

Practical Use of AUTOSAR in Safety Critical Automotive Systems

Simon Brewerton

Infineon Technologies UK Ltd

Rolf Schneider and Frank Grosshauser

Audi AG

Copyright © 2009 SAE International

ABSTRACT

With the increased adoption of AUTOSAR operating systems across the different automotive system domains a notable exception has been that of the safety critical systems. This domain has strict requirements on precise requirements capturing, proven design flow, robust implementation, exhaustive testing, detailed documentation and traceability, and project management processes. These requirements are normally prohibitive to adopt for commercial 'one size fits all' solutions due to the huge expense and resources required to meet such a strict regime. So under these constraints AUTOSAR is far from a perfect fit for safety systems. Nonetheless, the attractive features of reuse and portability still make AUTOSAR based systems highly desirable. This paper describes a solution to this dilemma by embedding a supervision subsystem inside a dual core AUTOSAR compliant microcontroller to give extremely deep levels of diagnostic coverage at runtime, not only for the hardware but also for the software and, by implication, the tooling and design specifications as well.

INTRODUCTION

AUTOSAR is more than just an operating system. It is a living, evolving, multi-party standard, which appears under many guises for different functional areas and applications. Although it was initially conceived as a solution to make portable, hardware independent software for 'soft real-time' applications such as central

body control ECUs and comfort functions, the allure of transportable and reusable software has made many OEMs and Tier 1 suppliers consider AUTOSAR for more stringent applications, where the major benefit of common communication layers make composing and testing an AUTOSAR based network of ECUs much quicker and more predictable.

AUTOSAR version 2 was not conceived to be necessarily suitable for safety related systems, so the software development process to which it was built does not directly meet the requirements of the IEC61508 part 3: Software Requirements. However, building a control system to meet IEC61508 Safety Integrity Level 3 is much more than just software. It involves a full safety life-cycle, from defining the hazards, the failure modes, detection, reactions, and implementation (hardware and software design and realization), through to testing, maintenance and documentation processes. The IEC61508 also has several drawbacks when applied to automotive systems, as it is a generic standard for the functional safety of any electrical / electronic / programmable electronic safety-related system. So this includes the electrical control of any machine which can potentially cause harm to the users, by-standers, or environment. This ranges from manufacturing machinery controlled by relays, PLCs, and mechanical sequencers, to the most complicated computer controlled automation system for robotics. So it requires a certain level of 'interpretation' to apply this standard to a motor vehicle. For this very reason a new standard is currently in preparation. This is the ISO26262, which is currently a

committee draft version. It will define an automotive safety lifecycle, based on the IEC61508, but tailored to the automotive industry for passenger cars, buses, commercial vehicles and towed vehicles. This ISO standard will define a new 'state-of-the-art' to clarify and remove any ambiguity for implementers and assessors of safety relevant automotive systems controlled by electronics or software.

For many OEMs the desire to make AUTOSAR 'fit for purpose' in a safety relevant system is clear, however the solution to a practical implementation has a very narrow solution space indeed. The work reported in this paper is based on several projects at Audi AG for chassis control systems employing an asymmetric dual core hardware solution. This consists of two different types of microprocessor cores on a single monolithic silicon device, together with a separate silicon with a small 'complex signature window watchdog' device. Such devices have been common place for some years in electronic throttle control systems (or EGAS systems).

SIL DECOMPOSITION

Assuming that the system hazard and risk analysis, safety requirements and scope have already been established, the first task that faces the electronics designer is to assign Safety Integrity Levels (SIL) to each of the safety relevant functions of the control system. When AUTOSAR (or any other integrated operating system such as OSEK) is used as the task scheduler, resource manager, communications manager and interrupt arbiter, it is highly likely that the operating system will be required to achieve the same SIL as the highest SIL control function, as the critical software block relies on OS services and communications to function correctly. So if an ECU has some hazards and risks that are deemed to be SIL3 then the OS would by implication also have to be SIL3 capable.

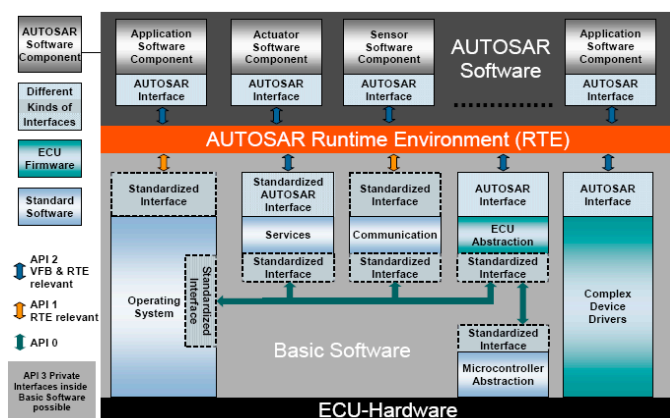


Figure 1: AUTOSAR (version 2) software structure [1]

The use of the AUTOSAR OS software structure (Fig 1) in a SIL3 application typically requires all the layers below the RTE to also be SIL3 capable.

However, the IEC61508 also allows the safety function to be achieved not only by the main control path of the system, but also by some other function, such as a monitor or external supervisor. So if a suitable SIL3 capable monitor or supervisor is put in place then the control (and hence operating system) no longer needs to necessarily be SIL3 capable (providing the monitoring is of suitable quality and has sufficiently low common cause failures). The problem is then shifted to that of showing a suitable quality of the monitoring system, and allowing the monitor to disable the system under control if the monitoring system detects a hazard.

