

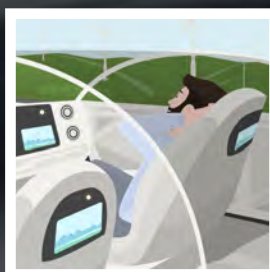
# Electronic Design® LIBRARY

A compendium of articles  
from *Electronic Design*

Sponsored by



## FOCUS ON: **TEST & MEASUREMENT** FOR **AUTOMOTIVE**



# FOCUS ON: TEST & MEASUREMENT FOR AUTOMOTIVE



## INTRODUCTION

Automotive design is quickly changing with the advent of greater connectivity, driver assistance, and infotainment systems. Meanwhile, the adoption of electric vehicles and hybrids continues apace. All of which adds up to challenges for test engineers who must deal with increasing complexity. Test and measurement systems are undergoing significant change, too, in order to stay ahead of the curve and provide insights, validate designs, ensure compliance and quality, and minimize design costs. This e-book explores the evolution of key automotive standards and highlights the latest test and measurement approaches gaining traction today.

*Karen Auguston Field, Content Director*

## TABLE OF CONTENTS

<b>CHAPTER 1:</b> COST-EFFECTIVE UNIT TESTING AND INTEGRATION IN ACCORDANCE WITH ISO 26262 .....	<b>2</b>
<b>CHAPTER 2:</b> ACHIEVING AUTOMOTIVE FUNCTIONAL SAFETY THROUGH RUNTIME MONITORING .....	<b>6</b>
<b>CHAPTER 3:</b> COMBINE AUTOSAR STANDARDS FOR HIGH-PERFORMANCE IN-CAR COMPUTERS ...	<b>9</b>
<b>CHAPTER 4:</b> INDEPENDENT THINKING: WHY THE “BLACK BOX” IS NEEDED FOR AUTONOMOUS VEHICLE DEPLOYMENTS.....	<b>14</b>
<b>CHAPTER 5:</b> TESTING TIMES FOR AUTOMOTIVE: 8 STANDARDS TO TRACK.....	<b>17</b>
<b>CHAPTER 6:</b> TAKING AUTOMOTIVE INTERNET FOR A DRIVE.....	<b>20</b>
MORE RESOURCES FROM <i>ELECTRONIC DESIGN</i> .....	<b>23</b>
VEHICLE ELECTRIFICATION: DISRUPTING THE AUTOMOTIVE INDUSTRY AND BEYOND <i>sponsored by NI</i> .....	<b>24</b>



Want more coverage like this?



## CHAPTER 1:

# COST-EFFECTIVE UNIT TESTING AND INTEGRATION IN ACCORDANCE WITH ISO 26262

Staying on track with ISO 26262 to meet automotive functional safety standards requires automation throughout the software lifecycle. The result is cost-effective production of high-assurance software.

The modern automobile is a maze of interactive electromechanical systems. Many of them, such as brakes, steering, airbags, powertrain, and adaptive driver assistance systems, are critical to human life and safety. Others—such as entertainment systems—not so much. However, they all rely on an exploding volume of software, and in many designs these component systems also share the same internal communication infrastructure. That means that functional safety from a systems perspective as well as at the independent unit level must be assured during the development and testing of this code.

## The Purpose and Scope of ISO 26262

ISO 26262 is a functional safety standard for road vehicles that defines requirements and processes to assure safety along a range of hazard classification levels called Automotive Safety Integrity Levels (ASIL). These specify functional safety measures for levels A (least hazardous) to D (most hazardous). ISO 26262 specifies a process that begins with general requirements, the specification of actual safety requirements, the design of the software architecture, and the actual coding and implementation of the functional units. There are also steps for testing and verification of each of these.

The necessary and detailed work of specifying system design requirements and safety requirements can be done at a fairly abstract level using spreadsheets, word processing tools, and more formal requirements-management tools. However, these requirements must also flow down to both the individual software components that implement them and the verification activities that prove them. Under ISO 26262, bi-directional traceability is critical to ensure a transparent and open lifecycle of development. Similarly, if code needs to be rewritten it is important to understand from which upstream requirement it was derived.

## Software Architecture Design and Testability

Design-for-testability is often an overloaded term, but the concept is clear under ISO 26262. The software architectural design, called out in Section 7 of the standard, specifically sets out to produce a software architecture that meets the software safety requirements. Modeling tools are often used during this early phase to explore the solution space for the software architecture. Some companies still rely on manual methods of high-level design, using documents or even high-level coding (that is, minus the detailed behavior). Regardless of the method, during design as well as during implementation, the architectural design must be verified.

For lower levels of safety integrity, i.e., levels A and B, techniques such as informal walkthroughs and inspections may suffice. For higher safety-integrity levels such as C and D, automation techniques allow developers to cost-effectively perform architectural analysis and review, including in-depth control flow and data-flow analysis. Ultimately, under ISO 26262 the architectural design should lead to a well-defined software architecture that is testable and easily traceable back to its functional safety requirements.

ISO 26262 also requires a hierarchical structure of software components for all safety-integrity levels. From a quality perspective, the standard includes example guidelines such as:

Software components should be restricted in size and loosely coupled with other components.

- All variables need to be initialized.
- There should be no global variables or their usage must be justified.
- There should be no implicit type conversions, unconditional jumps, or hidden data or control flows.
- Dynamic objects or variables need to be checked if used at all.

Without automation, the process of checking all these rules and recommendations against the unit under implementation would be painstaking, costly, and error-prone.

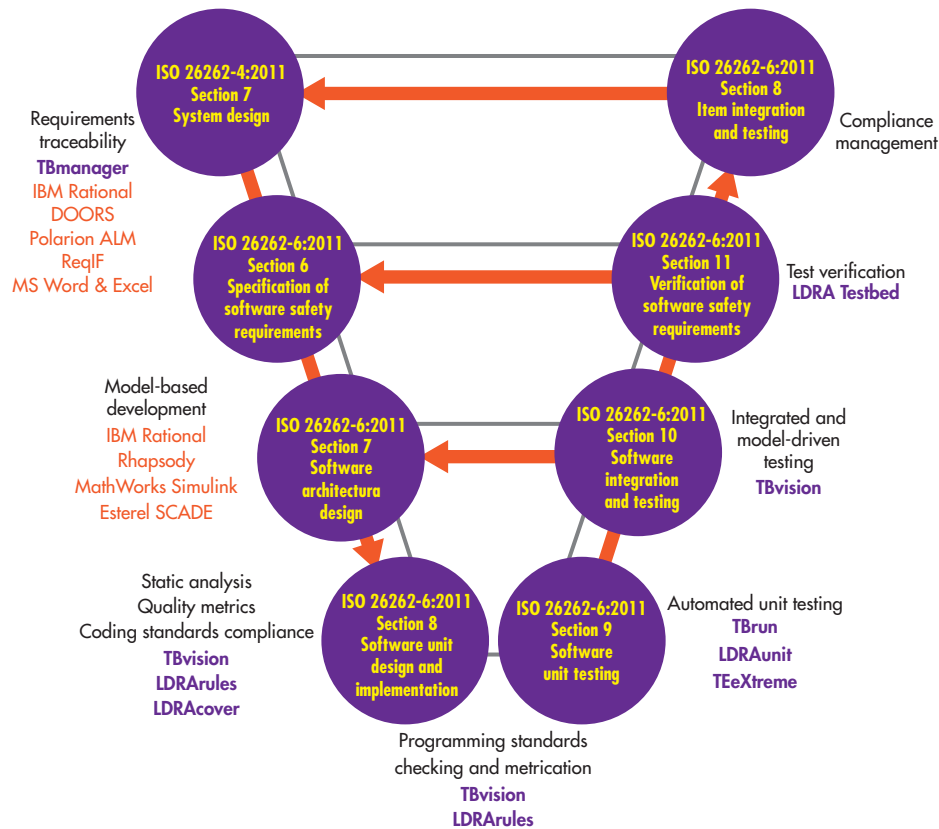
## ISO 26262

Within the requirements of ISO 26262, software unit implementation contributes to a more testable, high-quality application. While ISO 26262 does not specify a particular coding standard, it does require that one be employed. Appropriate standards and guidelines such as MISRA C:2012, MISRA C++:2008, SEI CERT C, CWE, and others share the goal of eliminating potential safety and security issues and are supported by automated tool suites. The coding guidelines can be regularly checked and enforced using an integrated static analysis tool that examines the source code and highlights any deviations from the selected standard.

Beyond adherence to coding standards, fully integrated software tools can check and enforce guidelines for quality design of software units and facilitate their integration and testing according to the defined software architecture and the system's requirements. Applying and enforcing such principles at the unit coding level makes it more certain that the units will fit and work together within the defined architecture. Ideally, an integrated tool suite should collate these automated facilities so that they can be applied to all stages of development in the standard "V" process model (Fig. 1) and can coordinate requirements traceability, analysis, and testing over all stages of product development.

## Coding Standards and Guidelines in the Context of

## Static Analysis and Software Unit Testing



1. Mapping the capabilities of an automated tool chain to the ISO 26262 process guidelines.

From a broad perspective, ISO 26262's practices are aimed at making code more understandable, more reliable, less prone to error, and easier to test and maintain. For example, restricting the size of software components and their interfaces makes them easier to read, maintain, and test, and therefore less susceptible to error in the first place. Static analysis can check a host of guidelines to ensure that variables are initialized, global variables are not used, and recursion is avoided. The existence of global variables, for instance, can cause confusion in a large program and make testing more difficult. Applying static analysis throughout the code-implementation phase can highlight violations as they occur, and ultimately confirm that there are none present. In addition, tools can generate complexity metrics to make it possible to measure and control software component size, complexity, cohesion, and coupling (Fig. 2).

Software unit testing demonstrates that each software unit (function or procedure) fulfills the unit design specifications and does not contain any undesired behavior. Once that is proven, unit integration testing

**Requirement based test case**

- Table 8 - Design principles for software unit design and implementation - Unfulfilled
- 1a - One entry and one exit point in subprograms and functions - Unfulfilled
- 1b - No global objects or variables or one online test during their creation - Unfulfilled
- 1c - Initialization of variables - Unfulfilled
- 1d - No multiple use of variable names - Unfulfilled
- 1e - Avoid global variables or else justify their usage - Unfulfilled
- 1f - Limited use of pointers - Unfulfilled
- 1g - No arbitrary type conversion - Unfulfilled
- 1h - No hidden state flow or control flow - Unfulfilled

**Unexecuted code for the given test case**

```

11 void rankSpeedCommand (t_T14 command)
12 {
13     switch (command)
14     {
15         case CALCULATE_CMD:
16             calculateAirSpeed (airSpeed);
17             break;
18         case DISPLAY_CMD:
19             displayAirSpeed (airSpeed);
20             break;
21     }
22 }
    
```

**Unexecuted data reference for the given test case**

Variable Name	Call Depth / Parameter Name	File	Procedure	Type Code	Address Code	Call in Sim.
airSpeed	0	AirSpeedCommand.cpp	rankSpeedCommand	Q	00000000	0
command	0	AirSpeedCommand.cpp	rankSpeedCommand	Q	00000000	0
data	0	AirSpeedCommand.cpp	rankSpeedCommand	Q	00000000	0

On line 39 the reference to airSpeed by displayAirSpeed is not executed with this test case

**2. Static analysis can examine control and data coupling of a software unit and relate it to the system architecture.**

demonstrates the continued correctness of that behavior as these units are deployed as part of a system, and then that this integration actually realizes the software architectural design laid out at the higher level. Perhaps the ultimate integration test is a system test, when all of the software is operated as a coherent whole.

