Recent Advances and Trends in On-Board Embedded and Networked Automotive Systems

Lucia Lo Bello; Riccardo Mariani; Saad Mubeen; Sergio Saponara

Abstract:
Modern cars consist of a number of complex embedded and networked systems with steadily increasing requirements in terms of processing and communication resources. Novel automotive applications, such as automated driving, rise new needs and novel design challenges that cover a broad range of hardware/software engineering aspects. In this context, this paper provides an overview of the current technological challenges in on-board and networked automotive systems. This paper encompasses both the state-of-the-art design strategies and the upcoming hardware/software solutions for the next generation of automotive systems, with a special focus on embedded and networked technologies. In particular, this paper surveys current solutions and future trends on models and languages for automotive software development, on-board computational platforms, in-car network architectures and communication protocols, and novel design strategies for cybersecurity and functional safety.

SECTION I.

Introduction

The size of embedded systems market is growing at a drastic pace. According to an estimate, it will be 258.72 billion USD by 2023 [1]. It is further estimated that automotive applications comprise more than 20% of this market. An embedded system consists of a hardware (HW), a processor, and peripherals, and the software (SW) that runs on the embedded processor [2]. In a modern car, the size of the embedded SW is in the order of millions of code lines. Many automotive embedded systems are real-time (RT) constrained, i.e., they must provide logically correct responses at correct times that are dictated by time-critical functionalities. Particularly, hard RT requirements apply to autonomous driving (AD) or autonomous machinery operation, according to a preplanned path/statement of work. Such functionalities are demanding in terms of both computational and environmental conditions. Challenging environmental requirements [3] have to be faced, such as temperature from −40 to 125 °C, mechanical and chemical stress and moisture resistance over 15-year lifetime, and electrostatic discharge protection of several kVs.

The current electric/electronic (E/E) on-board automotive architectures, with up to 100 dedicated electronic control units (ECUs), are no longer capable of answering the computing, communication, and memory requirements coming from innovations with increased needs of fail operational, functional safety (FuSa), cybersecurity, and RT behavior [4]–[7]. Such innovations include transition from internal combustion engines to full electric cars; introduction of advanced driver assistance systems (ADAS) and AD functions; increased level of on-board connectivity, mainly wired (e.g. FlexRay [8], CAN/CAN-FD [9], [10], and automotive Ethernet [11]); vehicle to everything (V2X) wireless connectivity for advanced services, such as fleet management, platooning, over-the-air SW updates; and stringent constraints in terms of FuSa and cyber security.

Rather than just increasing the number of basic ECUs, using 32-b microcontrollers (MCUs), new on-board E/E-architectures will exploit embedded high-performance computing (eHPC) platforms. Computational power in the order of tera operations per second (TOPS), see Fig. 1, is needed to implement RT perception and AD tasks in modern vehicles, particularly for high automation (Level 4, L4), in which the system can perform the driving task without human intervention, and full automation (Level 5, L5), in which the system takes over all the aspects of driving full time. As shown in Fig. 2, the automotive eHPC should sustain in RT with the following functions.

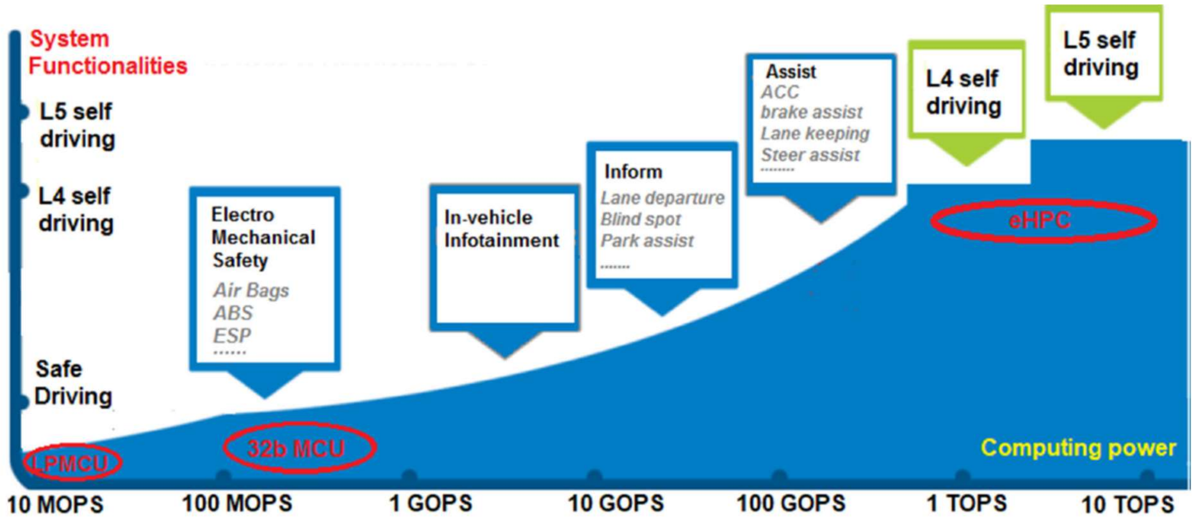


Fig. 1.

Computation needs versus AD/ADAS functions [4].

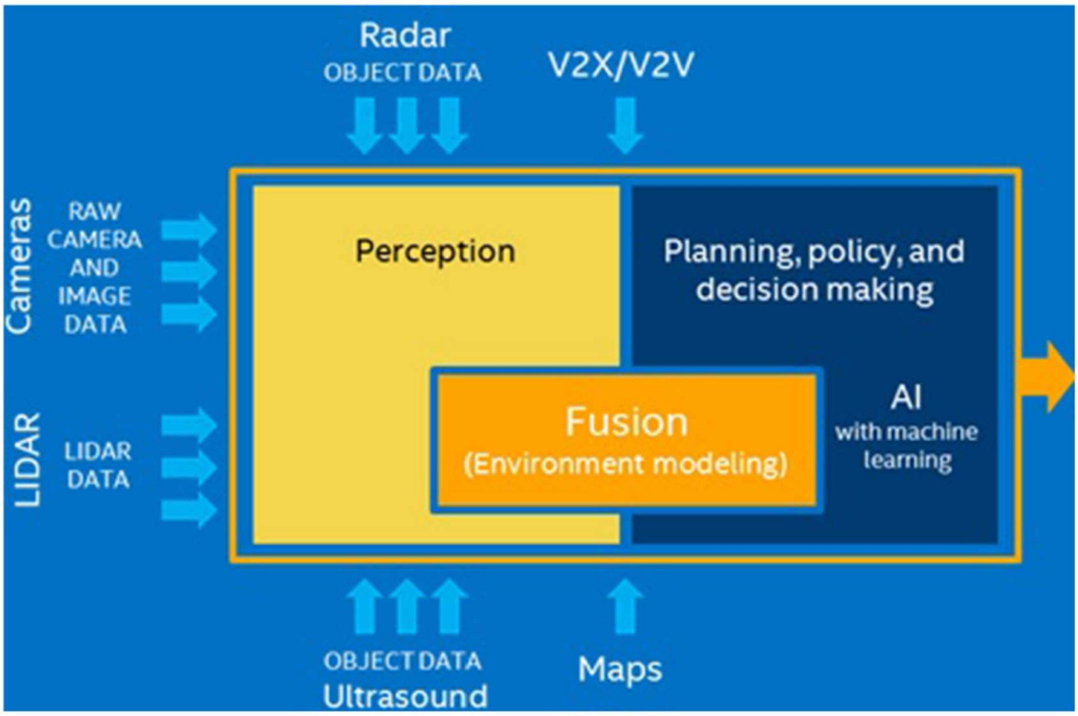


Fig. 2.

Autonomous platform at functional level.

1. *Observation:* building a model of the surrounding environment, where inputs are the direct observations produced by sensors [12] (cameras, radars, sonars, and lidars) or V2X wireless data, and the output is a geometrical and topological representation of the environment.

2. *Perception:* localization of the car, i.e., estimating its path, position, and orientation within a map, by fusion of global (satellite communication) and relative (gyro and accelerometer inertial sensors) data, detection of all static (landmarks, road, and traffic signs) and moving (vehicles, pedestrian) obstacles, and classification depending on how well they match up with a library of predetermined shape and motion descriptors.

3. *Planning and decision:* moving the car, which requires route planning and trajectory control, used to direct the car to its destination, while avoiding obstacles and following traffic rules.

All the above-mentioned phases will benefit from artificial intelligence (AI) techniques, which are widely addressed in the recent literature [13]–[21]. Online map data are required to provide long-range planning information, such as lane end, speed limits, construction sites, and other changing road conditions. All these operations have to be repeated in a time scale below 10 ms with stringent low-latency requirements. Perception results from fusion of all surround sensing and online map data into a single surround model. For data fusion, a grid-based approach may be used to determine the occupancy probability (Bayesian approach) of a cell, or the belief function (Dempster–Shafer approach), by evaluating the current sensor reading and the history from past cycle [22]. Grid occupancy is calculated from processed sensor data, with explicit modeling of uncertainties. Grid cells can bear additional information, such as moving object speed, which can be used to predict likely behavior.

In this new scenario, RT computational capabilities in the range of TOPS are required, as shown in Fig. 1. This also involves the connectivity through high-bandwidth time-sensitive networks of both general-purpose eHPC and number-crunching accelerators. In addition, high data storage capability in nonvolatile robust memories is required. As foreseen by Intel [4], from an average of 1.5 GB of traffic data per Internet user today, we will move toward 4000 GB of data generated per day by an AD car including technical data, personal data, crowd-sourced data, and societal data.

A. Paper Contribution

The aim of this paper is to provide an overview of the current technological challenges in on-board and networked automotive systems, reviewing the state-of-the-art design strategies and also pointing to the upcoming solutions. Unlike other surveys that focus on one specific challenge, e.g., in-vehicle communications [23]–[28], this paper aims to provide a full picture of cutting-edge topics in the addressed context. For this reason, this paper uniformly addresses core topics for on-board and networked automotive systems, which are as follows.

1. Models, languages, standards, and methodologies for automotive SW development.

2. High-performance on-board computation platforms.

3. On-board network architectures, protocols, and standards.

4. Design strategies for on-board cybersecurity.

5. Functional safety.

B. Paper Outline

The rest of this paper is organized as follows. Section II addresses the recent advancements in models, languages, architectures, and standards for automotive SW development. Section III discusses the automotive eHPC platforms. New trends and solutions for RT in-vehicle communications are presented in Section IV. Cybersecurity issues related to in-vehicle networking and possible countermeasures are analyzed in Section V, whereas design strategies for on-board functional safety are dealt with in Section VI. Finally, conclusions are drawn in Section VII.

SECTION II.

Automotive SW Development

Automotive industry has undergone a drastic shift from mechanic-intensive applications to SW-intensive applications in the last couple of decades [29]. According to Broy *et al.* [30], more than 80% of innovation in cars come from computer-controlled functionalities. The increasing demand for such functionalities and data-intensive applications in modern cars has led to the increasing size and complexity of automotive SW. According to an estimate in 2014, the amount of SW in a regular four-door car increased ten times in eight years reaching the size of approximately 1 GB [31]. Another estimate in 2009 predicted that a modern premium car shall contain nearly 100 million lines of code (MLoC) and was expected to reach 200–300 MLoC in the coming years. This estimate seems accurate, as Ford showcased their car containing 150 MLoC at the consumer electronics show (CES) in 2016 [32].

Model-based engineering (MBE) [33] and component-based SW engineering (CBSE) [34] have emerged as an attractive and cost-effective option to deal with the size and complexity of the SW. MBE uses models to describe functions, structures, and design artifacts throughout the SW development. CBSE allows to build large SW systems by reusing pre-existing SW components and their architectures. It is estimated that up to 90% of automotive SWs can be reused from previous releases or other projects if MBE and CBSE are used [35]. There exist several modeling languages and component models in the automotive domain that employ the principles of MBE and CBSE for the SW development [36], [37].

EAST-ADL [38] is an Architecture Description Language for automotive embedded systems. It has developed and evolved based on several European projects and research works, such as provided in [39]–[42]. The EAST-ADL methodology allows to model the SW architecture at four abstraction levels. These levels, together with some of the models, languages, and tools that are used for the SW development at each level, are depicted in Fig. 3. At the top level, called vehicle level, end-to-end requirements on the automotive functionality are captured. At the analysis level, the requirements are refined and expressed formally. Moreover, several different analyses can be performed including the requirements consistency analysis and the functions analysis. The design level defines the SW architecture, HW architecture, and SW to HW allocation model by abstracting implementation details. The concrete implementation of the SW architecture is performed at the implementation level. The language supports the modeling of an automotive SW architecture only at the top three levels in Fig. 3. The methodology recommends to employ standard or proprietary architectures and component models at the implementation level, e.g., AUTOSAR [43] and Rubus component model (RCM) [44], [45]. EAST-ADL is also aligned with the FuSa standard for road vehicles, ISO26262 [46], [47]. There are several variants and specific implementations of the language that are currently used in the automotive industry, e.g., Systemweaver and its variants SE tool, Rubus-EAST, and Fraunhofer ESK [48], that are also shown in Fig. 3. A detailed comparison of these tools and models is discussed in [49] and [50].

