A Platform to Configure and Monitor Safety-Critical Applications for Automotive Central Computers

Hadi Askaripoor, Morteza Hashemi Farzaneh, and Alois Knoll
Chair of Robotics, Artificial Intelligence and Real-time Systems
Technical University of Munich
Boltzmannstr. 3, 85748 Garching bei München
Email:{hadi.askari,morteza.hashemi}@tum.de, {knoll}@in.tum.de

2021 IEEE 26th International Conference on Emerging Technologies and Factory Automation (ETFA) | 978-1-7281-2989-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/ETFA45728.2021.9613692

***Abstract*—In the recent years, the new generation of vehicles has required high-performance computing units due to a large number of safety-critical applications and functionalities. The process of configuring and integrating critical applications into vehicle central hardware while satisfying safety requirements and optimization objectives, is a time-consuming, complicated and error-prone process. This study describes a novel platform to automate the mapping of safety-critical applications to hardware resources on an automotive high-performance central computer while taking into account predefined safety requirements and optimization goals. This platform contains a monitoring mechanism to verify the fulfillment of safety-critical requirements, measure system performance, assess operating system and middleware, and evaluate the deployed mapping. Three modules that make up our proposed platform: *E/E Designer*, *AHPCC*, and *Monitoring System*.**

***Index Terms*—Autonomous Driving, Functional Safety, Automated Mapping, Automotive High-Performance Central Computer.**

I. Introduction

In recent years, automotive control systems, such as Advanced Driving Assistance Systems (ADAS) and Automated Driving Systems (ADS), have witnessed significant growth in complexity and functionality. Furthermore, due to the high computational power required for ADS, [1], the automotive Electrical/Electronic (E/E) designs migrate from domain-oriented to cross-domain centralized and centralized architectures using Automotive High-Performance Central Computers (*AHPCC*). Meeting safety conditions (e.g., reliability assurance) in their design based on ISO 26262 and safety of the Intended functionality (SOTIF), is a vital prerequisite insofar as these systems comprise safety-critical applications, software components, and hardware [2], [3].

However, increasing demands and complexity growth in the automotive hardware and software architecture, create significant challenges for the current systems architecture, methods, and tools. First, a complete posterior analysis to ensure that the safety-critical requirements are met is no longer cost-effective. Second, manual design and development of safety-critical applications on *AHPCCs* in compliance with safety standards are elaborate, labor-intensive, and cost-inefficient.

Our work-in-progress study introduces a platform to automate the integration process of automotive applications into *AHPCCs* while meeting the preset safety requirements and predetermined optimization objectives. In addition, we introduce a monitoring mechanism to observe and verify the fulfillment of the predefined requirements after deploying to our platform at run-time.

Hence, we described the two following research questions:

- How do we facilitate assignment of the *AHPCC* resources to the safety-critical applications and verify the fulfillment of the specified safety requirements in the design phase to compute an *AHPCC*-widely verified configuration?
- How do we verify the fulfillment of the specified safety requirements after deployment of the derived configuration to the *AHPCC* and performance evaluation of the *E/E Designer* at run-time?

This study is outlined as follows: Section II discusses the related work. Section III describes our approach including *AHPCC*, *E/E Designer*, and *Monitoring system*. Finally, section IV describes further steps and the conclusion.

II. Related Work

Several studies have focused on design space exploration and embedded systems optimization. *Archepotrix* is an open-source tool that supports modeling software components, communication between software components, electronic control units (ECUs), buses, and services. Nevertheless, the tool can optimize the deployment of software components to ECUs based on various objectives, such as redundancy allocation, energy consumption, and cost [4]. Becker et al. [5] proposed an engineering approach, named *Mechatronic UML*, to model software/hardware components, specify constraints, and verify the models using model checker UPPAAL for distributed mechatronic systems [6], [7]. This approach aims at bringing model-based design formal analysis to the mechatronic area. Another open-source model-based development tool that supports architecture modeling from the requirements to code generation for embedded systems, presented as *AutoFOCUS3* in [8]. Furthermore, the tool can synthesize hardware platform architectures, including end-to-end latency calculation. It is also able to synthesize and explore optimal deployments and schedules. A domain-specific constraint language, named *AAOL*, utilized for modeling and optimization of automotive E/E architectures [9]. The supporting tool can be optimized based on user-defined objectives, and support design constraints, including memory capacity, Automotive Safety

Authorized licensed use limited to: Technische Universitaet Muenchen. Downloaded on May 11,2022 at 08:43:47 UTC from IEEE Xplore. Restrictions apply.