This method is the basis of the current 'state of the art' for the EGAS monitoring using an external complex window watchdog ASIC as the monitor of the system. The challenge is therefore to provide a monitor which is capable of securing the microcontroller hardware platform (functional independent monitoring) as well as the AUTOSAR OS, communications and services (functional dependent monitoring).

FUNCTIONAL INDEPENDENT MONITORING

Monitoring of the microcontroller hardware against faults which occur during runtime can be achieved in many different ways. The blocks to be checked and the maximum coverage levels that can be claimed by different methods are defined in the IEC61508 Part 2: Tables A4 to A12. The system defined by Audi includes software mechanisms to test the hardware mechanisms which are provided by the microcontroller sub-systems. The safety integrity relies on the correct operation of the hardware mechanisms to perform the safety function. These hardware mechanisms report errors via software hooks when a fault is detected. Other forms of checking include use of known test vectors that have been simulated (on gate level simulations) to show that DC faults and cross coupled faults can be detected. The monitoring functions are implemented as a set of low level drivers, built by an independent software contract company, to an IEC61508 capable software development process. With these software drivers activating the hardware check mechanisms, the functional independent monitoring of the basic hardware can achieve SIL3 requirements.

These blocks include:

- Processing Units (ALU, Register File, Pipeline)
- Invariable Memory (Flash, EEPROM)
- Variable Memory (SRAM)
- Instruction caches

FUNCTIONAL DEPENDENT MONITORING

A significant part of the safety function is checking the correct operation of the application specific functions. These include the requirements to check:

- I/O Units and Interfaces (SPI, FlexRay, CAN)
- Data Paths (Internal Buses and Bridges)
- Program Sequence (Flow and Time Budget Monitoring)
- Interrupt Monitoring (Vectoring, Rate and Missing Events)

Some of these requirements can be met by a combination of software and hardware resources, for example checking memories protected by Error Correcting Codes (ECC). Other requirements, such as program sequence monitoring, are enforced by the operating system but need to be monitored by a mechanism which is capable to having an independent time base and signature (flow) analyzer.

All these monitoring requirements demand that the microcontroller provide several mechanisms to the user to allow a logical split of application control path and monitoring path.

In the Audi system these are realized by two distinct CPU cores and sub-system comprising:

- Main CPU with local private code and data memories for execution of the control path
- Memory protection system (software encapsulation)
- Dual vectored interrupt units (one for each core)
- External complex signature window watchdog
- Second CPU with local private code and data memories for execution of the monitoring functions
- DMA engine with memory protection
- General purpose timer array (PWM & Input capture)
- Internal bus controllers
- SPI interface with automatic slave select line
- Memory error detection (ECC) blocks

Only a few automotive microcontroller families can provide all the required hardware to achieve this embedded monitoring function. One of the longest established devices is the TriCore family of microcontrollers from Infineon, which was also the validator platform used by the AUTOSAR consortium.

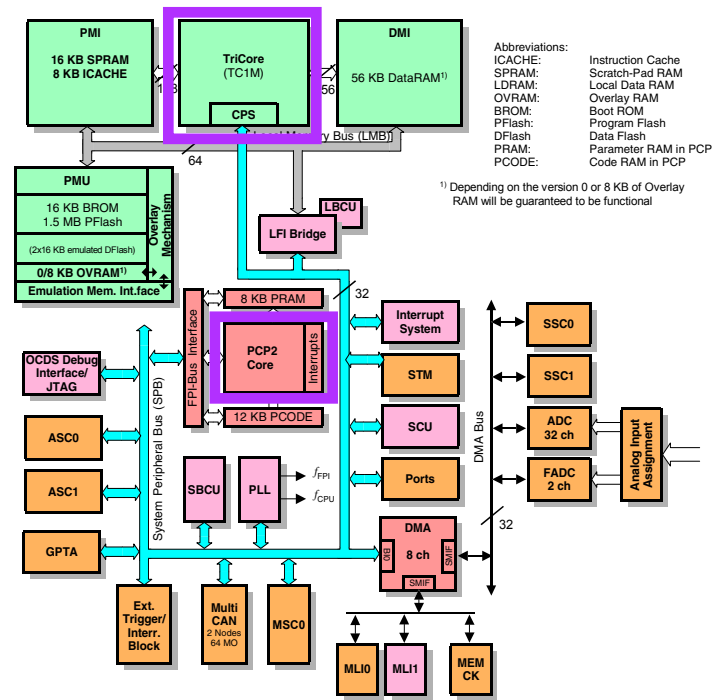


Figure 2: Block diagram of the Infineon TC1766 microcontroller which has a 80MHz main CPU (TriCore) for the application and a second 80MHz CPU (PCP) which is used for monitoring functions.

EXTENSION OF THE AUTOSAR SOFTWARE STRUCTURE

The standard AUTOSAR software (version 2) was taken over completely without changes, and two additional software blocks were added. One block is the application functional safety monitor and the other is the basis software provided from Infineon know as the PRO-SIL SafeTcore software. This PRO-SIL software provides the application independent functional safety and communicates to an external EGAS monitoring ASIC (a Bosch CY320) via SPI interface. The application and AUTOSAR runs on the TriCore CPU, together with part of the PRO-SIL driver software, and the rest of the PRO-SIL suite runs on the PCP processor.

The different software blocks are built into separate memory regions (via linker controls in the compiler). The memory protection hardware can then be used to enforce the protection boundaries, so that no software block can unintentionally (or indeed intentionally) corrupt another. The memory protection system on the TriCore processor provides a 'break-before-make' detection in case of violations. Any violation will trigger a trap, which is in turn routed to an error handler. The memory protection is quite distinct from a Memory Management Unit (MMU) as found in several processors from other vendors. An MMU can perform address translation, whereby one virtual address can be mapped (by software control) to any number of physical addresses in