication contributes with the description of a safety concept of a wind turbine mixed-criticality control system based on multicore partitioning that meets IEC-61508 and ISO-13849 industrial safety standards. The safety concept considers 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 wind turbine control and protection system. Section V describes a safety concept based on well tried safety principles and solutions (common practice) and Section VI an equivalent safety concept based on multicore partitioning. Finally, Section VII draws the overall conclusion and future work.

II. Background

A. *Certification standards*

IEC-61508 [8-10] is a generic international safety standard from which different domain specific standards have been derived. Safety Integrity Level (SIL) is a discrete level corresponding to a range of safety integrity values where 4 is the highest level and 1 is the lowest. As a rule of thumb, the highest the SIL the highest the certification cost.

B. Hypervisor

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 [11] 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.

III. RELATED WORK

Recent analysis [12-17], research projects [18] and publications [19-23] 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 / or virtualization technology can support the development of integrated architectures in mixed-criticality platforms 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. Multicore microcontrollers with inbuilt safety enhancement features are also offering promising solutions [24]. However, the migration of real-time software and development of safety critical embedded systems based on multicore and virtualization technology is a challenge as stated also by different experts in the field [25-28]. 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 [29, 30].

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 [8, 9]. 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 [9].

Therefore, spatial and temporal isolation / independence are key requirements in mixed-criticality systems in order to ensure interference freeness among safety and non safety partitions, and interference freeness among safety partitions. 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 processors [27] could simplify the collection of evidences for a certification process, since 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 [29, 30].

The avionics industry has widely adopted the Integrated Modular Avionics (IMA) [31] architecture, which allows integrating several applications on a single processing element. 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 [14, 15, 32-37].

IV. WIND TURBINE CONTROL AND PROTECTION

A wind park is composed of interconnected wind turbines and a centralized wind park control centre as shown in Figure 1. 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

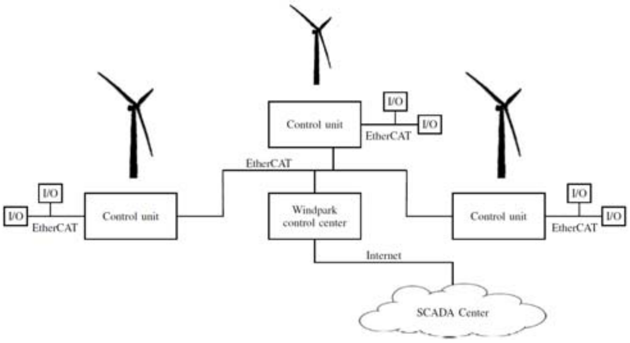


Figure 1: Simplified wind park diagram.

As shown in Figure 2, 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. The safety protection system must meet ’PLd’ level of ISO-13849 [38] and IEC-61508 SIL2/3.

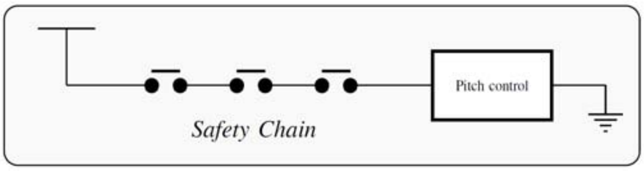


Figure 2: Wind turbine safety chain.

The ‘safety protection’ function 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. Most relevant simplified safety related requirements are described below:

- The 'safety protection' function must activate the 'safe state' if the 'rotation speed' exceeds the 'maximum rotation speed'

- The 'safety protection' function must ensure 'safe state' during system initialization (prior to the running state where rotation speeds are compared)
- 'Safety protection' function must be provided with a SIL3 integrity level (IEC-61508)
- The safe state is the de-energization of output 'safety relay(s)'
- Output 'safety relay(s)' is(/are) connected in serial within the safety chain
- A single fault does not lead to the loss of the safety function: HFT=1 and DC of the system >= 90\%.
- The Process Safety Time (PST) is 2 seconds. The reaction time must not exceed PST and detected 'severe errors' lead to a 'safe state' in less than PST
- The 'rotation speed' absolute measurement error must be equal or below 1 rpm to be used by 'safety protection'. If not it must be neglected
- The 'maximum rotation speed' must be configurable only during start-up (not running)

V. Safety Concept – Common Practice

This section describes a simplified safety concept for the “Wind Turbine control and protection” (Section IV) using well tried safety principles and solutions (common practice). The 1oo2 (D) dual channel architecture shown in Figure 4 is based on two independent processors, two shared diverse input sources (rotation speed) and two output relays connected in serial to the safety chain.

The safety node (SCPU) has a Hardware Fault Tolerance 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’ function. 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 functions only (’safety protection’ and ’diagnosis’) and ’P1’ processor executes all functions
- Each processor controls one independent safety relay
- One independent ’watchdog’ monitors each processor
- A ‘watchdog’ reset (e.g. due to timeout) implies de-energization of safety relays
- Local and cross-channel ’diagnosis’ are implemented in each processor
- EtherCAT ’communication stack’ is managed in P1 and the safety communication layer in ’safety protection’
- An IEC-61508 SIL2 system with HFT = 1 requires a Safe Failure Fraction (SFF) of 90% > SFF >= 60%

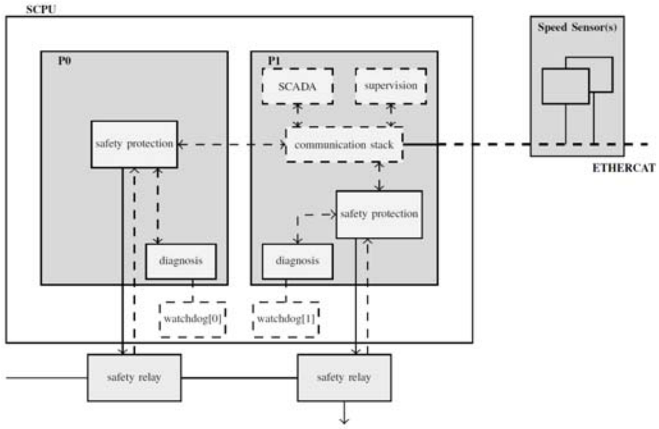


Figure 3: Safety concept (1002; 2 processors)

The future scalability of this approach is 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.).

VI. Safety Concept – Multicore Partitioning

This section describes the multicore partitioning based safety concept for the “Wind Turbine control and protection”, which aims to reach the same integrity level while providing a suitable solution that overcomes limitations described in the previous approach (Section V).

- The fault hypothesis identifies the assumptions regarding faults that the fault-tolerant safety system must tolerate (Section VI.A)
- The general description provides a textual description of the safety concept (Section VI.B)
- Most relevant safety techniques used to support the safety concept are described in Section VI.C
- The hypervisor compliant item is described in Section VI.D
- The overall diagnosis strategy is described in Section VI.E

A. Fault hypothesis

The fault-hypothesis [39] of this strategy consists of the following assumptions:

- Up to IEC-61508 SIL3 safety function(s).
- A single fault does not lead to the loss of the safety function: HFT=1 and Diagnostic Coverage (DC) of the system >= 90% (according to IEC-61508).
- The node (SCPU) forms a single Fault-Containment Region (FCR), can fail in an arbitrary failure mode

and meets IEC-61508-2 Annex E. The permanent failure rate is assumed to be in the order of 10-100 FIT (i.e. about one thousand years) and the transient failure rate is assumed to be in the order of 100.000 FIT (i.e. about one year), e.g. SEU [99].

- A heterogeneous quad-core processor is available (diverse core, x86 and LEON3 FT) and can fail in an arbitrary failure mode. Complete temporal isolation cannot be assured among cores (if the processor does not provide enough guarantees, sufficient evidence), but bounded temporal interference can be provided.
- The multicore processor (ASIC) is developed taking fault avoidance measures required for the targeted SIL (as described in IEC 61508-2 Annex F) into account. Parts may be integrated as Proven-In-Use IP or so called compliant items.
- A partition can fail in an arbitrary failure mode when it is affected by a fault, both in the temporal as well as the spatial domain.
- The hypervisor provides interference freeness (bounded time and spatial isolation) among partitions, fails in an arbitrary failure mode when it is affected by a fault and is a compliant item as defined in Section D.
- Safety over EtherCAT provides a safety communication layer with the required integrity level (SIL3) according to IEC-61784-3 [40].

B. *General description*

The safety concept is described in a top-down approach. Figure 4 shows a partitioned solution allocated to a heterogeneous quad-core processor, based on two LEON3 FT soft core processors and two x86 cores. The core allocation has been equivalent to the processor allocation described in the previous safety concept (Section V, Figure 3). Each functional group from Section V corresponds to one or more partitions, “Supervision” functional group is divided in multiple partition(s).

The hypervisor is a compliant item (see Section D) that ensures temporal and spatial isolation among partitions, ensuring interference freeness among safety and non safety partitions, and interference freeness among safety partitions. Partitioning and multicore allocation enables performance maximization and interference freeness.

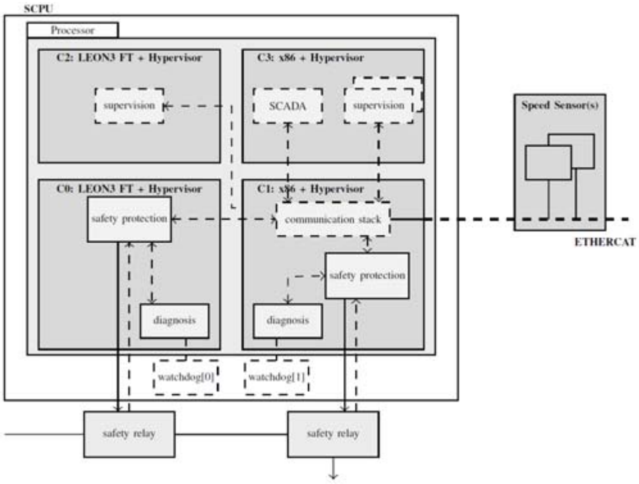


Figure 4: Simplified safety concept (1oo2, multicore).