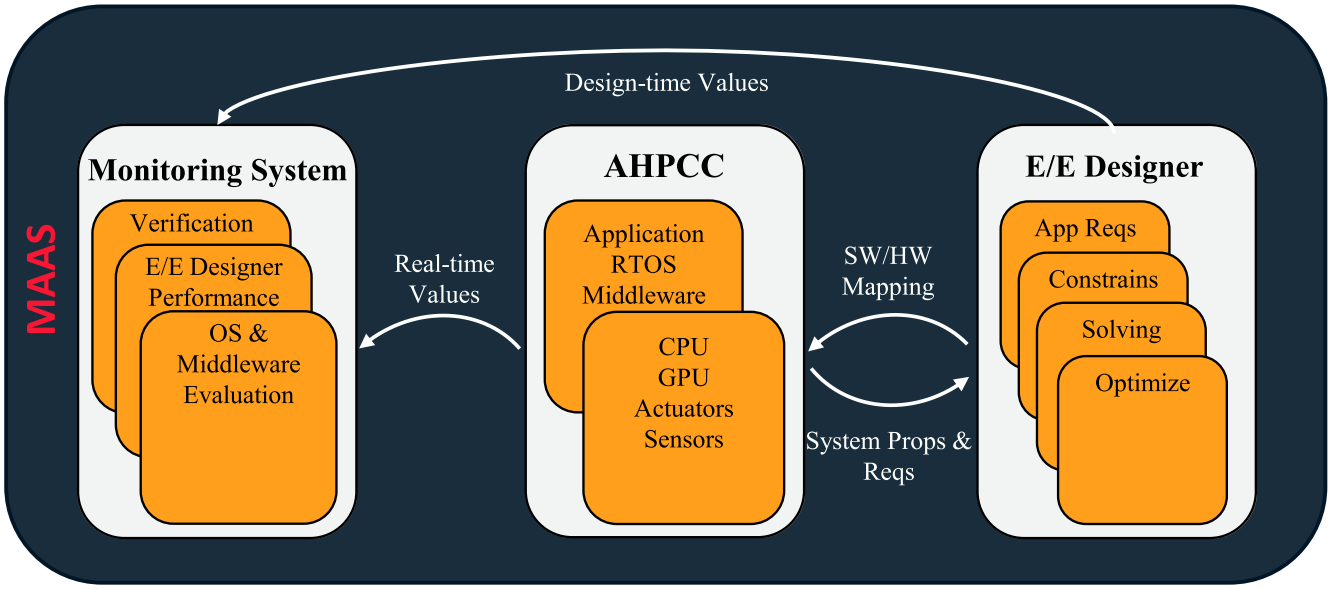

Figure 1. The MAAS (Mapping Automotive Applications Safely) platform overview to configure and monitor safety-critical applications including three modules.

Integrity Level (ASIL) requirements, and predefined deployments. *OSATE* is another open-source tool that implements the architecture analysis and design language (AADL) and supports modeling both aerospace and automotive systems. It supports model checking, schedulability analysis, and flow latency analysis; however, it does not support any optimization goals [10].

An approach has been introduced [11] to solve and optimize mapping issues (i.e., deployment of a safety-critical application on avionics hardware) for distributed systems in the aviation domain. To facilitate the software integration process for multi-core and many-core embedded systems, an open-source tool, called *AAP4MC*, was proposed [12]. It contributes to the mapping and partitioning of embedded multi-core systems while optimizing the final solution regarding preset optimization goals such as load balancing and energy consumption.

