EMBEDDED SOLUTIONS
FOR SAFETY-CRITICAL APPLICATIONS

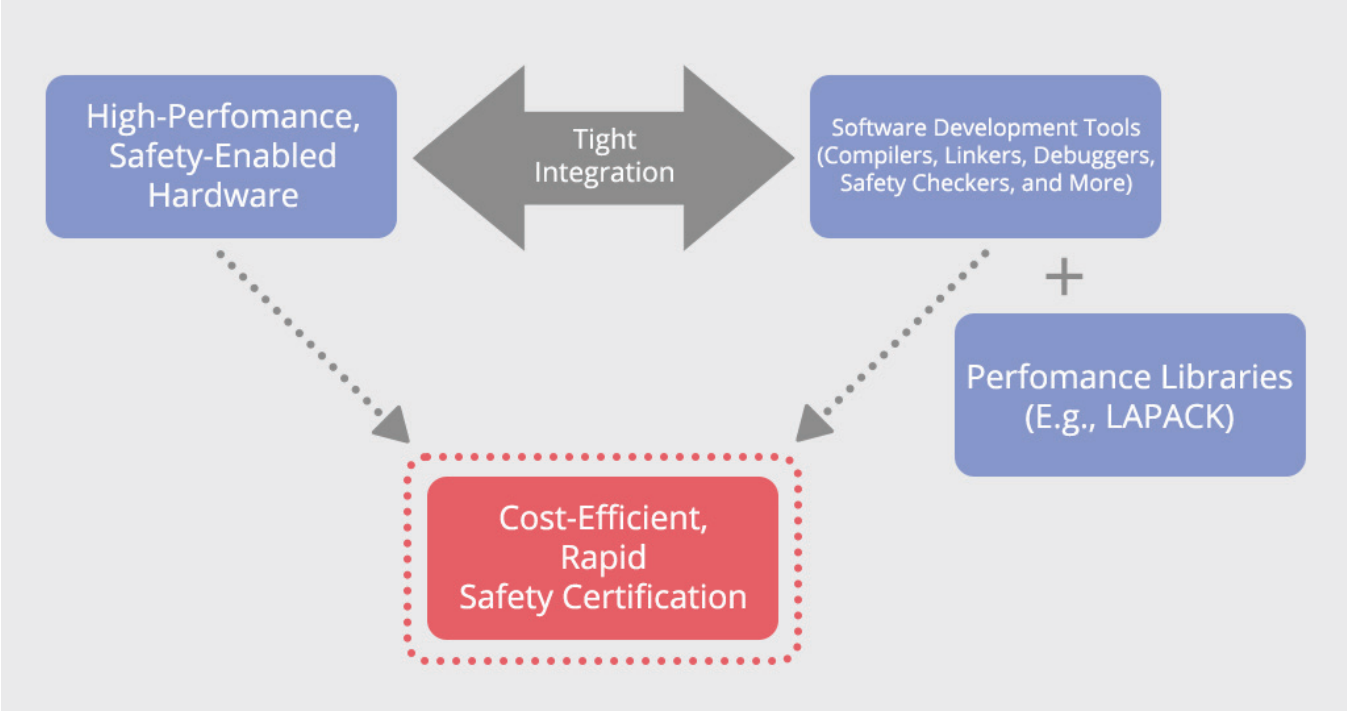
INTRODUCTION

Cars used to be mechanical creatures. But today, more and more of a car's functions are controlled by embedded solutions that consist of tightly coupled hardware and software. For example, powertrain control modules (PCMs) control fuel injection, the transmission, braking, and more. Automotive manufacturers are investing in a growing portfolio of advanced driver assistance systems (ADAS) features, such as adaptive cruise control and blind-spot detection.

The ISO 26262 standard was created to help guarantee proper management of safety throughout automotive applications. The standard defines several Automotive Safety Integrity Level (ASIL) classifications. An embedded solution's combination of hardware and software must be built with safety in mind, such as using fallback paths, redundancy, and self-monitoring, and the components must be tested according to the ASIL classification for the application that uses them. The ASIL classification for a particular application is determined by the severity of the harm a failure of the application (or parts thereof) could create in different scenarios. Components that have no impact on any safety-critical function (such as the graphical user interface for a car's radio channels) have a lower ASIL classification than ADAS functions that can have a significant impact on safety. Because ADAS functions make high-level decisions for the driver, they usually must be certified to a high ASIL classification, such as ASIL-D.