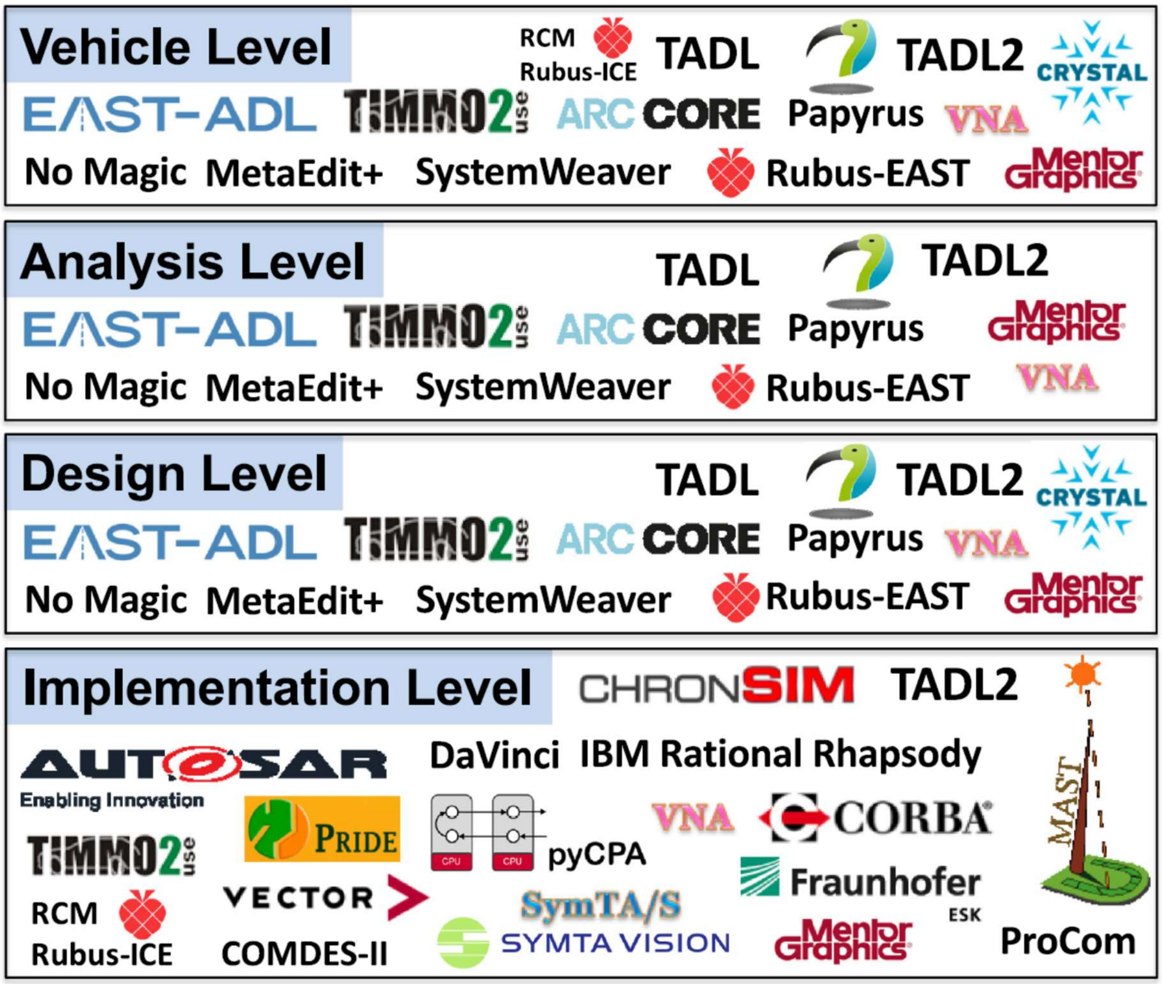


Fig. 3.

Abstraction levels considered during the automotive SW development.

There are several middleware approaches and component models that are used for the development of automotive SW at the implementation level. For example, CORBA [51] and iLAND [52]. COMDES [53] and ProCOM [54] represent examples of component models from academia, whereas RCM and AUTOSAR are the examples of industrial component models. AUTOSAR is a worldwide development partnership of vehicle manufacturers, suppliers, and companies from the electronics and information and communications technology (ICT) industry [43]. AUTOSAR-based SW is widely used by all original equipment manufacturers (OEMs) in Europe and is gaining momentum in the USA, Japan, and Korea. The initial version of AUTOSAR did not account for modeling timing information, which is of utmost importance in vehicular safety-critical systems. The support for timing modeling in AUTOSAR is provided by the TADL [55] and TADL2 [56] languages. These languages were developed within two large EU initiatives, i.e., TIMMO and TIMMO2USE [57]. The AUTOSAR standard comprises a way to define the in-car network infrastructure and communication matrix, the necessary exchange formats as well as an operating system infrastructure for embedded ECUs (Classic Platform) and performance ECUs (Adaptive Platform). To support automotive requirements, the SW environment and development kits must provide these functionalities, see Fig. 4.

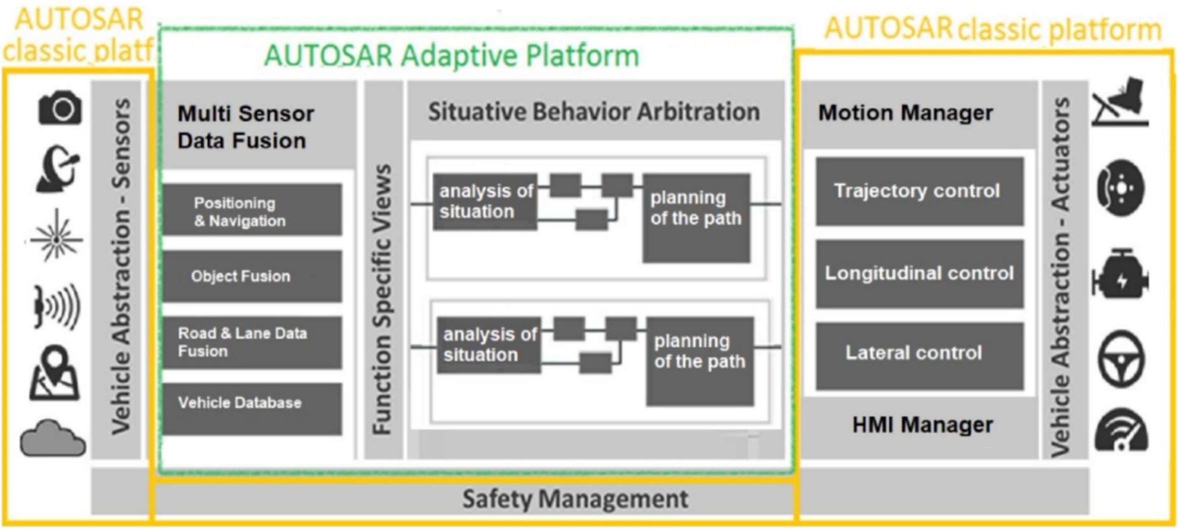


Fig. 4.

AD architecture with the AUTOSAR platform.

Implementations of the AUTOSAR Classic Platform can be subject to safety certification according to the automotive safety integrity level (ASIL) A-D in the ISO26262 FuSa standard. Moreover, the requirements on the response times of runtime entities (tasks) in these implementations are often in the lower us range. The recent AUTOSAR Adaptive Platform defines a service-oriented middleware as well as system health monitoring for automotive performance ECUs, which can run on POSIX PSE51-compatible operating systems (e.g., Linux, QNX, and Integrity OS). A new standard interface is defined to access HW accelerator units, which is planned to be based on the widely accepted OpenCL standard. The Adaptive Platform is expected to become the automotive standard for performance and number-crunching ECUs. This is because the service-oriented network protocols are the same in both the Classic and the Adaptive Platforms. The interoperability between the two platforms is also supported.

A common requirement for performance ECUs is the strict separation of specific SW domains. The introduction of hypervisor supports safety and security, so that in a mixed-criticality environment, SW functions with different ASIL can be easily separated. Moreover, a hypervisor can separate small monitoring apps as well as complete specialized operating systems and driver stacks in virtual machines. This can enhance security and secure communication to back-end systems and Internet. To be compatible with a large range of existing SW packages, especially from the HPC domain, a Linux-based operating system environment will be chosen as basis for the performance ECU SW. Since the AUTOSAR Adaptive Platform is available on Linux, this opens the door to the world of Linux-based infrastructure SW.

To develop SW for the eHCP platform, an SW development kit together with well-defined exchange formats has to be provided. AUTOSAR ARXML, as the industry-standard format for exchanging information about ECUs, ECU communication, integrated self-describing SW services, and SW-components, makes the system complexity manageable. In the future, the term “system” in automotive will be redefined from a single ECU up to complete cars, and even extended to fleets.

Prototyping environments for AD development extend the automotive eHPC SW environment into a production environment for AD. Examples of such prototyping environments include robot operating system 2.0 from Open Source Robotics Foundation and EB robinos, which is an SW framework and architecture for highly automated driving based on open interfaces implementing the open robinos specifications. Optimized application libraries should be provided for use by the perception tasks, sensor fusion, and situative behavior analysis. Lidar sensor processing, which involves representation of data and segmentation into objects, requires efficient implementations of the Point Cloud Library and Fast Library for Approximate Nearest Neighbors. Camera processing implies a variety of computer visions that are prototyped with the OpenCV library and moved to the OpenVX programming environment to meet the performance requirements. Sensor fusion and other high-performance functions of computer vision are implemented in OpenCL when CUDA is not available. Dense linear algebra libraries, such as BLAS/BLIS and Eigen (C++ templated library), must be available and optimized, as they are required by machine learning algorithms and standard deep learning frameworks (e.g., Caffe and TensorFlow).

SECTION III.

High-Performance Computation Platforms

Today's embedded automotive-qualified processors, with capabilities of hundreds of MOPS, see Fig. 1, cannot handle AD functions. There is a need for more powerful HW platforms, such as eHPC data fusion platform, see Fig. 5, which are designed by combining an automotive-certified RT MCU with general purpose HPC processors and accelerators. The latter are used to increase the power efficiency and to act as safe number crunchers with direct access (not shown in Fig. 5) to sensor data through Ethernet or low-voltage differential signaling (LVDS). A multi-Gb/s time sensitive networking (TSN) link should be used to connect the safe MCU supervisor, the accelerators, and the general-purpose HPC processors. This type of interaction will require reliable and secure communication channels, proper identity management, and assurance while providing adequate data and identity privacy. Next-generation AD systems require that the whole perception process be qualified at ASIL-D level according to the ISO 26262 automotive FuSa standard [47]. This can be achieved by performing redundant computations with possibly dissimilar implementation techniques on the “safe number crunchers.” These safe number crunchers are qualified at ASIL-B by implementing a range of safety mechanisms, such as error-detection and error-correction codes (EDC/ECC) in memory, parity in caches, and cyclic redundancy check (CRC) in network-on-chip (NoC). The redundant results are then compared by the safe MCU qualified for ASIL-D, which monitors the computations and decides whether the results can be trusted. The eHPC platform will be connected to the car backbone with a runtime environment compliant with AUTOSAR. For the safe MCU, already available 32-b cores, such as Infineon Aurix or ST SPC5/Freescale MPC56, can be adopted. The ST SPC5 and Freescale MPC56 families are built in 40-nm technology on 32-b PowerPC instruction set, with up to 4 cores (with dual lock-step approach), single instruction multiple data floating point unit, 8 MB of embedded flash, multichannel 12-b analog to digital converter, interfacing data rate up to 10 Mb/s with FlexRay, I2C, LIN [58], CAN, and SPI [59]. Similarly, the Infineon Aurix ranges from a 300-MHz triple-core device with 720 millions of instructions per second (MIPS) and 8 MB of embedded flash down to an 80-MHz single core with 130 MIPS and 0.5-MB embedded flash.

