DIGITAL INDUSTRIES SOFTWARE

Methodologies for achieving functional safety using AUTOSAR in E/E architecture

Executive summary
As the prevalence of automated driving, electrification, and connected vehicle applications increases, vehicle safety requirements are becoming more stringent. At the same time, electrical and electronic (E/E) vehicle architecture complexity is increasing. As these trends converge, solution architects and engineers are looking for ways to manage design complexity, meet design constraints for functional safety, achieve international compliance, and deliver a rich and flexible solution.

Dr. Ahmed Majeed Khan, Senior Product Manager, Integrated Electrical Systems (IES) Embedded/ AUTOSAR, and Jeffrey Hancock, Senior Product Managers, Embedded Platform Solutions, Siemens

SIEMENS

Contents

Introduction

As the prevalence of automated driving, electrification, and connected vehicle applications increases, vehicle safety requirements are becoming more stringent. At the same time, electrical and electronic (E/E) vehicle architecture complexity is increasing. As these trends converge, solution architects and engineers are looking for ways to manage design complexity, meet design constraints for functional safety, achieve international compliance, and deliver a rich and flexible solution.

The Siemens' AUTOSAR solution is at the forefront of addressing these challenges. Capital® VSTAR™ is Siemens' implementation of the AUTOSAR standard and is a complete offering with tools and software platform to meet all ECU platform needs. Capital VSTAR enables a holistic approach for achieving functional safety and helping E/E software developers meet new and evolving requirements for automotive software.

In this whitepaper, we'll explore:

- A comparison of the 2011 and 2018 releases of the ISO 26262 "Road Vehicles – Functional Safety" standard
- Design considerations for AUTOSAR-based electronic control units (ECUs)
- Probable fault scenarios and strategies to eliminate interference among software components and functions with different safety integrity levels (ASIL)
- How a multicore framework enables Mixed Safety Criticality on a multicore system-on-chip (SoC) by maintaining separate domains via interprocessor communications

Mega trends in the automotive industry

The automotive industry is being driven by four key mega trends:

1. **Autonomous:** Today's autonomous vehicles require the interoperability of complex functions that only E/E design can accomplish. As such, the market for E/E design is growing at a CAGR of 7% and is projected to reach $469 billion by 2030.
2. **Connected:** Connected vehicles require a focus on security and data privacy, as each autonomous car produces 4 terabytes of data every day.
3. **Electrified:** For electric vehicles, performance hinges on battery weight and range. As performance challenges are overcome, popularity will increase, and electric vehicles could account for as much as 55% of new car sales by 2030.
4. **Shared Mobility:** Environmental initiatives will continue to emphasize shared mobility, but operational uptime and user availability will be key to success. McKinsey predicts this market segment will generate $1.5 trillion in services revenue by 2030.

These numbers underscore an enormous opportunity – and an existential threat. Market players must be disruptors or have the ability to respond quickly to changing market conditions and trends.

Safety and security are of paramount Importance

Today's automotive vehicles consist of numerous complex systems, and any system failure or misfire can lead to serious consequences. Not only can automotive manufactures face costly lawsuits and reputational damage, peoples' lives are at stake. As such, automakers place functional safety at the top of their list of priorities in every phase of development, beginning with E/E system design.

To that end, AUTOSAR was created. AUTOSAR is a global partnership of more than 300 leading companies in the automotive and software industry that has established a standardized software framework and open system architecture for intelligent mobility. The organization brings together 31 International automotive OEMs, including the 20 top-selling OEMs. Together with other Tier1 players and suppliers, these AUTOSAR partners are collaborating to shape the future of intelligent mobility, and working to define an automotive open system architecture standard to support the needs of future automotive applications.

AUTOSAR ensures that there are standardized interfaces in the application layer to enable software components to communicate via specific ports using a virtual function bus. The framework supports the development of safety-related systems by using various safety measures and mechanisms.

ISO26262 specifies protocols for capturing system requirements, coding and development, as well as test strategy and execution and system documentation. It provides the means to verify that teams are following repeatable, reliable design and test processes.

The standard is not static, however; in recent years, many changes have been introduced. From a software development standpoint, the most recent modification pertains to safety content, new technologies, and a shift in focus:

- **Safety content:** The requirement for more safety content, such as safety management and plan. Repetitions of "refined" work products have been removed.
- **New Technologies:** Changes related to the introduction of multicore, model-based development and Agile methodologies.
- **Shift in Focus:** More focus has been placed on software architecture, software safety analysis, failure analysis, fault injection, and coupling factors.

Methodologies for ensuring functional safety

The AUTOSAR standard is constantly evolving to accommodate state-of-the-art industry requirements. Let's take a look at the methodologies it provides to comply with the safety landscape:

- **Memory partitioning:** AUTOSAR separates the software components so that none of those can change or corrupt the memory associated with others (see Figure 1). The modules from the basic software (BSW) layer below the RTE are separated by ASIL neads in the model. Depending on the assigned ASIL, each software component is grouped into partitions. A multicore system can be used to increase the possible allocation of features to an ECU, and AUTOSAR supports this use case. A memory-partitioned system can be deployed on a multicore MCU. Context switches between cores can be aligned with context switches between memory partitions, enabling implementations for use cases in which BSW functions are provided separately at each core, while providing the functionality at many cores (see Figures 2,3).