The global automotive ECU market, which includes PCMs, is expected to reach nearly $46 billion by 2020 (1), while the global ADAS market is forecasted to experience double-digit growth between 2014 and 2024, from a baseline estimate of $18.2 billion (2). That's plenty of incentive for automotive OEMs and other innovators to devote more time and effort to developing these systems.

However, safety certification of ADAS can be a long and arduous process. Using pre-certified and deeply tested hardware components (such as microprocessors) and software components (such as compilers and performance libraries like LAPACK) can make system-level certification much easier and contribute to a greater level of safety.



SAFETY-CRITICAL APPLICATIONS RELY ON ADVANCED EMBEDDED HARDWARE

When designing safety-critical applications such as PCM and ADAS functions, automotive developers must be certain that calculations are executed without error and completed within a predefined maximum period of time. It is also important that one section of code does not interfere with other sections' execution, such as by changing data or consuming needed resources. In addition, certain instructions must be completed before or after other instructions – getting the timing right is crucial. Data from multiple sensors, such as radar and cameras, must be collected and fused in real-time to create a coherent and reliable map of a car's surroundings.

These stringent requirements mean that the first step in any successful PCM or ADAS development is to choose the right hardware. Doing so can help speed safety certification and bring a product to market on time and on budget. For example, the innovative Infineon AURIX TC3xx Family TriCore™ processor is expressly suited to the needs of the automotive industry in terms of performance and safety. Its innovative multi-core architecture, based on up to six independent 32-bit TriCore CPUs, has been designed to meet the highest safety standards while significantly increasing performance. Developers using the AURIX TC3xx Family TriCore processor will spend less effort achieving the ASIL-D standard than with legacy processor architecture. Customers wanting to reduce their time-to-market can now cut down their microcontroller unit (MCU) safety development cost by as much as 30 percent. (3)

The TriCore processor is specifically optimized for a safety-critical environment, offering several safety-related features, as discussed below.

Detect Errors Faster with Multi-core Lockstep Architecture

With the AURIX TC3xx Family TriCore processor, lockstep execution (in which two cores run the same instructions in parallel to detect errors) is available on up to four of the six cores (Figure 1). This enables a new level of ISO 26262 functional safe computational power on a single integrated device—reducing development effort and cost while still achieving ASIL-D compliance.

AURIX TC2xx Family lockstep architecture is designed to control and mitigate failures caused by physical isolation, instruction-level execution diversity, and circuit-level design and timing diversity. Special features include a two-cycle delay, unique design of clock and reset networks, and an innovative design of the lockstep comparator. The AURIX TC2xx Family lockstep cores have been additionally transformed to provide architectural hardware diversity and further reduce common-cause failures.