Unit testing and unit integration testing uses this framework to provide a harness to execute a subset of the code base, and verify that the functionality of the software interfaces is in accordance with software design specification and requirements. That includes ensuring that only those requirements are met, and that the software does not include any unrequired (that is, undesired) functionality. This same unit test facility can be used to create fault-injection tests for functional safety, measure resource usage, and where applicable, ensure that auto-generated code behaves in accordance with the model from which it was derived.

While static analysis can perform an automated “inspection” of the source code to verify adherence to the ISO 26262 guidelines for coding and unit implementation, beyond that, the information derived from that static analysis can be used to provide a framework for dynamic analysis—the analysis of executed code. Ideally, all dynamic analysis should be implemented using the target hardware so that any issues resulting from its limitations are highlighted as early as possible. If target hardware is not available in the early phases of a project, the code should be executed in a simulated environment based on the verification specification. That can allow development to proceed with the caveat that testing on the actual target hardware will ultimately be needed.

**Ensuring Traceability to Software Safety Requirements**

Integration testing, then, is designed to ensure that all units work together and in accordance with the architectural design and the requirements. In the case of an ISO 26262-compliant project, that implies the verification of functions relating to the ISO 26262 software safety requirements as well as more generic functional

requirements. Again, these tests can initially use a simulated environment, but ISO 26262 then calls for analysis of the differences between source and object code and between the test and the target environments in order to specify additional tests for ultimate use in the target environment. In any event, testing on the target hardware must be completed prior to certification.

Where tests fail, it is likely that code will need to be revised. Similarly, requirements can change part-way through a project. In either case, all affected units must be identified and all the associated unit and integration tests must be re-run. Fortunately, such regression tests can be automated and systematically re-applied to assure that any new functionality does not adversely affect any that is already implemented and proven.

This constant attention to the faithful representation of requirements by the code is especially relevant to unit test and integration. The inputs and expected outputs for these tests are derived from the requirements, as are tests for fault testing and robustness (Fig. 3). As units are integrated into the context of their associated call trees, the same test data can be reused.

Unit and integration testing with dynamic analysis ensures the software functions correctly, both as a unit and “playing well with others” when connected to other units in the overall program. However, in the latter context it is also necessary to evaluate the completeness of such testing as well as to make sure there is no unintended functionality. Function and call-coverage analysis tests to see that all calls have been made and all functions have been called. It is, however, necessary to more thoroughly examine the structure by executing statement and branch coverage, which assures that each statement has been executed at least once and that each possible branch from each decision point is taken at least once.

Where there are multiple conditions to consider at such a decision point, the number of possible combinations can soon lead to a situation where testing each of them is impractical. Modified Condition/Decision Coverage (MC/DC) is a technique that reduc-

**File Explorer**

- IdentifyProduct
- removeLastProduct
- startSession
- CashRegister\_barcode
- CashRegister\_cancel
- by Calls
- Global Variables
- Return Type - void
- Combined Coverage Run
  - Statement Coverage - 100%
  - Branch/Decision Coverage - 100%
- Current Coverage Run
  - Statement Coverage - 100%
  - Branch/Decision Coverage - 100%
  - Branch/Decision / Decision Coverage
- CashRegister\_code
- CashRegister\_end
- CashRegister\_key

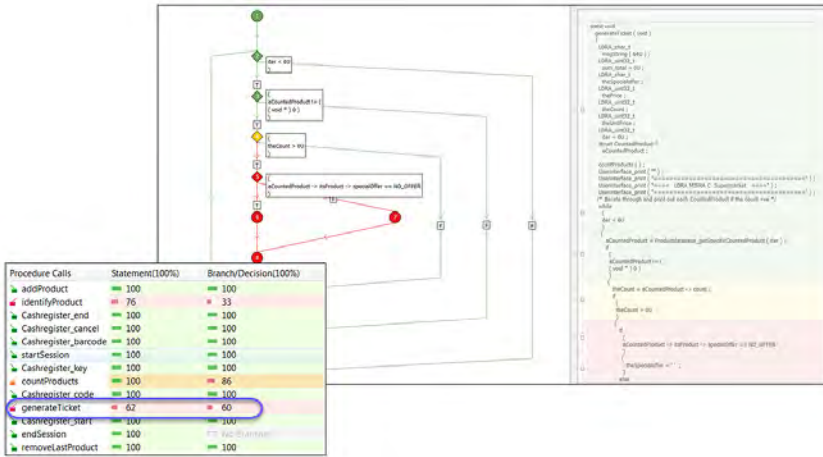
**Test Case View**

Test Case	Regression P / F	Procedure	Object
14	PASS	removeLastProduct	
15	PASS	removeLastProduct	
16	PASS	startSession	
17	PASS	CashRegister_barcode	
18	PASS	CashRegister_barcode	
19	PASS	CashRegister_cancel	
20	PASS	CashRegister_cancel	
21	PASS	CashRegister_cancel	
22	PASS	CashRegister_cancel	

**Variable I/O View**

Value	Name	Type	Use	Regression Analyst	Regression
0U	scannedProducts	LDRA_uint32_t	Input global	Assigned	
1	state_Active	state	Input global	Assigned	
0U	scannedProducts	LDRA_uint32_t	Output global	Compare + Write	1+
0	state_Site	state	Output global	Compare + Write	1+

**3. Requirements-based testing allows the developer to enter inputs and expected outputs in the LDRA tool suite. The outputs are captured along with structural coverage data and compared with the expected outputs.**



**4. Structural coverage analysis of the application correlates the internal functions of the units and their interfaces with the architectural design of the system.**

es the number of test cases required in such circumstances, calling only for testing to demonstrate that each condition can independently affect the result.

Coverage analysis at the unit level will verify the conditions within that unit but will obviously not exercise calls outside that unit. Unit tests, integration tests, and system tests can all contribute to the degree of coverage over the entire project (Fig. 4).

To place all of this into context: the functions of the software units are determined by the software architectural design, which is in turn determined by the requirements. Both the requirements and architecture, which define the units, also define the testing required by each of those units. In turn, when the units are integrated, they are tested to verify their functional interaction as well as their compliance to the software architectural design and—for ISO 26262—both functional and safety requirements.

The requirements at the top of the V-model shown in Fig. 1 are often defined using specialized requirements tools such as IBM

Rational DOORS, or modeling tools such as MathWorks Simulink. Having a software tool suite that interfaces with such tools can be an advantage in verifying the bidirectional traceability that is required by ISO 26262, even where modeling tools are used to automatically generate source code (Fig. 5). These auto-generated units are subject to the same rigorous testing, verification, and integration procedures as hand-coded units, legacy code, and open-source code.

**Automated Testing and Verification Tools Keep ISO 26262 Projects on Track**

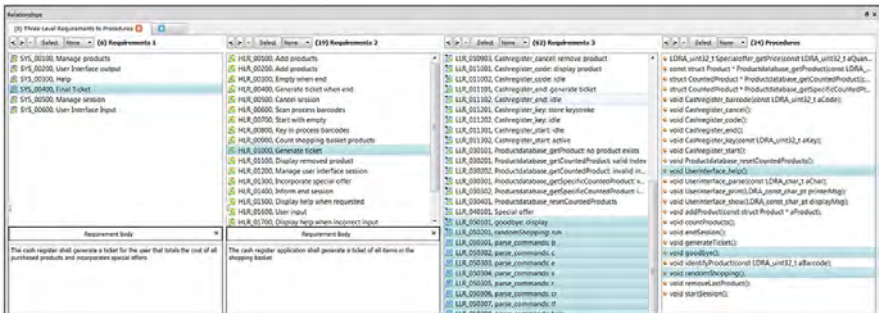
It is easy to think of development as a stage-by-stage process, with test coming somewhere after coding. However, regular testing during development complete with bidirectional requirements tracing is vital because the later a failure shows up, the higher the cost in both time and money.

With all this concurrent activity, maintaining an up-to-date handle on project and traceability status by traditional means is a logistical nightmare. For example, the possible cause of an integration test failure might be a contradiction in requirements, something that is much easier to deal with if recognized early. If it is later and the requirements need to be modified, it will have an inevitable ripple effect through the project. What other parts of the software are affected? How far back do you have to modify and test to be sure the change is covered?

A similar unhappy scenario accompanies a coding error discovered late in the day. What other units are dependent on that code? What if there is an incorrect specification in one of the requirements but unit tests have already been run and are now at least suspect? How do you find your way to know that everything has been fixed?

In such situations, manual requirements tracing will at some point break down. At the very best, it will still leave a sense of uncertainty. However you collate your requirements, whatever design approach you adopt, whether you develop model-generated or hand-generated code, automated testing and verification tools can do more than just report on the status of a particular development stage. An integrated tool suite can provide traceability between the requirements, the design, and the source code—from the lowest-level functions and their test results up to the specified requirements. Staying on track with ISO 26262 from bright idea to a reliable and safely running system needs constant attention with a flair for detail that only automated tools can deliver.

*to view this article online, [click here](#)*



**System requirements      Software high-level requirements      Software low-level requirements      Source code**

**5. The LDRA traceability function shows a detailed design linked upstream to software requirements and downstream to software units.**

**BACK TO TABLE OF CONTENTS**

Want more coverage like this?



STEPHEN PATERAS, Product Marketing Director,  
Tessent Group - Mentor, a Siemens Business

## CHAPTER 2:

# ACHIEVING AUTOMOTIVE FUNCTIONAL SAFETY THROUGH RUNTIME MONITORING

**As the amount and complexity of automotive safety-critical functions continues to expand, the need for regular in-line monitoring of electronic systems will grow as well. Here is an example of a chip-level test architecture supporting system-wide monitoring.**

The accelerating march towards autonomous vehicles is driving a fundamental change in the nature of automotive semiconductor parts. Gone are the days where automotive devices had the luxury of being developed and manufactured using mature and robust processes. Nor can competing brands or their suppliers afford the automotive industry's traditionally long product renewal cycles.