Figure 5 shows the partitioned solution allocated to a heterogeneous quad-core processor with all hardware resources of relevance: communication buses, memory, shared resources, clocks and synchronization mechanisms, etc.

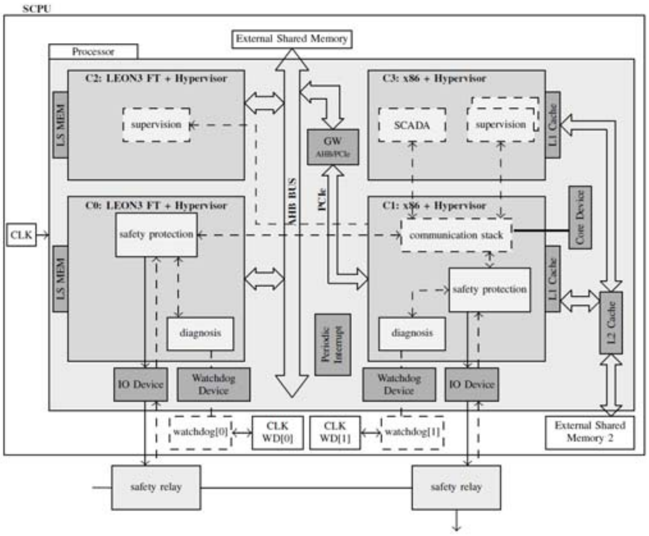


Figure 5: Safety concept (1oo2), multicore with shared resources).

The safety node (SCPU) supports / provides:

- The SCPU has two independent watchdogs with different clock sources controlled by the CPU. They reset the SCPU if not refreshed correctly.
- The SCPU has two external shared memories (‘External Shared Memory’ and ‘External Shared Memory 2’)
- The SCPU provides additional safety techniques for a IEC-61508 SIL3 (HFT = 1 and DC >= 90%), e.g. Power Failure Monitor (PFM)

- The SCPU has a heterogeneous multicore processor
- The SCPU meets IEC-61508-2 Annex E. (Note: If the (COTS) SCPU would not meet IEC-61508-2 Annex E, the core dedicated to safety partitions should be and independent processor within the same ECU)

The safety node (SCPU) processor supports / provides:

- The processor is quad-core, two LEON3 FT cores and two x86 cores (diverse). LEON3 FT and x86 cores are connected via PCIe gateway to AHB bus
- Each LEON3 FT core has internal L1 memory, not shared (‘LS MEM’). An AHB bus connects LEON3 FT cores and an external shared memory (‘External Shared Memory’)
- Each x86 core has internal L1 cache memory. All x86 cores share an L2 memory cache and an external shared memory (‘External Shared Memory 2’)
- The processor has an internal synchronization hardware that generates a periodic interrupt of configurable period to all cores.

A certifiable hypervisor according to Section D supports the processor and the configuration of the hypervisor ensures that:

- “Safety Protection” partitions control associated safety relays (command digital output and confirm state with digital input) and performs reciprocal comparison of results by software
- Diagnosis partitions perform checks of the CPU or possibly of I/Os in case they are not dedicated (exclusive access) to other partitions, using reciprocal comparison of results by software and control associated watchdog. Diagnosis partitions also manage health monitoring diagnosis information.
- The “communication stack” partition manages the EtherCAT communication bus
- Inter-partition communication is supported by the hypervisor using shared communication buses (e.g. PCIe<->AHB) and external shared memories.
- The system configuration is static and defined during the design stage: e.g. the allocation of ‘partitions’ and ‘system partitions’ to the platform; the configuration of ‘partitions’, ‘system partitions’ and ‘hypervisor’; the scheduling of partitions and resources; etc.

C. *Safety techniques*

The safety concept includes a detailed selection and analysis of safety techniques. This section summarizes most relevant safety techniques used to support the safety-concept:

- Measures to reduce the probability of systematic faults (Section VI.C.1)
- FMEAs, measures to control errors and system reaction to errors (Section VI.C.2))

1) Measures to reduce the probability of systematic faults

The usage of a Functional Safety Management (FSM) compliant with IEC-61508 and required SIL level:

- The overall system is conceived, developed and certified using a SIL3 FSM compliant with IEC-61508.
- The hypervisor meets the requirements of a certifiable hypervisor as described in Section D.
- Safety partitions are conceived, developed and certified using a SIL3 FSM compliant with IEC-61508
- Tools associated to the development, validation, verification, configuration and parameterization of safety partitions are qualified tools according to IEC-61508-3 (see chapter 7.4.4)
- The system configuration is static and defined during the design stage

2) FMEAs, measures to control errors and system reaction to errors

The safety concept includes detailed FMEAs, measures to control errors, error reaction definitions and it is complemented with a detailed assessment of the platform [41]. The overall diagnosis is based on the diagnosis strategy described in Section E.

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:

- 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
- 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
- Interrupts: Some interrupts in the Atom processor cannot be rerouted and this can influence the timing behaviour of the hypervisor, the maximum temporal interference suffered by a partition is measured
- 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.

Example fault avoidance techniques:

- 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 timeslot 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.

Example fault control techniques:

- Shared-resources: Safety partitions are executed in two diverse cores (’x86’ and ’LEON3 FT’) with different hypervisor configuration. Each ’diagnosis’ partition refreshes 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.

D. *Hypervisor as compliant item*

The strategy assumes that the hypervisor ported to the given platform is 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

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 these 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

VII. Conclusions 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 concept for a wind turbine mixed-criticality control system based on multicore partitioning.

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 unexpected violation occurs, diagnosis should lead the system to safe-state. Therefore, the lack of complete temporal isolation would reduce the availability of the system but should not jeopardize safety.

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.

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 Prof. Peter Puschner from 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] D. Buttle, "Real-Time in the Prime-Time - ECRTS (KEYNOTE TALK)," ETAS GmbH2012.

[3] J. Leohold, "Communication Requirements for Automotive Systems," in *5thIEEE Workshop on Factory Communication Systems (WCFS 2004)*, ed. Wien (Austria), 2004.

[4] ERRAC, "Joint Strategy for European Rail Research 2020," ERRAC - The European Rail Research Advisory Council2001.

[5] H. Kirrmann and P. A. Zuber, "The IEC/IEEE Train Communication Network," *IEEE Micro*, vol. vol. 21, no. 2, pp. 81-92, March/April 2001.

[6] F. Corbier, L. Kislin, and E. Fourgeau, "How Train Transportation Design Challenges can be addressed with Simulation-based Virtual Prototyping for Distributed Systems," in *3rdEuropean congress ERTS - Embedded Real Time Software*, Tolouse (France), 2006, p. 6.

[7] 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.

[8] IEC, "IEC 61508-1: Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 1: General requirements," ed, 2010.

[9] IEC, "IEC 61508-2: Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 2: Requirements for electrical / electronic / programmable electronic safety-related systems," ed, 2010.

[10] IEC, "IEC 61508-3: Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 3: Software requirements," ed, 2010.

[11] 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.

[12] "Mixed Criticality Systems," European ComissionFebruary 3 2012.

[13] "MULCORS - Use of Multicore Processors in airborne systems (Research Project EASA.2011/6)," EASA16th December 2012.

[14] EASA, "Certification Memorandum - Software Aspects of Certification - EASA CM – SWCEH – 002," 9th March 2013.

[15] EASA, "Development Assurance of Airborne Electronic Hardware," ed, 2011.

[16] S. Balacco and C. Rommel, "Next generation embedded hardware architectures: Driving Onset of Project Delays, Costs Overruns and Software Development Challenges," Klockwork, Inc.September 2010.

[17] "2013 - Embedded Market Study," UBM Tech2013.

[18] S. Trujillo, R. Obermaisser, K. Gruettner, F. Cazorla, and J. Perez, "European Project Cluster on Mixed-Criticality Systems," in *Design, Automation and Test in Europe (DATE) Workshop 3PMCES (Performance, Power and Predictability of Many-Core Embedded Systems)*, Dresden (Germany), 2014.

[19] M. S. Mollison, J. P. Erickson, J. H. Anderson, S. K. Baruah, and J. A. Scoredos, "Mixed-Criticality Real-Time Scheduling for Multicore Systems," presented at the Proceedings of the 2010 10th IEEE International Conference on Computer and Information Technology, 2010.

[20] R. Ernst, "Certification of trusted MPSoC platforms," in *MPSoC Forum*, 2010.

[21] 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.

[22] D. Gonzalez, J. M. Garate, A. Trapman, L. Monsalve, and S. Trujillo, "Mixed-Criticality in Wind Power: The MultiPARTES Approach," in *ESReDA Conference 2012*, Glasgow, UK, 2012, p. 9.

[23] X. Jean, M. Gatti, G. VBerthon, and M. Fumey, "The use of multicore processors in airborne systems," Thales Avionics2011.

[24] S. P. Brewerton, N. Willey, S. Gandhi, T. Rosenthal, C. Stellwag, and M. Lemerre, "Demonstration of Automotive Steering Column Lock using Multicore AutoSAR® Operating System," in *Society of Automotive Engineering (SAE)*, 2012.

[25] J. Schneider, M. Bohn, and R. Röbger, "Migration of Automotive Real-Time Software to Multicore Systems: First Steps towards an Automated Solution," in *22nd EUROMICRO Conference on Real-Time Systems*, Brussels, 2010.

[26] 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.

[27] 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.

[28] J. Abella, F. J. Cazorla, E. Quinones, A. Grasset, S. Yehia, P. Bonnot, *et al.*, "Towards improved survivability in safety-critical systems," in *IEEE 17th International On-Line Testing Symposium (IOLTS)*, 2011, pp. 240-245.