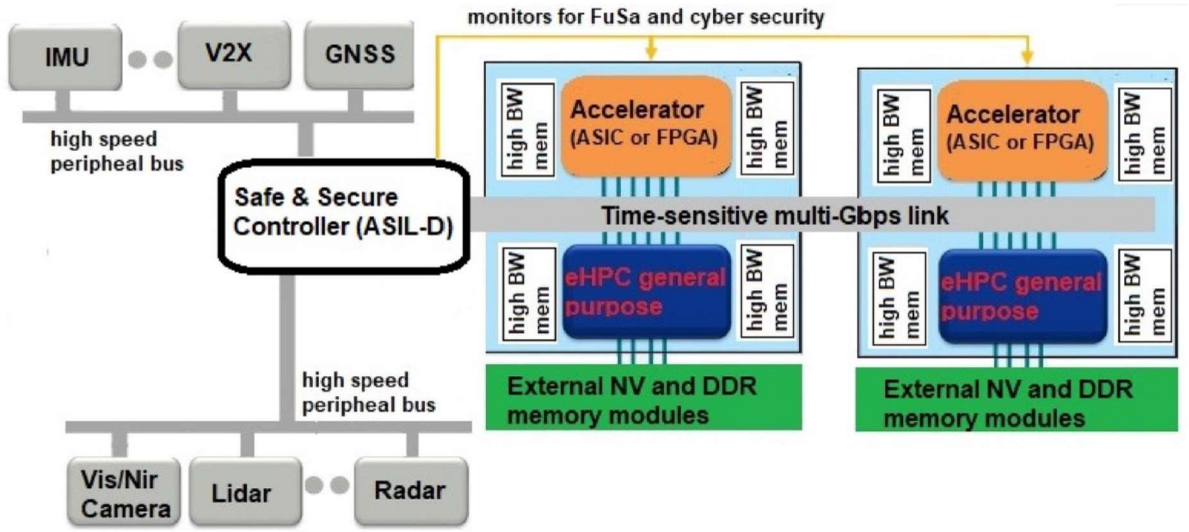


Fig. 5.

Automotive eHPC platform.

Instead, for the HPC units in Fig. 5, massively parallel platforms are appearing in the car market. Mass production of the Renesas R-Car H3 in 16-nm technology is expected in 2018 [60]. R-Car H3 includes 9 ARM Cortex cores (8 64-b A57/A53 engines with L1/L2 cache plus a 32-b R7 with L1 cache), offering 40k MIPS plus a PowerVR GX6650 graphics engine with 192 arithmetic logic unit (ALU) cores for three-dimensional graphics (more than 100 GLOPS and 4K video display/streaming) and dedicated video coprocessors (H.26x/MPEGx codec, distortion compensator, IMP-X5 image recognition). The R-Car H3 is ASIL-B and has a rich set of high-rate interfaces, such as Ethernet, USB, DVD/blue-ray SATA, SD card, and audio/video I/O, besides I2C and CAN. The power consumption amounts to tens of watts. New actors are entering this application domain, such as Nvidia and Intel, to bring on-board TOPS capability and AI technologies with multichip automotive supercomputers. They have signed core partnerships with mass market car makers, Nvidia with Audi and Intel with BMW, to have on the roads AI cars by 2020. To this aim, a new European Processor Initiative for eHPC in AD has recently been started [61]. NVIDIA has recently presented the Xavier AI car's computer, which features 30 TOPS capability for a power consumption of 30 W, thanks to eight ARM 64-b cores plus a 512-core Volta graphics processing unit (GPU), and a video processing unit supporting 8K video decode and encode and high dynamic range, as well as a computer vision accelerator. The Xavier AI is fabricated in 16-nm TSMC fin field-effect transistor (FinFET) technology with an estimated complexity of 7 billion transistors. The power consumption of such HPC platforms will be in the range from tens to hundreds of watts, e.g., from 30 W of Xavier chip to 500 W of the 320 TOPS Drive PX Pegasus board announced at GTC Europe 2017 [62]. Due to the high environmental temperature of under-the-hood car electronics, passive cooling systems are not enough. Hence, the design of low-cost/low-size active cooling systems for HPC ECUs is a new emerging challenge.

Also Intel is developing several platforms for automated driving: a first multichip platform, so called Intel Go, has been made available using an Aurix ASIL-D 32-b MCU enhanced for computation capability by an ATOM C3000 core in 14-nm technology, and by Arria 10 field programmable gate array (FPGA) [3]. The FPGA accelerator includes an embedded dual-core 1.5-GHz ARM A9 core, more than 1 million logic elements (a 64-b six-LUT with four FFs at the output), and 1.7 million user flip-flops, and 64 MB of embedded memory. The Arria 10 family includes hardened single-precision IEEE 754 floating point units, with an aggregate throughput of 1.3 TOPS. This platform supports driving automation L3, in which the system performs the driving task, but a human driver will intervene when requested. The next evolution of the platform, suitable for all AD levels, will combine one or more EyeQ5 [63] accelerators (by Mobileye, an Intel company) and one or more ATOM-based general-purpose processors (e.g., Denverton). The EyeQ5 will enable processing of more than 16 multimegapixel cameras and other sensors. Its computational power targets 15 TOPS while drawing only 5–6 W in a typical application. It implements high performance NoC interconnect and multichannel low-power DDR interfaces, to support high-computational and data bandwidth requirements. Another Intel platform is announced that will use powerful Xeon processors and 2 multichip boards connected with a 16-port 10-GB Ethernet to sustain L4 and L5 AD levels, mainly targeting fleets.

According to the scheme in Fig. 5, high-performance functions that need time predictability, such as perception functions in automated cars, need to be implemented on high-performance accelerators that also provide response time guarantees. Time-predictability capabilities start with the core, then the local memory hierarchy, then the global interconnect, and finally external memory and I/O interfaces. In the Intel Go proposal, the accelerator role is managed through an Arria10 FPGA or EYEQ5 chip. As an alternative, the architecture extensions of the reduced instruction set computer (RISC-V) accelerator cores have already proven to be suitable for scalable computing capabilities with high power efficiency [64], including also machine-learning tasks [65]. RISC-V is developed according to an open HW-SW model, thus easing interoperability of eHPC solutions.

The accelerator should ensure timing compositional property, which means that any global worst case execution time (WCET) is composed of local WCETs. This also implies that the WCET in a core experiencing resources conflicts, e.g., accesses to the memory hierarchy, is safely approximated by adding the resource interference times to the WCET on the core executing without interferences [66]. The timing compositional property requires in-order instruction pipeline and is compatible with caches, provided they have an LRU replacement policy. The basis for the RT accelerator architecture in Fig. 5 can be a very-long instruction word (VLIW) extension of the RISC-V ISA. VLIW execution, opposed to superscalar execution, is a core implementation technique that enables multiple instruction issues while being compatible with the timing compositional property. This VLIW extension approach ensures that any standard RISC-V binary will execute correctly, but in single-issue mode on such a VLIW core. A simple recompilation will enable to achieve multiple instruction issues on this core. In both the accelerator and the general-purpose HPC architecture in Fig. 5, an NoC interconnect is responsible for arbitrating access to shared resources, such as an I/O or a memory. One main issue when using multicore or many-core architectures for designing safety critical systems is to master the impact of contentions that can arise due to parallel requests for a shared resource, on the estimation of the WCET of tasks. Current approaches either rely on an HW approach, for instance time-division multiple access (TDMA), to ensure no contention can arise at runtime, or on SW approaches through the use of specific execution models, such as predictable execution model that explicitly separates data accesses from computation. The need to integrate functionalities with different level of criticalities on such multicore or many-core architectures has led to the design of mixed-criticality systems. Extension of these approaches to such mixed-criticality systems is currently based on a technique that drops noncritical tasks whenever a given threshold contention level has been reached. However, more flexible strategies are required at the interconnect level to maximize the utilization level of such multicore or many-core platforms. At the multicore level, the introduction of an HW contention manager to monitor the slack activity at the interconnect level will improve the system capability to allocate resources to noncritical tasks and adapt the scheduling of requests to shared resources, such that critical tasks still meet their deadline while the number of requests from noncritical tasks is maximized. At the many-core level, an NoC is used to interconnect cores or tiles. NoC is often designed for a given type of application and specific characteristics when targeting RT systems should be developed. Experimentally checking the behavior of an NoC in case of contention between flows is still an open topic. Designing a way to execute routers of an NoC in which a stream would systematically compete with other flows would facilitate the observation of contentions within an NoC. The HW mechanisms for regulating streams in contention could then be enriched to interface with the system SW, in order to dynamically adapt the control performed versus the target latency.

The eHPC platform in Fig. 5 should be equipped with HW resources to sustain V2X connectivity needed for RT HD map download and infotainment, over-the-air diagnostic and SW update, sensor-data upload from the vehicle for machine learning. To this aim, two solutions can be adopted [5], [6]: IEEE 802.11p or Cellular-V2X. IEEE 802.11p uses 10-MHz channels within the (5.85–5.925 GHz) band to achieve data rates of several Mb/s for V2X. IEEE 802.11p transceivers are already available on the market (e.g., STM-Autotalks chipset) and, as discussed in [5], they can be implemented at low cost in mature and already automotive-qualified CMOS technologies. With 33 dBm of effective isotropic radiated power, a single-hop connectivity of 1 km can be achieved. Cellular-V2X connectivity can be achieved with multilayer multiple input multiple output (MIMO) transmission according to emerging 5G transceivers. Operating both in sub-6 GHz and 28-GHz millimeter wave (mmW) bands, data rates of up to several Gb/s can be achieved [67]. However, high-end technology nodes are required to sustain mmW and massive MIMO 5G operations and the way to achieve low-latency guaranteed performance is still an open issue. A first 5G modem has been announced by Intel at CES 2017 [68], although its automotive qualification is still on-going and the 5G standardization is still not settled.

SECTION IV.

In-Vehicle Network Architectures

Vehicles are becoming increasingly smart, connected, and part of the Internet. While new functionalities, such as natural speech recognition and cloud-based services, are developed, in-vehicle legacy systems have to be maintained and integrated with the new developments for the sake of cost effectiveness. As a consequence, traditional signal-based communication, mainly consisting of cyclic message broadcasts, such as in LIN, CAN/CAN-FD [69], [70], and FlexRay, has to coexist with service-based communication, made up of event-based message unicasts, such as the ones typical of IP-based networks (e.g., Ethernet and Wi-Fi) [71]. Scalable service-oriented middleware over IP allows the introduction of service-oriented transmission of information, in which a sender only transmits data when at least one receiver in the network needs this data, thus avoiding to load the network and all connected nodes with unnecessary traffic.

Dynamic distribution of functions, virtualization of ECUs, and the network controlled by virtual machine and network hypervisor are in the roadmap of future automotive network architectures, which today are migrating from the current central-gateway structure to a domain-based architecture. The vehicle E/E-architecture of tomorrow will be therefore characterized by an automotive Ethernet backbone connecting different domains, isolated and protected by domain controllers [72]. The central Ethernet switch will be also connected to a smart antenna, being LTE/5G, WiFi/BLE, and V2X/DSRC, the most likely technologies. Following the development of the IEEE standards within the TSN Working Group, the vision for the automotive Ethernet backbone connecting different domains, each with its TSN control unit, provides for master MCUs integrating Ethernet PHYs and TSN switch functionalities with security modules and various protocol converters for local legacy serial networks.

Ethernet switches implement separate collision domains and offer several features that can be used to increase security: VLANs, unicast filtering, multicast filtering, and access control lists. However, many state-of-the-art attacks from the information technology world can be applied to in-vehicle Ethernet, so special care must be taken and multiple levels of defense should be in place. For instance, performing deep packet inspection in the switches represents an efficient solution to avoid forwarding malicious packets to the host controller that would be therefore entrusted with security checks only on a second stage of inspection that would be required for specific frames only, e.g., those coming from the external of the vehicle.

The automotive Ethernet backbone will likely be a TSN-enabled implementation of 802.3 Ethernet. The recent standards 100BASE-T1 [73] and Gigabit PHY (IEEE 802.3bp-2016) [74] already allow the use of a light unshielded twisted pair of copper wires for automotive usage. Also, a broad spectrum of bit rates are envisaged for automotive Ethernet nowadays, also including 10 Mb/s and 2.5, 5, and 10 Gb/s (for the backbone and for raw sensor data transmission). For Multi-Gig Automotive Ethernet PHY, various options from shielded cables to coax to optical fiber (for 10 Gb/s) are under consideration.

The recent Layer 2 TSN standards are expected to be dominating the scene for AD. In fact, although current ADAS systems already require processing of high-resolution data originating from video cameras, radars, and lidars, self-driving cars require a significantly higher number of sensors, more network connections and better networking solutions for video links than the current technologies based on point-to-point connections, which will not be able to support the packet-based data transport needs of self-driving cars.