Therefore, to the best of our knowledge, no platform enables the *AHPCC* resource mapping, verification, and optimization at design-time while ensuring the attainment of the safety requirements at run-time. As a result, we present a novel platform that fills this gap and addresses the aforementioned research questions.

III. Approach

As indicated in Figure. 1, our proposed platform consists of three modules comprising *E/E Designer*, *AHPCC*, and *Monitoring System*. Also, Figure. 1 illustrates how the modules interact with each other. Each module is explained in detail in the following:

A. AHPCC

As mentioned earlier, the automotive E/E architecture moves toward centralized architecture demanding a high-performance central computer or *AHPCC* to be capable of processing a huge quantity of data. Our proposed platform consists of an *AHPCC* having two sub-modules such as software (SW) and hardware (HW). The SW comprises applications, real-time operating systems (RTOS), and middleware whereas the HW comprises a multi-core processor, Graphics Processing Unit (GPU), actuators, and sensors. The *AHPCC* module acts as the center of our proposed platform sending its system properties and requirements to the *E/E Designer*. Accordingly, SW/HW mapping, calculated by the *E/E Designer*, is deployed on the *AHPCC* (See Figure. 1). Finally, the *AHPCC* sends values of the mapping-related parameters to the *Monitoring System* after SW/HW mapping deployment as demonstrated in Figure. 1. There are several AHPCCs, which have been developed by different companies, that we introduce the most relevant products to our work in the following.

Figure. 2. (a) and 2. (b) demonstrated *DRIVE AGX Xavier* and *Pegasus Developer Kits*, respectively. It is claimed to provide standard SW, HW, and sample applications for the development of self-driving vehicles. They support various Input (I)/Output (O) interfaces, such as camera, lidar, Radar, and vehicle IO. Both kits have two Xavier systems-on-a-chip (SoCs) that can cover six various types of processors, such a CPU (including 8 cores), GPU, Deep Learning Accelerator (DLA), Programmable Vision Accelerator (PVA), Image Signal Processor (ISP), and stereo/optical flow accelerator. Besides, Pegasus (Figure. 2. (b)) employed the power of other Turing GPUs to achieve a higher tera operations per second (TOPS) rate [13].

Authorized licensed use limited to: Technische Universitaet Muenchen. Downloaded on May 11,2022 at 08:43:47 UTC from IEEE Xplore. Restrictions apply.

MPPA-DEV4 development platform, indicated in Figure. 2. (c), can be considered as another *AHPCC*. It prepares a ready-to-use environment to evaluate, develop, and optimize applications in automotive, data-centric, robotics, and communication domains [14]. Figure. 2. (d) described *R-Car H3* and *M3 Starter Kits* for supporting automotive software development. Also, the process of establishing open-source automotive Linux environments was facilitated by these products [15]. *AVA-3501* is a computing platform suitable for autonomous

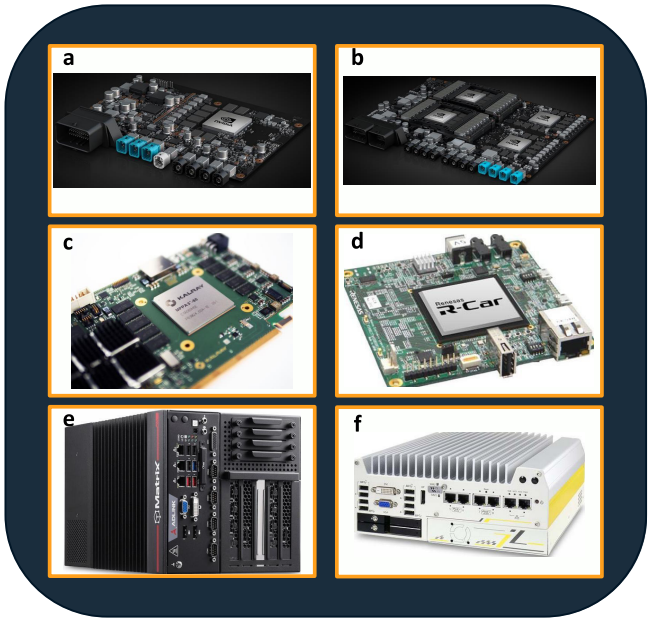

Figure 2. The six relevant *AHPCC* for the MAAS platform.

cars including Intel Xenon 9th Gen CPU and RTX8000 GPU (See Figure. 2. (e)) [16]. Finally, the last related *AHPCC* ,with product information *Nuvo-7208VTC*), contains a 8-core processor (See Figure. 2. (f)) [17].