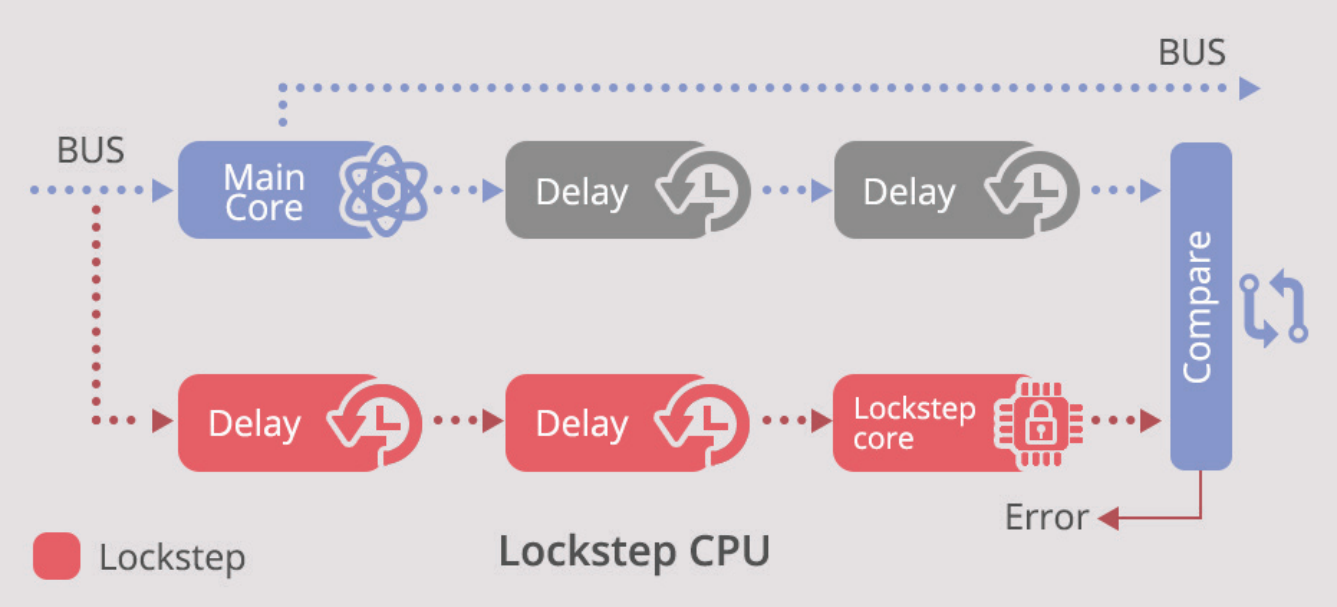


Figure 1. Architectural hardware diversity between the main core and the lockstep cores helps reduce common-cause failures. Courtesy of Infineon.

Declare Your Freedom from Interference

While multi-core architecture provides many performance and safety advantages, it is important that the architecture also provides freedom from interference – that is, the software assigned to one ASIL classification should not influence or interfere with the software with a higher ASIL classification. This freedom from interference must be achieved across the spatial domain (memory accesses), communication domain, and the time domain. (4) That is, software should not corrupt data owned by other software components, nor should software applications that are sharing peripherals corrupt those peripherals' configurations or outputs. As cores can share many resources, like buses and memory, the timing behavior of all applications must take into account the effects of these shared resources.

The TriCore processor provides freedom from interference through several methods:

- In the memory and communication domains, each core's CPU memory protection unit (CPU-MPU) monitors outgoing traffic to the bus.
- The bus MPUs enable partitioning of memory through static assignment of memory ranges to specific cores.
- Register Access Protection (RAP) allows static assignment of peripherals to specific cores at the bus level, thereby preventing other cores from gaining access to those peripherals.
- All safety-related registers of global peripherals on the processor are SendInit protected. In order to write to them, the writing core must also have access to the global safety watchdog.
- Error handling includes the following features:
    - User-defined, core-specific trap-handlers for illegal operation strings and addressing errors
    - Global "alarms," such as memory bit-flips or die overheating, with a direct connection to the processor's Safety Management Unit (SMU)
    - Support for user-defined software alarms
    - An emergency-stop feature that is capable of switching pins into a predefined state automatically without software input

Get the Timing Right with an Integrated Generic Timer Module (GTM)

The AURIX TC3xx Family TriCore processor uses a GTM with a unique architecture that supports both pre-implemented hardware functions as well as a multi-channel sequencer (MCS), which is a simple programmable processing core. In this way, the GTM can execute complex functions in software while guaranteeing low latency and full timing predictability.

The GTM's modular design makes it easy to add and remove features in a system without affecting the existing ones—simplifying development and moving toward platform independence. The GTM can be programmed using C as well as assembly, provided a tool set is chosen that offers compilers that support the GTM.

Compute Results Quickly with Hardware Fast Fourier Transform (FFT)

The radar systems used by many ADAS applications use the FFT capabilities of the microprocessor, which transform the radar signal frequencies into spatial information. Likewise, PCMs may use FFT to monitor engine health and perform vibration analysis. By nature, FFT functions are computationally intensive; the TriCore processor provides highly efficient FFT processing that can enhance the performance of radar systems. Embedded ADAS applications using the TriCore processor's FFT capabilities comply with the ISO 26262 standard.

INCREASING SAFETY WITH MULTI-CORE PROGRAMMING

There are many software development tools available to automotive developers; choosing between them can be difficult, but is paramount to the success of embedded solution development.

Choose a Compiler that Is Tightly Integrated with the Hardware

Automotive developers using the AURIX TC3xx Family TriCore processor can choose between several C compilers (as well as several assemblers). Developing a set of criteria for choosing a compiler can help identify the best choice.