Automotive devices are no longer only for simple functions like controlling windows or light signaling but are now required for complex functions related to advanced driver-assist systems (ADAS) and increasingly for autonomous driving applications. The processing power required for these advanced functions results in the need for very large and complex devices manufactured in the most advanced process nodes. This, coupled with the need for these devices to meet the notoriously stringent safety requirements defined by the ISO 26262 standard (a standard for functional safety of electrical and electronic components in road vehicles up to 3500 kg.), is resulting in a perfect storm for automotive device and systems manufacturers. Solutions are needed to ensure new complex automotive electronic systems operate safely

at all times throughout the life of the vehicle.

An emerging approach that is gaining broader adoption is to make use of a set of embedded monitoring functions distributed throughout each semiconductor device and tied together through a global communication infrastructure that enables rapid detection and reporting of random failures anywhere in the system. The monitors must operate without interfering with normal functional operation and have the flexibility to provide varying degrees of failure coverage based on the end-application of the semiconductor device and the associated Automotive Safety Integrity Level (ASIL) classification defined by the ISO 26262. There are four ASILs identified by the standard: ASIL A, ASIL B, ASIL C and ASIL D. ASIL D dictates the highest integrity requirements and ASIL A the lowest. The intervening levels are simply a range of intermediate degrees of hazard and degrees of assurance required.

An example chip-level test architecture supporting distributed system-wide monitoring is illustrated in *Figure 1*.

A standard IEEE 1149.1 test access port (TAP) provides a portal to all on-chip test resources for manufacturing test. The TAP connects to a reconfigurable serial access network based on the IEEE 1687 standard (often referred to as the IJTAG standard). This

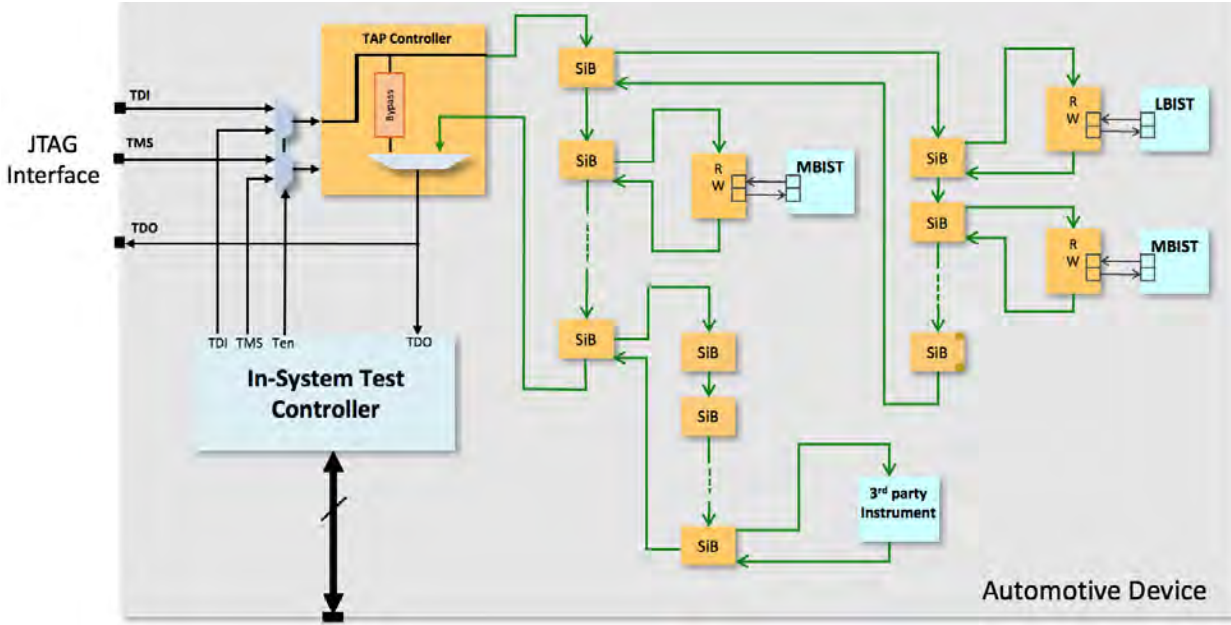


Figure 1: Chip-level test architecture for in-system test

IJTAG network is made up of switches called segment insertion bits (SIBs). Each SIB allows a sub-network to be switched-in or bypassed, allowing for optimized access to any test resource within the network. The IJTAG network is also accessed by an In-System Test (IST) controller. The IST controller communicates to the out-

side world through a CPU interface and performs the parallel to serial and serial to parallel data conversion necessary to transport information between the external CPU bus and the internal IJTAG network. This IST controller enables a system-level communication architecture as illustrated in *Figure 2*.

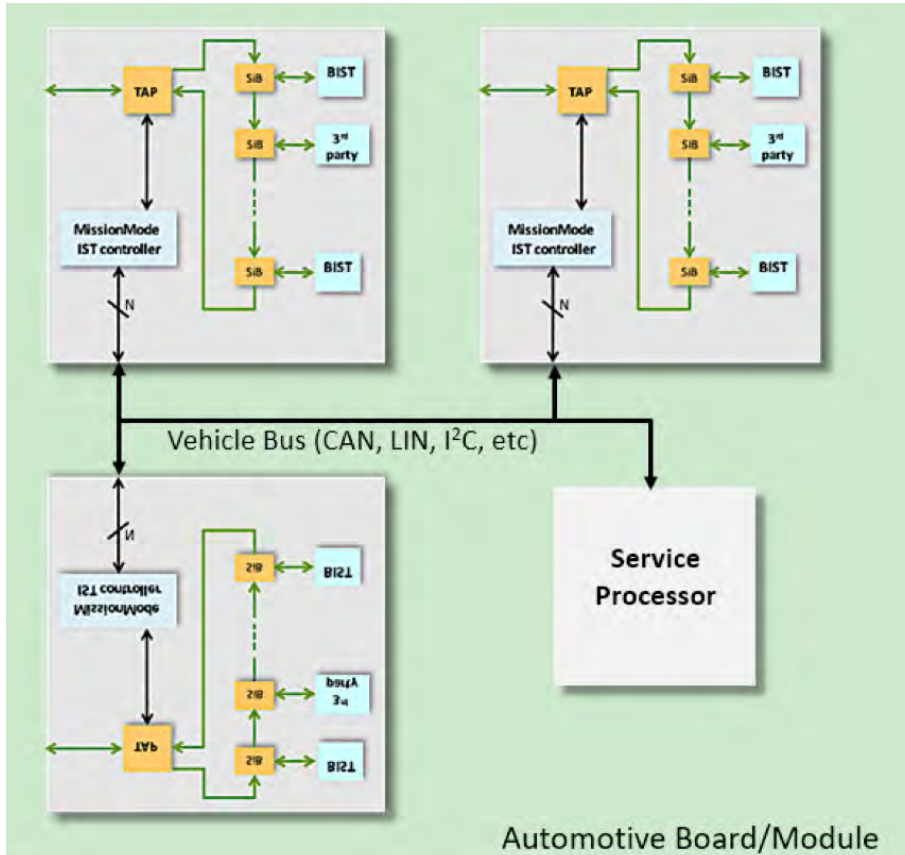


Figure 2: System-level test architecture

A service processor can access each chip's IST controller and hence any on-chip test resource through whatever backplane vehicle bus implemented such as CAN (Controller Area Network) or I<sup>2</sup>C (Inter Integrated Chip).

The effectiveness of this distributed systems depends on the test resources implemented within the various devices. Probably the most common form of on-chip test resource is Memory Built-In Self-Test (BIST). An MBIST engine fully tests an embedded memory by algorithmically generating a sequence of read and write operations that covers the entire address space. A major difficulty in running such a memory test during vehicle operation is that the memory must first be taken offline to allow the BIST engine to take control. It may also be necessary to back up the memory contents before running the test and restoring the contents afterwards as the memory test will destroy any pre-test memory content. Another complication is that taking the memory offline will also likely degrade the system's performance, which may not be acceptable in some applications.

A new non-destructive MBIST technique has been developed to avoid all of these problems.



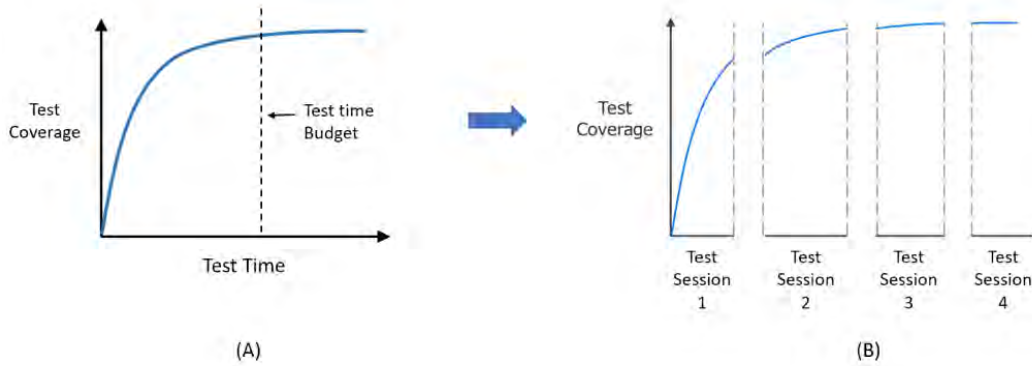


Figure 3: Managing Logic BIST test time

In this approach, the MBIST engine tests the memory using a series of short sequences of transactions, often referred to as bursts. A burst will typically only last for a small number of clock cycles (perhaps 20 to 30) and targets different memory locations each time. The entire memory is therefore tested over a large number of short MBIST sessions. The approach is non-destructive because the memory locations that are modified by a burst are saved and restored during each burst by the MBIST engine. Functional performance is not significantly affected because the bursts are only initiated when arbitration logic implemented between the MBIST engine and the functional logic determines the memory is free.

Logic BIST is another popular form of in-system test resource that can be accessed through the IST controller. This test solution involves the on-chip generation of random patterns that are applied to scan chains to test the logic portion of a chip. The circuit responses to all of the random patterns are accumulated into a signature, which is examined at the end of the test for a pass/fail result. The test coverage achieved by applying an increasing number of random patterns grows logarithmically as shown in *Figure 3a*.

A common challenge in using this approach is achieving a high enough test coverage within a given time budget. A solution to this problem is to break up the test into multiple sessions as shown in *Figure 3b*, above. Each successive session is applied during an available break in the functional operation. For example, in an image processor used to process visual data, each test session could be applied in between processing individual image frames. Implementation of the multiple test session solution requires careful coordination between the IST controller and logic BIST engine. The IST controller must keep track of which test session is to be applied next, initialize the logic BIST engine to have it generate the correct set of random patterns, and then retrieve and compare the intermediate signature to determine pass or fail status.