[29] 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.

[30] 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.

[31] "RTCA DO-297 Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations," ed, 2005.

[32] X. Jean, M. Gatti, G. VBerthon, and M. Fumey, "The use of multicore processors in airborne systems," Thales Avionics2011.

[33] 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," ed, 2013, pp. 970-975.

[34] L. M. Kinnan, "Use of multicore processors in avionics and its potential impact on implementation and certification," *SAE Technical Papers*, 2009.

[35] 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.

[36] J. Nowotsch and M. Paulitsch, "Leveraging Multi-core Computing Architectures in Avionics," in *Dependable Computing Conference (EDCC), 2012 Ninth European*, 2012, pp. 132-143.

[37] S. Fisher, "Certifying Applications in a Multi-Core Environment: a New Approach Gains Success," SYSGO AG.

[38] IEC, "ISO 13849-1: Safety of machinery - Safety-related parts of control systems —," ed, 2002, p. 58.

[39] H. Kopetz, "On the Fault Hypothesis for a Safety-Critical Real-Time System," in *Automotive Software - Connected Services in Mobile Networks*. vol. 4147, M. Broy, I. Krüger, and M. Meisinger, Eds., ed: Springer Berlin Heidelberg, 2006, pp. 31-42.

[40] *IEC-61784-3: Industrial communication networks - Profiles - Part 3-3: Functional safety fieldbuses* 2003.

[41] C. Helpa and H. Isakovic, "D3.5 - Assesment of the MultiPARTES platform," TU Wien2013.

11th International TÜV Rheinland Symposium
Functional Safety in Industrial Applications
May 13 – 14, 2014, Cologne - Germany

Jon Perez
IK-4 Ikerlan - Spain

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

Ein Sicherheitskonzept für Windkraftanlagen. Embedded Systeme unterschiedlicher Kritikalität basierend auf einer multicore Partitionierung.

IK4 IKERLAN
Research Alliance

TÜVRheinland®
Precisely Right.

Acknowledgement

- Research projects:

FP7 MULTIPARTES

FP7 DREAMS

FP7 PROXIMA

- Based on:
  - Perez J., Gonzalez D., Trujillo S. (IK4-Ikerlan)
  - Trapman T., Garate J. M. (Alstom Renewables)
  - “A safety concept for a wind power mixed-criticality embedded system based on multicore partitioning”, Workshop on Mixed Criticality Systems (WMC) at RTSS 2013
  - http://www-users.cs.york.ac.uk/~robdavis/wmc/paper15.pdf

Agenda

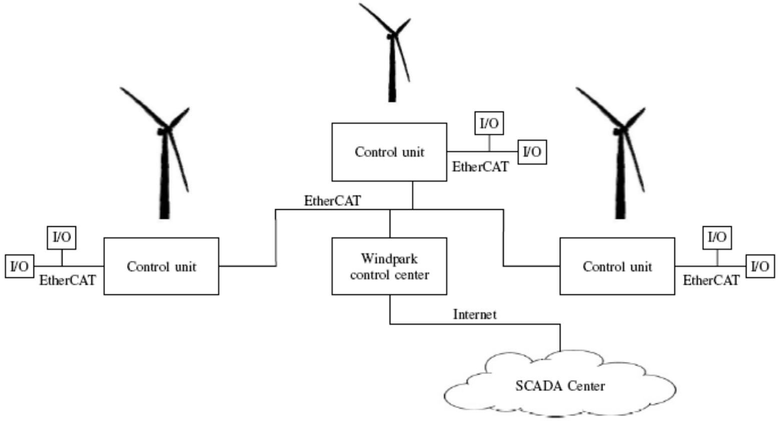
- Introduction
- Safety Concept
  - Requirements
  - Safety Concept (A) – 'Traditional Approach'
  - Safety Concept (B) – Multicore partitioning
    - Fault hypothesis
    - General description
    - Safety Techniques
- Conclusion
- Questions

Introduction - 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 presentation describes a safety concept for a wind turbine mixed-criticality control system based on multicore partitioning that meets IEC-61508 and ISO-13849 industrial safety standards.
- The safety concept has been reviewed and approved by a certification body.

Introduction - Off-shore Wind Turbine

- A modern off-shore wind turbine dependable control system manages [1]:
  - **I/Os**: up to three thousand inputs / outputs
  - **Function & Nodes**: several hundreds of functions distributed over several hundred of nodes
  - **Distributed**: grouped into eight subsystems interconnected with a fieldbus
  - **Software**: several hundred thousand lines of code



[1] Perez, Gonzalez et al.: "A safety concept for a wind power mixed-criticality embedded system based on multicore partitioning". Real Time Systems Symposium (RTSS) - MCS Workshop Vancouver, December 2013

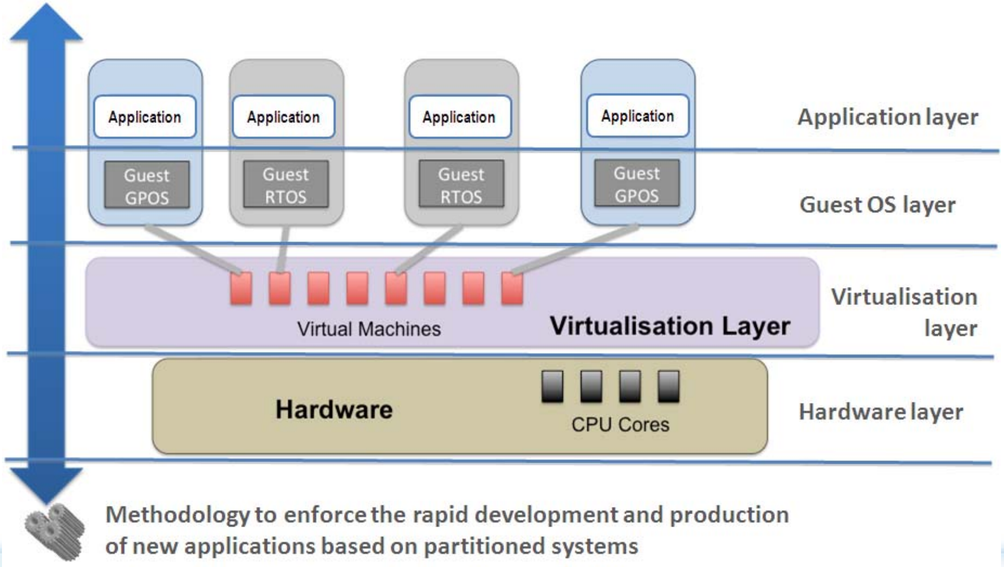
Introduction – Mixed Criticality

- "Modern electronic systems used in industry (avionics, automotive, etc) combine applications with different security, safety, and real-time requirements. Systems with such mixed requirements are often referred to as mixed-criticality systems"

  [Baumann, 2011]

- "*A mixed criticality system is a system that can execute several applications guaranteeing their mixed requirements of different real-time, security and safety*"

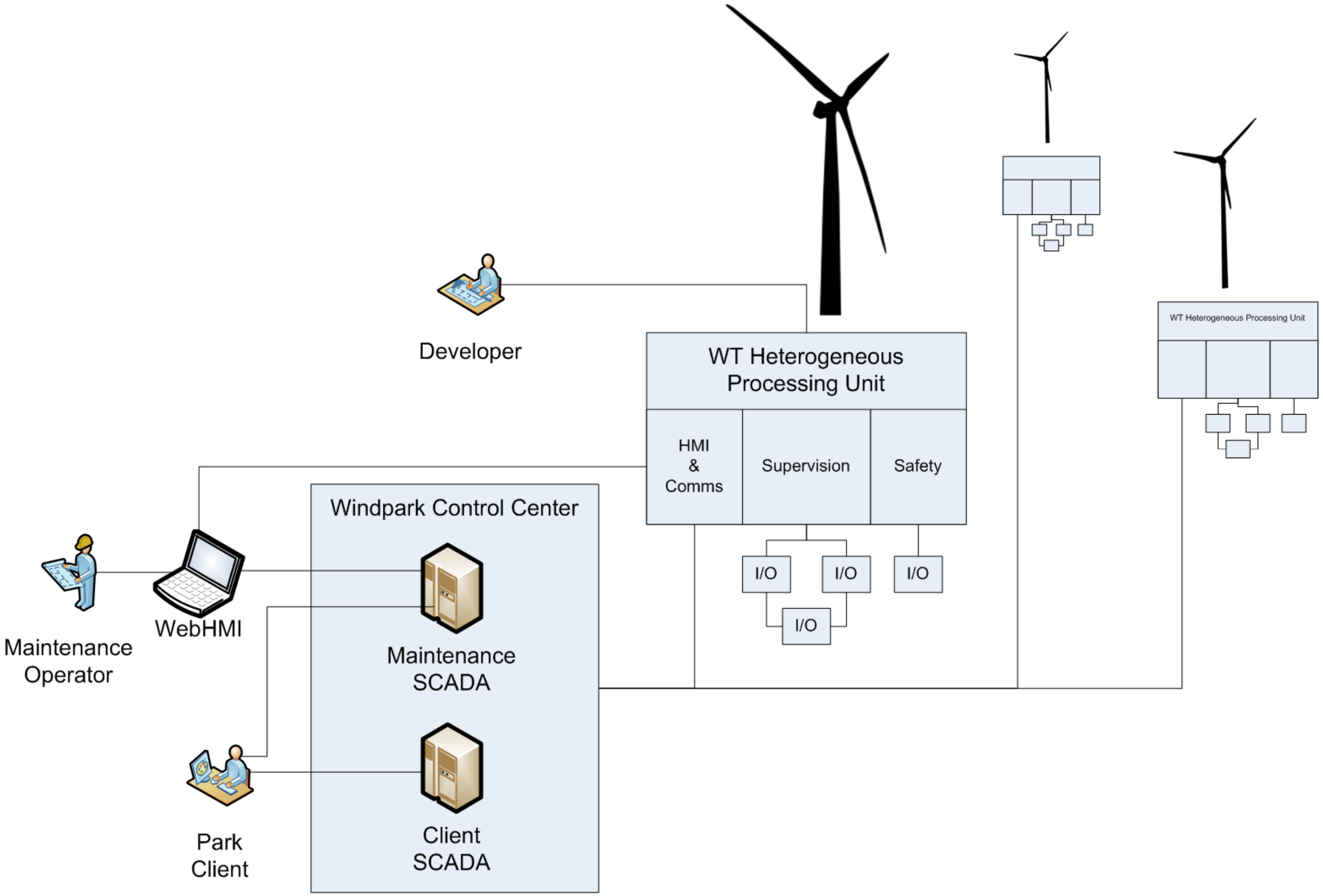
  [MultiPARTES, 2013]