A. Automotive Ethernet From AVB to TSN

The IEEE 802.1 audio video bridging (AVB) is a set of technical standards that provides the specifications for time-synchronized low-latency streaming services through IEEE 802.1Q [75] networks. The AVB documents include: the IEEE 802.1AS-2011 [76]—timing and synchronization for time-sensitive applications in bridged local area networks (whose revision is in progress as IEEE P802.1AS-Rev project), the IEEE 802.1Qav-2009, forwarding and queueing enhancements for time-sensitive streams, which specifies the credit-based shaper (CBS); and the IEEE 802.1Qat-2010 and Stream Reservation Protocol (SRP). The last two amendments have been rolled into the IEEE 802.1Q-2014 standard [75]. Finally, the IEEE Std 802.1BA-2009 specifies a set of usage-specific profiles to help interoperability between networked devices using the AVB specifications. AVB introduces a number of new and important concepts to IEEE 802.1 networks to provide quality of service. The first is the support for priority, to distinguish between time-sensitive flows and ordinary traffic and handle them differently. The second is bandwidth reservation, to set aside a certain amount of guaranteed bandwidth across a portion of the network for handling the high-priority traffic. Last but not least, AVB provides a set of protocols to manage the network time for supporting synchronized operations (i.e., A/V playback). For seven hops within the network, AVB guarantees a fixed upper bound for latency. In particular, two stream reservation (SR) classes are defined, i.e., Class A, which provides a maximum latency of 2 ms, and Class B, which provides a maximum latency of 50 ms. With AVB, the IEEE has moved Ethernet into the RT applications domain. AVB is expected to replace (or is already gradually replacing) the Media Oriented Systems Transport (MOST) protocol [77] in the multimedia/infotainment domain, and the LVDS cables in camera-based ADAS. This paper [78] deals with the CBS of AVB and the use of priorities as defined in IEEE 802.1Q in automotive cases studies. The AVB suitability for automotive usage is addressed in [79] and [80]. In particular, Alderisi *et al.* [79] provide a comparative performance evaluation of AVB and TTEthernet, a well-known technology standardized by Society of Automotive Engineers as AS6802 [81], for ADAS, multimedia and infotainment traffic. The comparison was obtained through OMNeT++ simulations based on realistic traffic patterns on star-based networks under a high and varying workload. Results show that both AVB and TTEthernet meet the requirements of ADAS and multimedia flows. The two technologies complement each other, as TTEthernet allows for completely deterministic transmission and offline verification of time-triggered messages for safety-critical applications while AVB allows for online SR, thus fitting entertainment applications with varying bandwidth demand. The problem of routing AVB streams to minimize their worst-case end-to-end delay is addressed in [82], which proposes an effective solution, based on a search-space reduction technique and a greedy randomized adaptive search procedure based heuristic.

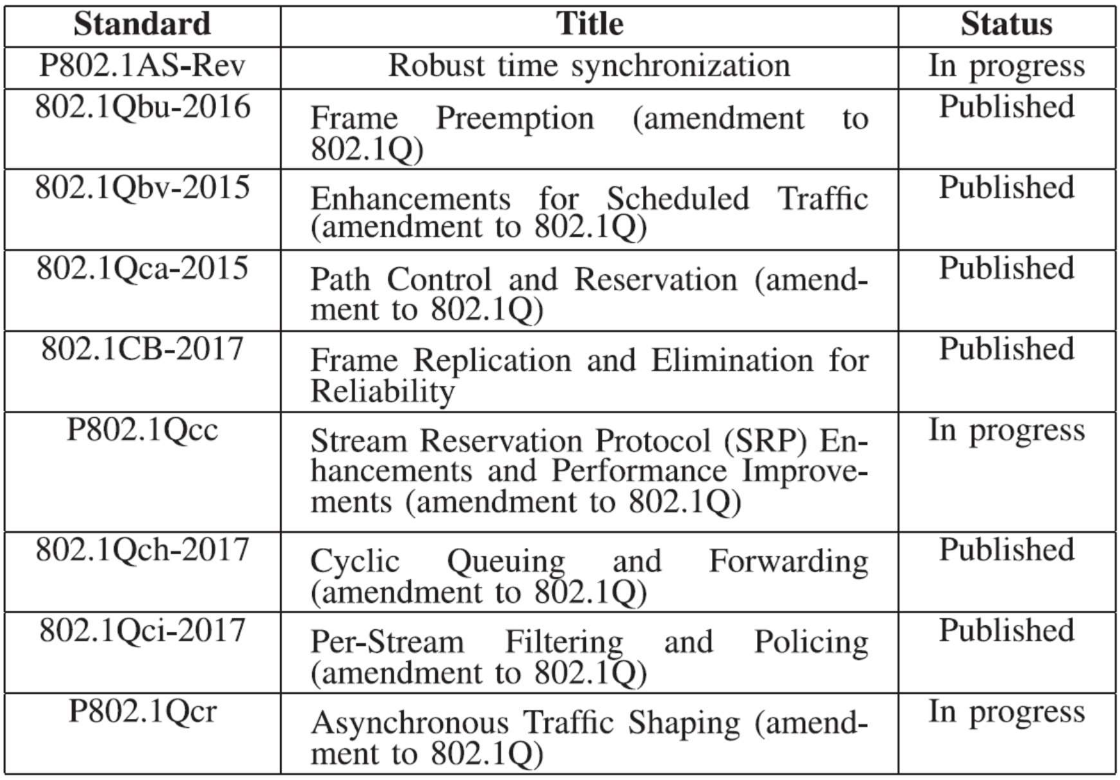
Despite its advantages, AVB does not provide support for scheduled traffic (ST), i.e., high-priority small-size time-sensitive traffic (e.g., control traffic) that has to be transmitted according to a time schedule without interference from other traffic. In fact, as AVB provides only two RT traffic classes, a mutual interference problem raises if multiple time-critical traffic flows in the same network are mapped on the same SR class, with nonnegligible effects on delay. In particular, if ST is handled in the same queue as large video frames mapped on the same SR class (e.g., Class A), it will experience very variable latency and high jitter. Moreover, SR class frames undergo the CBS algorithm, and shaping blocks frame transmission for a given class if the credit of the class is below zero. For this reason, a more effective way of handling ST in AVB networks was proposed in [83] and [84]. The new approach, called AVB_ST, adds a new, separate traffic class on top of the AVB SR Classes A and B, which is called ST class. ST frames are tagged with the highest priority TAG according to the IEEE 802.1Q standard, whereas SR Classes A and B take the second highest priority and the third highest priority, respectively. ST traffic is handled in a separate queue and does not undergo credit-based shaping, thus avoiding the undesirable effects of shaping on the flow latency. SR Classes A and B are handled by CBS, whereas best effort traffic by strict priority. Comparative performance assessments between standard AVB and AVB_ST in a realistic automotive scenario in [84] showed that the AVB_ST is able to support ST, offering low and predictable latency values without significantly affecting SR traffic. This is due to: the introduction of a separate class for ST combined with offset-based scheduling for ST flows; the temporal isolation provided by the time-aware shaper mechanism; and strict priority scheduling, that offers low and bounded latencies to ST even under a high SR traffic load. A response time analysis for multihop AVB ST networks that is also applicable to multihop AVB networks is presented in [85]. The analysis uses a bandwidth overreservation concept and overcomes the limitations of previous analysis approaches for AVB networks [86], [87], which, in most of the cases, do not lead to a schedulable result due to the tight bandwidth allocation imposed by the AVB standard. AVB_ST is similar to the IEEE 802.1 Qbv-2015 standard [88], with differences in the way that the ST window is sized and for the rate at which the increase of one of the CBS parameters (i.e., the Idleslope) for the SR classes is determined.

B. Time-Sensitive Networking

The TSN standard family provides precise time synchronization, deterministic communications, ultralow latency, zero congestion loss, reliability, and fault tolerance. These properties are foundational for the next generation of AD vehicles. TSN offers other notable advantages. One is the ability to support both RT and best effort traffic over the same network in a flexible way. Changes in the time-critical flows can be accommodated without the need for offline reconfigurations and the best effort traffic can use any bandwidth left over by TSN flows. In addition, TSN offers fast startup, thanks to preconfigured values for timing and bandwidth reservation, and faster firmware updates time than other protocols (e.g., CAN), thanks to the higher data rate. Table I summarizes the TSN standards published so far and some of the ongoing projects that are relevant to automotive applications. A brief description of each of them is provided in the following.

TABLE I TSN Standard Overview

Timing and synchronization for time-sensitive applications



Standard	Title	Status
P802.1AS-Rev	Robust time synchronization	In progress
802.1Qbu-2016	Frame Preemption (amendment to 802.1Q)	Published
802.1Qbv-2015	Enhancements for Scheduled Traffic (amendment to 802.1Q)	Published
802.1Qca-2015	Path Control and Reservation (amendment to 802.1Q)	Published
802.1CB-2017	Frame Replication and Elimination for Reliability	Published
P802.1Qcc	Stream Reservation Protocol (SRP) Enhancements and Performance Improvements (amendment to 802.1Q)	In progress
802.1Qch-2017	Cyclic Queuing and Forwarding (amendment to 802.1Q)	Published
802.1Qci-2017	Per-Stream Filtering and Policing (amendment to 802.1Q)	Published
P802.1Qcr	Asynchronous Traffic Shaping (amendment to 802.1Q)	In progress

The IEEE 802.1AS-Rev [89] improves redundancy, allowing for configuring multiple grandmaster clocks and multiple synchronization spanning trees.

Frame preemption

The IEEE 802.1Qbu-2016 [90] amendment enables a bridge port to suspend the ongoing transmission of a *preemptable* frame to allow one or more *express* (time critical) frames to be transmitted before transmission of the preemptable frame is resumed. This standard works in combination with the IEEE 802.3br-2016 amendment, which allows critical data packets to break up into smaller fragments the noncritical packets in transit over a single physical link.

ST

The IEEE 802.1Qbv-2015 [88] amendment defines policies that enable a bridge or an end station to schedule transmission from each queue based on a known timescale, thanks to a *transmission gate* associated with each queue on a port. When the transmission gate is open, the queued frames are selected for transmission, whereas when the gate is closed, the queued frames are blocked. An ordered list of gate operations (gate control list) is associated with each port and is cyclically repeated. Building the gate control list is a scheduling problem. As the Qbv standard is quite novel, there is still not much work on this specific topic. The work in [91] and [92] proposes a formal description of scheduling constraints for building the gate control list and the adoption of satisfiability modulo theories solvers for the synthesis of communication schedules for Qbv.

Path control and reservation

The IEEE 802.1Qca-2015 amendment [93] provides for explicit path control, bandwidth reservation, and data flow redundancy (protection and restoration).

Frame replication and elimination for reliability

The IEEE 802.1CB-2017 [94] standard provides for identification and replication of frames, redundant transmission, identification, and elimination of duplicate frames.

SRP enhancements and performance improvements

The IEEE P802.1Qcc project [95] provides support for more SR streams, configurable SR classes and streams, Layer 3 streaming, deterministic SR convergence, and a user network interface for routing and reservations.

Cyclic queuing and forwarding

The IEEE 802.1Qch-2017 [96] amendment specifies a transmission selection algorithm that allows deterministic delays through a bridged network to be easily computed regardless of network topology, thus allowing for much simpler determination of network delays and reduced delivery jitter. Synchronized cyclic enqueuing and queue draining procedures enable bridges and end stations to synchronize their frame transmission to achieve zero congestion loss and deterministic latency.

Per-stream filtering and policing

The IEEE 802.1Qci-2017 [97] amendment specifies procedures for a bridge to perform frame counting, filtering, policing, and service class selection for a frame based on the particular data stream to which the frame belongs. Such policing and filtering functions allow the detection and mitigation of disruptive transmissions by other systems in a network, improving its robustness and security. When unexpected traffic is present, policing prevents the intruder from impairing the network.

Asynchronous traffic shaping

P802.1Qcr [98] specifies asynchronous traffic shaping mechanisms to achieve deterministic latency and zero congestion loss without using network topology information or relying on synchronous communication, thus allowing for higher link utilization. Relevant to this standard are the works given in [99], which introduces the urgency-based scheduler (UBS), and [100], which addresses the UBS synthesis when assigning queues and priority levels to hard RT data flows.

SECTION V.

Cybersecurity Issues and Countermeasures for In-Vehicle Networking

Since this survey addresses on-board embedded and networked automotive systems, this section is focused on cybersecurity issues and countermeasures for in-vehicle networks. Other cybersecurity aspects, such as those related to cars external connectivity, cloud-based traffic, and fleet managements, just to name a few, are out of scope of this paper.