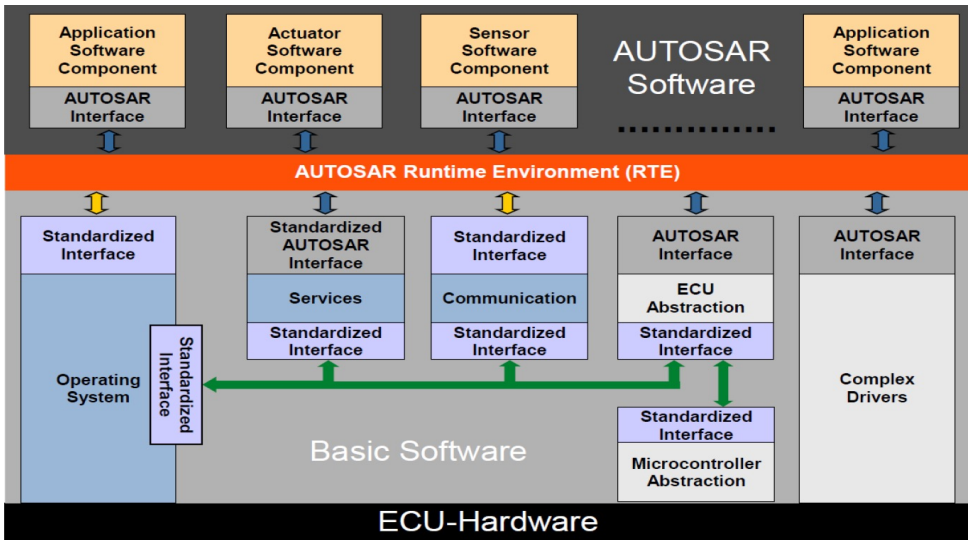


Figure 1: Spatial isolation at the memory level.

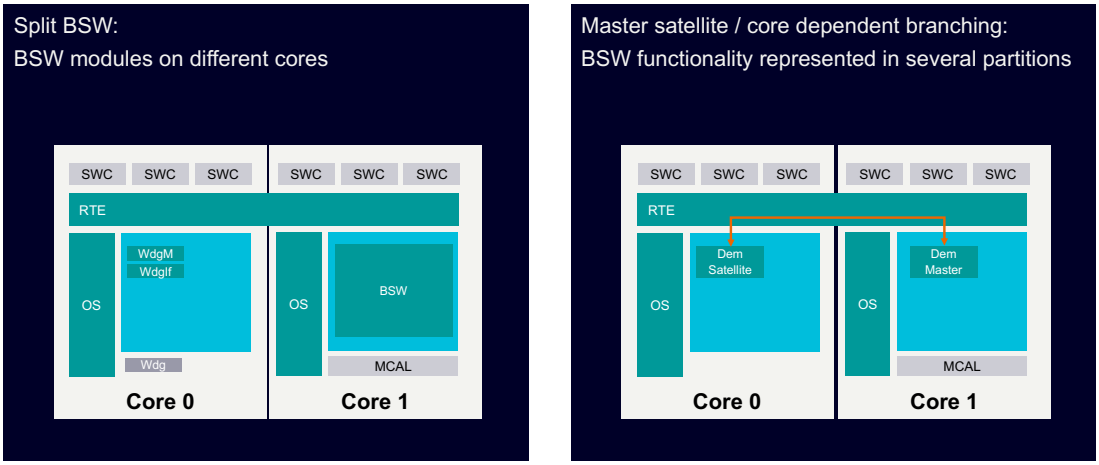


Figure 3: Spatial isolation at the core level.

- **Freedom from interference:** Freedom from interference between different ASILs is the foundation for a safety system (See Figure 4). It relies on the separation in spatial aspects provided by a safe memory, as we just saw. The other main providers are temporal separation, providing safe execution and data integrity, provided by safe communication. An automotive system, however, also relies on synchronized mode management. And to ensure that the system is operating on the same conditions, it is essential to provide safe synchronization through properly protected and propagated system states. In an AUTOSAR system, this is insured by the runtime environment, basic software mode manager and ECU state manager.

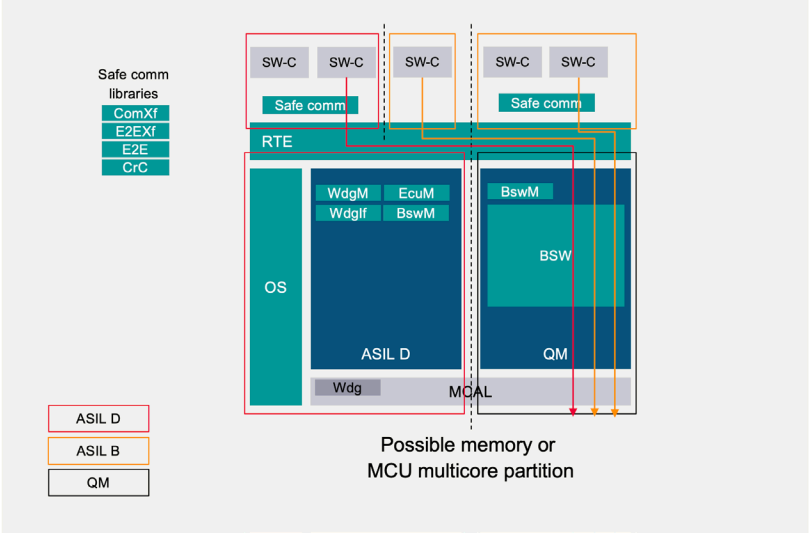


Figure 4: Freedom from Interference.

- **Safe communication:** Safe communication requires integrity of data exchange between senders and receivers (see Figure 5). Therefore, data protection mechanisms are required to safeguard against faults in communication links. Examples for such faults include random hardware faults (e.g. corrupt registers of a CAN transceiver), interference due to EMC, and systematic faults. The end-to-end communication protection library in AUTOSAR helps detect and handle errors in communication links at runtime, which satisfies requirements for ASIL D-qualified safe communications.