One major differentiator is whether a compiler is open source or proprietary. Open source compilers can be inexpensive, but have several drawbacks that can impede safety-critical embedded solution development. The three major issues with open source compilers are:

1. **Compiler not controlled by the provider:** The foremost problem with open source compilers is that the compiler is not fully under control of the compiler provider. If a bug is found, the compiler provider's customer wants that bug fixed within a predefined timeframe. But with open source, the compiler provider may not have anyone in-house who knows how to fix it, because only certain parts of the open source compiler are developed in-house while the rest reuses open source technology. Therefore, fixing bugs may take longer than desired.
2. **Open-source compilers may not be aware of hardware:** Generic compilers built using open source are not usually fully aware of specific hardware capabilities such as those offered by the AURIX TC3xx Family TriCore processor, or are not capable of fully exploiting those features. For example, many open source compilers do not support the TriCore processor's GTM, while others lack the ability to optimize compiled code for multi-core architecture. Compilers support these types of features using language extensions, and it is easier to add language extensions if the compiler is not open source.
3. **Custom compilers generally yield better code.** Finally, compilers that are built entirely in-house often yield better performance of the compiled code. For example, to gain the smallest memory footprint and the fastest code possible, the linker you chose must enable you to control which code accesses which memory blocks. The TriCore processor's memory architecture offers several types of memory, including ECC protection, for the most flexibility in application design (Figure 2). Scratchpad memory (also known as "near memory") can be accessed on each core without using the bus, resulting in extremely fast memory access. On-die flash memory and RAM add extra memory capacity and capabilities, although access is not as fast as near memory access. Unless the compiler and linker support the ability to easily map the right objects into each of these types of memory, the TriCore's flexible memory architecture is useless, hindering application performance.

The same is true for programming for multi-core architecture. Writing code for a multi-core processor is far more complicated than programming for a single core, and requires the linker to allow the developer to assign specific instructions to execute on specific cores, using the cores' unique IDs.

Another advantage of choosing a compiler that is tightly integrated with the hardware is the availability of expert support—from both the compiler and hardware companies. Instead of having to "go it alone," the developer can ask questions. In addition, it is even sometimes possible to get rapid product enhancements – something that rarely happens in the open source community, which tends to adopt new technology at a slower pace.