Secure by design, in-vehicle networking should ensure several properties, such as data integrity, confidentiality, authentication, and availability. However, several security vulnerabilities [101]–[108] characterize current in-vehicle networking technologies, using CAN and/or CAN-FD as a backbone, and a plethora of other interconnecting technologies for specific subsystems (e.g., LIN for local interconnection of low data rate nodes, MOST for infotainment with USB and Bluetooth user interfaces, and FlexRay for latency-critical functions).

The net-spanning data exchange via various gateway devices potentially allows access to any vehicular bus from every other existing bus system. In principle, each LIN, CAN, or MOST controller is able to send messages to any other existing car controller [109], [110]. Without particular preventive measures, a single compromised bus system endangers the whole vehicle communication network. Whereas attacks on LIN or multimedia networks may result in the failure of power windows or navigation SW, successful attacks on CAN or FlexRay networks may result in malfunctioning of some important driving assistance functions, which leads to serious impairments of driving safety [111], [112].

While the use of CRC ensures data integrity, the broadcast nature of CAN/CAN-FD or FlexRay is a risk in terms of confidentiality, as an attacked ECU can monitor all data passing on the bus. Moreover, since new ECUs can be added in a plug-and-play way (assigning them a new identifier) without modifying the already installed ECUs, and since the data link layer does not provide any signature mechanism, there is a high risk of authentication vulnerability. Similarly, the multimaster feature with an arbitration based on identifier priority poses risks in terms of availability. For example a hacker can attack a bus and behave as a new ECU, reading all data on the bus and generating false packets. Using a high priority identifier, the malicious ECU can win the arbitration and then continuously send invalid messages thus making a jamming attack. Even though these invalid frames will be discarded by the receiving controllers, the attack makes the bus unavailable to other ECUs connected to the bus. Denial of service attacks may affect the backbone bus or the local bus. In the first case, they will lead to system failure, whereas in the second case, they will lead to functional failure. The malicious ECU, after reading a message from the bus, can also impersonate another ECU for replay attacks, with a potential for harmful consequences for the vehicle occupant.

Due to the lack of signature mechanisms for authenticity and transmission encryption, it is easy for an attacker to emulate a protocol-compliant behavior. As a consequence, controllers are not able to verify whether an incoming message comes from an authorized or unauthorized and/or malicious sender. Controllers just check rules, such as bit stuffing, CRC, and data length code consistency, which may be enough for data integrity, but not for cybersecurity. Moreover, utilizing the CAN mechanisms for automatic fault localization, malicious CAN frames can determine the disconnection of every single controller by posting several well-directed error flags. Similar to the CAN automatic fault localization, the bus guardian in FlexRay can be utilized for the well-directed deactivation of any controller by appropriate faked error messages. Attacks on the common time base, which would make the FlexRay network completely inoperative, are also feasible by posting proper malicious SYNC messages on the bus. Moreover, the introduction of well-directed sleep frames deactivates the corresponding power-saving capable FlexRay controllers.

As possible countermeasures, the following techniques are foreseen and are likely to appear in the new generation of car connectivity devices.

1. To cluster the subnetworks and related subsystems in security islands, separated by gateways with embedded cybersecurity functionalities, so that an attack on a nonsafety related bus, such as LIN or MOST, cannot propagate to the safety-related functions connected to Flexray or CAN [103]. This approach will also be applied to the future architectures based on automotive Ethernet [113].

2. To embed cybersecurity HW accelerators in new automotive computing units to sustain message encryption in RT. This is the reason why in the literature new digital macrocells are appearing, which are implemented in RT security techniques, such as the Advanced Encryption Standard, with different cipher modes, used in symmetric cryptography [114] or more complex algorithms, such as the Elliptic Curve Digital Signature Algorithm, for asymmetric cryptography [115], [116]. The use of HW-based coprocessors is required by stringent latency and energy-efficiency requirements that are not achievable with SW-based implementations.

3. To embed signature mechanisms for controller authentication in new automotive computing units. Authentication of all senders is needed to ensure that only valid controllers are able to communicate on automotive bus systems [103], [115], [117], [118]. All unauthorized messages may then be processed separately or immediately discarded. Every controller therefore needs a certificate to authenticate itself against the gateway as a valid sender. For example, as proposed to [103], a certificate may consist of the controller identifier ID, the public key, and the authorizations of the respective controller. The gateway, in turn, should securely hold a list of public keys of all accredited OEMs for the considered vehicle. Each controller certificate is digitally signed by the OEM with the relevant secret key. The gateway again uses the corresponding public key of the OEM to

verify the validity of the controller certificate. If the authentication process succeeds, the relevant controller is added to the gateways list of valid controllers.

4. To cluster the ECUs in different trustable classes depending on how easily they can be attacked. For example, in [119], a security framework for vehicular systems, called VeCure, is proposed, which can fundamentally solve the message authentication issue of the CAN bus. Each node that sends a CAN packet needs to also send the message authentication code packet (8 B). The ECUs are split into two categories, namely the low-trust and the high-trust groups. ECUs that have external interfaces, e.g., OBD-II or telematics, are put in the low-trust group. The high-trust group ECUs share a secret symmetric key to authenticate each incoming and outgoing message.

5. To implement intrusion detection mechanisms based on the physical or packet layer features, for example, a clock-based intrusion detection system at physical layer is proposed in [105]. Similarly, an in-vehicle network traffic monitoring technique is proposed in [120] to detect the increased transmission rates of manipulated message streams.

6. To implement gateway firewalls, for example, as proposed in [103], if the vehicular controllers are capable of implementing digital signatures, the firewall rules are based on the authorizations given in the certificates of every controller. Therefore, only the authorized controllers are able to send valid messages to the high safety-critical in-vehicle bus systems. If the vehicular controllers do not have the abilities to use digital signatures, the firewall can be established only on the authorizations of each subnet. However, controllers of less restricted networks, such as LIN or MOST, should generally be prevented from sending messages to the high safety-relevant bus systems as CAN or FlexRay. Simplified firewalllike functionalities can be also implemented in each end node and not only in the gateways, with the so-called digital data diode [121]. The idea is to interpose a digital unit between the CAN controller and the CAN transceiver to detect and block unauthorized access. When a frame is detected as malicious, the digital unit corrupts the CRC sequence modifying the CRC-field bits. Therefore, the transmission and reception of a frame that is targeted as malicious generates an error condition that is detected by all the nodes in the CAN network (i.e., each node that has received the corrupted malicious frame transmits an error frame). Furthermore, the digital unit conceals the corruption operation from the sender of the malicious frame. As a result, the sender cannot detect the CRC sequence corruption. Hence, the sender will not attempt to retransmit the malicious frame.

SECTION VI.

Functional and Responsibility Safety

The new world of SW-defined autonomous things brings both technical challenges and liability concerns [122]. Particularly, AD vehicles are composed of electronic platforms with many sensing inputs and many complex processing elements (see Fig. 2), which involves millions of SW lines of code. As a consequence, HW and SW may go wrong and this may cause hazards if no countermeasures are taken. On top of HW and SW failures, cars operate in a very complex environment with many variants, e.g., AD cars share the road with human-driven vehicles. Last but not least, the increase in connectivity through V2X opens possibility for security attacks. Consequently, several potential issues and requirements need to be considered by the automotive manufacturers. One such requirement is FuSA, which is mainly concerned with making the safe from HW failures and SW bugs.

A. FuSa in the Context of the ISO26262 Standard

The first edition of the ISO26262 safety standard consisted of nine normative parts and a guideline as the tenth part. The second edition of the standard, to be published within 2018, will consist of ten normative parts and two guidelines, one (the part 11) is specific to the application of ISO26262 to semiconductor components. The goal of the standard is to provide an automotive safety lifecycle (management, development, production, operation, service, and decommissioning) and support tailoring of the necessary activities during the lifecycle. The standard also covers the functional safety aspects of the entire development process (requirements specification, design, implementation, integration, verification, validation, and configuration). Moreover, the standard provides requirements for validation and confirmation measures to ensure that an acceptable level of safety is achieved. The standard covers both *systematic* and *random* failures. The systematic failure (either in HW or SW) is related in a deterministic way to a certain cause that can only be eliminated by changing the design, manufacturing process, operational procedures, documentation, or other relevant factors. Whereas, the random HW failure is one that can unpredictably occur during the lifetime of an HW element and that follows a probability distribution.

The standard provides an automotive-specific risk-based approach for determining risk classes (ASIL), where “D” and “A” represent the highest and lowest safety integrity levels, respectively. Note that ASIL is as a classification for the overall system, but the safety requirements specified to the HW and SW elements, in general, inherit the same level. For example, today SW-defined cockpit systems require ASIL-B (trending to ASIL-C) while ADAS and AD require ASIL-D. To give an idea of the implications, in terms of HW random failures, ASIL-D means that 99% of the faults potentially violating the safety goal shall be either detected or safely managed and that the overall system shall have a probability of residual (i.e., unmanaged) HW random failures less than 10 FIT (10 faults in one billion hours of operation). An important concept of ISO26262 is the safety mechanism, which is a technical solution implemented to detect and mitigate (tolerate, control, or avoid) failures in order to achieve/maintain the intended functionality or a safe state in the case of a failure without an unreasonable level of risk. The second edition of the standard emphasizes not only on fail-safe systems but also on fault-tolerant systems. Here, the goal is to guarantee the normal (or reduced) operation after a fault has occurred.

Despite FuSa is measured at system level, there are specific requirements for semiconductors. The second edition of ISO26262 will include a new part (part 11) with more than 150 pages of guidelines for digital and analog macrocells, FPGAs, and sensor circuits. Herein, some of the most important topics and challenges are as follows.

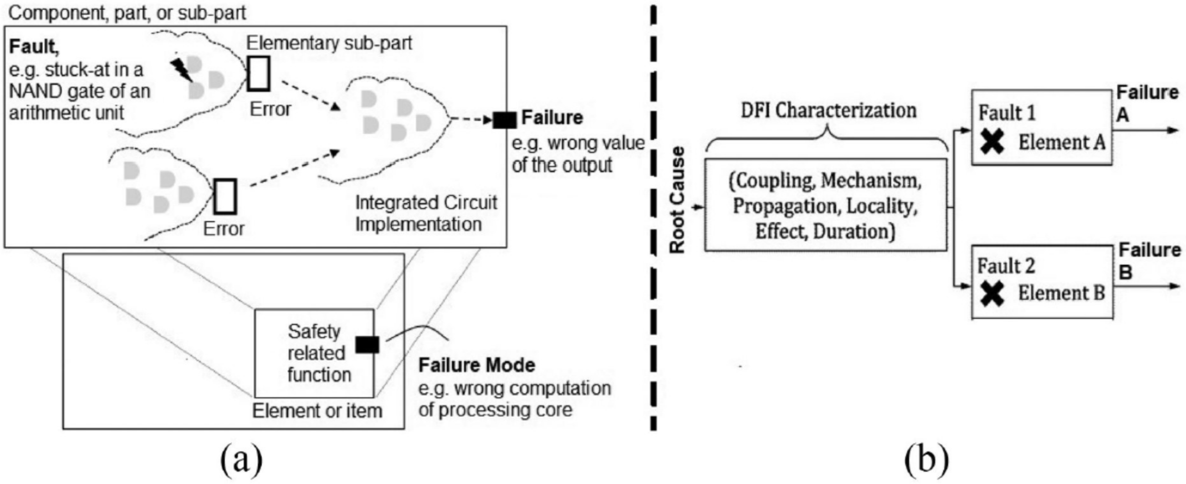


Fig. 6.

(a) Link between fault, error, and failure. (b) Dependent failures.

1. *How to consider safety aspects of semiconductor components?* The aspects of interest include in-context versus safety element out of context and definition of the assumption of use (AoU). The AoU refers to the usage modes or countermeasures that the system maker has to consider if using the safety-related semiconductor component.

2. *How to define the level of details of the safety analysis as a function of the safety concept, the stage of the analysis, and the safety mechanisms used?*

3. *How to determine the correlation between fault, error, and failures?* The relationship among the fault, error, and failures is depicted in Fig. 6(a). This challenge is also concerned with the definition of fault models, failure modes, and distribution of failure rate across failure modes. In order to address this challenge, guidelines are required to derive a consistent set of failure modes and consider new fault models (e.g., multiple stuck-at) caused by modern technologies.

4. *How to handle all kinds of macrocell (hard or soft) with or without embedded safety mechanisms embedded?* This challenge also extends to legacy macrocells.

5. *How to determine base failure rate for both permanent and transient faults?* Another challenge in this regard is to deal with nonconstant failure rates and advanced packaging.

6. *How to perform fault injection?* The scope of this challenge spans over different abstraction levels that support evaluation of the HW architectural metrics, pre-silicon verification of safety requirements, and detection of faults and control their effects.