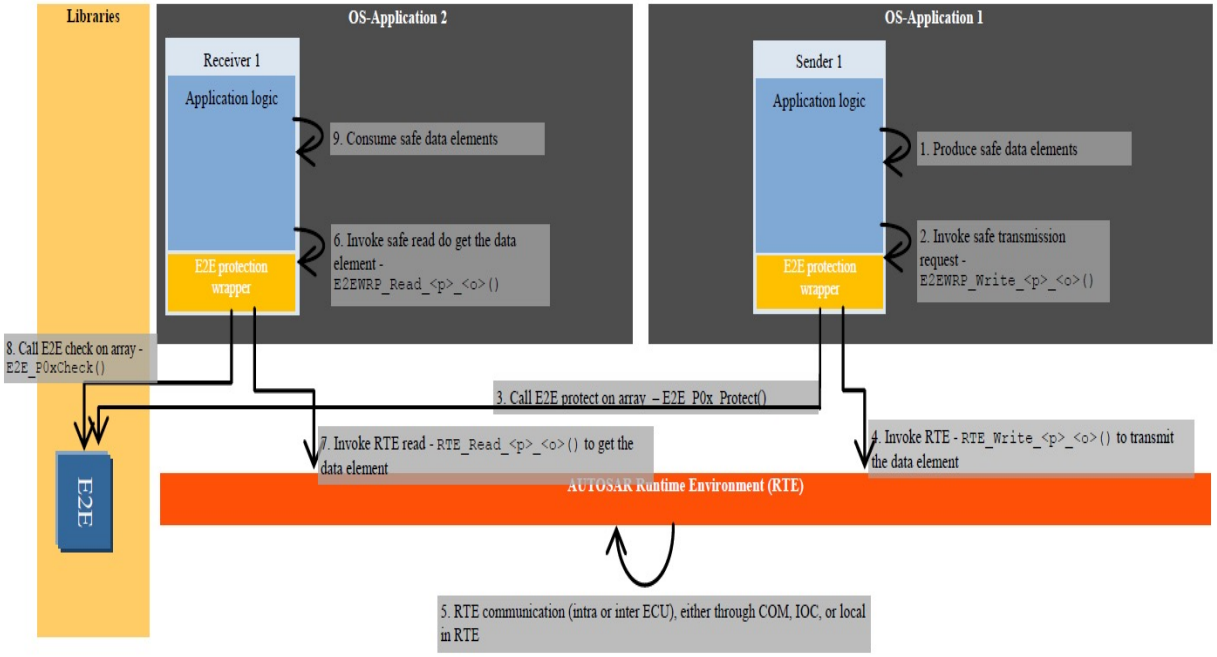


Figure 4: End-to-end communication protection.

- **Temporal isolation:** For safe execution, we determine the timing such that the system's action and reactions are performed within the allocated slots (see Figure 5). The right time can be described in terms of a set of timing constraints. Temporal partitioning is not enforced in AUTOSAR, due to the fixed-priority preemptive scheduler. However, the OS provides mechanisms for timing faults. The timing protection budget is a combination of execution time, resource lock time, and inter-arrival time budgets, which are configured statically. With this mechanism, the interference between tasks is minimal, meeting ISO 26262 requirements for the absence of error propagation (see Figure 5). Worst-case execution time, or WCET, is the longest execution time for a task on a target procession. For critical tasks with higher ASIL requirements, developers will spend more effort to improve the WCET.

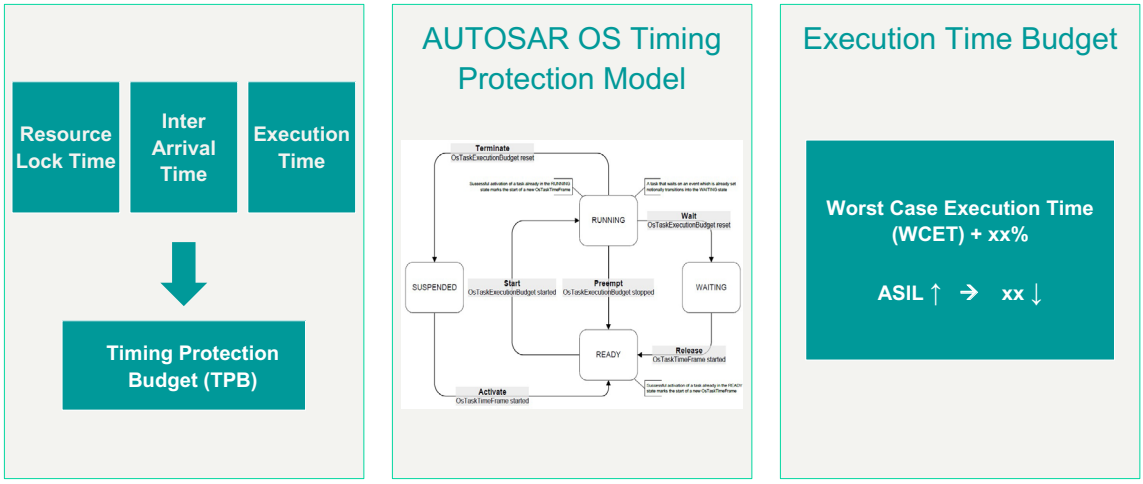


Figure 4: End-to-end communication protection.

- **Schedulabilty:** Embedded systems often have real-time constraints. Soft real-time systems can miss their deadline once in a while. However, when hard real-time systems that are safety-critical miss their deadlines, they can fail, resulting in the loss of human lives or environmental hazards. OS tasks scheduled by algorithms such as "rate monotonic" or "earliest deadline" help realize a schedulability analysis, such that the tasks meet their deadline on a processor. Apart from the software execution path, the resulting timing analysis is also affected by a processor architecture, along with a number of cores or partitions. AUTOSAR supports an OS provisioning, which relies on scheduling tables for reliable task execution.