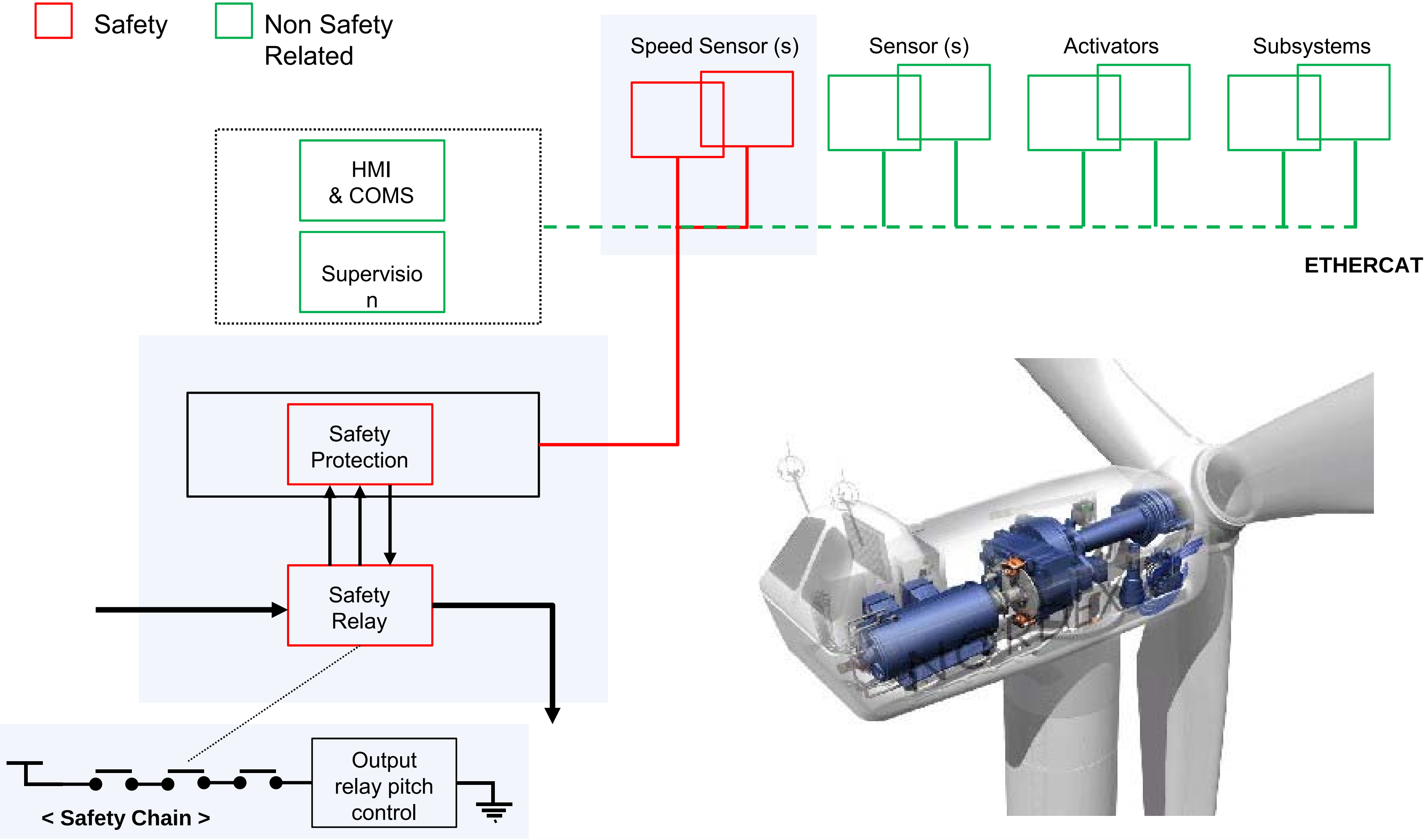
IK4 IKERLAN Research Alliance

TÜVRheinland® Precisely Right.