Based on the computational power, core's number, variety of automotive applications and interfaces, customer support, and documentation, one of the aforementioned HW will be chosen as our *AHPCC*.

B. E/E Designer

As demonstrated in Figure. 1, *E/E Designer* includes four sub modules. After deploying the system properties and requirements from *AHPCC* to *E/E Designer*, the SW/HW mapping is computed by the *E/E Designer* utilizing the initial idea of the proposed approach in [18]. Figure. 3 illustrates the procedures to generate SW/HW mapping. The system requirements and properties, extracted from selected *AHPCC*, include the number of CPU cores, type of core's architecture, RTOS properties, ASIL level of the HW components, GPU properties, safety-criticality level of the applications, mean time to failure (MTTF) limit, reliability, freedom from interference (FFI) using isolation technique (e.g., partitioning), redundancy, execution time, memory usage, CPU/GPU utilization, energy consumption, latency, dropped message/frame

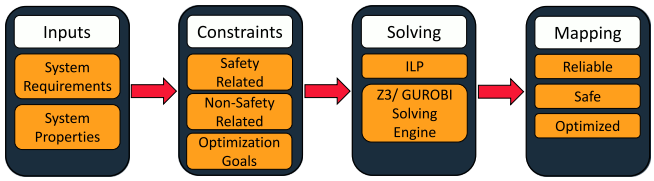

Figure 3. The *E/E Designer* approach generates a mapping, fulfills the preset requirements, and optimization objectives.

rate, etc. These requirements and properties as the inputs of the *E/E Designer* are automatically transformed, through performing the model-driven development (MDD) approach, into safety and non-safety constraints that represent their mathematical formulation using integer linear programming (ILP) [19]. Furthermore, the optimization objectives based on system requirements are defined (e.g., improve MTTF, and system performance) and integrated into the constraints system (See Figure. 3). The constraints are then solved utilizing *Z3/GUROBI* solving engine indicated in Figure. 3 [20], [21]. Consequently, a design-time mapping/configuration solution is created (i.e., assignment of an application to the *AHPCC* resource based on predetermined requirements) and deployed to the *AHPCC* to be conducted in the real-world environment to identify the behavior of mapping solution in run-time (See Figure. 3 and Figure. 1).

C. Monitoring System

As far as the run-time behavior of the operating system, and middleware in the *AHPCC* is uncertain due to event-based activities (e.g., application service discovery, and other dynamic and interacting processes causing non-deterministic system resource usage), consideration and creation of the relevant constraints at the design-time by the *E/E Designer* are unrealistic. To establish the fulfillment of the safety requirements at run-time after deploying the mapping solution, computed by the *E/E Designer*, to the *AHPCC*, the *Monitoring System* module is developed in our proposed platform (See Figure. 1).

To proceed with this system, the approach described in [22] is followed. The approach has introduced a monitoring mechanism for identifying the timing violations in autonomous driving platforms. As illustrated in Figure. 4, to mitigate the risk in case of violation of safety-critical requirements, the *Monitoring System* receives the predefined requirements from the *E/E Designer* module. It also acquires the run-time values from the *AHPCC* module. In the next step, the design-time requirements and run-time status regarding the requirements are compared and continuously verified. As an example, a safety-critical application, *A*, must be isolated as well as mapped on a CPU core, *C*, which meets the ASIL D based on HW properties. After taking this requirement into account in the constraints system of the *E/E Designer*, it is deployed to the *AHPCC* for a real-world scenario. Therefore, the *Monitoring System* verifies in real-time whether application *A* has been isolated and assigned to core *C* or not. This observation is

Authorized licensed use limited to: Technische Universitaet Muenchen. Downloaded on May 11,2022 at 08:43:47 UTC from IEEE Xplore. Restrictions apply.