All of these methodologies require extensive tooling support. For example, one of the most challenging configurations is bus communication distributed over a multicore architecture, which has thousands of signals and hundreds of PDUs. An adopted AUTOSAR-based workflow ensures that the system behaves as expected (see Figure 6). In addition to the automated workflow, it is also essential to ensure setup is correct and that the engineering tasks have been implemented successfully, including the correct assignment to ASIL partitions, where different views provide ways to analyze dependencies.

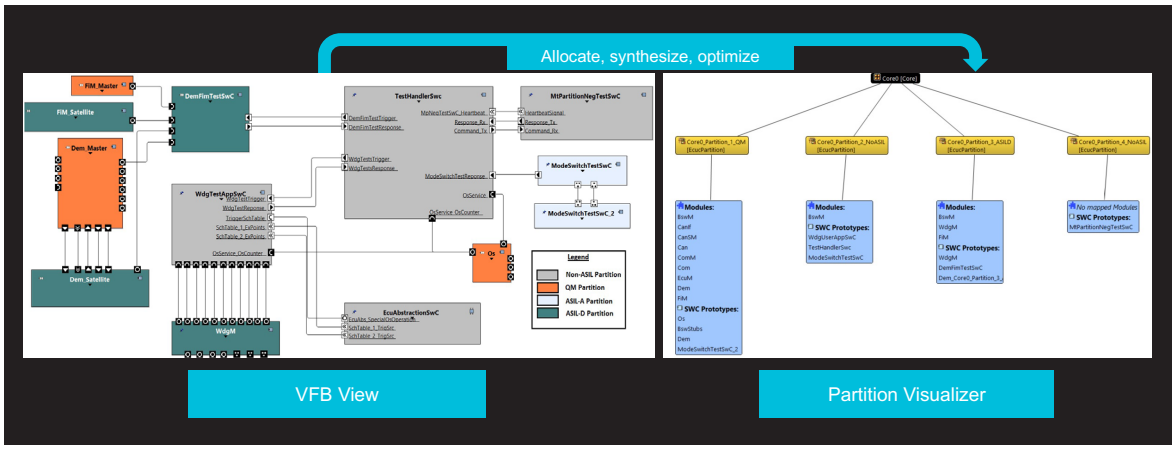


Figure 6: Tooling support for distribution of BSWs and SWCs on different partitions.

Ensuring system consistency

A final consideration is system consistency. The model-based systems engineering approach is a multi-stage development process of the preliminary system that brings together all aspects of the ISO26262 specification – system, hardware and software development – to satisfy the functional safety goals that are elicited in the system risk analysis. The first stage is identifying key requirements for carrying out multi-domain system modeling.

Once that multi-domain model is complete, the E/E aspects can be extracted. This allows developers to build a functional definition of the E/E system and to define the system architecture to allocate functions correctly, down to the full embedded software flow through the software running on an ECU. The flow should also cover the electrical system design harness engineering and publication creation. Continuous verification and validation for data, with full automation and deep integration across all these stages, is desirable. This enables developers to regulate in terms of enterprise PLM and ALM, as well as to the overall program management (see Figure 7).

Consistancy assurances across the whole vehicular system

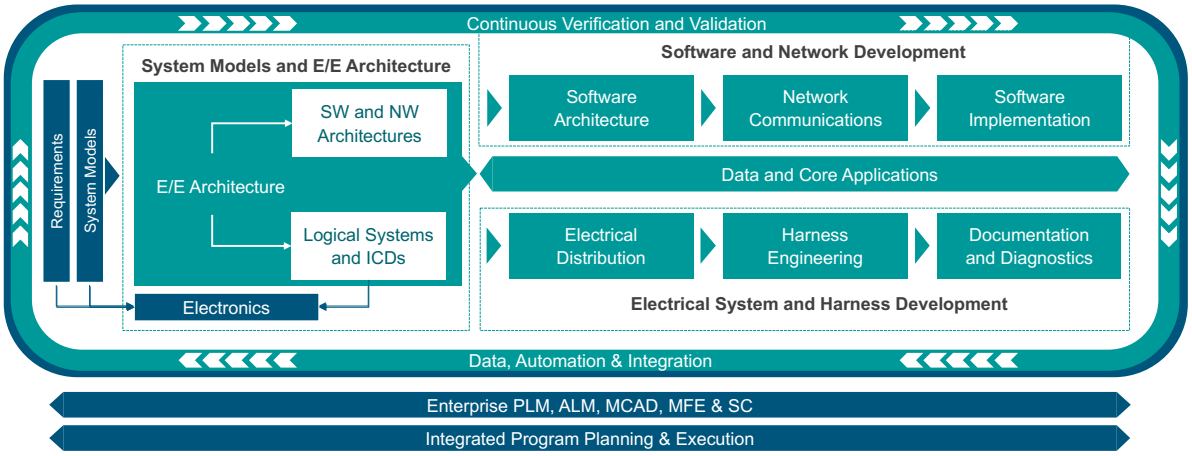


Figure 7: Systems-driven design for functional safety compliance.

Siemens Capital VSTAR for functional safety
Advanced integrated E/E system development tools are required today to deliver tomorrow's products. Deploying the model-based system development approach using a rules-based design process provides the best conditions to ensure that functional design abstractions are synthesized into current by-construction safe systems. Built around AUTOSAR, Capital VSTAR combines best-in-class AUTOSAR-compliant design tools and software, simplifying the ECU software generation for a functionally safe system.

Mixed-safety criticality

Before we dive into the concept of mixed-safety criticality, let's examine how to develop a Mixed Safety Criticality solution.

Over the years, multicore has evolved from dual-core to quad-core. Since then, CMA providers have listened to consumers and provided additional resources and computer power to ECU developers. While they're still providing additional cores, they're also adding specialties, such as real-time cores, DSPs, and soft cores. This has led to the development of heterogeneous hardware being consolidated onto a single system on a chip (SOC) (see Figure 8).

To fully leverage heterogeneous hardware, we need to look at it from a software perspective.