7. *How to identify dependent failure initiators [DFI, see Fig. 6(b)]?* A related challenge is how to perform the dependent failure analysis.

8. *How to define and apply fault models, failure modes, safety mechanisms, and avoidance of systematic failures, with respect to ISO26262, for HW platforms?* The platforms include digital and analog components, memories, programmable logic devices (PLDs)/FPGAs, sensors/micro electro mechanical systems (MEMS), multicores, and modern system on chip (SoCs). The SoCs used in the automotive domain include a combination of the following HW and SW features.

    1. EDC/ECC for memories, including caches and registers.
    2. Built-in self-test for arrays and logic, which are operated both at key-on/off and at periodic intervals.
    3. Safety mechanisms for on-chip interconnects, including coherent fabrics (e.g., information redundancy, data/address codes, firewalls, and timeouts).
    4. Different redundancy types for processing cores (see Fig. 7).
    5. End-to-end safety protocols for peripherals. These protocols are combinations of CRC, time stamp, and frame counter.
    6. SW test libraries to address permanent failures in the logic not covered by other safety mechanisms.
    7. Dedicated HW cores for fault handling (e.g., Safety island).

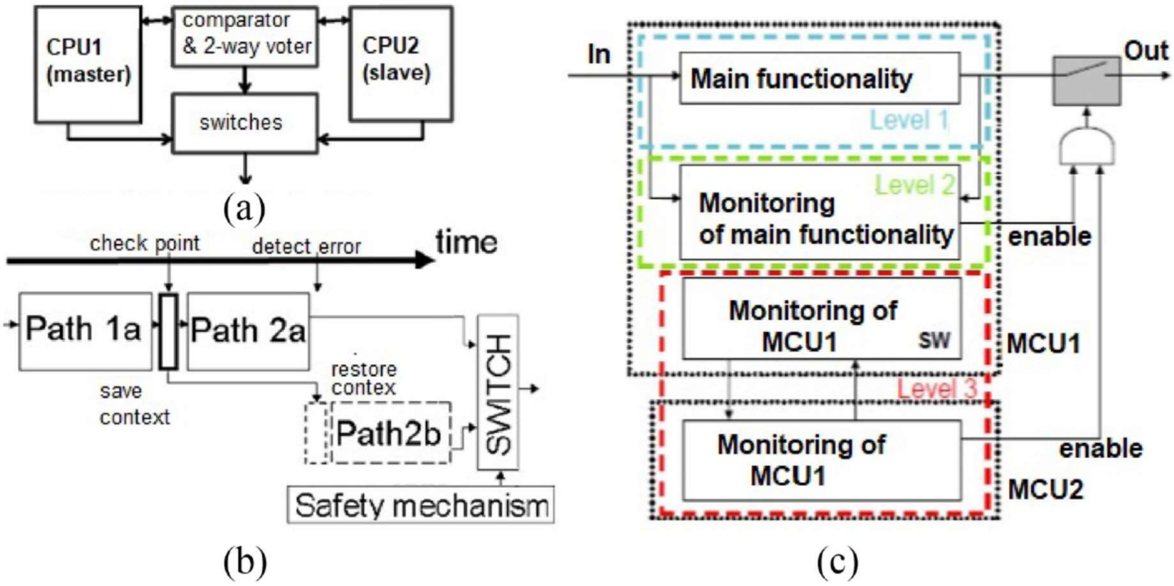


Fig. 7.

Different redundant architecture solutions.

B. Responsibility-Sensitive Safety

The most recent trend in FuSA is responsibility-sensitive safety (RSS). Introduced by Shalev-Shwartz *et al.* [123], the RSS model formalizes the common sense of human judgment under a comprehensive set of road situations. It sets clear definitions for what it means to drive safely versus to drive recklessly. With human drivers, the interpretation of responsibility for collisions and other incidents is fluid. Today, in the case of an accident, the blame is determined based on imperfect information and other factors interpreted afterward. With machines, the definitions can be formal and mathematical. Machines have highly accurate information about the environment around them; they always know their reaction time and braking power, and are never distracted or impaired. We do not need to interpret machines' actions after the fact. Instead, we can program them to follow a determined pattern—as long as we have the means to formalize that pattern. At its core, the RSS model is designed to formalize and contextualize today's driving dilemmas, such as notions of safe distance and safe gaps when merging and cutting in, which agent cuts in, and thus assumes responsibility to maintain a safe distance. Moreover, this model allows to specify the right of way, define safe driving with limited sensing (e.g., when road users are hidden behind buildings or parked cars and might suddenly appear), and more. Clearly, human judgment includes avoiding accidents and not merely avoiding blame. The RSS model attempts to build a formal foundation that sets all aspects of human judgment in the context of driving with the goal of setting a formal "seal of safety" for autonomous cars. More details on the RSS model can be found in [123].

SECTION VII.

Conclusion

This paper analyzed recent technological challenges and HW/SW solutions for on-board embedded and networked automotive systems. In this context, this paper mainly focused on automotive SW, advanced execution platforms, on-board network communications, on-board cybersecurity, and functional safety with respect to SW and HW. This paper identified the need for new E/E architectures, exploiting eHPC and number-crunching accelerators, supervised by a safe and secure MCU, to meet the computation and memory requirements in the order of TOPS and TB, respectively, for perception and fusion tasks. Besides HW, also the automotive SW complexity has drastically increased in the recent years. Model- and component-based SW development techniques have proven helpful and cost effective in managing the size and complexity of automotive SW, which is often in the range of several tens to hundreds MLoC. The SW complexity is expected to grow further in time. Hence, there is a strong need to develop efficient models and languages for the automotive SW development. Moreover, the existing standard technologies for the SW development (e.g., AUTOSAR) need to adapt according to the evolution in the car industry, with respect to advanced computer-controlled functionality, AD, ADAS, and V2X. There is also a strong need to support interoperability and automation among the state-of-the-art and state-of-the-practice languages, models, and tools that are used for the automotive SW development at various abstraction levels. In-car communications require new network architectures. Ethernet, with a broad choice of data rates, and the TSN standards will be key enablers for upcoming automotive scenarios, including AD. Current in-vehicle networks suffer from several vulnerabilities in terms of confidentiality, authentication, and availability. While some possible countermeasures have been already found, vehicular communications (V2X) and automated driving are fostering the steady rise of novel challenging vehicular cybersecurity issues. In addition to security, other open research topics that deserve investigation include traffic planning response-time analysis of TSN networks, and the use of Ethernet for 5G mobile fronthaul.

REFERENCES

1."Embedded system market size by application by product. Industry outlook report regional analysis application development potential price trends competitive market share & forecast 2016 to 2023", 2016.

2.F. Salewski and S. Kowalewski, "Hardware/software design considerations for automotive embedded systems", *IEEE Trans. Ind. Inform.*, vol. 4, no. 3, pp. 156-163, Aug. 2008.

3.S. Saponara, G. Pasetti, F. Tinfena, P. Dabramo and L. Fanucci, "HV-CMOS design and characterization of a smart rotor coil driver for automotive alternators", *IEEE Trans. Ind. Electron.*, vol. 60, no. 6, pp. 2309-2317, Jun. 2013.

4."Technology and computing requirements for self-driving cars", *doc. no. 0514/RH/CMD/PDF*, 2016.

5.S. Saponara, G. Ciarpi and B. Neri, "System-level modelling/analysis and LNA design in low-cost automotive technology of a v2x wireless transceiver", *Proc. 3rd IEEE Int. Forum Res. Technol. Soc. Ind. Leveraging Better Tomorrow*, pp. 1-5, Sep. 2017.

6.P. Guturu, "Explosive wireless consumer demand for network bandwidth-fifth generation and beyond [future directions]", *IEEE Consum. Electron. Mag.*, vol. 6, no. 2, pp. 27-31, Apr. 2017.

7.S. Brunner, J. Roder, M. Kucera and T. Waas, "Automotive e/e-architecture enhancements by usage of Ethernet TSN", *Proc. IEEE 13th Workshop Intell. Solutions Embedded Syst.*, pp. 9-13, Jun. 2017.

8.J. Dvorak and Z. Hanzalek, "Using two independent channels with gateway for FlexRay static segment scheduling", *IEEE Trans. Ind. Inform.*, vol. 12, no. 5, pp. 1887-1895, Oct. 2016.

9."ISO 11898-1, Road Vehicles–Interchange of Digital Information–Controller Area Network (CAN) for High-Speed Communication, ISO Standard-11898", Nov. 1993.

10.G. M. Zago and E. P. de Freitas, "A quantitative performance study on can and can FD vehicular networks", *IEEE Trans. Ind. Electron.*, vol. 65, no. 5, pp. 4413-4422, May 2018.

11.B. Kraemer, "Automotive Ethernet", *IEEE Commun. Mag.*, vol. 54, no. 12, pp. 4-4, Dec. 2016.

Show Context View Article Full Text: PDF (131KB) Google Scholar

12.S. Saponara and B. Neri, "Radar sensor signal acquisition and multidimensional FFT processing for surveillance applications in transport systems", *IEEE Trans. Instrum. Meas.*, vol. 66, no. 4, pp. 604-6125, Apr. 2017.

Show Context View Article Full Text: PDF (1424KB) Google Scholar

13.A. Lucas, M. Iliadis, R. Molina and A. K. Katsaggelos, "Using deep neural networks for inverse problems in imaging: Beyond analytical methods", *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 20-36, Jan. 2018.

View Article Full Text: PDF (1755KB) Google Scholar

14.M. Al-Qizwini, I. Barjasteh, H. Al-Qassab and H. Radha, "Deep learning algorithm for autonomous driving using GoogLeNet", *Proc. IEEE Intell. Veh. Symp.*, pp. 89-96, Jun. 2017.

View Article Full Text: PDF (533KB) Google Scholar

15.W. Shi, M. B. Alawieh, X. Li, H. Yu, N. Arechiga and N. Tomatsu, "Efficient statistical validation of machine learning systems for autonomous driving", *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, pp. 1-8, Nov. 2016.

Access at ACM Google Scholar

16.C. Vallon, Z. Ercan, A. Carvalho and F. Borrelli, "A machine learning approach for personalized autonomous lane change initiation and control", *Proc. IEEE Intell. Veh. Symp.*, pp. 1590-1595, 2017.

View Article Full Text: PDF (489KB) Google Scholar

17.N. Gallardo, N. Gamez, P. Rad and M. Jamshidi, "Autonomous decision making for a driver-less car", *Proc. 12th Syst. Syst. Eng. Conf.*, pp. 1-6, Jun. 2017.
View Article Full Text: PDF (1350KB) Google Scholar

18.C. Ilas, "Perception in autonomous ground vehicles", *Proc. Int. Conf. Electron. Comput. Artif. Intell.*, pp. 1-6, Jun. 2013.
View Article Full Text: PDF (287KB) Google Scholar

19.G. Prabhakar, B. Kailath, S. Natarajan and R. Kumar, "Obstacle detection and classification using deep learning for tracking in high-speed autonomous driving", *Proc. IEEE Region 10 Symp.*, pp. 1-6, Jul. 2017.
View Article Full Text: PDF (861KB) Google Scholar

20.M. Giering, V. Venugopalan and K. Reddy, "Multi-modal sensor registration for vehicle perception via deep neural networks", *Proc. IEEE High Perform. Extreme Comput. Conf.*, pp. 1-6, Sep. 2015.
View Article Full Text: PDF (3079KB) Google Scholar

21.C. Laugier and J. Chartre, "Intelligent perception and situation awareness for automated vehicles", *Proc. Conf. GTC Eur.*, pp. 1-22, 2016.
Show Context UniPi Google Scholar

22.G. Tanzmeister and D. Wollherr, "Evidential grid-based tracking and mapping", *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 6, pp. 1454-1467, Jun. 2017.
Show Context UniPi Google Scholar

23.S. C. Talbot and S. Ren, "Comparision of fieldbus systems CAN TTCAN FlexRay and lin in passenger vehicles", *Proc. 29th IEEE Int. Conf. Distrib. Comput. Syst. Workshops*, pp. 26-31, 2009.
View Article Full Text: PDF (228KB) Google Scholar

24.U. Keskin, "In-vehicle communication networks: A literature survey", 2009.
UniPi Google Scholar

25.S. Tuohy, M. Glavin, E. Jones, M. Trivedi and L. Kilmartin, "Next generation wired intra-vehicle networks a review", *Proc. IEEE Intell. Veh. Symp.*, pp. 777-782, Jun. 2013.
View Article Full Text: PDF (172KB) Google Scholar

26.N. Navet and F. Simonot-Lion, "In-vehicle communication networks A historical perspective and review" in Industrial Communication Technology Handbook, Boca Raton, FL, USA:CRC Press, vol. 96, 2013.
UniPi Google Scholar

27.S. Tuohy, M. Glavin, C. Hughes, E. Jones, M. Trivedi and L. Kilmartin, "Intra-vehicle networks: A review", *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 534-545, Apr. 2015.
View Article Full Text: PDF (376KB) Google Scholar

28.W. Zeng, M. A. S. Khalid and S. Chowdhury, "In-vehicle networks outlook: Achievements and challenges", *IEEE Commun. Surv. Tut.*, vol. 18, no. 3, pp. 1552-1571, Jul.–Sep. 2016.
Show Context View Article Full Text: PDF (3850KB) Google Scholar

29.C. Ebert and J. Favaro, "Automotive software", *IEEE Softw.*, vol. 34, no. 3, pp. 33-39, Jun. 2017.
Show Context View Article Full Text: PDF (841KB) Google Scholar

30.M. Broy, I. Kruger, A. Pretschner and C. Salzmann, "Engineering automotive software", *Proc. IEEE*, vol. 95, no. 2, pp. 356-373, Feb. 2007.
Show Context View Article Full Text: PDF (277KB) Google Scholar

31.J. Schroeder et al., "Predicting and evaluating software model growth in the automotive industry", *Proc. IEEE Int. Conf. Softw. Maintenance Evol.*, pp. 584-593, Sep. 2017.
Show Context View Article Full Text: PDF (671KB) Google Scholar

32.I. Baas, "A glimpse into the future of travel and its impact on marketing", Jan. 15 2016, [online] Available: http://www.thedrum.com/opinion/2016/01/11/glimpse-future-travel-and-its-impact-marketing.