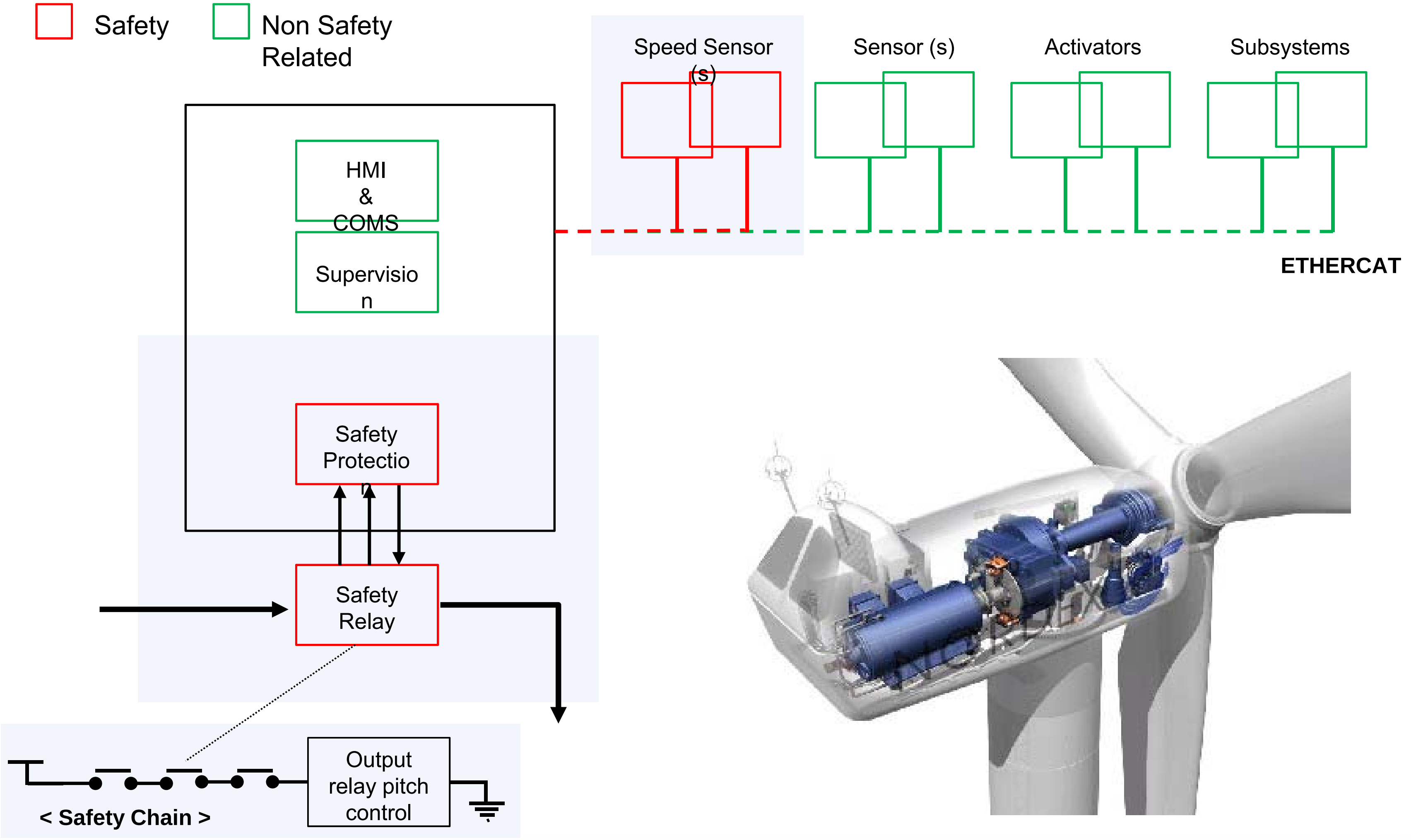
Introduction – Context Diagram



Introduction – Context Diagram



Introduction – Proposed Solution

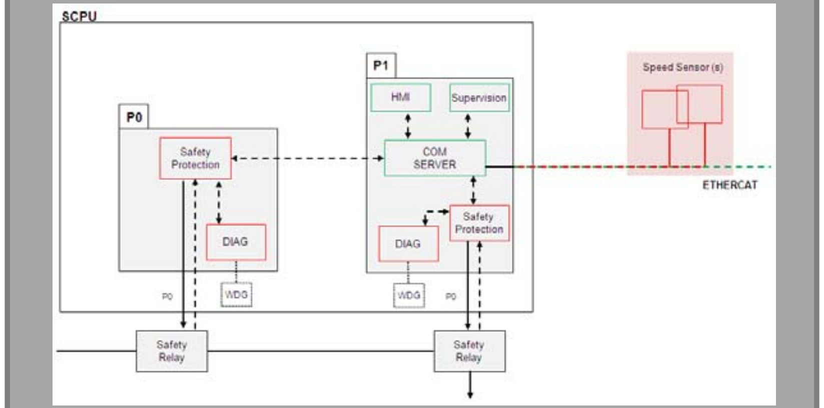


Safety Concept - Requirements

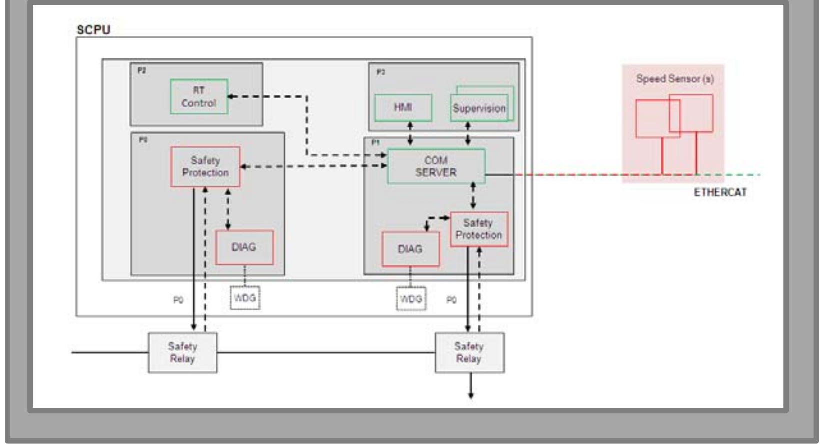
ID	Requirement
SR_WT_4	The <Protection System> safety function must activate the "safe state" if the "rotation speed" exceeds the "maximum rotation speed"
SR_WT_5	The <Protection System> safety function must ensure "safe state" during system initialization (prior to the running state where rotation speeds are compared)
SR_WT_6	<Protection System> safety function must be provided with a SIL3 integrity level (IEC-61508).
SR_WT_7	The safe state is the de-energization of output "safety relay(s)"
SR_WT_8	Output "safety relay(s)" is(/are) connected in serial within the safety chain.
SR_WT_9	A single fault does not lead to the loss of the safety function: HFT=1 and Diagnostic Coverage (DC) of the system >= 90% (according to IEC-61508).
SR_WT_10	The reaction time must not exceed PST (SW_WT_14)
SR_WT_11	Detected 'severe errors' lead to a "safe state" in less than PST (SW_WT_14).
SR_WT_12	The "rotation speed" absolute measurement error must be equal or below 1 rpm to be used by <Protection System>. If measurement error ≥ 1 rpm it must be neglected.
SR_WT_13	The "Maximum Rotation Speed" must be configurable only during start-up (not running).
SR_WT_14	The Process Safety Time (PST) is 2 seconds.

Safety Concept – The approach…

DUAL PROCESSOR – 1oo2



SINGLE PROCESSOR – 1oo2, partitioned, heterogeneous quad-core

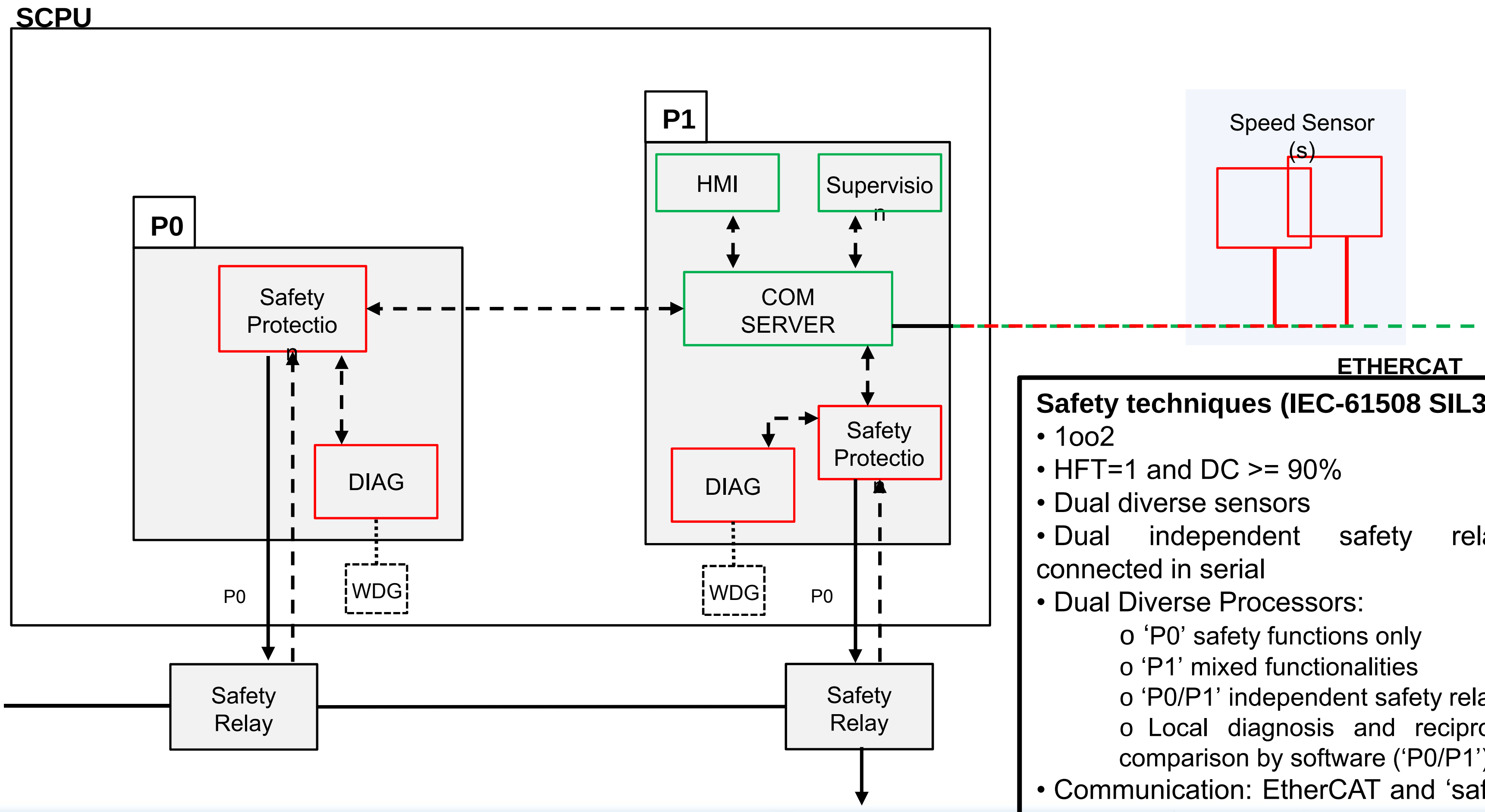


- Safety concept based on 'common practice in industry'
- Serves as a reference, not detailed

- Analogous safety concept using heterogeneous multicore and hypervisor
- The MultiPARTES contribution

Safety Concept (A – ‘Traditional’)

DUAL-PROCESSOR – 1oo2

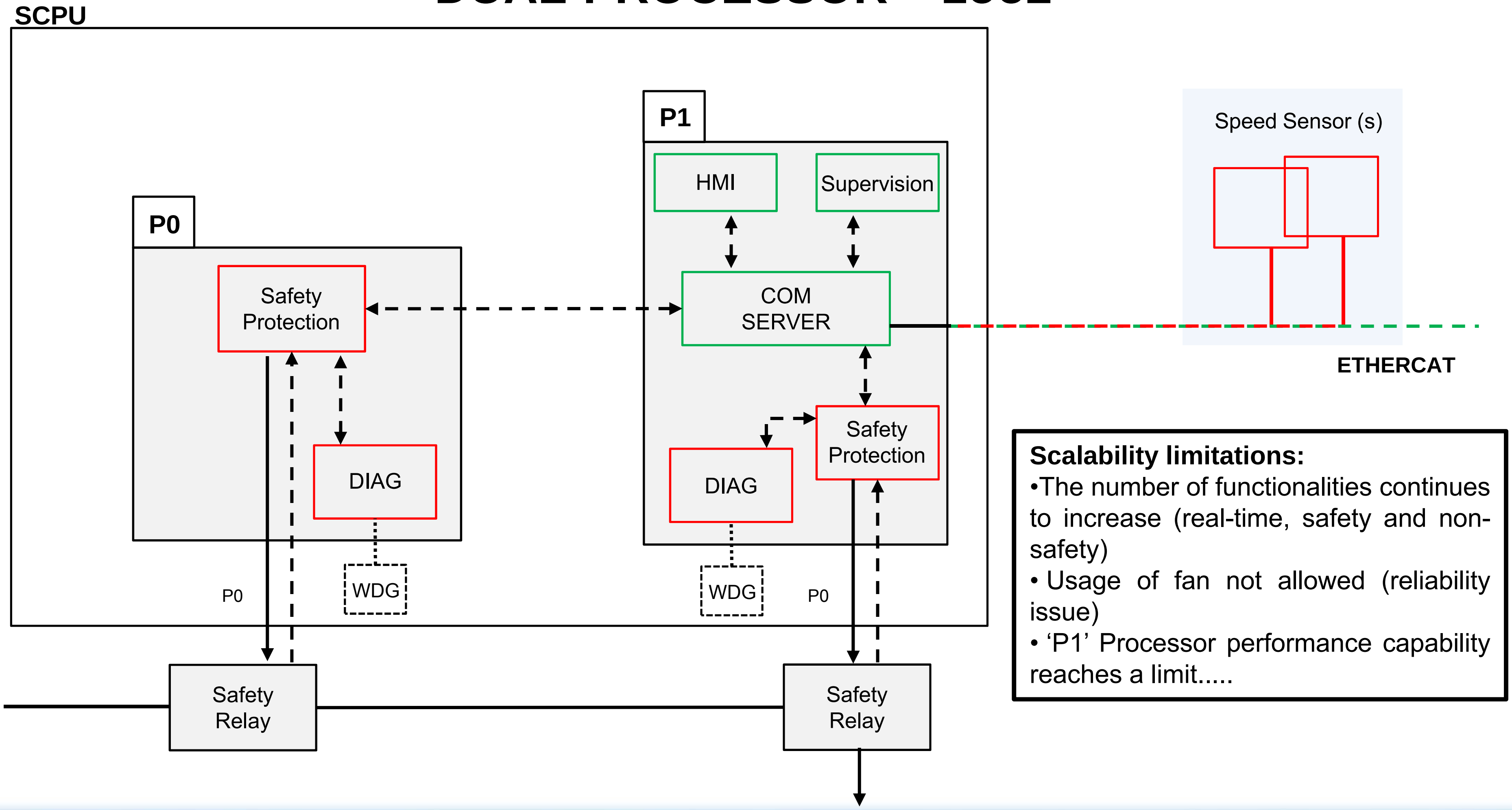


Safety techniques (IEC-61508 SIL3):