A distributed monitoring capability as described above enables any number of system level safety-related functions to be implemented. Key-on and key-off tests can easily be accomplished by sending out commands to all IST controllers to have all test resources run their most comprehensive tests. Any test failures can

be reported back to the system software, which can use the results to drive some form of corrective action from something as simple as displaying a warning message on the dashboard to powering down the vehicle for further service. The IST controllers can also be instructed to run intermittent tests while the vehicle is operating on portions of the electronic system that are involved in safety-critical functions. Failing results from these tests would likely drive immediate actions that would place the vehicle into some safe operational state.

The need for regular in-line monitoring of automotive electronic systems will no doubt continue to grow as the amount and complexity of safety-critical functions continue to expand. Some commercial solutions to address this need have already been introduced and no doubt will continue to evolve over time.

*Stephen Pateras is product marketing director within Mentor's Tessent group and has responsibility for the company's BIST and automotive test solutions. His previous position was VP marketing at LogicVision where he was instrumental in defining and bringing to market several generations of LogicVision's semiconductor test products. From 1991 to 1995, Stephen held various engineering lead positions within IBM's mainframe test group. He received his Ph.D. in Electrical Engineering from McGill University in Montreal, Canada.*

to view this article online, [click here](#)

[BACK TO TABLE OF CONTENTS](#)



## CHAPTER 3:

# COMBINE AUTOSAR STANDARDS FOR HIGH-PERFORMANCE IN-CAR COMPUTERS

**Adaptive AUTOSAR paves the way for expandability in ECU software in vehicles. By replacing the trusted domain controller architectures with powerful control units via new processors, Ethernet, and potentially a Linux OX, flexibility is greatly increased across domains.**

Developments such as the trend toward alternative drive concepts and automated driving place high demands on the E/E architecture inside vehicles. Automated driving functions, in particular, call for more and more powerful control units. In addition, functions are becoming increasingly networked with each other and with connected infrastructure, such as back-end systems (Fig. 1). Using data from various vehicle sensors as well as external sources (such as highly precise map material), for example, the software creates accurate environment models that are used by multiple driving functions simultaneously.

At the same time, it must be possible to update the software over the lifetime of the vehicle to incorporate new functions or defend against new security risks. The more driving tasks the software assumes, and the more connections there are to the outside world, the greater the demands are on the functional safety, security, reliability, and integrity of control units.

Given these complex requirements, today's largely static systems are soon found wanting. For that reason, a small number of very powerful control units are replacing the trusted domain control-

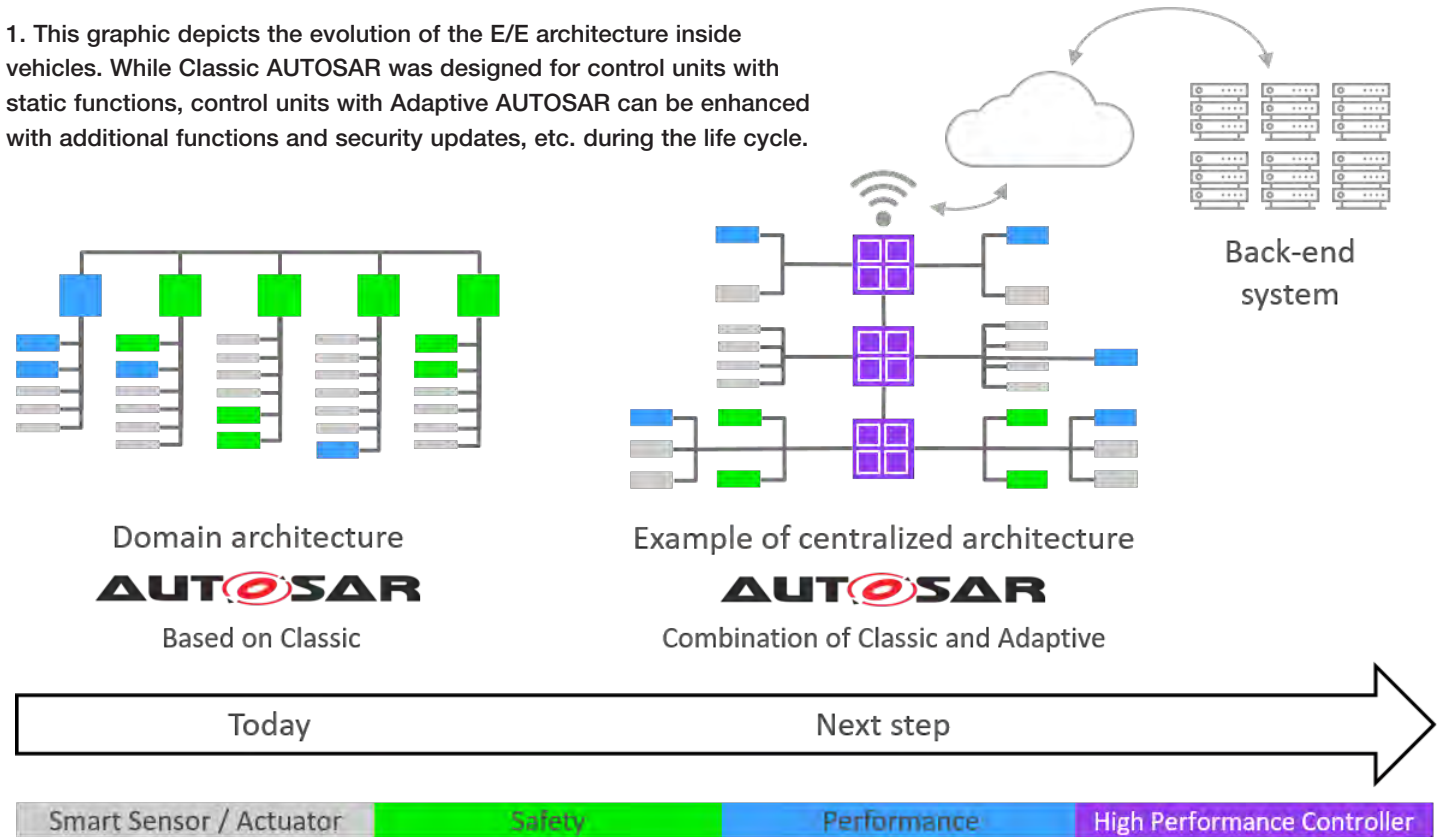
ler architectures. This increases flexibility and allows a dynamic distribution of functions across domain boundaries. This new architecture is enabled on the hardware side by the development of ever better (multi-core) processors, as well as the availability of Automotive Ethernet, which removes the bandwidth limitations for the exchange of information between individual modules.

## Software Must Become More Flexible

The corresponding software requires a more flexible architecture that can represent these dynamics. That is why Classic AUTOSAR, which has established itself as the standard architecture for control units, is being supplemented with the Adaptive AUTOSAR platform. This approach enables a dynamic software configuration. With service-based communication and heterogeneous computation, it also provides mechanisms that ensure the necessary performance. In addition, it is easier to update or add software functions without having to restart the entire system.

Nevertheless, Classic AUTOSAR is not obsolete. This standard was designed for control units with a limited computing capacity. Unlike Adaptive AUTOSAR, it offers only a static configuration of

1. This graphic depicts the evolution of the E/E architecture inside vehicles. While Classic AUTOSAR was designed for control units with static functions, control units with Adaptive AUTOSAR can be enhanced with additional functions and security updates, etc. during the life cycle.



the operating system. Aside from the disadvantages of the major configuration effort and the limited possibilities for software enhancements, however, this static configuration offers numerous advantages regarding the implementation of safety-relevant software components.

It therefore makes sense to combine a Classic AUTOSAR system with Adaptive AUTOSAR to achieve the necessary performance, as well as the security of the E/E architecture for technologies like autonomous driving. But how can both standards work together meaningfully on a central control unit?

### Multi-Core Architecture with Performance and Safety Cores

The combination of particularly safety-relevant and CPU-intensive functions begins at the hardware level. The central control unit, which in this example uses Elektrobit software, contains a high-performance computer comprising a combination of several multi-core processors (Fig. 2). These processors are divided into performance cores with integrated security hardware and safety cores. Running on the performance cores are multiple performance partitions, in which CPU-intensive vehicle and user functions are executed. They also have a security partition, which guarantees a secure startup and ensures that applications are authenticated.

The safety cores, in turn, enable the execution of safety-critical functions, plausibility checks, monitoring, and validation of the

results of the performance cores. The quantity and composition of the performance and safety cores are flexible in principle and based on the project requirements of the control unit. The central control unit is also connected to other control units via an Ethernet switch with multiple Gigabit Ethernet channels.

There are already initial hardware solutions available for a high-performance central control unit of this nature, such as Renesas' R-Car H3, Intel's Denverton, and Nvidia's Parker (T186). They integrate a combination of powerful performance processors with a safety controller.

A central control unit of this kind forms the basis for a software architecture that fulfills five key requirements:

1. Integration of vehicle functions on one control unit
2. Execution of safety-relevant functions
3. Secure startup of the overall system
4. Optimized communication
5. Updating and addition of vehicle functions

### Integration of Vehicle Functions on One Control Unit

Functions that previously ran on various (individual) control units can now be bundled on one central unit. The hardware resources of the performance cores are separated by a hypervisor. That hypervisor virtualizes the hardware and, in so doing, provides the partitions as virtual machines. In this way, the integrator creates various Adaptive AUTOSAR partitions as well as a Classic AUTOSAR partition. The latter uses an operating system and

basic software based on Classic AUTOSAR. Vehicle functions that exist as software components (SWCs) can be integrated just like in a Classic AUTOSAR control unit. The Adaptive AUTOSAR partitions use a POSIX-compatible operating system and Adaptive AUTOSAR basic software. This structure allows vehicle functions based on Classic AUTOSAR, as well as those based on Adaptive AUTOSAR, to be integrated on a control unit.

In contrast to the performance cores, the hardware of the safety core is designed for a higher safety level (Automotive Safety Integrity Level, ASIL) pursuant to ISO 26262. It offers special mechanisms for detecting errors. Established safety concepts of existing Classic AUTOSAR control units are applied.

Controllers designed to ASIL D combined with a safety-certified operating system and other certified basic software components (for run-time monitoring and securing communication) enable the integration of functions with the highest safety requirements according to ASIL D. Thanks to an overarching concept for monitoring the performance cores, these powerful cores can also satisfy the required safety requirements despite not being designed with safety in mind.