Show Context UniPi Google Scholar

33.T. A. Henzinger and J. Sifakis, "The embedded systems design challenge", *Proc. 14th Int. Symp. Formal Methods*, pp. 1-15, 2006.

Show Context CrossRef UniPi Google Scholar

34.I. Crnkovic and M. Larsson, Building Reliable Component-Based Software Systems, Norwood, MA, USA:Artech House, 2002.

Show Context UniPi Google Scholar

35.P. Thorngren, "Keynote talk: Experiences from EAST-ADL Use" in , Gothenberg, Sweden, Oct. 2013.

Show Context UniPi Google Scholar

36.I. Crnkovic, S. Sentilles, A. Vulgarakis and M. Chaudron, "A classification framework for software component models", *IEEE Trans. Softw. Eng.*, vol. 37, no. 5, pp. 593-615, Sep. 2011.

Show Context View Article Full Text: PDF (1464KB) Google Scholar

37.K. Petersen et al., "Choosing component origins for software intensive systems: In-house COTS OSS or outsourcing?—A case survey", *IEEE Trans. Softw. Eng.*, vol. 44, no. 3, pp. 237-261, Mar. 2018.

Show Context View Article Full Text: PDF (1133KB) Google Scholar

38."EAST-ADL Domain Model Specification V2.1.12", [online] Available: http://www.east-adl.info/Specification/V2.1.12/EAST-ADL-Specification_V2.1.12.pdf.

Show Context UniPi Google Scholar

39.P. Cuenot et al., "Managing complexity of automotive electronics using the EAST-ADL", *Proc. 12th IEEE Int. Conf. Eng. Complex Comput. Syst.*, pp. 353-358, Jul. 2007.

View Article Full Text: PDF (111KB) Google Scholar

40.D. Chen et al., "Integrated safety and architecture modeling for automotive embedded systems", *e&i Elektrotechnik und Informationstechnik*, vol. 128, no. 6, pp. 196-202, Jun. 2011.

CrossRef UniPi Google Scholar

41.D. Chen, L. Feng, T. Qureshi, H. Lönn and F. Hagl, "An architectural approach to the analysis verification and validation of software intensive embedded systems", *Computing*, vol. 95, no. 8, pp. 649-688, 2013.

CrossRef UniPi Google Scholar

42.R. T. Kolagari et al., "Model-based analysis and engineering of automotive architectures with EAST-ADL: Revisited", *Int. J. Conceptual Struct. Smart Appl.*, vol. 3, no. 2, pp. 25-70, 2015.

Show Context CrossRef UniPi Google Scholar

43.S. Fürst and M. Bechter, "AUTOSAR for connected and autonomous vehicles: The AUTOSAR adaptive platform", *Proc. 46th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. Workshop*, pp. 215-217, Jun. 2016.

Show Context View Article Full Text: PDF (295KB) Google Scholar

44.K. Hänninen, J. Maki-Turja, M. Nolin, M. Lindberg, J. Lundback and K. Lundback, "The Rubus component model for resource constrained real-time systems", *Proc. IEEE Symp. Ind. Embedded Syst.*, pp. 177-183, 2008.

Show Context UniPi Google Scholar

45.S. Mubeen, H. Lawson, J. Lundbäck, M. Gålnander and K. L. Lundbäck, "Provisioning of predictable embedded software in the vehicle industry: The Rubus

approach", *Proc. IEEE/ACM 4th Int. Workshop Softw. Eng. Res. Ind. Pract.*, pp. 3-9, May 2017.
Show Context View Article Full Text: PDF (1019KB) Google Scholar

46."Road Vehicles – Functional Safety, ISO 26262-1:2011", [online] Available: http://www.iso.org/.
Show Context Google Scholar

47.G. Bahig and A. El-Kadi, "Formal verification of automotive design in compliance with ISO 26262 design verification guidelines", *IEEE Access*, vol. 5, pp. 4505-4516, 2017.
Show Context View Article Full Text: PDF (7516KB) Google Scholar

48."Future vehicle software architectures", Jan. 15 2018, [online] Available: https://www.esk.fraunhofer.de/en/research/projects/adaptives_bordnetz.htm l.
Show Context Google Scholar

49.S. Mubeen, J. Mäki-Turja and M. Sjödin, "Communications-oriented development of component-based vehicular distributed real-time embedded systems", *J. Syst. Arch.*, vol. 60, no. 2, pp. 207-220, 2014.
Show Context CrossRef Google Scholar

50.S. Mubeen, T. Nolte, M. Sjödin, J. Lundbäck and K.-L. Lundbäck, "Supporting timing analysis of vehicular embedded systems through the refinement of timing constraints", *Softw. Syst. Model.*, Jan. 2017, [online] Available: https://doi.org/10.1007/s10270-017-0579-8.
Show Context CrossRef Google Scholar

51.D. Schmidt and F. Kuhns, "An overview of the real-time CORBA specification", *Computer*, vol. 33, no. 6, pp. 56-63, Jun. 2000.
Show Context View Article Full Text: PDF (265KB) Google Scholar

52.M. G. Valls, I. R. Lopez and L. F. Villar, "iLAND: An enhanced middleware for real-time reconfiguration of service oriented distributed real-time systems", *IEEE Trans. Ind. Inform.*, vol. 9, no. 1, pp. 228-236, Feb. 2013.
Show Context View Article Full Text: PDF (799KB) Google Scholar

53.X. Ke, K. Sierszecki and C. Angelov, "COMDES-II: A Component-based framework for generative development of distributed real-time control systems", *Proc. 13th IEEE Int. Conf. Embedded Real-Time Comput. Syst. Appl.*, pp. 199-208, Aug. 2007.
Show Context View Article Full Text: PDF (301KB) Google Scholar

54.S. Sentilles, A. Vulgarakis, T. Bures, J. Carlson and I. Crnkovic, "A component model for control-intensive distributed embedded systems", *Proc. 11th Int. Symp. Compon. Based Softw. Eng.*, pp. 310-317, 2008.
Show Context CrossRef Google Scholar

55.*TADL: Timing Augmented Description Language*, Oct. 2009.
Show Context Google Scholar

56.*Timing Augmented Description Language (TADL2) Syntax Semantics*, Aug. 2012.
Show Context Google Scholar

57."TIMMO-2-USE Project", [online] Available: https://itea3.org/project/timmo-2-use.html.
Show Context Google Scholar

58.*Local Interconnect Network (LIN) Specification*, [online] Available: www.lin-subbus.org.
Show Context Google Scholar

59.F. Pieri, C. Zambelli, A. Nannini, P. Olivo and S. Saponara, "Is consumer electronics redesigning our cars? Challenges of integrated technologies for sensing computing and storage", *IEEE Consum. Electron. Mag.*, vol. 7, no. 5, pp. 8-17, Sep. 2018.
Show Context View Article Full Text: PDF (2218KB) Google Scholar

60.R. Saussard, B. Bouzid, M. Vasiliu and R. Reynaud, "A robust methodology for performance analysis on hybrid embedded multicore architectures", *Proc. IEEE 10th Int. Symp. Embedded Multicore/Many-Core Syst.-on-Chip*, pp. 77-84, Sep. 2016.
Show Context View Article Full Text: PDF (644KB) Google Scholar
61.M. Valero, "European Processor Initiative & RISC-V", *Proc. RISC-V Workshop*, May 2018.
Show Context UniPi Google Scholar
62.J. Huang, "NVIDIA CEO Keynote", *Proc. GPU Technol. Conf. Eur.*, Oct. 2017.
Show Context UniPi Google Scholar
63. Jan. 30 2018, [online] Available: https://newsroom.intel.com/wp-content/uploads/sites/11/2018/01/intel-mobileye-ads-product-brief.pdf.
Show Context UniPi
64.M. Gautschi et al., "Near-threshold RISC-V core with DSP extensions for scalable IOT endpoint devices", *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 25, no. 10, pp. 2700-2713, Oct. 2017.
Show Context View Article Full Text: PDF (2247KB) Google Scholar
65.E. Azarkhish, D. Rossi, I. Loi and L. Benini, "Neurostream: Scalable and energy efficient deep learning with smart memory cubes", *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 2, pp. 420-434, Feb. 2018.
Show Context View Article Full Text: PDF (2756KB) Google Scholar
66.F. Cazorla, J. Abella, E. Mezzetti, C. Hernandez, T. Vardanega and G. Bernat, "Reconciling time predictability and performance in future computing systems", *IEEE Des. Test*, vol. 35, no. 2, pp. 48-56, Apr. 2017.
Show Context View Article Full Text: PDF (511KB) Google Scholar
67.S. Saponara, F. Giannetti, B. Neri and G. Anastasi, "Exploiting mm-wave communications to boost the performance of industrial wireless networks", *IEEE Trans. Ind. Inform.*, vol. 13, no. 3, pp. 1460-1470, Jun. 2017.
Show Context View Article Full Text: PDF (748KB) Google Scholar
68.Y. Huo, X. Dong and W. Xu, "5G cellular user equipment: From theory to practical hardware design", *IEEE Access*, vol. 5, pp. 13992-14010, 2017.
Show Context View Article Full Text: PDF (4370KB) Google Scholar
69."CAN with flexible data-rate (CAN FD)", *White Paper Ver. 1.1.*, 2011.
Show Context UniPi Google Scholar
70.G. M. Zago and E. P. de Freitas, "A quantitative performance study on CAN and CAN FD vehicular networks", *IEEE Trans. Ind. Electron.*, vol. 65, no. 5, pp. 4413-4422, May 2018.
Show Context View Article Full Text: PDF (449KB) Google Scholar
71.S. Singer, "High performance compute architecture supporting revolutionary requirements" in , San Jose, CA, USA, Nov. 2017.
Show Context UniPi Google Scholar
72.D. Reinhardt and M. Kucera, "Domain controlled architecture - A new approach for large scale software integrated automotive systems", *Proc. Int. Conf. Pervasive Embedded Comput. Commun. Syst.*, pp. 221-226, Feb. 2013.
Show Context UniPi Google Scholar
73."IEEE Standard for Ethernet Amendment 1: Physical Layer Specifications and Management Parameters for 100 Mb/s Operation over a Single Balanced Twisted Pair Cable (100BASE-T1), IEEE Std 802.3bw-2015 (Amendment to IEEE Std 802.3-2015)", Mar. 2016.
Show Context UniPi Google Scholar

74.*IEEE Standard for Ethernet Amendment 4: Physical Layer Specifications and Management Parameters for 1 Gb/s Operation over a Single Twisted-Pair Copper Cable*, Sep. 2016.

Show Context UniPi Google Scholar

75."IEEE Standard for Local and Metropolitan Area Networks, Bridges and Bridged Networks, IEEE Std. 802.1Q", 2014.

Show Context UniPi Google Scholar

76."IEEE Standard for Local and Metropolitan Area Networks - Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks, IEEE Std 802.1AS-2011", Mar. 2011.

Show Context UniPi Google Scholar

77."MOST Specification Revision 3 Version 2", 2010, [online] Available: http://www.mostcooperation.com/.