Heterogeneous software solutions and architectures use heterogeneous hardware. For example, there may be general-purpose operating systems, RTOSes, and bare metal applications all running on different types of cores. Some key considerations include how the system will boot, how communications will transpire across a shared workload, and so on.

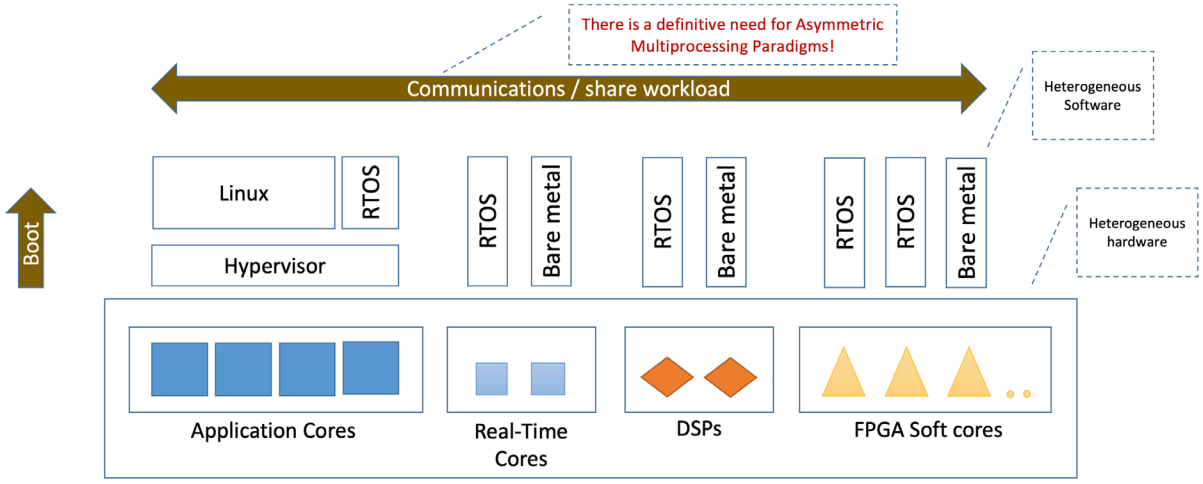


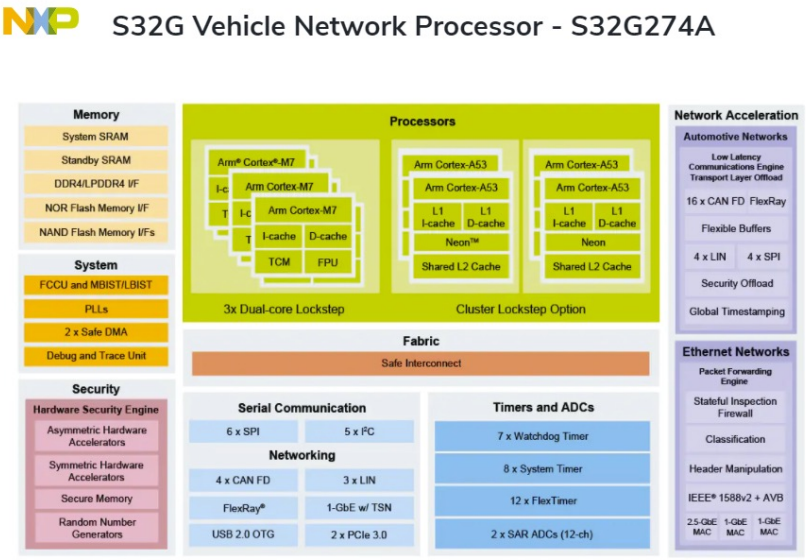
Figure 8: Heterogeneous Multiprocessing in Embedded System On Chips

"Software will account for nearly 30% of total vehicle content by 2030. "

McKinsey & Company

These chips have a high degree of complexity. Take, for example, the S32G Vehicle Network Processor . This processor consists of a set of Cortex A53 Application Processors and a number of Cortex M7 devices. In some cases, they have dual A72s, some R5s, various types of memory in peripherals, and communication blocks throughout the chip (see Figure 9).

With this complexity, the amount of software required grows. With additional compute power, you can lower your cost by consolidating everything onto a single SOC. This brings about new challenges – not the least of which is achieving mixed-safety criticality.



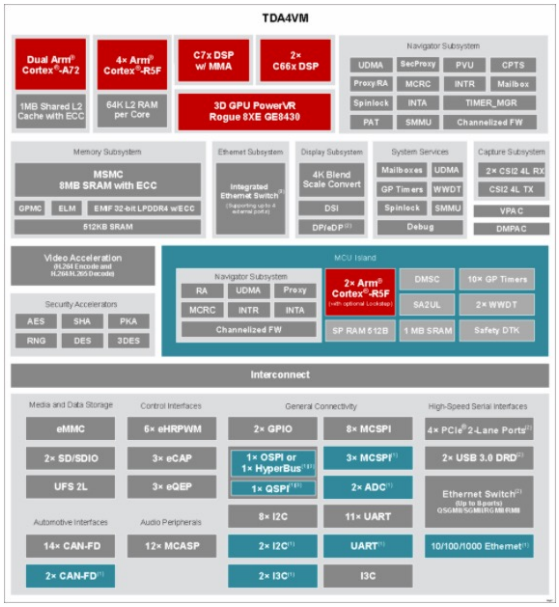


Figure 9: S32G Vehicle Network Processor.

Complexity brings new challenges

In the past, you may have had a system that was specifically focused on providing functional safety running along with other systems running on Linux that were not required to provide functional safety. Thanks to multicore processing systems, consolidation is now common, and on a single SOC, you'll have mixed-safety criticality. There will be some mission-critical subsystems with real-time requirements, some secure subsystems, and some safety-certified subsystems. To ensure the same level of software reliability and safety when integrating these systems onto a single SOC, you must isolate the safe and non-safe domains, and establish reliable safe and secure communications between those domains.

