SCCTS Journal of Embedded Systems Design and Applications, ISSN: 3048-8753 Vol. 3, No. 2, 2026 (pp. 26-35)
RESEARCH ARTICLE ECEJOURNALS.IN

Hardware-Assisted Intrusion Detection System for Automotive Embedded ECUs

Yip Mum Wai[1]*, Mohamad Bin Abdul Hamid[2]
[1]Tunku Abdul Rahman College, Malaysia
[2]Universiti Kebangsaan Malaysia, Malaysia

Keywords:
Intrusion Detection System, ECU, Automotive Security, CAN Bus, Hardware Security, ISO/SAE 21434, Real-Time Monitoring, Embedded System, ARM Cortex-R5

Author's Email:
yipmw@mail.tarc.edu.my, mdah@ukm.my

https://doi.org/10.31838/ESA/03.02.04

Received : 07.01.2026
Revised : 13.02.2026
Accepted : 25.03.2026

Abstract

Modern automotive systems progressively rely on Electronic Control Units (ECUs) to take control of vital vehicle functionality and are heavily interlinked on in-vehicle networks, e.g. the Controller Area Network (CAN). With the increased vehicle sophistication into a complex cyber-physical system, the car is more exposed to cyber-related attacks such as message injection, spoofing, and denial-of-service (DoS) attacks that have the potential to affect safety and performance. Implementations of Intrusion Detection Systems (IDS) using software, can be not very effective when it comes to real-time detection, because of the computational overhead imposed by them and the shared CPU usage. The paper proposes a new hardware-assisted intrusion detection system (H-IDS) designed to work on the automotive platforms and embedded devices (ECUs) in particular but is focused on the low-latency and real-time nature of the system and its capability to deploy on resource-constrained devices. The described architecture uses a light on-chip co-processor in the ECU as a special security engine to observe CAN traffic and observe abnormal behavior based on temporal and statistical characteristics. The H-IDS performs detection tasks on a separate processor-offloading payload so it does not interfere with other processes or even stop when the vehicle does not perform its tasks. This was implemented on an ECU platform based on the ARM Cortex-R5 in a hardware prototyping system with FPGA and then tested in a range of simulated attack conditions. The input of the experiment shows that there is a significant reduction in the detection latency that the H-IDS displays an average latency of 14 microseconds that is much faster than software-only solutions. Also, the design has less than 2 percent CPU overhead and its power and memory requirements fit squarely within the power and memory budget of most automotive embedded systems. The provided proposed H-IDS is aligned with ISO/SAE 21434 standards of cybersecurity, and it is scalable to be deployed on various heterogeneous ECU systems. Through the increased detection responsiveness and negligible resource consumption, the work is a step towards realistic hardware-enforced security measures, which can be incorporated in the next generation of automotive architectures to guarantee resilient and trustful in-vehicle networks.

How to cite this article: Wai YM, Hamid MBA (2026). Hardware-Assisted Intrusion Detection System for Automotive Embedded ECUs. SCCTS Journal of Embedded Systems Design and Applications, Vol. 3, No. 2, 2026, 26-35

Introduction

Automotive industry is experiencing high rate of digital transformation, which is because of development of electronic control, autonomous driving and vehicle connectivity. Contemporary vehicles represent complicated cyber-physical systems capable of having tens of Electronic Control Units (ECUs) that control important functionalities, such as engine control, braking, steering, infotainment, and advanced driver-assistance systems (ADAS). They can exchange data with others of their own such as in-vehicle networks (IVNs) mainly such as the Controller Area Network (CAN) bus to coordinate real-time activities and to guarantee efficiency operations. Nevertheless, the resulting connectivity and interconnectivity subjects vehicles to various cybersecurity risks, which are on the rise to pose a bigger challenge to in-vehicle systems of being hacked.

Some types of cyberattacks on automotive ECUs are spoofing, message injection, denial-of-service (DoS) and replay attacks. Such attacks may breach safety of vehicles as well as privacy of users. Events of Real life, including the hijacking of a Jeep Cherokee wherein its control was taken remotely in 2015 have highlighted how serious a well functioning security construct is imperative in vehicular networks. The conventional Intrusion Detection Systems (IDS) are mainly designed to work at the software level but are bound by the computational capability of the embedded processors and may end up in excessive latency and false positive behavior in low resource systems.