- 1oo2
- HFT=1 and DC >= 90%
- Dual diverse sensors
- Dual independent safety relays connected in serial
- Dual Diverse Processors:
  - o ‘P0’ safety functions only
  - o ‘P1’ mixed functionalities
  - o ‘P0/P1’ independent safety relay
  - o Local diagnosis and reciprocal comparison by software (‘P0/P1’)
- Communication: EtherCAT and ‘safety over EtherCAT’

Safety Concept (A – 'Traditional')

DUAL-PROCESSOR – 1oo2

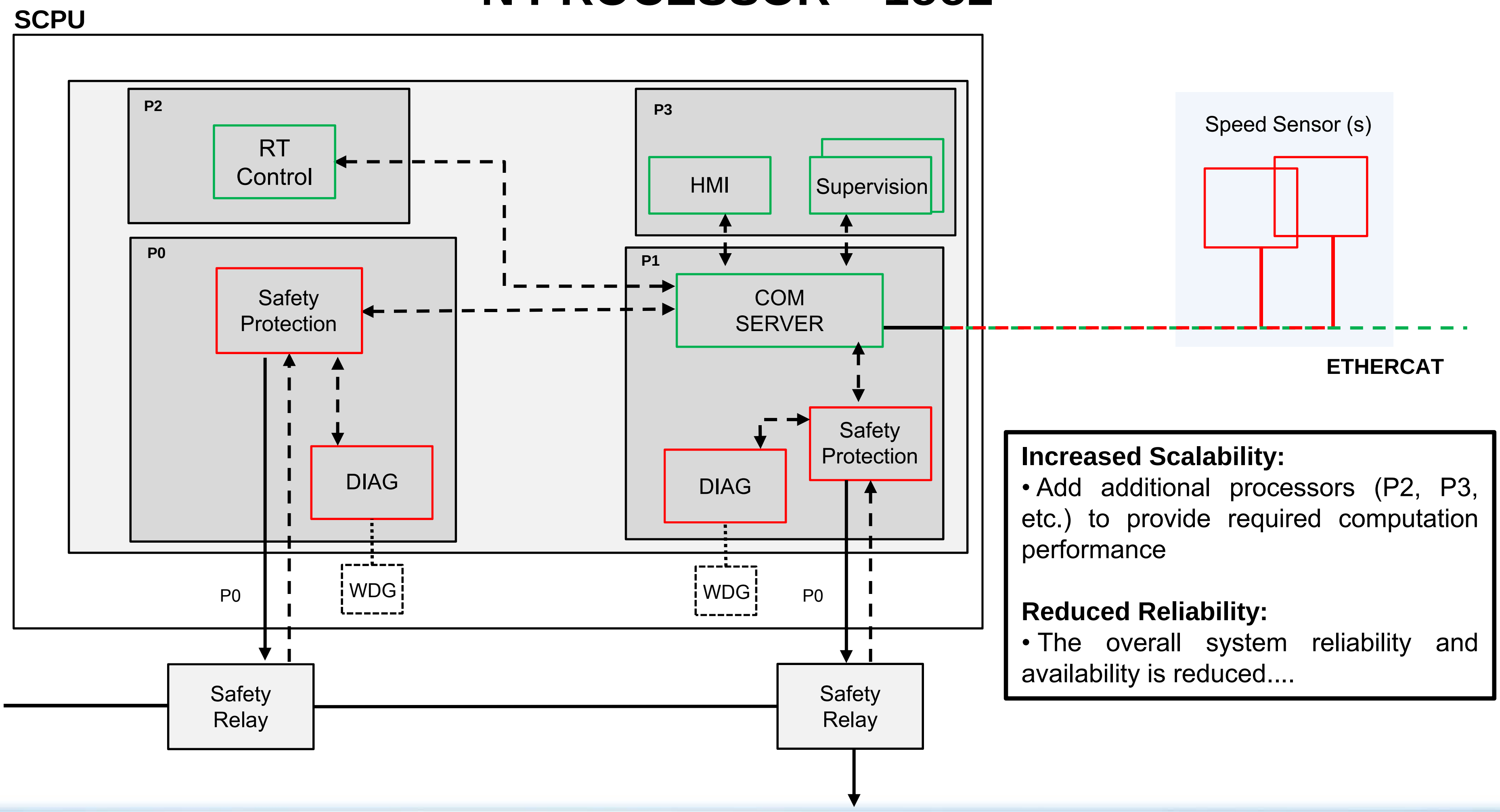


Scalability limitations:

- The number of functionalities continues to increase (real-time, safety and non-safety)
- Usage of fan not allowed (reliability issue)
- 'P1' Processor performance capability reaches a limit.....

Safety Concept (A – ‘Traditional’)

N PROCESSOR – 1oo2



Increased Scalability:

- Add additional processors (P2, P3, etc.) to provide required computation performance

Reduced Reliability:

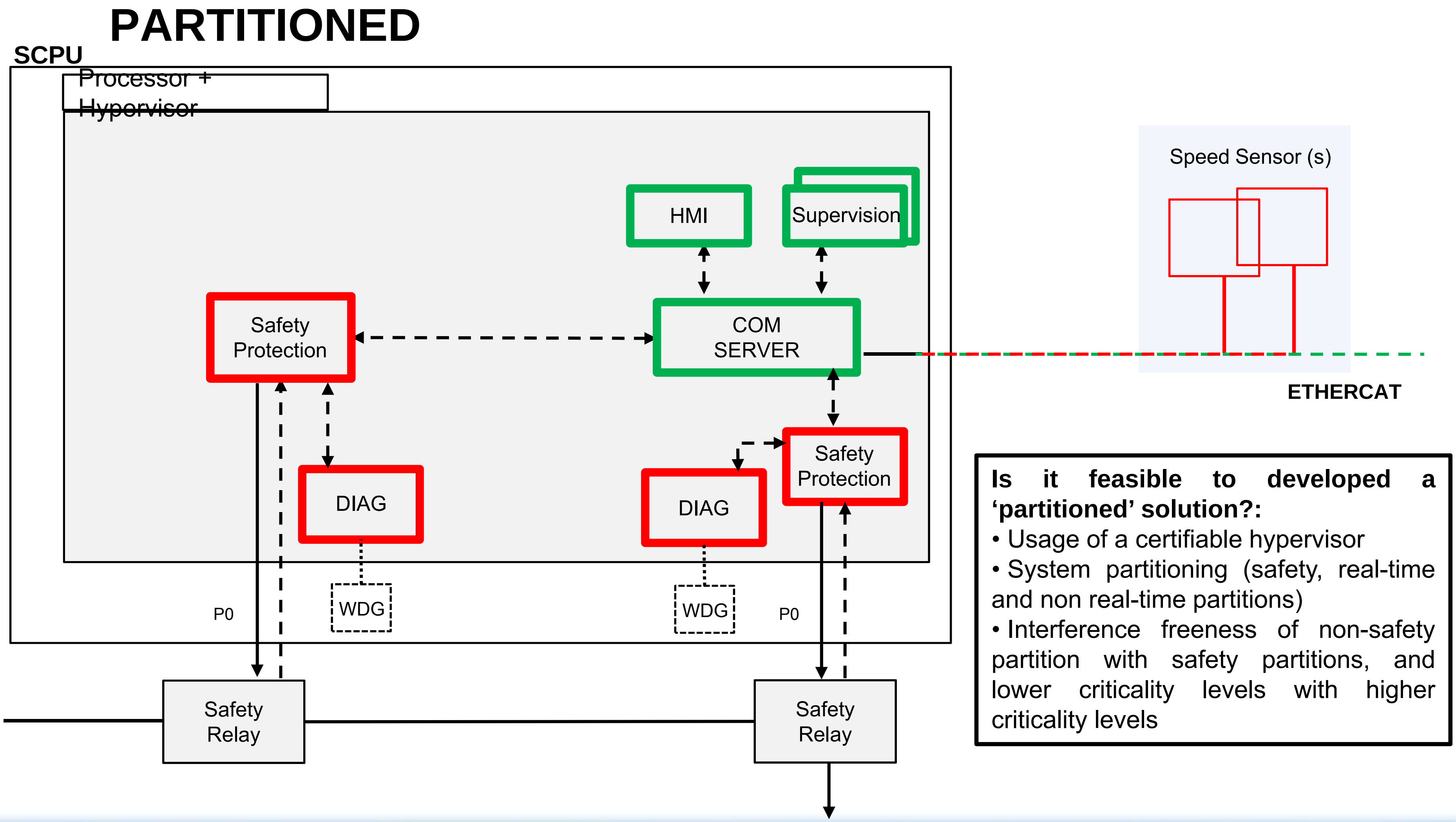
- The overall system reliability and availability is reduced....

Safety Concept (B – 'Multicore partitioning')

- The **fault-hypothesis** [1] 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 and the transient failure rate is assumed to be in the order of 100.000 FIT
  - **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 a compliant item and fails in an arbitrary failure mode when it is affected by a fault. Qualified tools.
  - **Partition**: A partition can fail in an arbitrary failure mode, both in the temporal as well as the spatial domain

[1] 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.