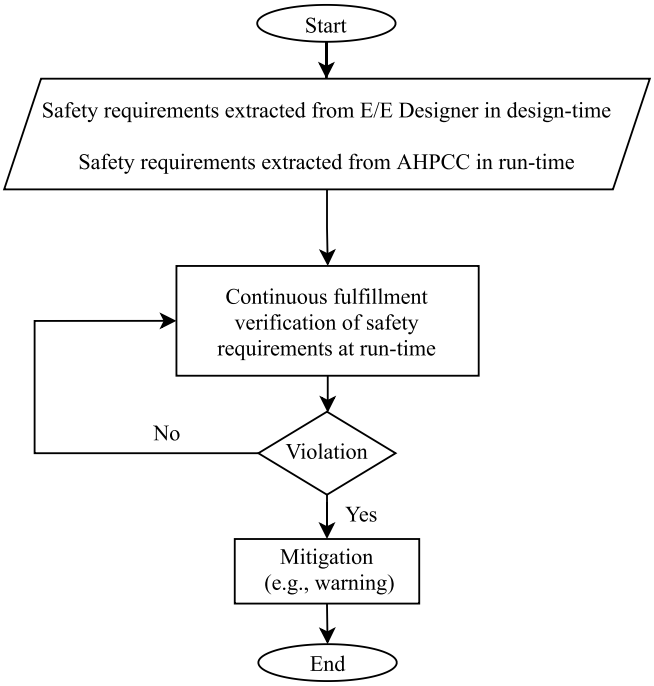

Figure 4. The flow chart shows the action of the Monitoring System in case of violation.

visualized as a warning in a graphical user interface (GUI) in case of violation (See Figure. 4). In addition, the *Monitoring System* evaluates the *E/E Designer* performance in terms of various metrics on account of utilizing its mapping solution and a random mapping for the *AHPCC*. This system also checks the possible safety violations created by an unpredictable operating system and middleware procedures such as multi-threading, automatic resource allocation, etc (See Figure. 1).

IV. Further Steps & conclusion

In this paper, we proposed a novel platform, including three modules: *E/E Designer*, *AHPCC*, and *Monitoring System*, to (semi-) automate the mapping procedure of the safety and non-safety critical applications on an automotive high-performance central computer (AHPCC) fulfilling the predefined static safety and non-safety requirements as well as considering the *AHPCC* properties. We presented the *E/E Designer* module to create a SW/HW mapping based on the requirements and properties using integer linear programming (ILP) method. We also analyzed the most related existed developer kits to be used in our platform as the *AHPCC* module. Moreover, we presented a *Monitoring System* module to verify the meeting of the requirements dynamically, measure the *AHPCC* performance, and evaluate the performance analysis depended on the prespecified static requirements displaying on a graphical user interface (GUI).

Our future work will focus on choosing a *AHPCC* based on the aforementioned analysis, developing the *E/E Designer* module including several safety requirements (e.g., redundancy, FFI, ASIL level, and reliability), and carrying out the *Monitoring System* following its specifications.

References

[1] V. Bandur, G. Selim, V. Pantelic, and M. Lawford, "Making the case for centralized automotive e/e architectures", *IEEE Transactions on Vehicular Technology*, vol. 70, no. 2, pp. 1230–1245, 2021. DOI: 10.1109/TVT.2021.3054934.

[2] ISO 26262:2018, "Road vehicles – functional safety", en, Tech. Rep. [Online]. Available: https://www.iso.org/standard/43464.html.

[3] ISO/PAS 21448:2019, "Road vehicles – safety of the intended functionality", en, Tech. Rep. [Online]. Available: https://www.iso.org/standard/70939.html.

[4] A. Aleti, S. Bjornander, L. Grunske, and I. Meedeniya, "Archeopterix: An extendable tool for architecture optimization of aadl models", in *2009 ICSE Workshop on Model-Based Methodologies for Pervasive and Embedded Software*, IEEE, 2009, pp. 61–71.