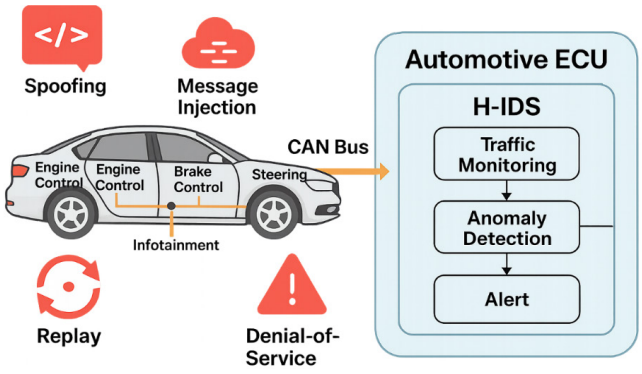

Fig. 1: Overview of Automotive Cybersecurity Threats and the Proposed Hardware-Assisted Intrusion Detection System (H-IDS) Architecture.

Since automotive applications are highly deterministic, critical and real-time, there is a sense of urgency to develop lightweight, efficient, and low-latency security mechanisms, which can be deployed into the vehicle ECUs themselves. In order to solve these problems, the current paper presents a new hardware supported intrusion detection system (H-IDS) that targets the automotive ECUs. The H-IDS allows real-time monitoring of CAN traffic to be carried out continuously without excessive distraction of the main ECU processes as it passes the detection load to a dedicated on-chip co-processor.

This paper has the following principal contributions to make:

- We offer a design and implementation of the prototype of a real-time H-IDS capable of identifying a wide range of known intrusion types in a network-based system with high degrees of accuracy and little overhead.
- The system is incorporated naturally into an ARM Cortex-R5 based automotive ECU.
- We test the performance of the offered solution against practical automotive attack settings where we simulated the benefits in detection latency, resource consumption, and alignment to the ISO/SAE 21434.

The proposed H-IDS is a notable step towards the security of the future of smart, connected cars since it allows or enables hardware-based monitoring and pre-emptive threats.

Related Work

Automobiles today mostly depend on Electronic Control Unit (ECU) and in-vehicle network (IVN) to control real-time functions and to handle communications within different subsystems. Examples of common IVNs are Controller Area Network (CAN), Local Interconnect

Network (LIN), FlexRay and Automotive Ethernet. The most commonly used is the CAN protocol because it is simple to use, although it lacks any security capability that makes it more prone to be attacked by hackers.

Cyberattacks targeting in-vehicle network are generally based on CAN broadcast nature of signal and trust-based model of communication. Even the likes of spoofing (transducing phony CAN frames), replay (rewtransmitting received valid messages), flooding (draining the network with elevated-priori frames), and denial of service (DoS) attacks (denying access to vital functions of ECUs) are worth mentioning. Such weaknesses require installation of intrusion detection system (IDS) in the automotive communication stack.

Rule-based, and anomaly-based are the general categories of the existing IDS solutions. Rule-based systems deploy a pre-defined patterns or signatures to identify familiar attacks,[1] where anomaly-based systems identify the normal pattern of traffic and warn possible threats as deviations.[2] By contrast, software-based IDS frameworks have gained a lot of attention in the literature but are not widely used in embedded automotive ECUs because of processing power, memory, and real-time limitations. Furthermore, potential tampering of software-only solutions is possible and latency that can be introduced by software-only solutions can break safety-critical timing constraints.

In response to these drawbacks, the hard-based IDS mechanisms have become popular. Such solutions get dedicated co-processors, FPGA fabric or custom logic blocks to track traffic and assess anomalies on low latency lanes. As an example, Wang et al.[3] have described a CAN IDS that monitors packets by using FPGA to significantly shorten their detection time. In the same line of thought, Cho and Shin[4] generated a physical fingerprinting scheme based on clock-skew to authenticate ECUs that provides a new design of anomaly detection with the incorporation of hardware.