Figure 10 shows an example of mixed-safety criticality in an 2022 Cadillac Escalade. On one hand, the console contains your radio, heating controls, and other subsystems that enhance the driver experience. If something were to fail on one of the non-critical subsystems such as the radio, it would cause an annoyance but no real risk. On the other hand, the console also includes the backup camera. If the backup camera were to fail, the driver – and other drivers and pedestrians on the road – could be endangered. This is a real-world example of mixed-safety criticality, where multiple operating systems with different levels of functional safety

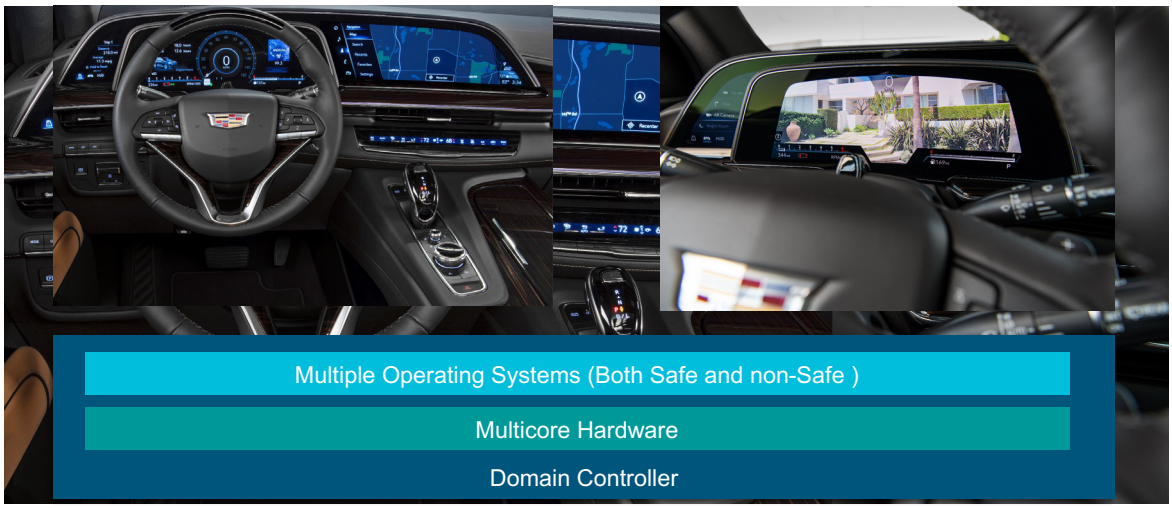


Figure 10. The interior of a 2022 Cadillac Escalade.

requirements are running on one or maybe two pieces of multicore hardware leveraging a cockpit domain controller.

Figure 11 illustrates the various components of a system with mixed-safety criticality. In addition to a critical certified functional RTOS, for which AUTOSAR is the perfect fit, you may be running less critical or non-critical software on Linux, bare metal, Nucleus or other OSes. You need to ensure communications between the various domains and processes for booting software. Other key components are isolation, hardware verification and monitoring, error handling and resource management.

Some typical functional safety concepts that must be considered when designing a mixed criticality system include:

- **System failures that affect functional safety goals:** Typical systematic faults may arise due to errors in manufacturing and development processes and can affect both the hardware and software. They can originate from a failure to verify intended functionality, from manufacturing test escapes, or by operating outside rated conditions.
- **Random faults:** Inherent to silicon aging and environmental conditions, random faults may be permanent, such as those impacting RAM memory, or temporary, such as memory corruption due to SEU.

Figure 11: Components of a Mixed Criticality System.

- **Hardware-provided safety features:** It's important to leverage any safety features provided by software designed for functional safety: The software should be designed with features such as safe boot, separation, and isolation of safe and non-safe systems, as well as safety-certified software components.

Let's look at a few of these concepts in greater detail.

Isolation

Isolation – both temporal and spatial – between safe and non-safe domains is critical. Temporal isolation can be achieved by providing dedicated independent cores, while spatial isolation is possible with hardware protection units. Fortunately, silicon vendors such as Xilinx and others are providing these capabilities on the silicon itself. By leveraging some of these hardware capabilities, we can ensure isolation between these different environments from a software standpoint.

Communications

Hardware goes a long way toward achieving isolation, but there are several areas that must be maintained by software (see Figure 12):

- **Buffer validation:** Buffer parameters such as address, size, and permissions must be validated before being used by the safe domain, including checking bounds on the buffer and discarding any buffer that falls outside the valid range. Buffer validation must be paired with the proper error response to give the user insight into system interactions for detecting malicious activity.
- **Interrupt flooding validation:** The non-safe world has the potential to flood the communication channel with interrupts. Without special handling, this unanticipated load can violate the temporal isolation requirements of the system. Implementing a mechanism to throttle excessive interrupts from the non-safe side, or to support polling mode on the safe side, is often necessary.

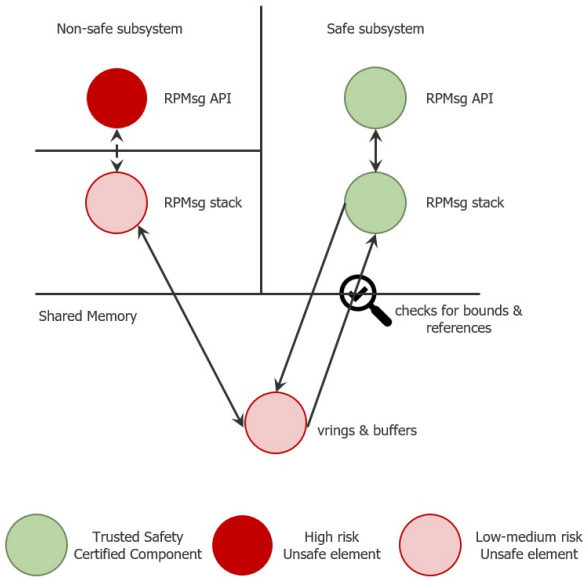


Figure 12: Maintaining safe communications on a mixed-safety criticality system.