Show Context UniPi Google Scholar

78.J. Migge, J. Villanueva, N. Navet and M. Boyer, "Insights on the performance and configuration of AVB and TSN in automotive networks", *Proc. Embedded Real-Time Softw. Syst.*, pp. 1-10, Jan. 2018.

Show Context UniPi Google Scholar

79.G. Alderisi, A. Caltabiano, G. Vasta, G. Iannizzotto, T. Steinbach and L. Lo Bello, "Simulative assessments of IEEE 802.1 Ethernet AVB and time-triggered Ethernet for advanced driver assistance systems and in-car infotainment", *Proc. Veh. Netw. Conf.*, pp. 187-194, Nov. 2012.

Show Context View Article Full Text: PDF (329KB) Google Scholar

80.G. Alderisi, G. Iannizzotto and L. Lo Bello, "Towards 802.1 Ethernet AVB for advanced driver assistance systems: A preliminary assessment", *Proc. IEEE 17th Conf. Emerg. Technol. Factory Automat.*, pp. 1-4, Sep. 2012.

Show Context View Article Full Text: PDF (436KB) Google Scholar

81.*Time-Triggered Ethernet AS6802*, Nov. 9 2016.

Show Context UniPi Google Scholar

82.S. M. Laursen, P. Pop and W. Steiner, "Routing optimization of AVB streams in TSN networks", *SIGBED Rev.*, vol. 13, no. 4, pp. 43-48, 2016.

Show Context UniPi Access at ACM Google Scholar

83.G. Alderisi, G. Patti and L. Lo Bello, "Introducing support for scheduled traffic over IEEE audio video bridging networks", *Proc. 18th IEEE Conf. Emerg. Technol. Factory Automat.*, pp. 1-9, Sep. 2013.

Show Context View Article Full Text: PDF (453KB) Google Scholar

84.L. Lo Bello, "Novel trends in automotive networks: A perspective on Ethernet and the IEEE Audio Video Bridging", *Proc. 19th IEEE Int. Conf. Emerg. Technol. Factory Automat.*, pp. 1-8, Sep. 2014.

Show Context View Article Full Text: PDF (384KB) Google Scholar

85.M. Ashjaei, G. Patti, M. Behnam, T. Nolte, G. Alderisi and L. Lo Bello, "Schedulability analysis of Ethernet audio video bridging networks with scheduled traffic support", *Real-Time Syst.*, vol. 53, no. 4, pp. 526-577, Jul. 2017.

Show Context CrossRef UniPi Google Scholar

86.U. D. Bordoloi, A. Aminifar, P. Eles and Z. Peng, "Schedulability analysis of Ethernet AVB switches", *Proc. 20th IEEE Int. Conf. Embedded Real-Time Comput. Syst. Appl.*, pp. 1-10, Aug. 2014.

Show Context View Article Full Text: PDF (424KB) Google Scholar

87.J. Diemer, D. Thiele and R. Ernst, "Formal worst-case timing analysis of Ethernet topologies with strict-priority and AVB switching", *Proc. 7th IEEE Int. Symp. Ind. Embedded Syst.*, pp. 1-10, Jun. 2012.
Show Context View Article Full Text: PDF (801KB) Google Scholar

88."IEEE Standard for Local and Metropolitan Area Networks – Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic, IEEE Std 802.1Qbv-2015 (Amendment to IEEE Std 802.1Q))", Mar. 2016.
Show Context UniPi Google Scholar

89.*Official Project Website of 802.1AS-Rev – Timing and Synchronization for Time-Sensitive Applications*, 2016, [online] Available: https://1.ieee802.org/tsn/802-1as-rev/.
Show Context UniPi Google Scholar

90."IEEE Standard for Local and Metropolitan Area Networks – Bridges and Bridged Networks – Amendment 26: Frame Preemption, IEEE Std 802.1Qbu-2016 (Amendment to IEEE Std 802.1Q-2014)", Aug. 2016.
Show Context UniPi Google Scholar

91.S. S. Craciunas, R. Serna Oliver and W. Steiner, "Formal scheduling constraints for time-sensitive networks", Sep. 2017, [online] Available: https://arxiv.org/abs/1712.02246.
Show Context CrossRef UniPi Google Scholar

92.W. Steiner, S. S. Craciunas and R. S. Oliver, "Traffic planning for time-sensitive communication", *IEEE Commun. Standards Mag.*, vol. 2, no. 2, pp. 42-47, Jun. 2018.
Show Context View Article Full Text: PDF (216KB) Google Scholar

93."IEEE Standard for Local and Metropolitan Area Networks – Bridges and Bridged Networks - Amendment 24: Path Control and Reservation, IEEE Std 802.1Qca-2015 (Amendment to IEEE Std 802.1Q-2014)", Mar. 2016.
Show Context UniPi Google Scholar

94."IEEE Standard for Local and Metropolitan Area Networks–Frame Replication and Elimination for Reliability, IEEE Std 802.1CB-2017", Oct. 2017.
Show Context UniPi Google Scholar

95.*P802.1Qcc – Stream Reservation Protocol (SRP) Enhancements and Performance Improvements*, 2013, [online] Available: https://1.ieee802.org/tsn/802-1qcc/.
Show Context UniPi Google Scholar

96."IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks–Amendment 29: Cyclic Queuing and Forwarding, IEEE 802.1Qch-2017 (Amendment to IEEE Std 802.1Q-2014)", Jun. 2017.
Show Context UniPi Google Scholar

97."IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks–Amendment 28: Per-Stream Filtering and Policing, IEEE Std 802.1Qci-2017 (Amendment to IEEE Std 802.1Q-2014)", Sep. 2017.
Show Context UniPi Google Scholar

98.*P802.1Qcr – Asynchronous Traffic Shaping*, 2016, [online] Available: https://1.ieee802.org/tsn/802-1qcr/.
Show Context UniPi Google Scholar

99.J. Specht and S. Samii, "Urgency-based scheduler for time-sensitive switched Ethernet networks", *Proc. 28th Euromicro Conf. Real-Time Syst.*, pp. 75-85, Jul. 2016.
Show Context View Article Full Text: PDF (910KB) Google Scholar

100.J. Specht and S. Samii, "Synthesis of queue and priority assignment for asynchronous traffic shaping in switched Ethernet", *Proc. IEEE Real-Time Syst. Symp.*, pp. 178-187, Dec. 2017.
Show Context View Article Full Text: PDF (889KB) Google Scholar

101.D. K. Nilsson, U. E. Larson, F. Picasso and E. Jonsson, "A first simulation of attacks in the automotive network communications protocol flexray", *Proc. Int. Workshop Comput. Intell. Secur. Inf. Syst.*, pp. 84-91, 2009.

CrossRef UniPi Google Scholar

102.C. W. Lin and A. Sangiovanni-Vincentelli, "Cyber-security for the controller area network (CAN) communication protocol", *Proc. Int. Conf. Cyber Secur.*, pp. 1-7, Dec. 2012.

View Article Full Text: PDF (295KB) Google Scholar

103.M. Wolf, A. Weimerskirch and C. Paar, Secure In-Vehicle Communication, Berlin, Germany:Springer, pp. 95-109, 2006.

Show Context UniPi Google Scholar

104.O. Avatefipour and H. Malik, "State-of-the-art survey on in-vehicle network communication can-bus security and vulnerabilities", *Int. J. Comput. Sci. Netw.*, vol. 6, no. 6, pp. 720-727, Dec. 2017.

UniPi Google Scholar

105.K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection", *Proc. 25th USENIX Secur. Symp.*, pp. 911-927, 2016.

Show Context UniPi Google Scholar

106.E. dos Santos, A. Simpson and D. Schoop, "A formal model to facilitate security testing in modern automotive systems", *Proc. Joint Workshop Handling IMPlicit EXplicit Knowl. Formal Syst. Develop. Formal Model-Driven Techn. Develop. Trustworthy Syst.*, pp. 95-104, Nov. 2017.

CrossRef UniPi Google Scholar

107.T. Hoppe, S. Kiltz and J. Dittmann, "Security threats to automotive can networks– practical examples and selected short-term countermeasures", *Rel. Eng. Syst. Saf.*, vol. 96, no. 1, pp. 11-25, 2011.

CrossRef UniPi Google Scholar

108.M. Lukasiewycz, P. Mundhenk and S. Steinhorst, "Security-aware obfuscated priority assignment for automotive can platforms", *ACM Trans. Des. Automat. Electron. Syst.*, vol. 21, no. 2, 2016.

Show Context UniPi Access at ACM Google Scholar

109.T. Eisenbarth, T. Kasper, A. Moradi, C. Paar, M. Salmasizadeh and M. T. M. Shalmani, "On the power of power analysis in the real world: A complete break of the KeeLoq code hopping scheme" in Advances in Cryptology – CRYPTO 2008, Berlin, Germany:Springer, pp. 203-220, 2008.

Show Context CrossRef UniPi Google Scholar

110.K. Koscher et al., "Experimental security analysis of a modern automobile", *Proc. IEEE Symp. Secur. Privacy*, pp. 447-462, May 2010.

Show Context View Article Full Text: PDF (1771KB) Google Scholar

111.R. Currie, "Hacking the CAN bus: Basic manipulation of a modern automobile through CAN bus reverse engineering", *SANS Reading Room*, Jun. 2017, [online] Available: https://www.sans.org/reading-room/whitepapers/threats/paper/37825.

Show Context UniPi Google Scholar

112.F. Li, L. Wang and Y. Wu, "Research on CAN network security aspects and intrusion detection design", Nov. 2017.

Show Context CrossRef UniPi Google Scholar

113.S. Shreejith et al., "VEGa: A high performance vehicular Ethernet gateway on hybrid FPGA", *IEEE Trans. Comput.*, vol. 66, no. 10, pp. 1790-1803, Oct. 2017.

Show Context View Article Full Text: PDF (1144KB) Google Scholar

114.B. Carnevale, L. Baldanzi, L. Pilato and L. Fanucci, "A flexible system-on-a-chip implementation of the advanced encryption standard", *Proc. 20th Int. Conf. Syst. Theory Control Comput.*, pp. 156-161, Oct. 2016.
Show Context View Article Full Text: PDF (771KB) Google Scholar

115.C. Patsakis, K. Dellios and M. Bouroche, "Towards a distributed secure in-vehicle communication architecture for modern vehicles", *Comput. Secur.*, vol. 40, pp. 60-74, 2014.
Show Context CrossRef UniPi Google Scholar

116.A. Sghaier, M. Zeghid and M. Machhout, "Fast hardware implementation of ECDSA signature scheme", *Proc. Int. Symp. Signal Image Video Commun.*, pp. 343-348, Nov. 2016.
Show Context View Article Full Text: PDF (4410KB) Google Scholar

117.H. Ueda, R. Kurachi, H. Takada, T. Mizutani, M. Inoue and S. Horihata, "Security authentication system for in-vehicle network" in SEI Tech. Rev. 81, Osaka, Japan, 2015.
Show Context UniPi Google Scholar

118.P. Mundhenk et al., "Security in automotive networks: Lightweight authentication and authorization", *Trans. Des. Automat. Electron. Syst.*, vol. 22, no. 2, pp. 25:1-25:27, 2017, [online] Available: http://doi.acm.org/10.1145/2960407.
Show Context UniPi Access at ACM Google Scholar

119.Q. Wang and S. Sawhney, "VeCure: A practical security framework to protect the can bus of vehicles", *Proc. Int. Conf. Internet Things*, pp. 13-18, Oct. 2014.
Show Context View Article Full Text: PDF (870KB) Google Scholar

120.P. Waszecki, P. Mundhenk, S. Steinhorst, M. Lukasiewycz, R. Karri and S. Chakraborty, "Automotive electrical and electronic architecture security via distributed in-vehicle traffic monitoring", *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 36, no. 11, pp. 1790-1803, Nov. 2017.
Show Context View Article Full Text: PDF (1422KB) Google Scholar

121.H. Okhravi, F. T. Sheldon and J. Haines, "Data diodes in support of trustworthy cyber infrastructure and net-centric cyber decision support" in Optimization and Security Challenges in Smart Power Grids., Berlin, Germany:Springer, pp. 203-216, 2013.
Show Context CrossRef UniPi Google Scholar

122.G. Xie, G. Zeng, Y. Liu, J. Zhou, R. Li and K. Li, "Fast functional safety verification for distributed automotive applications during early design phase", *IEEE Trans. Ind. Electron.*, vol. 65, no. 5, pp. 4378-4391, May 2018.
Show Context View Article Full Text: PDF (1188KB) Google Scholar

123.S. Shalev-Shwartz, S. Shammah and A. Shashua, "On a formal model of safe and scalable autonomous vehicles", Dec. 2017.