### Secure System Startup

An overarching boot concept allows the central control unit to be started up securely within a defined timeframe (Fig. 2). The sequence and interaction of the units in the boot process are particularly important for the following: fulfilling the time require-

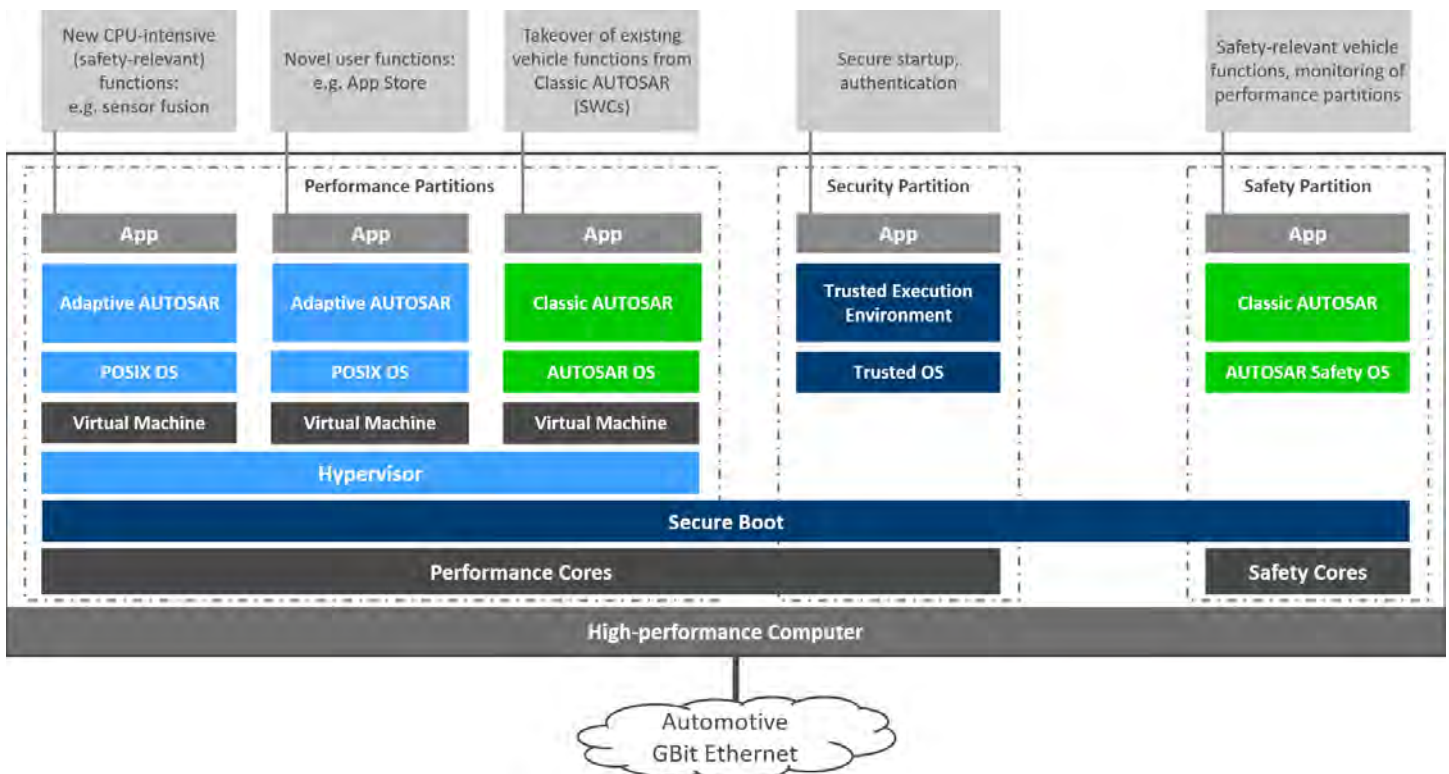
ments in terms of the availability of the systems and enabling the security specifications to be configured as quickly as possible. To this end, the roles of individual cores are defined such that the slaves and their assigned components are started from a master.

The safety core is started first for logical reasons—namely, its monitoring function and shorter start time. The need to provide multiple Ethernet ports requires the use of a high-performance Ethernet switch. This is connected via the safety controller in order to enable quick availability of Ethernet communication. The performance cores are then started, beginning with the security partition that serves as the anchor point for protecting all the lower-level and higher-level applications.

### Optimized Communication

To improve communication between the vehicle functions, both Classic AUTOSAR and Adaptive AUTOSAR use the service-based communication concept SOME/IP. To ensure regulated access by the various partitions to the Ethernet hardware switch within the central control unit, a special virtual Ethernet switch driver is required. In addition to regulating communication with other control units, it simultaneously regulates efficient internal communication between the partitions (Fig. 3).

One of the main features of Adaptive AUTOSAR is the ability to update individual functions on the control unit retroactively and during run-time. In contrast to Classic AUTOSAR, this can be done without replacing and restarting the entire software



2. Shown is the architecture of a central control unit with performance and safety cores.

Want more coverage like this?

11 | [electronicdesign.com](http://electronicdesign.com) | Subscribe to *Electronic Design's* newsletter *Automotive Electronics*

SPONSORED BY NATIONAL INSTRUMENTS

of the control unit. However, it does need to be done in a controlled manner to prevent any faulty or even detrimental updates. Cryptographic processes that run on the security partition are therefore used for all Adaptive AUTOSAR applications. They review the signature of the functions to be loaded and only allow them to be updated once they have been authenticated successfully.

### More Flexibility with Embedded Linux?

In Classic AUTOSAR, the operating system was specified, whereas the use of existing POSIX operating systems is a fundamental component of Adaptive AUTOSAR. This forms the basis for flexible software development. The standardized programming interface enables developers to create applications independently of one another and to distribute them freely to the control units in the vehicle.

The prerequisite here is that the operating system provides the applications with interfaces in accordance with the POSIX profile PSE51 of IEEE 1003.13. Alongside proprietary operating systems—such as Wind River’s VxWorks, Green Hills’ Integrity, and QNX—the free software Linux is a promising alternative. But is it suitable for use in Adaptive AUTOSAR systems?

Unlike commercial POSIX operating systems, which tend to be supplied precompiled as binary code, Linux is available as source code during development and for integration. This enables a more transparent development process, partly through improved debugging for the customer. The Linux kernel is also open to expansion, which means that customers have the opportunity, for example, to add their own kernel modules. In addition, Linux provides functionality far in excess of the required POSIX standard.

Linux is already in use in a variety of industries. It is optimized using feedback from billions of installations and users. The automotive sector, too, already uses Linux, with the focus to date on infotainment and human machine interfaces. The kernel itself is constantly optimized and enhanced comprehensively by the Linux community.

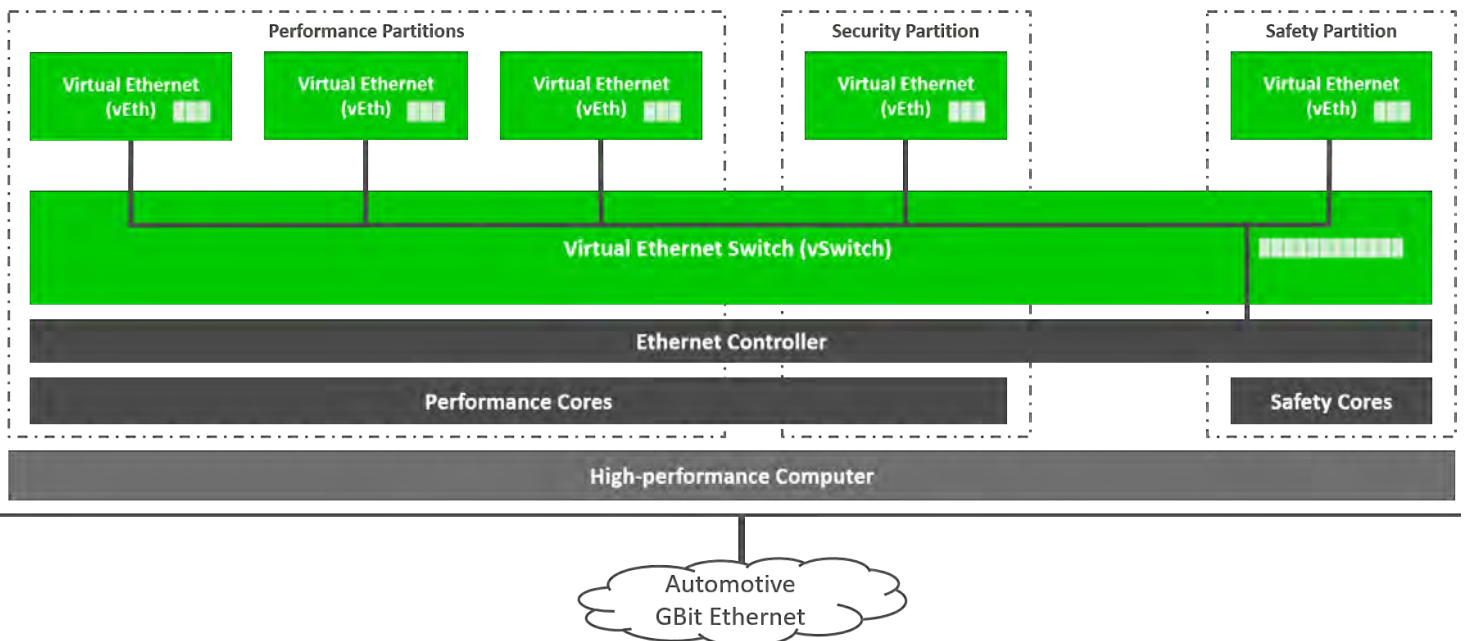
Because Linux is available under a free software license (GPLv2), there are no license fees. Costs are incurred by the customer through services like configuration, customization, and provision, as well as the qualification and maintenance of customized deliveries. Nevertheless, using Linux with Adaptive AUTOSAR does present a number of automotive-specific challenges, as encountered by Elektrobit in its pilot project (Fig. 4).

### Software Updates and Security

A control unit with a connection to the internet is exposed to persistent attacks. These may come from outside, but they can also be triggered by harmful applications on the control unit itself. Alongside the cryptographic processes described above, special extensions, such as seccomp-bpf, are added to the kernel to restrict the system calls of applications.

In addition to the development releases, a special version of the Linux kernel is brought out every year. Long-term support is provided for two years. For a vehicle with a typical development cycle of four years, the vehicle system must be designed by the manufacturer and supplier so that the kernel can be replaced during development and in later operation while the system retains binary compatibility with existing applications.

The use of Linux containers ensures consistency at levels like the memory and CPU, as well as for shared resources. It also



3. A virtual Ethernet switch driver regulates communication with other control units and between the partitions.

enables the separate replacement of individual containers. This solution has proven its worth in other industries and is being used in this project in the automotive industry for the first time.