Safety Concept (B – ‘Multicore partitioning’) 1/3

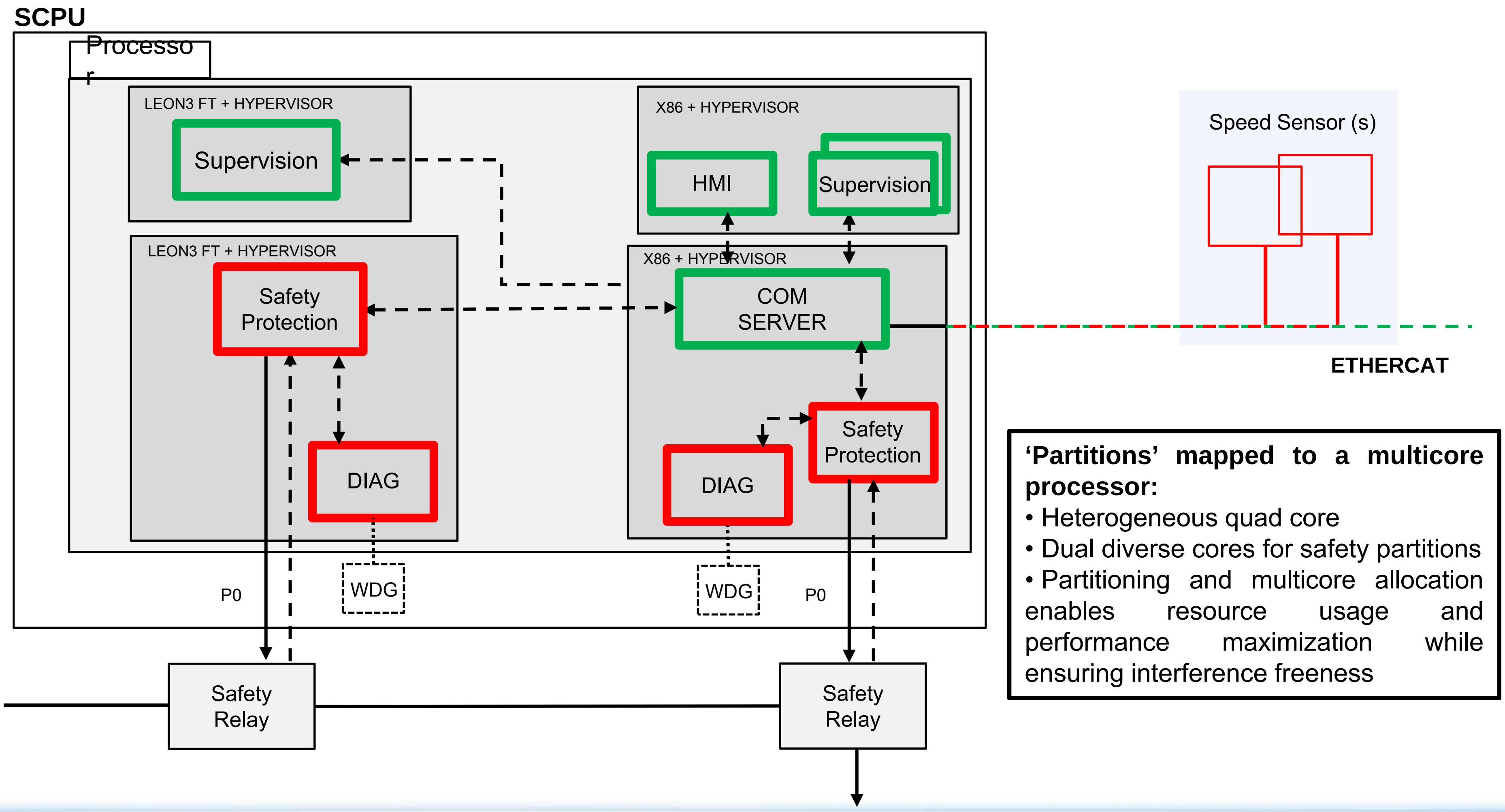


Is it feasible to developed a ‘partitioned’ solution?:

- Usage of a certifiable hypervisor
- System partitioning (safety, real-time and non real-time partitions)
- Interference freeness of non-safety partition with safety partitions, and lower criticality levels with higher criticality levels

Safety Concept (B – 'Multicore partitioning') 2/3

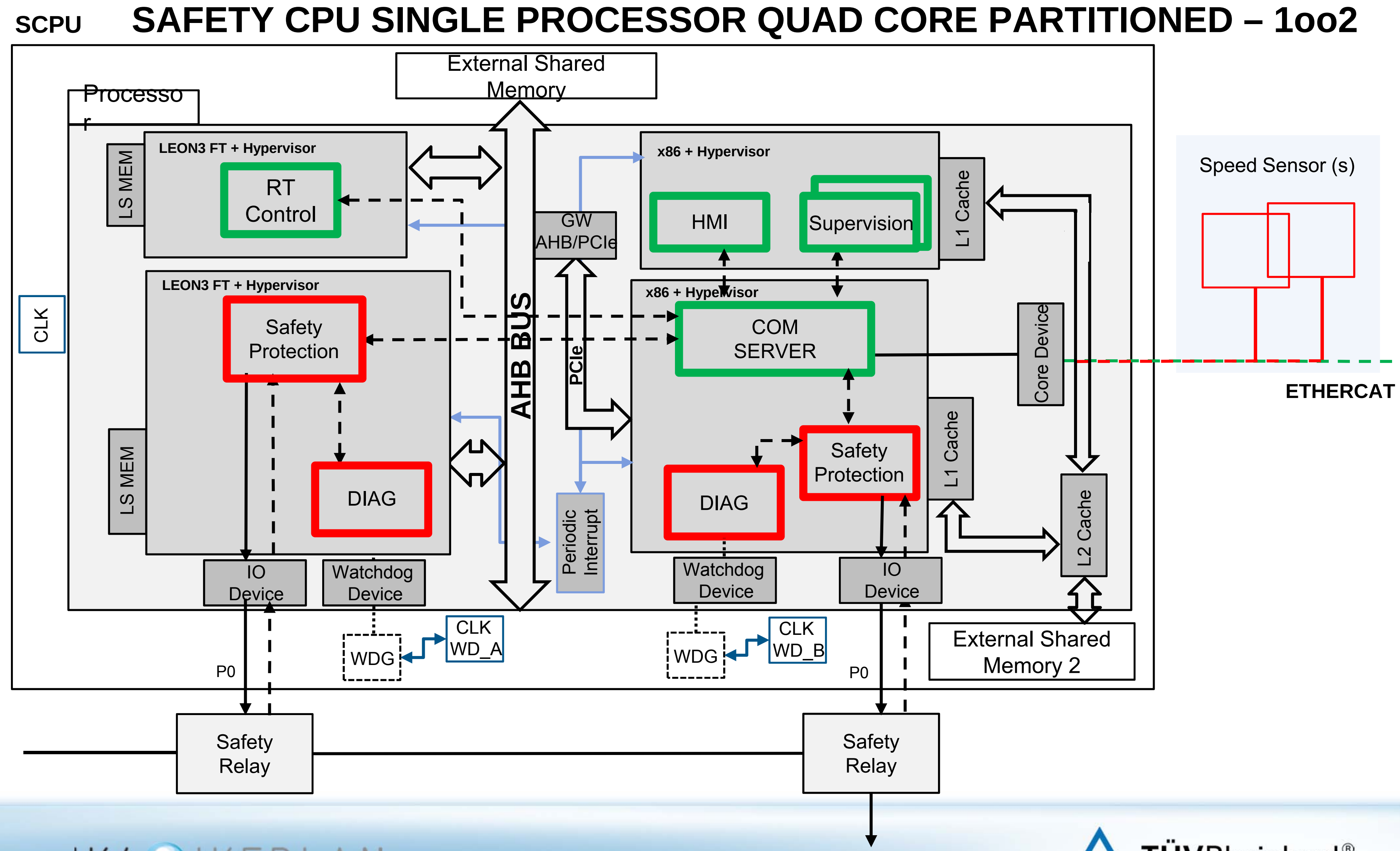
SAFETY CPU SINGLE PROCESSOR QUAD CORE PARTITIONED – 1oo2



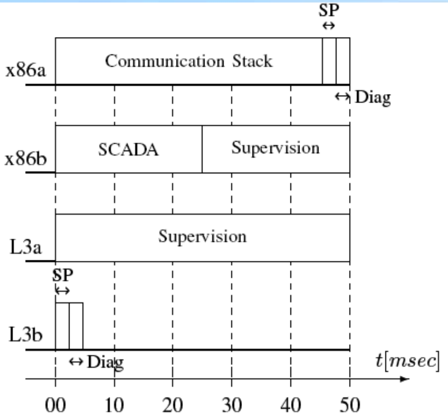
'Partitions' mapped to a multicore processor:

- Heterogeneous quad core
- Dual diverse cores for safety partitions
- Partitioning and multicore allocation enables resource usage and performance maximization while ensuring interference freeness

Safety Concept (B – ‘Multicore partitioning’) 3/3



Safety Concept (B – ‘Multicore partitioning’)