These capabilities are critical to ensure isolation necessary for mixed-safety criticality systems.

Figure 13 is an example of a mixed-safety criticality SOC, with AUTOSAR-compliant critical systems running on real-time/safety cores, and non-critical systems on the left that run on a variety of application cores. Linux, Sokol Flex, and Android OSes can run on top of those application cores, as well as a Nucleus RTOS or even a hypervisor. This model enables developers to leverage the hardware separation, along with software-enabled isolation and communications. That's where a multicore framework comes into play, enabling this heterogeneous environment.

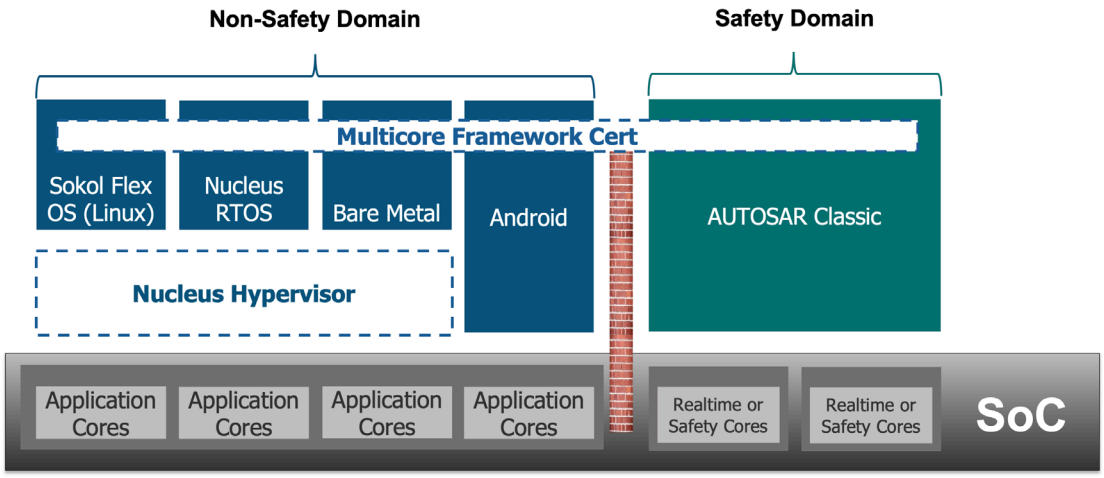


Figure 13. A mixed-safety criticality SOC enabled by a multicore framework.

Siemens multicore framework safety certifiable solution

Designed to enable multiple operating systems, Siemens' multicore Framework Safety Certifiable Solution is based off of the OpenAMP project. OpenAMP stands for Open Asymmetric Multi-Processing, which seeks to standardize the interactions between operating environments in a heterogeneous embedded system through open-source solutions for Asymmetric MultiProcessing (AMP). Siemens has extended the solution to enable mixed safety criticality and provide safe communications between domains.

Our multicore framework supports mixed-safety criticality systems with the following features:

- The ability to run multiple remote OSes
- Native execution and performance
- Distributed system management
- The ability to leverage hardware isolation
- Comprehensive documentation and artifacts
- Eclipse-based IDE with GCC/G++ tools

Siemens is currently in the process of completing ISO26262 ASIL D certification, as well.

Siemens also offers embedded Linux. We have two Linux distributions, depending on your needs:

- **Sokol Flex OS:** This distribution is highly customizable and aligned with the latest semi-provided long-term support kernels.
- **Sokol Omni OS:** This distribution is enterprise-class, providing a long-term supported kernel with pre-built Debian binary packages and real-time functionality.

Siemens' embedded Linux distributions simplify development and increase ROI, while enabling developers to secure and maintain devices. They are portable and customizable, and integrated with the Sokol IoT framework.

End-to-end E/E systems development

Siemens Capital VSTAR is a comprehensive AUTOSAR solution that provides enhanced performance, virtual validation, functional safety, cybersecurity, and fast deployment. We offer a certified and non-certified version of our multicore framework, as well as a hypervisor (see Figure 14):

- **Hypervisor:** The Nucleus Hypervisor enables developers to reduce costs with high performance and secure consolidation of disparate runtimes on multicore application processors, providing a comprehensive solution built for embedded systems.
- **Multicore Framework:** The Multicore Framework add-on to Nucleus enables developers to easily configure and deploy multiple operating systems and applications across cores on heterogeneous multicore processors.
- **Multicore Framework Certified:** The Multicore Framework Certified add-on to Nucleus SafetyCert enables developers to create systems of mixed safety-criticality on homogeneous or heterogeneous multicore processors, simplifying implementation and lowers certification costs.

Siemens' end-to-end offering for AUTOSAR includes general-purposes operating systems, hypervisors, and cloud connectivity, for everything from traditional CPUs to complex heterogeneous SOCs (see Figure 15).

Siemens also provides an array of services, such as Security and Vulnerability Monitoring to free up teams to work on tasks that add value.