Many sensitive functions and applications in the vehicle, such as the speedometer and reversing camera, need to be available or executed within a time span of well under two seconds. This is achieved through configuration and optimization of the Linux kernel, removal of unnecessary services, and organization of the startup process based on urgency or demand for availability.

The ability to test and secure such complex systems is a prerequisite for the use of Linux in the automotive industry. The options include freely available and commercial test environments or subcontracting to specialized companies. Organizations such as the Open Source Automation Development Lab (OSADL) offer support services.

## The Road Ahead for Adaptive AUTOSAR

With its four main versions, Classic AUTOSAR has matured over 10 years. The goal is to develop Adaptive AUTOSAR into a mature standard much more quickly. To this end, every delivery of the specification comes with a reference implementation, which is used as a proof of concept. In addition, the use of existing and sometimes free implementations, such as Linux, saves time and effort.

The challenge now is to transform the implementation parts of Adaptive AUTOSAR into software suitable for series production. The quality of the implementation parts will be checked using established methods from Classic AUTOSAR. These include requirements management and the definition of quality features for code and test coverage, which must be used to check the implementation parts according to the release process. On the basis of these results, the decision will be taken to adopt and extend or to use one's own implementations of Adaptive AUTOSAR parts.

The Adaptive AUTOSAR standard is still being developed. For use in a series project, it is vital that the necessary functional components are provided on time. Any delays or the absence of these elements in the planned AUTOSAR releases will impact the progress of the projects. In addition, further requirements or requirements that have yet to be defined fully must be provided to the projects and fed back into the standard.

It is crucial that the new developments needed for Adaptive AUTOSAR, and their possible uses, are evaluated with the customer quickly in order to continue to drive forward the development of the standard. There is obvious potential for a clever combination of Classic and Adaptive AUTOSAR on the central control unit. At the same time, many established solutions from Classic AUTOSAR will be used, such as the certified safety oper-



4. To use Linux with Adaptive AUTOSAR, a number of automotive-specific requirements must be met.

ating system. This will significantly speed up the introduction of Adaptive AUTOSAR for high-performance computers of future generations of vehicles.

*Dr. Roman Pallierer studied computer engineering at the Vienna University of Technology (TU Wien) and wrote his doctoral thesis on the validation of embedded communications protocols. Starting in 2002, he was responsible for tool development for FlexRay projects at Elektrobit. Since 2007, he has been working as Product and Solution Manager for the AUTOSAR basic software.*

*Börge Schmelz studied electrical engineering at the Technical University of Cologne (TH Köln). Since 2005, he has been involved in the development of AUTOSAR control units at Elektrobit. In 2010, he took over the management of global service projects with a view to successfully realizing series production ramp-ups globally with the use of EB products. In 2017, he switched to product management for high-performance computers and Adaptive AUTOSAR.*

to view this article online, [click here](#)

[BACK TO TABLE OF CONTENTS](#)



PAUL HIGHTOWER  
CEO of Instrumentation Technology Systems

## CHAPTER 4:

# INDEPENDENT THINKING: WHY THE “BLACK BOX” IS NEEDED FOR AUTONOMOUS VEHICLE DEPLOYMENTS

While many of the technologies to deliver self-driving vehicles are in use or demonstration phases, regulatory issues are not resolved.

Realization of autonomous vehicles (AVs) rely on artificial intelligence (AI) coupled with machine learning (ML). The decisions an AI system makes from instant to instant will depend on its perception of the local environment (objects in “view,” road conditions, the condition of the vehicle, the load the ve-

hicle is carrying, local rules of the road, and the experience accumulated since being put into service). The variables are so many that even the same vehicle may not behave the same way when traversing from A to B, depending upon the conditions. In other words, autonomous cars will have a personality of sorts.

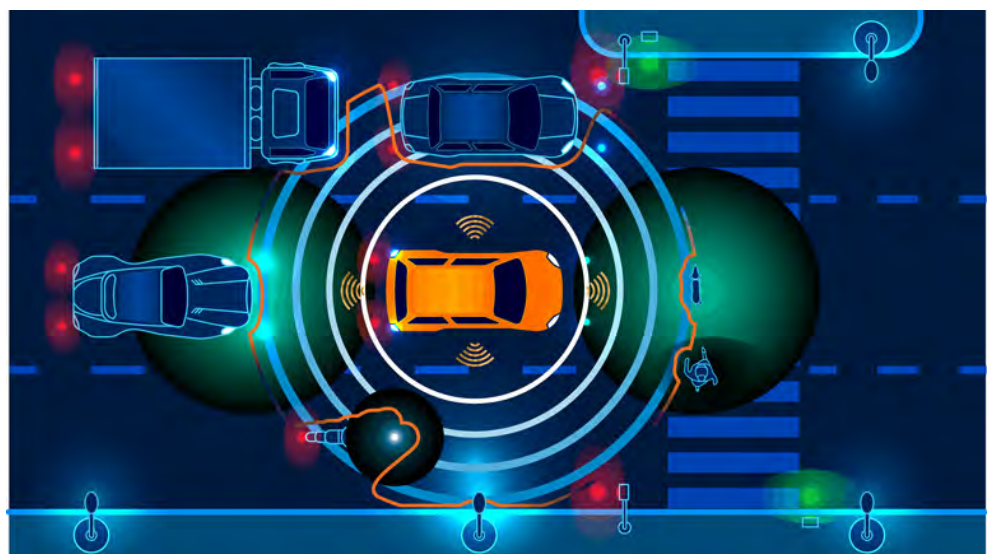
How does one evaluate AVs, then? Would you trust the manufacturer who has a vested interest in getting to market and avoiding liability? Moreover, in driving many variables can affect success

Data input from multiple sensors replace our five senses to inform us what is going on around the vehicle.

(getting to your destination). The objective may not be just “getting there.” In real life, it is often “getting there at a specific time.”

## How Vehicles Are Tested

The time objective adds parameters that vary with the travel itself. Since the vehicle is making the pathway, segment speed,



Want more coverage like this?

obstruction avoidance, collision avoidance, travel time decisions, and more, testing an AV is not just about braking performance or how a vehicle crushes in a collision. One must evaluate how the vehicle responds to its environment with pressures from its objectives and the desired driving style of the chief occupant. Testing decision-making capability and appropriateness of a vehicle is a completely new world. It should be a necessary practice to ensure vehicle performance and safety in the extremely wide range of conditions that AVs will be subjected to.

At present, Event Data Recorder regulations only require data to be collected for a short time (less than a second) around an impact sufficient to deploy the airbag system. However, I can think of many driving incidents in my own life that never deployed the airbag. These incidents were caused by human error. In the future, we can replace the cause with AI decision error (although the hope is far fewer).

Since humans may be passive and not even paying attention, it will likely be important to include an independent witness of events to understand the circumstances of an incident. If the AV has human controls (steering wheel, brake, accelerator), what had control at the time of the accident? The witness may need to independently detect lesser incidents and keep a record for future analysis.

An independent witness compiling vehicle sensor information, video, and control data would be valuable to evaluate whether the AI/ML system is properly interpreting sensor data. Additionally, the independent witness could help determine whether the decisions made are at least within the acceptable range defined by expert drivers.

Even with semi-autonomy, how is control handed off between the AI/ML driver and the human? Do both see the same danger and select the same corrective action? Which has the liability if the corrective action is insufficiently effective? How do you know who (AI/ML or human) has control at the instant of the incident?

Scenarios of this type have been investigated by the University of Leeds in the UK. In situations such as this, who has the liability for injury and property damage? Did the human take control away from the AI processor? Did the AI processor override human input? On the other hand, was the ultimate response to a situation a combination of both? If control was handed back to the human, were they paying attention and given enough time to respond?

## Sensors for Modern Testing

The sensors that replace your eyes and ears to tell you what is going on around you and what your car is doing are many. They range from an array of cameras, radars, sonar, laser imagers, and



Vehicle makers need differentiation of control systems. A result may be a myriad of artificial intelligence and machine learning behaviors or personalities.

accelerometers to the on-board GPS and even the clock. The data coming from this collection of sensors must fuse together to form a real-time equivalent of what you see and understand in any instant of a driving situation. To do so makes the AI processing very busy.

What happens if a sensor malfunctions? What backup is there should one or more sensors fail and the AI/ML system is partially blinded? What happens if a sensor is displaced (functioning or not) and thus information needed by the AI processor is flawed or missing altogether?

What happens during a collision when sensors may progressively be damaged or destroyed while the AI processor is trying to navigate the vehicle to safety? Can it still construct and properly classify objects and predict trajectories? If it determines that the probability of failure is high, what does it do? If there are no human controls, does it just stop? How does it find a safe place? In an accident, how could you know what went wrong without an independent black box that collects raw, unprocessed sensory information being sent to the AI/ML system?

All these questions suggest that the nature of vehicle testing is due for a huge change. We can also expect that all of the AI/ML systems will be different as vehicle manufacturers try to find their competitive edge and satisfy the demographics of their target marketplace.

## Onboard Sensing Technology

This all builds a case for an onboard independent witness that can not only provide evidence in an accident investigation, but also can lead to the development of new criteria to improve AI





If there is an incident, how and when, if ever, is control handed off between the AI/ML driver and the human?

processors, including experience and skill loaded into the ML of new vehicles.

It also suggests that the instrumentation at vehicle test ranges may not be adequate to evaluate AVs, test changes and modes, and certify a design as ready for release. Instead, tools that can capture and record data synchronized with video should be required so that we can capture each picture taken by the cameras, each sound bite, each radar image, each action taken by the AI processor, each response by the vehicle systems (brakes, steering, engine), and each input by the chief occupant.

Such information is particularly important in fault analysis. We can evaluate the decisions made by the AI processor and how the chief occupant interacted that resulted in the incident. Armed with this information, all stakeholders can learn the cause of an incident and plot a path to prevent it from happening again.

At test ranges, data/video collection would serve as a tool to independently evaluate if the data from all of the sensors and cameras were correctly integrated to form an “image” of the scene similar or even better than a human would have assembled. This comparison should be through the eyes of the sensors themselves so that expert evaluators can see what information and imagery was presented to the AI processor to create its interpretation of its physical environment.

In real-world environments, this data/video should be easily accessible to first responders and accident investigators. It will be essential for them to collect all relevant information about the AV’s behavior and influence in the resulting event.

Autonomous cars are different from any other. They are decision makers. How we test them, introduce them to the marketplace, investigate accidents, and manage maintenance will be very different. An independent black box will be necessary to improve

decision-making and collect the data needed to understand the causes and consequences of incidents.