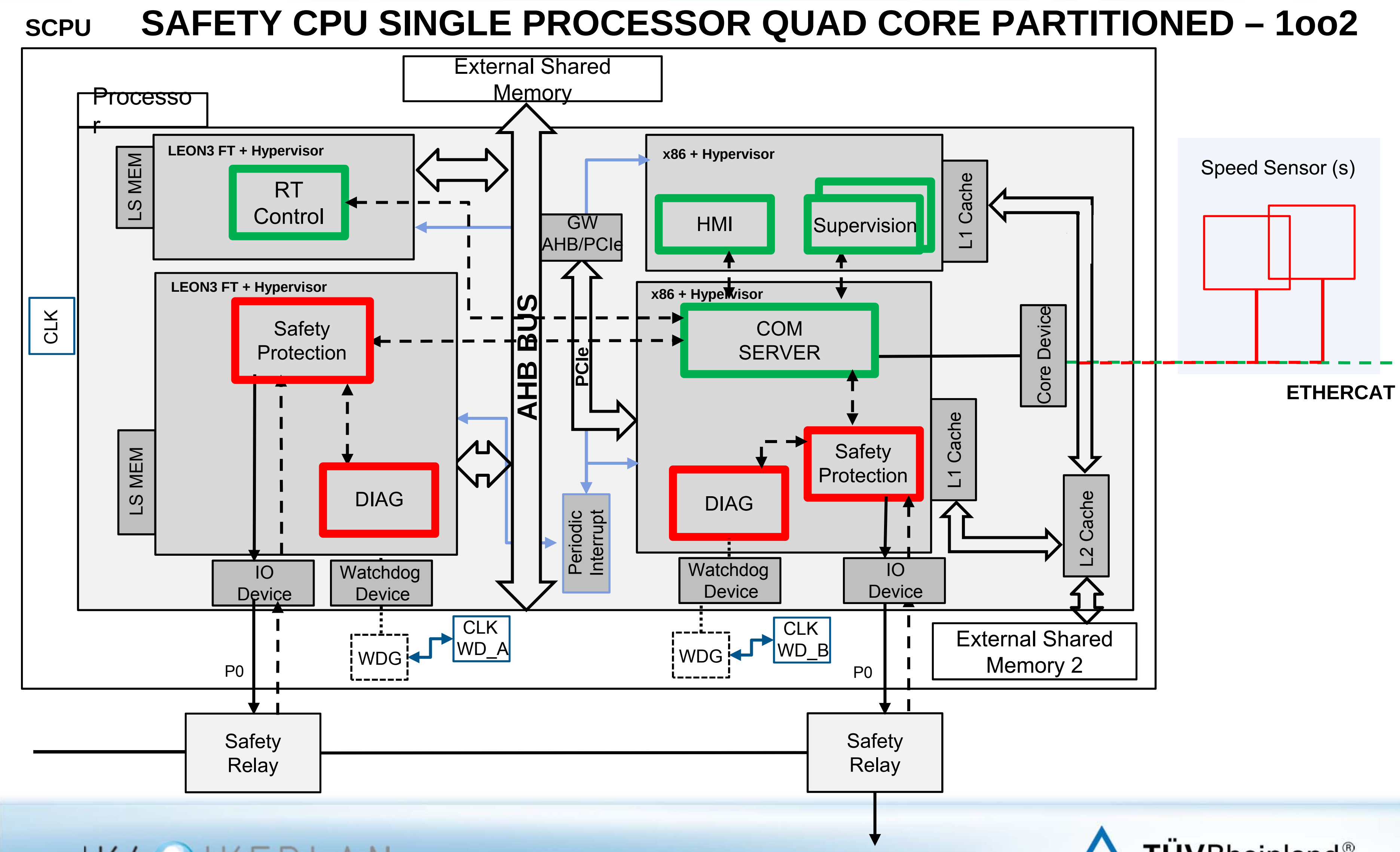
- Scheduling:
  - Static cyclic scheduling algorithm
  - pre-assigned guaranteed time slots
  - defined at design time
  - synchronized based on the global notion of time
- Diagnosis:
  - The partition should be self contained and should provide safety life-cycle related techniques and platform independent diagnosis abstracted from the details of the underlying platform
  - The hardware provides autonomous diagnosis and diagnosis components to be commanded by software
  - The hypervisor and associated diagnosis partitions should support platform related diagnosis
  - The system architect specifies and integrates additional diagnosis partitions required to develop a safe product taking into consideration all safety manuals

[1] 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.

Safety Concept (B – ‘Multicore partitioning’)

- Safety techniques:
  - Measures to reduce the probability of systematic faults
    - The overall system is developed and certified using a SIL3 FSM compliant with IEC-61508.
    - The hypervisor is a compliant item
    - Qualified tools according to IEC-61508-3 (see chapter 7.4.4)
  - Detailed FMEAs, measures to control errors and system reaction to errors

Safety Concept (B – ‘Multicore partitioning’) 3/3



Conclusions and future work

- It is possible….
- Mixed-criticality paradigm based on COTS multicore and partitioning provides multiple potential benefits but certification is a challenge
- Temporal independence must be met according to IEC-61508.
- The lack of complete temporal isolation could reduce the availability of the system but should not jeopardize safety (fault avoidance and control)
- 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.

Adobe Acrobat Document

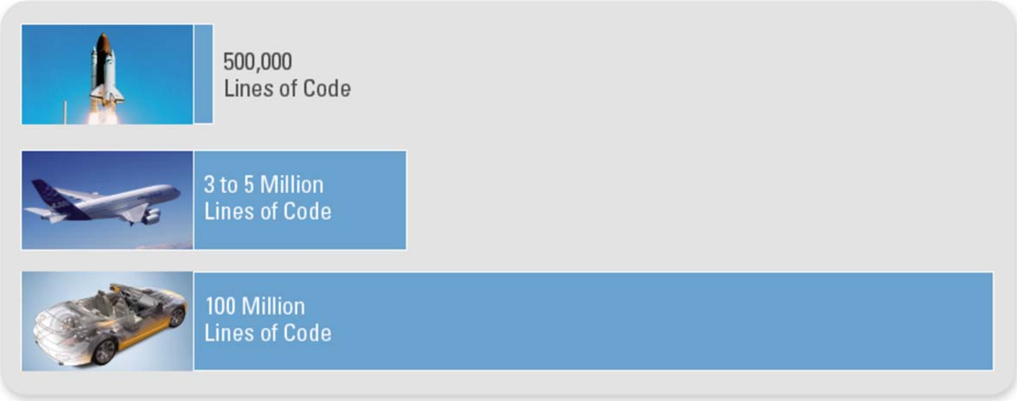
Questions

Backup slides

Introduction - Automotive

- Automotive domain:
  - The software component in high-end cars currently totals around 20 million lines of code, deployed on as many as 70 ECUs [1] that accounts for some 30% of overall production costs and is rising steadily [1]
  - a premium car implements about 270 functions that a user interacts with, deployed over 67 independent embedded platforms, amounting to about 65 megabytes of binary code [2]

[3]

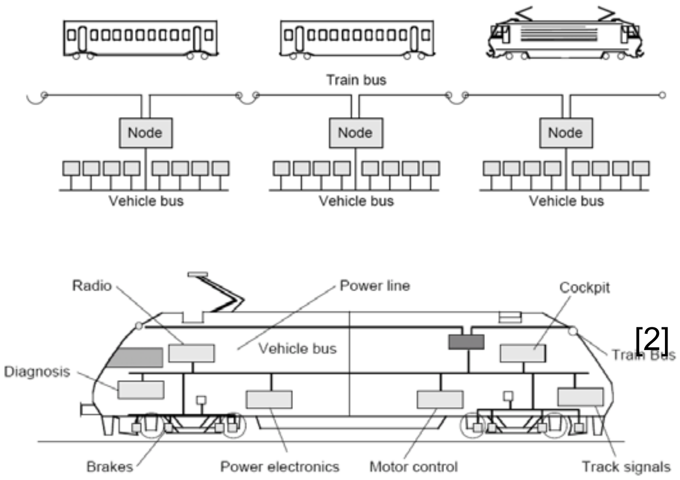


[4]

[1] Darren Buttle, ETAS GmbH, Germany, Real-Time in the Prime-Time, ECRTS (KEYNOTE TALK), 2012
[2] Christian Salzmann and Thomas Stauner. Automotive software engineering. In Languages for System Specification, pages 333–347. Springer US, 2004.
[3] Leohold, J. Communication Requirements for Automotive Systems. 5thIEEE Workshop on Factory Communication Systems (WCFS). Wien, 2004
[4] National Instruments, How engineers are reinventing the automobile,, http://www.ni.com/newsletter/51684/en/ , 2013

Introduction - Railway

- Railway domain (on-board):
  - The ever increasing request for safety, better performance, energy efficient, environmentally friendly and cost reduction in modern railway trains have forced the introduction of sophisticated dependable embedded systems [1]
  - The number of ECUs (Electric Control Units) within a train system is of the order of a few hundred [2,3]
  - Groups of distributed embedded systems:
    - Train Control Unit
    - Railway Signalling (e.g. ETCS)
    - Traction Control
    - Brake Control
    - Etc.



[2]

[1] The European Rail Research Advisory Council (ERRAC), Joint Strategy for European Rail Research 2020

[2] Kirrmann, H. and P. A. Zuber (2001). "The IEC/IEEE Train Communication Network." IEEE Micro vol. 21, no. 2: 81-92.

[3] F. Corbier, et al, *How Train Transportation Design Challenges can be addressed with Simulation-based Virtual Prototyping for Distributed Systems,* 3rdEuropean congress Embedded Real Time Software (ERTS), France, 2006

Introduction - Opportunities

- 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 [1]:
  - A considerable complexity growth [2]
  - Increase number of subsystems -> increase cost, size, weight and power consumption
  - Increase number of connectors and cables -> reliability reduction, e.g. automotive field, 30-60% of electrical failures attributed to connectors [3]
  - Scalability limitations
- On the other hand, COTS multicore and virtualization technology could enable an integrated architectural approach ..... Mixed-criticality.

[1] Perez, Gonzalez et al.: "A safety concept for a wind power mixed-criticality embedded system based on multicore partitioning". Real Time Systems Symposium (RTSS) - MCS Workshop Vancouver, December 2013

[2] H. Kopetz. The complexity challenge in embedded system design. In 11th IEEE International Symposium on Object Oriented Real-Time Distributed Computing (ISORC), pages 3–12, 2008.

[3] 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.