The table 1 summarizes key recent works in this domain:

These studies portray an interest towards incorporating hardware accelerators and lightweight detecting engines in order to counter the resource and time limits of embedded ECUs. Nevertheless, only limited solutions have shown compatibility with automotive-grade MCUs, including ARM Cortex-R5, which a real-time and a co-processor based hardware-assisted IDS is provided in this paper.

System Architecture

ECU Platform and Network Configuration

The suggested hardware-assisted intrusion detector (H-IDS) is deployed on the current vehicle-grade embedded frontier according to the ARM Cortex-R5 processor, a high-performance, real-time microcontroller core focused on functional safety applications and commonly used in the automotive ECUs. The Cortex-R5 will be a dual-issue pipeline architecture, low interrupt latency, and built-in ECC (Error Correction Code), which makes Cortex-R5

Table 1: Literature Review Summary of IDS Techniques fo6r Automotive ECUs

Ref	Methodology	Platform	Dataset/Tool	Pros	Cons
[1] Miller & Valasek, 2015	Rule-based signature matching	CAN	Custom CAN logs	Effective for known attacks	Ineffective against zero-day exploits
[2] Taylor et al., 2016	Anomaly detection using SVM	Embedded Linux ECU	CAN-in-the-Middle Toolchain	Detects unknown attacks	High false-positive rate
[3] Wang et al., 2018	FPGA-based H-IDS	Xilinx Spartan-6 FPGA	CANoe Simulator	Low latency, high throughput	Complex integra-tion
[4] Cho & Shin, 2016	Clock-skew fingerprinting	Custom ECU testbed	Physical test setup	No protocol modification required	Sensitive to environmental changes
[5] Wu et al., 2020	Deep Learning-Based Anomaly Detection	Raspberry Pi 3	Car-Hacking Dataset	Adaptive learning, good detection accuracy	Computationally intensive for ECUs

more suitable to automotive control applications, powertrains, chassis, and advanced driver-assistance systems (ADAS) that are safety-critical applications. In a design which uses a family of processors, the target ECU hosts this processor in a system-on-chip (SoC) design, which may also include on-chip memory, peripheral interfaces and optional on-chip FPGA fabric to co-process security functions. The in-vehicle communication takes place under the Controller Area Network (CAN) protocol and it is set up with a baud rate of 500 kbps, the industry standard of medium- to high-speed car information transmission. Many ECUs are linked together via the CAN bus with a common differential bus line, with strong real-time messaging capability and in-built message prioritization through identification-based arbitration. Communication protocol stack ISO 11898-1 standard which defines the data link and physical layer of CAN is extended with a light weight CAN driver that interfaces to the hardware abstraction layer (HAL) with a real time operating system (RTOS). The ECU is computer-programmable to accept standard frames of CAN 2.0A/B and those frames incorporate hooks to allow application-layer diagnostics (UDS over CAN) and telemetry. IDS co-processor will be used to tap into the receive buffer of the CAN controller without interrupting data flow to monitor and analyze all log entries in real time.

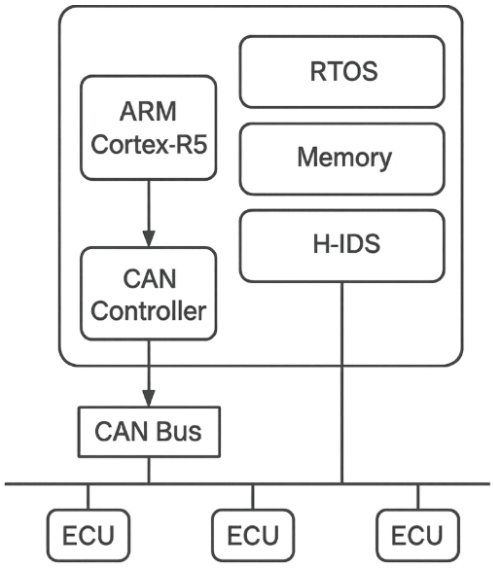

Fig. 2: Block Diagram of ARM Cortex-R5-Based ECU with Integrated Hardware-Assisted Intrusion Detection System (H-IDS).