*Paul Hightower is CEO of [Instrumentation Technology Systems \(ITS\)](#), the market leading supplier of HD-SDI video-data fusion products. While bringing traditional text and graphics overlay and accurate timestamping capability to the HD-SDI engineering test market, the company has pioneered the use of metadata to collect image relevant data in real time.*

to view this article online, [click here](#)

[BACK TO TABLE OF CONTENTS](#)



## CHAPTER 5:

# TESTING TIMES FOR AUTOMOTIVE: 8 STANDARDS TO TRACK

While autonomous vehicles hold sway with the masses, engineers are busy making sure what we have now continues to operate safely, effectively, and in compliance with recognized standards from bodies such as the Society of Automotive Engineers (SAE). That said, technology is changing rapidly and the SAE is going full tilt to keep up. Here is a selection of electronic hardware and software SAE standards, both new and updated, that you should be watching to ensure you don't get "dieseled."

## 1. (SAE) Recommended Practice for Pass-Thru Vehicle Programming (J2534/1)

Last revised in December 2004, the update of the Society of Automotive Engineers (SAE) [Recommended Practice for Pass-Thru Vehicle Programming](#) (J2534/1) is particularly interesting given the recent Volkswagen diesel emissions debacle. The RP describes a standardized interface that connects between a PC and vehicle to enable the reprogramming of emission-related control modules. It allows each vehicle manufacturer to control the programming sequence for the electronic control units (ECUs) in their vehicles.

### *Rationale for 2015-10-2 Update*

Concerns from vehicle OEMs, regulators, and application programmers prompted updating of the J2534-1 document to address a wide range of items, from ensuring testability and consistency among compliant devices and clarification of APIs, to message handling, timing/control settings, and the addition of a pass-through device discovery mechanism.

## 2. Class B Data Communications Network Interface (J1850)

The [Class B Data Communications Network Interface](#) (J1850) standard is widely used for diagnostics and data sharing in applications such as engine, transmission, ABS, and instrumentation. Class A supports rates up to 10 Kbit/s, Class B supports rates of up to 100 Kbits/s, and Class C supports up to 1 Mbits/s. The standard defines a minimum set of data communication requirements such that the resulting network is cost-effective for simple applications and flexible enough to use in complex applications.

### *Rationale for 2015-10-14 Update*

Last updated in 2006, J1850 is being revised to detail a High-Speed Mode (83.3 Kbits/s) using pulse-width modulation (PWM), as well as 4x Speed Mode (41.6 Kbits/s) and Block Mode (unlimited data length) for the VPW protocol (popular with GE).

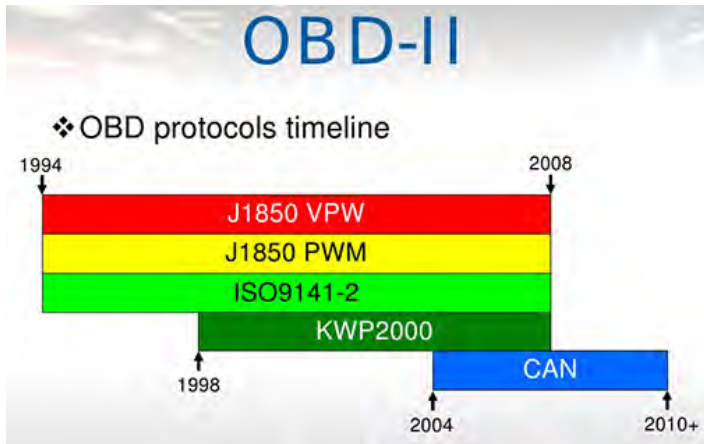
## 3. SAE Standard J551/15 for Vehicle Electromagnetic Immunity

Nothing spells disaster quite like an electrostatic discharge (ESD) at the wrong time, such as at a gas station, or when an internal electronic module is in use at a critical time. The SAE standard J551/15 for [Vehicle Electromagnetic Immunity—Electrostatic Discharge \(ESD\)](#) specifies the ESD test methods and procedures necessary to evaluate electronic modules intended for use in complete vehicles.

### *Rationale for 2015-09-17 Update*

The document has been finalized based on the conclusion that it contains basic and stable technology that is not dynamic in nature. As such, it is a good starting point for designers of any automotive electronic module.

## 4. SAE J1962 Diagnostic (OBD) Connector Standard



(Image courtesy of On-Board Diagnostics)

SAE J1962 details the requirements of an on-board [Diagnostic \(OBD\) Connector](#) as defined by U.S. On-Board Diagnostic regulations. It sets forth the functional requirements of the vehicle connector as well as the external test equipment.

*Rationale for 2015-09-11 Update*

When the California Air Resources Board (CARB) asks for clarification, it's time to provide it. In this case, CARB wants clarification regarding the access area to the vehicle diagnostic connector that is intended to provide proper clearance for the mating of the scan-tool connector to the vehicle. The SAE committee has responded in kind and approved the review of J1962.

### 5. SAE J1939-73 Diagnostics Application Layer

SAE J1939-73 [Diagnostics Application Layer](#) defines the SAE J1939 messages to accomplish diagnostic services and identifies the diagnostic connector to be used for the vehicle service tool interface.

*Rationale for 2015-08-28 Update*

Changes were made to address issues identified during the development and of the SAE J1939-84 test capability. Among others, changes include adding text to limit the number of SPNs that need to be reported in DM24 for data stream, freeze frame, or test results.

### 6. SAE J2600 Compressed Hydrogen Surface Vehicle Fueling Connection Devices

Hydrogen fuel cells for automotive applications continue to evolve, but it has to do so carefully given the storage pressures involved. SAE J2600 [Compressed Hydrogen Surface Vehicle Fueling Connection Devices](#) helps in this evolution by setting the standard for the design and testing of compressed hydrogen surface vehicle (CHSV) fueling connectors, nozzles, and receptacles.

*Rationale for 2015-10-21 Update*

Only three years after the last revision in 2012, J2600 was revised in October of this year to reflect the incorporation of additional geometries for different pressures, and to modify, add, or delete testing requirements based on lessons learned as the state of the art



Re-fueling a Hyundai ix35 with hydrogen fuel. (Image courtesy of Bexi81 via Wikimedia Commons)

has progressed. Also, the document moved from a Recommended Practice to Standard.

Re-fueling a Hyundai ix35 with hydrogen fuel. (Image courtesy of Bexi81 via Wikimedia Commons)

### 7. J1939/16 Automatic Baud Rate Detection Process

J1939/16 [Automatic Baud Rate Detection Process](#) is a brand-new one from the SAE. It is tasked with defining a process enabling devices to detect the baud rate of a network automatically without disrupting existing devices.

*Rationale for its issuance on 2015-10-04*


Manufacturers of devices capable of operating on SAE J1939 networks with different baud rates need to reliably detect the baud rate of that network segment—without interrupting network communications. The goal of J1939/16 is to establish a common method of automatically detecting the baud rate of an SAE J1939 network.

### 8. SAE's J3029 Forward Collision Warning and Mitigation Vehicle Test Procedure—Truck and Bus

The name itself is a mouthful, but we're thankful it now exists: the SAE's J3029 [Forward Collision Warning and Mitigation Vehicle Test Procedure—Truck and Bus](#) is also a brand new Recommended Practice to establish a uniform test procedure for forward collision avoidance and mitigation (FCAM) systems used in highway commercial vehicles and coaches greater than 10,000 lb.

*Rationale for issuance on 2015-10-22*

With widening commercial availability of FCAM systems, a vehicle test procedure to evaluate the effectiveness of these systems is justified. J3029 outlines a basic test procedure to be performed under specified operating and environmental conditions. It does not define tests for all possible operating and environmental conditions. Minimum performance requirements are *not addressed* in this document.

to view this article online,  [click here](#)

 [BACK TO TABLE OF CONTENTS](#)



WILLIAM WONG  
Technology Editor

## CHAPTER 6:

# TAKING AUTOMOTIVE ETHERNET FOR A TEST DRIVE

**Automotive Ethernet utilizes new interface requiring new testing protocols and test equipment.**

**A**utomotive Ethernet has a hidden gem at the Consumer Electronics Show and it will be much more noticeable at this year's show. It is also something supported by the major test and measurement vendors since not all Ethernet tools will work with the new standards.

The standards include [IEEE 802.3bw 100BASE-T1](#) and [IEEE P802.3bp 1000BASE-T1](#). These run the 100 Mbit/s and 1 Gbit/s Ethernet protocols we have come to know and love. They run over a single twisted pair and can have up to four inline connectors, not including the endpoints. There is a tradeoff though. The run length is only 15 m. That is more than enough to handle most automotive and many transportation applications.

Automotive Ethernet will replace the Media Oriented Systems Transport (MOST) bus found on many vehicles today. It will also compete with systems like [Maxim Integrated Products' Gigabit Multimedia Serial Link \(GMSL\)](#).

Chips and PHYs are now readily available for automotive Ethernet as are switches. [Marvell's 8-port, 88Q5050 Secure network switch support the 100Base-T1 standard](#). The switch supports [time-sensitive networking \(TSN\)](#). [Audio video bridging \(AVB\)](#) ingress policy and rate limiting that are part of TSN. It also has 802.1Qav/Qbv queue-shaping support for AVB and TSN. The switch is compatible with IEEE 802.1AS time-synchronization protocols like the precision time protocol (PTP), aka [IEEE 1588](#). These

1. Tektronix's 8-channel, 5 Series MSO has a large, 15.6-in capacitive touch display, 12-bit resolution and bandwidth up to 2 GHz.



Want more coverage like this?



2. Tektronix’s FlexChannel sockets sport eight digital inputs and one high resolution analog input.

timing standards can be found on conventional Ethernet networks as well but they highlight why the test and measurement tools need to be more sophisticated.

Tektronix’s 8-channel, 5 Series MSO (mixed signal oscilloscope) has a large, 15.6-in capacitive touch display (Fig. 1). The scope’s 12-bit ADCs can deliver up to 16 bits of vertical resolution and systems are available with a bandwidth from 350 MHz to 2 GHz. The sample rate on all channels is 6.25 Gsamples/s. Pricing starts at \$12,000.

Tektronix 5 Series MSO systems utilize FlexChannel inputs (Fig. 2). These combine an analog BNC connector with eight digital inputs. What combination of inputs is used will depend upon the probe that is plugged into the socket. The BNC connector can be used directly for coax inputs as well. The 8-input 5 Series MSOs provide up to 64 digital inputs.

The 5 Series MSO has two option packages for the automotive market. The 5-SRAUTO protocol option package addresses standards like CAN, CAN FD, LIN and FlexRay. This includes CAN FD support for non-ISO and ISO versions of the standard. The systems provide complete serial triggering and analysis of these major buses.