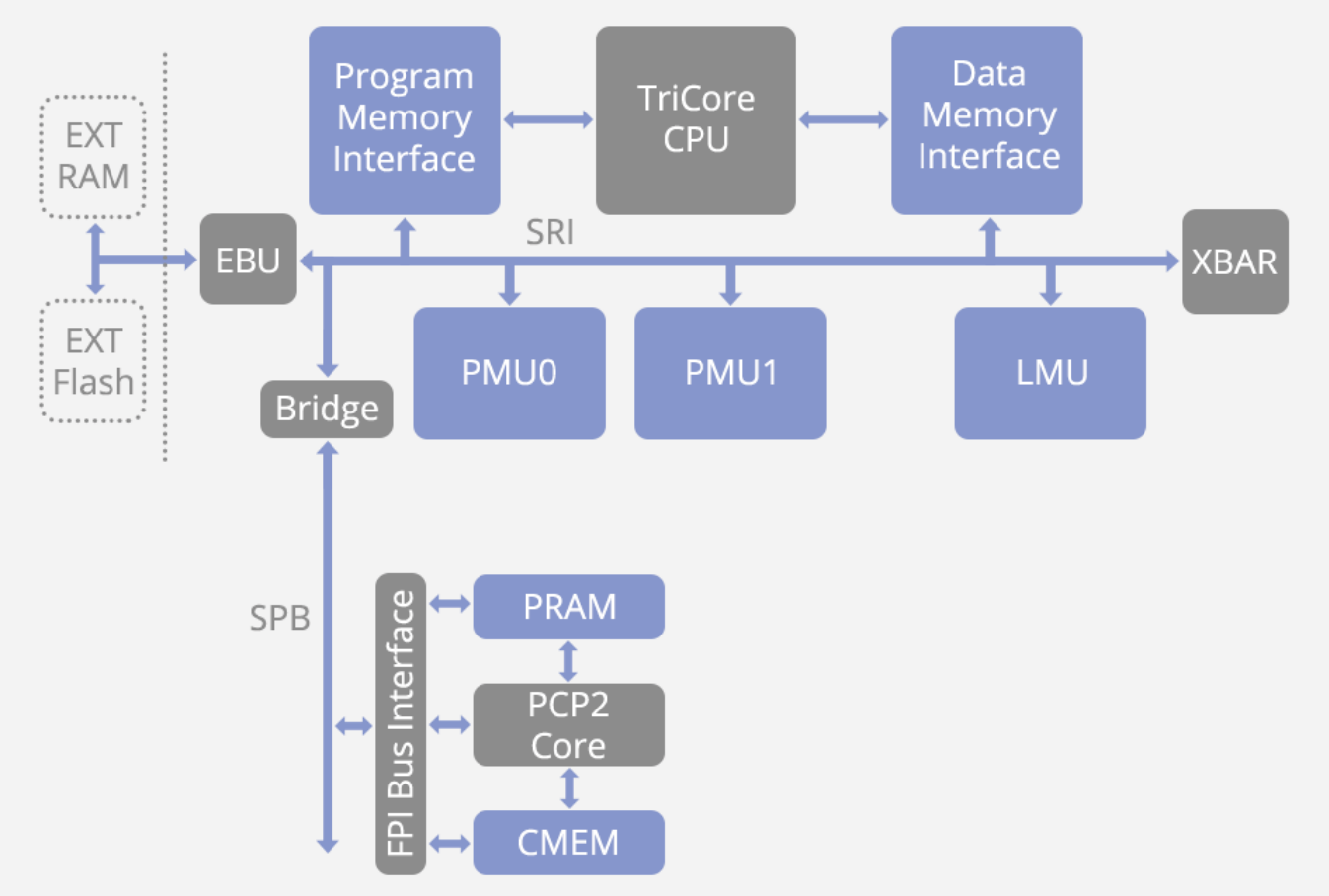


Figure 2. Using a compiler/linker combination that can take advantage of the processor's memory architecture helps software architects optimize the performance of their applications.

Code rarely runs perfectly the first time. And even the best automotive developers don't think of every possible scenario. That is why it is important to invest in powerful debugging and safety-check software development tools. For example, products are available that perform safety checking on code. These safety checkers help ensure freedom from interference by finding MPU traps that are very difficult to cover with testing.

WHY ASPICE?

Automotive SPICE® (ASPICE) is a standard framework for designing and assessing automotive software development processes. Effective implementations lead to better processes and better product quality. Choosing a compiler that is ASPICE-certified at Level 2 means that you can rest assured the product has been developed according to processes that are proven and enable you to meet the required safety standards, which means your application is more likely to be error-free.

Not only does using an ASPICE-certified compiler result in better code, it can also save money. Error-correction costs climb steeply the further along into a project they are found (Figure 3). (5)

Concept	$ 1,300
A sample	$ 4,550
B sample	$ 5,200
C sample	$ 7,800
PV series	$ 84,500
Production	$ 104,000
Post Production	$ 117,000

Figure 3. Using reliable, ASPICE-certified software development tools to correct errors early in a project can significantly reduce error correction costs (data gathered from Audi, BMW, Daimler, Porsche, and Volkswagen).

PUT YOUR SAFETY-CRITICAL APPLICATION ON A SOLID FOUNDATION

Safety is an increasingly important topic in the automotive industry, as ADAS and PCM functions become more numerous and rely on increasingly complex embedded processing and embedded software. The industry is responding by defining stringent safety standards and specifications for both hardware and software. Budget is also a concern for most OEMs and safety-critical application developers. They must create safe solutions cost-efficiently and quickly to maintain a competitive edge. For optimal performance in a safety-critical environment, developers need access to safety-certified hardware combined with ASPICE-certified compilers that are optimized for that hardware.

REFERENCES

(1) "Automotive ECU Market Analysis By Application And Segment Forecasts To 2020," https://www.marketresearchandstatistics.com/ad/automotive-ecu-market-analysis-by-application-powertrain-chassis-electronics-safety-security-entertainment-communication-navigation-and-segment-forecasts-to-2020/

(2) "ADAS: On the road to widespread adoption," http://embedded-computing.com/articles/adas-the-road-widespread-adoption/

(3) "TriCore™ Architecture & Core," http://www.infineon.com/cms/en/product/microcontroller/32-bit-tricore-tm-microcontroller/tricore-tm-architecture-and-core/channel.html?channel=ff80808112ab681d0112ab6b73d40837

(4) Infineon, AURIX TC2xx Family Safety Manual AP32224 V1.2, Infineon Munich, 2015

(5) "ASPICE Made Easy-Case Studies and Lessons Learned," https://www.ibm.com/developerworks/community/files/basic/anonymous/api/library/9e66f5de-701e-4994-9291-75646d558240/document/d88a5328-3d2a-4e82-9a3f-e4e5bf9f41be/media/Session22-A-SPICE%20compliance%20made%20easy%20Case%20studies%20and%20lessons%20learned.pdf

LEARN MORE...

https://adas.org.au/need-adas-qualification/

https://www.lifewire.com/advanced-driver-assistance-systems-534859