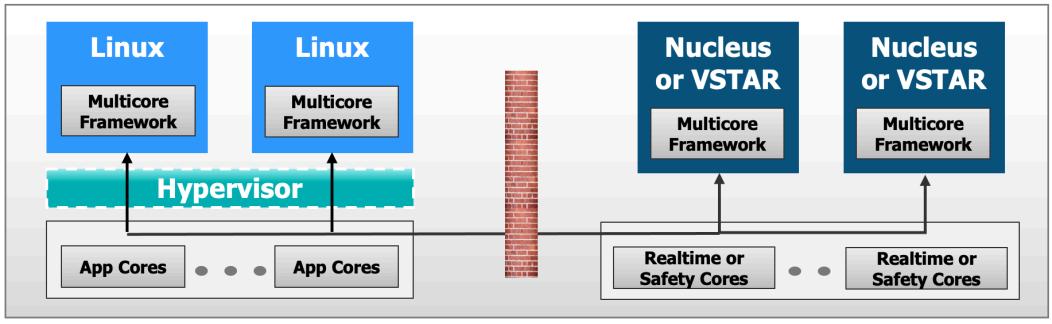


Figure 14: Siemens provides all the necessary components for developing systems with mixed- safety criticality.

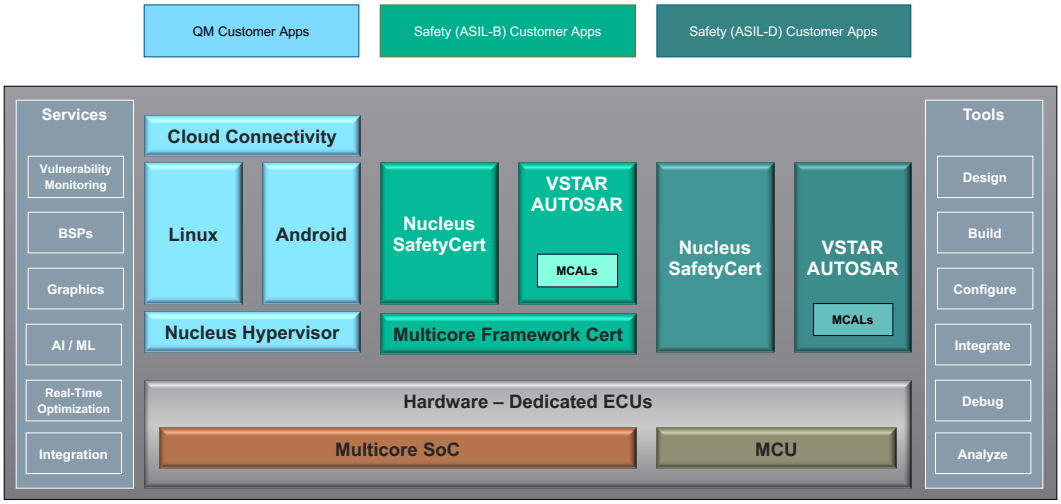


Figure 15: Siemens Automotive Embedded Solutions.

Siemens Services: Embedded and AUTOSAR

- Global development and project management
- Production-grade QA infrastructure
- Platform expertise examples for:
  - Mutli-core and multi-architecture support
  - Real-time implementation and optimization
  - Memory footprint optimization
  - Boot time optimization
  - In-depth knowledge of AUTOSAR, Linux and OSS
  - Compiler optimization and extension
  - BSPs/Custom driver implementation and optimization
- Other Areas of expertise
  - Safety and Security
  - Languages: C, C++, Python
  - Graphics/UI: Linux, Eclipse, GCC, LLVM
  - Machine Learning

Conclusion

With the increased compute power made possible with heterogeneous multicore SoC design come new challenges and opportunities, such as how to meet the demand for mixed-safety criticality components and functionality. To comply with the newest changes to ISO 26262 and the safety landscape, developers must accommodate additional safety content, support new technologies, and focus on reducing faults. Spatial isolation at the memory and core levels is critical to ensure freedom from interference, temporal isolation, schedulability, and end-to-end communication protection. This requires extensive tooling support, model-based transformations, and a system-driven design approach.

Siemens' Capital AUTOSAR platform empowers E/E development teams with the tools, capabilities and services necessary to meet evolving functional safety requirements for modern electric vehicles today and for years to come.

Learn more about Siemens solutions for the Automotive industry here.

Siemens Digital Industries Software

Americas: 1 800 498 5351

EMEA: 00 800 70002222

Asia-Pacific: 001 800 03061910

For additional numbers, click here.

About Siemens Digital Industries Software

Siemens Digital Industries Software is driving transformation to enable a digital enterprise where engineering, manufacturing and electronics design meet tomorrow. Xcelerator, the comprehensive and integrated portfolio of software and services from Siemens Digital Industries Software, helps companies of all sizes create and leverage a comprehensive digital twin that provides organizations with new insights, opportunities and levels of automation to drive innovation. For more information on Siemens Digital Industries Software products and services, visit siemens.com/software or follow us on LinkedIn, Twitter, Facebook and Instagram. Siemens Digital Industries Software – Where today meets tomorrow.

About the authors

Dr. Ahmed Majeed Khan, Senior Engineering Manager

Dr. Ahmed Majeed Khan, is an engineering enthusiast, experienced in working with cross-functional groups to push the envelope of technology implemented in diverse automotive and consumer electronic domains. Having a proficiency to manage onshore and off-shore development of innovative and disruptive products, he led teams around the globe to produce several high volume, high quality system-level solutions. Currently, Dr. Khan is a Senior Engineering Manager at Siemens, where he assists in creation of a market-leading automotive-grade product portfolio. He is also Siemens focal point towards international automotive software consortium of AUTOSAR. He holds a doctorate in Engineering Management from George Washington University, an MS in Electrical Engineering from Michigan State University and has over a decade of experience working with embedded systems.

Jeff Hancock, Senior Product Marketing Manager

Jeff Hancock is a Senior Product Manager in Siemens Embedded, a segment of Siemens Digital Industries Software. Jeff oversees the Nucleus® RTOS and the embedded Hypervisor runtime product lines, as well as associated middleware, and professional services. Jeff earned his Bachelor of Science degree in Electrical Engineering Technology from Purdue University.

84407-D4 2/22 A