The 5-CMAUTOEN Automotive Ethernet package targets the automotive Ethernet standards. There is an automated compliance solution that includes test software that runs while performing Physical Media Attachment (PMA)



3. Rohde & Schwarz’s RTO digital oscilloscopes support automotive Ethernet testing.

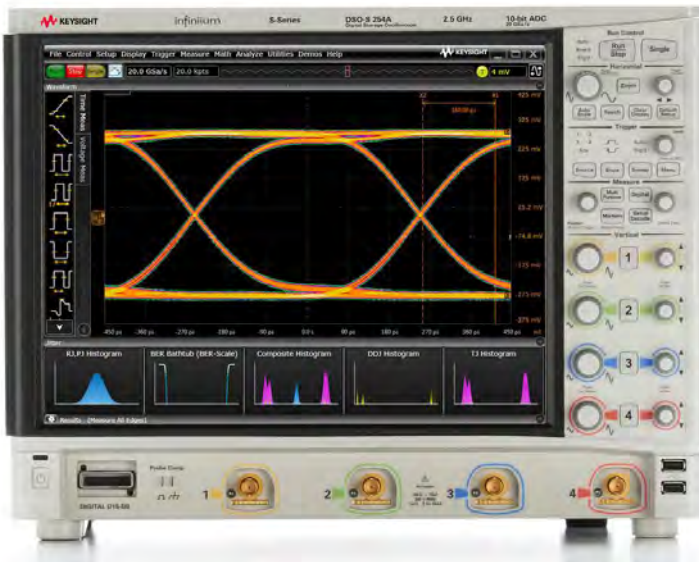


4. Keysight Technologies’ has bundled hardware and software solutions that address the automotive test and measurement market.

transmitter Group 1 electrical measurement compliance tests, as defined by the OPEN Alliance Special Interest Group (SIG) standard for automotive Ethernet.

Rohde & Schwarz’s RTO digital oscilloscopes (Fig. 3) support automotive Ethernet testing. Versions with up to four analog channels

Want more coverage like this?



5. The 4-channel, 2.5 GHz DSOS254A is part of one of Keysight's automotive bundles.

are available with bandwidths that range from 600 MHz to 6 GHz. The sample rate is 20 Gsamples/s with a 16-bit vertical resolution.

The RTO scopes can be combined with triggering and decoding support for LIN, CAN, CAN-FD and FlexRay. The R&S RTO-K24 compliance test option addresses 100Base-T1 while the RTO-K87 compliance test option address 1000Base-T1.

[Keysight Technologies'](#) has bundled hardware and software solutions for the automotive market (*Fig.4*). The bundles can support automated set-up and testing for the transmitters, receivers and link segments. They can handle BroadR-Reach (that the standards were based on), 100Base-T1 and 1000Base-T1. This includes end-to-end functional and standards-compliance conformance testing support.

The bundles come with a number of hardware components like the DSOS254A oscilloscope (*Fig. 5*), the 44 GHz N9010A EXA Signal analyzer and the 2-channel 81150A Pulse Function Arbitrary Noise Generator. The DSOS254A is a 4-channel, 2.5 GHz scope. It has a 15-in touchscreen display. The system operates at 20 Gsamples/s using a 10-bit ADC.

to view this article online, [click here](#)

[BACK TO TABLE OF CONTENTS](#)

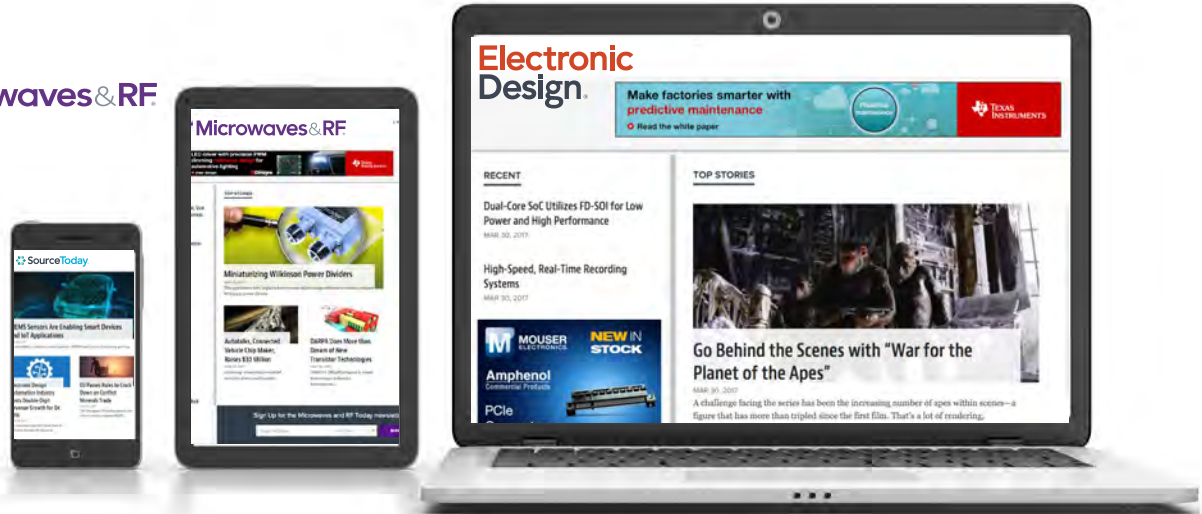
# CHECK OUT THESE RESOURCES FROM ELECTRONIC DESIGN AND OUR SISTER BRANDS

## WEBSITES

**Electronic Design** **Microwaves & RF**

**sourceesb**


**Power Electronics**



## MAGAZINES

You can also apply for a subscription to one of our traditional magazines available in both print and digital formats.


**ELECTRONIC DESIGN – complimentary in US and Canada -  Subscribe Now**  
*Non-qualified or Outside the US and Canada:* you will be prompted to pay based on your location.

**MICROWAVES & RF – complimentary internationally-  Subscribe Now**  
*Non-qualified:* you will be prompted to pay based on your location.



## NEWSLETTERS

Stay current with the industry with newsletters created by engineers and editors with inside connections! You'll find coverage on a wide variety of electronics technologies from Analog & Mixed Signal to Embedded and more.

 **Click Here** to check out what more than 200,000 engineers are reading now.



## ABOUT US

A trusted industry resource for more than 50 years, the Penton Electronics Group is the electronic design engineer's source for design ideas and solutions, new technology information and engineering essentials. Individual brands in the group include *Electronic Design*, *Microwaves & RF* and *Power Electronics*. Also included in the group is a data product for engineers, *SourceToday.com*.

An  
**informa**  
*business*



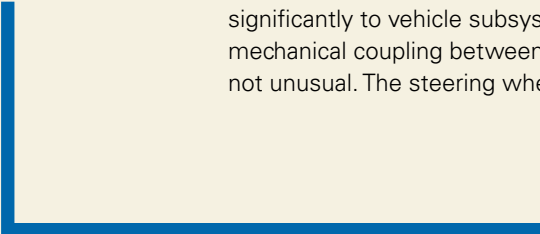


# Vehicle Electrification: Disrupting the Automotive Industry and Beyond



Around the globe, governments are announcing mandates that will bring about the demise of the internal combustion engine. China has led the charge by requiring 8 percent of new vehicles on the road to be “new energy” or zero emission vehicles in 2018, a huge growth over the current 2 to 3 percent on the road today. Similar strong government regulations limiting the future of the internal combustion engine have passed around the world, and the importance and growth of the hybrid and fully electric automobile industry can’t be overstated. Volvo has possibly taken the strongest stance of the automotive manufacturers by pledging to make only hybrid or fully electric cars by 2019 and committing to sell more than 1 million electric vehicles by 2025. “This announcement marks the end of the solely combustion engine-powered car,” said Hakan Samuelsson, president and CEO of Volvo, in a July 2017 statement.

## More Than Just EV/HEV



The move from internal combustion to hybrid and then fully electric power plants represents only the most visible portion of the aggressive growth of power electronics systems in vehicles. Electrification applies just as significantly to vehicle subsystems. As recently as 10 years ago, a fully mechanical coupling between the steering wheel and the front wheels was not unusual. The steering wheel connected to a shaft that connected to a

“This announcement marks the end of the solely combustion engine-powered car.”

—Hakan Samuelsson, President and CEO, Volvo

rack-and-pinion system that turned the wheels, and even a more efficient hydraulic version of the system still maintained a mechanical coupling between the steering wheel and the tires. The story is similar for the accelerator pedal and manual transmission.

The explosion of drive-by-wire technology throughout the modern vehicle has changed this paradigm. A sensor, a remote actuator, and multiple control systems have replaced the mechanical linkage. Instead of a direct connection between the steering wheel and the front tires, a sensor on the steering column now measures the angle of the wheel. An embedded controller then translates that measurement into an angle and sends the value to the vehicle’s communication bus. Elsewhere on the communication bus, another controller picks up the value, translates that into an angle of the wheel potentially based on vehicle speed and driver settings, and then commands an actuator to move the wheel to a desired angle. In many vehicles, a safety system sits in the middle of this drive-by-wire steering system to make sure the vehicle stays in the traffic lane and avoids obstacles in the roadway. As the number of power electronics subsystems in the vehicle grows, the automobile itself begins to look like an electrical microgrid with a common power bus connecting a growing list of sources and sinks of power, each managed by an independent embedded control system.

## The Broader Impact

Taking a slightly broader look at the implications of government automotive mandates, the exponential growth in electrification and the impending end of internal combustion engines represent a radical change in the infrastructure required to support the shift in vehicle power plants. A car with an internal combustion engine requires roughly 10 minutes at nearly any street corner's gas station to fill up its tank for another 300 miles of driving. However, even with a dedicated supercharger, a similar pit stop requires at least an hour for a fully electric vehicle to charge. Even for the slow recharge associated with a daily commute, the required charging hardware needs some thought. For homeowners, installing an overnight charging station might be as simple as putting in a high-current circuit in the garage, but this becomes more complicated for a renter in a house or an apartment. If a car owner happens to live in a city and parks on the street, the concept of a home-charging station might be completely impossible.

Looking at the future of vehicle electrification from the prospective of the electrical utility, the cyclic demands based on the daily workforce schedule combined with the high-load demands of fast charging present incredible new challenges for the electrical grid. If an entire workforce returns home at 5:00 p.m. and plugs in its electric vehicles around the same time, this shifts the timing of the typical peak demand on the grid and refocuses the regional peak consumption from heating or cooling toward transportation. On the larger scale of a gas station, a collection of the superchargers for fast charging will require an amount of energy similar to that of a medium-sized neighborhood.

The government-mandated trend of electric vehicles directly leads to growth in the complexity of vehicles and indirectly leads to an immediate need for growth in infrastructure. The future of the automotive industry will drive the future of the grid, which will require smarter control systems. Turning this into

reality represents a truly interdisciplinary challenge to build safe and reliable control systems among other needs. To get to market quickly, this will require an increased reliance on real-time test, production test, and ecosystem partners who have vertical expertise building tools on top of an industry-leading, flexible, and open platform. With the right tools, engineers can adapt to the disruptive technologies vehicle electrification will require.