[5] S. Becker, S. Dziwok, C. Gerking, C. Heinzemann, W. Schäfer, M. Meyer, and U. Pohlmann, "The mechatronicuml method: Model-driven software engineering of self-adaptive mechatronic systems", in *Companion Proceedings of the 36th International Conference on Software Engineering*, 2014, pp. 614–615.

[6] S. Burmester, H. Giese, and M. Tichy, "Model-driven development of reconfigurable mechatronic systems with mechatronic uml", in *Model Driven Architecture*, Springer, 2004, pp. 47–61.

[7] J. Bengtsson, K. Larsen, F. Larsson, P. Pettersson, and W. Yi, "Uppaal—a tool suite for automatic verification of real-time systems", in *International hybrid systems workshop*, Springer, 1995, pp. 232–243.

[8] V. Aravantinos, S. Voss, S. Teufl, F. Hölzl, and B. Schätz, "Autofocus 3: Tooling concepts for seamless, model-based development of embedded systems.", *ACES-MB&WUCOR@ MoDELS*, vol. 1508, pp. 19–26, 2015.

[9] S. Kugele and G. Pucea, "Model-based optimization of automotive e/e-architectures", in *Proceedings of the 6th International Workshop on Constraints in Software Testing, Verification, and Analysis*, 2014, pp. 18–29.

[10] P. Feiler, "The open source aadl tool environment (osate)", Carnegie Mellon University Software Engineering Institute Pittsburgh United, Tech. Rep., 2019.

[11] R. Hilbrich and L. Dieudonné, "Deploying safety-critical applications on complex avionics hardware architectures", 2013.

[12] R. Höttger, H. Mackamul, A. Sailer, J.-P. Steghöfer, and J. Tessmer, "App4mc: Application platform project for multi-and many-core systems", *it-Information Technology*, vol. 59, no. 5, pp. 243–251, 2017.

[13] NVIDIA. (2021). Nvidia drive hardware, [Online]. Available: https://www.nvidia.com/en-us/self-driving-cars/drive-platform/hardware/.

[14] KALRAY. (2020). Safe compute acceleration for automotive, [Online]. Available: https://www.kalrayinc.com/automotive/.

[15] RENESAS. (2021). R-car-h3-m3-starter-kit, [Online]. Available: https://www.renesas.com/jp/en/products/automotive-products/automotive-system-chips-socs/r-car-h3-m3-starter-kit.

[16] ADLINK. (2021). Adlink ava-3501, [Online]. Available: https://www.adlinktech.com/en/Connected-Autonomous-Vehicle-Solutions.

[17] Neosys. (2021). Nuvo-7208vtc, [Online]. Available: https://omtec.de/industrie-pc/rugged-embedded/nuvo-7000-serie/nuvo-7208vtc.

[18] H. Askaripoor, M. H. Farzaneh, and A. Knoll, "Considering safety requirements in design phase of future e/e architectures", in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, 2020, pp. 1165–1168.

[19] C. Atkinson and T. Kuhne, "Model-driven development: A metamodeling foundation", *IEEE software*, vol. 20, no. 5, pp. 36–41, 2003.

[20] L. Gurobi Optimization, *Gurobi optimizer reference manual*, 2021. [Online]. Available: http://www.gurobi.com.

[21] L. De Moura and N. Bjørner, "Z3: An efficient smt solver", in *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, Springer, 2008, pp. 337–340.

[22] H. Askaripoor., S. Shafaei., and A. Knoll., "A flexible scheduling architecture of resource distribution proposal for autonomous driving platforms", in *Proceedings of the 7th International Conference on Vehicle Technology and Intelligent Transport Systems - VEHITS,*, INSTICC, SciTePress, 2021, pp. 594–599.

Authorized licensed use limited to: Technische Universitaet Muenchen. Downloaded on May 11,2022 at 08:43:47 UTC from IEEE Xplore. Restrictions apply.