This transparent interface to the ECU and the CAN stack means there is extreme visibility of network traffic without impacting on normal control flow, or changing interrupt load imposed on the main processor.0

Proposed Hardware-Assisted IDS Framework

The hardware-aided intrusion monitoring system (H-IDS) is obdurated to be a modular co- processor added to the main ECU processor in the system-on-chip (SoC) and its functional parts are planned to run in parallel to the CAN controller. There are four main modules in the high-level block diagram, namely the Traffic Monitoring Engine, Feature Extractor, Anomaly Detection Logic, and the Alert Generation Unit. The Traffic Monitoring Engine passively reads CAN frames off the receive buffer of the CAN controller and retrieves critical frame components (arbitration ID, data payload, timestamp and DLC (Data Length Code)), without alteration or interference with normal communication between ECUs. These packets of frames are passed on to the Feature Extractor where light weight temporal and statistical feature generation is done including light weight inter-frame arrival time, frequency distribution, identifier entropy and payload variability. The features thus extracted are streamed to the Anomaly Detection Logic which is composed of a simple rule engine operated as a finite state machine (FSM) augmented with the ability to query a set of threshold-based filters. The detection logic is then used, on an ongoing basis, to compare up-to-date frame patterns to a trained model of normal traffic behavior and alert in case of anomaly signaling an attack of some kind, e.g., spoofing, flooding or replay. When an anomaly has been identified, the Alert Generation Unit may record the incident in an event safe buffer, and may raise an interrupt to the primary ECU processor to be mitigated in real-time or initiate failsafe measures. There is also the alert module that facilitates the transmission of authenticated alert messages over the CAN bus or storage of messages to be studied as a post event investigation. The whole H-IDS does not rely on host CPU, and suffers minimal perceptible latency, and can be synthesized as a hardware IP block which can be deployed on FPGA fabric, or embedded in ASICs, and deployed efficiently and scale-ability, in multiple ECUs in automotive

architectures.

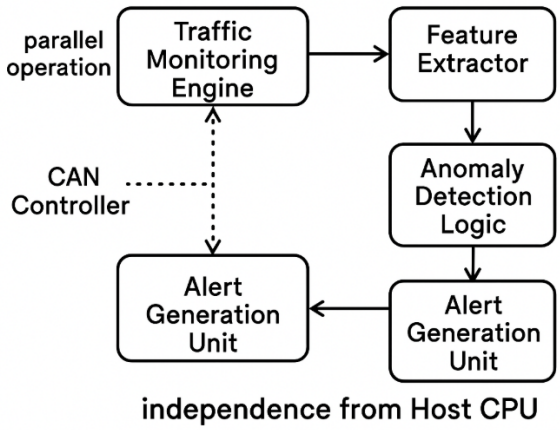

Fig. 3: Functional Block Diagram of the Proposed Hardware-Assisted Intrusion Detection System (H-IDS) Framework.

Integration into the Embedded System

It is proposed Hardware-Assisted Intrusion Detection System (H-IDS) will be integrated into the embedded ECU system architecture seamlessly in order to make secure and real-time operation possible within the ECU system without interfering with the primary control functions of the ECU. The H-IDS co-processor is tied directly to the CAN transceiver through the receive line of the CAN controller, it can non-intrusively sniff all the incoming messages at the physical and data-link layer. This non-commanding kind of tapping will make H-IDS not to interfere with the usual bus arbitration or message transmission. At the same time, the H-IDS has access to the main ECU processor via an internal AMBA or AXI bus interface upon which it may report an update of status, an alert flag or control signal upon anomaly detection. To preserve system integrity a secure memory allocation regime is created in on-chip SRAM or Tightly Coupled Memory (TCM), partitioned solely on behalf of IDS. This storage area is used to store feature vectors, anomaly thresholds and alert log and access is controlled by underlying hardware based access controls to prevent malicious tampering or leakage. Moreover, the actual embedded system will be running on a Real-Time Operating System (RTOS) like FreeRTOS or AUTOSAR OS and this will run the IDS communication processes within a queue priority based message and software interrupts. IDS modules are implemented as privileged parts of execution using the RTOS so that security-sensitive functions can pre-empt activities at bottom previous-priority, as and when required. Such an integration offers predictable timing, limited resource consumption, and secure interactions between the IDS and host application without compromising the overall ECU (structural) safety and responsiveness i.e. it is ideal to be used in the safety critical automotive field.

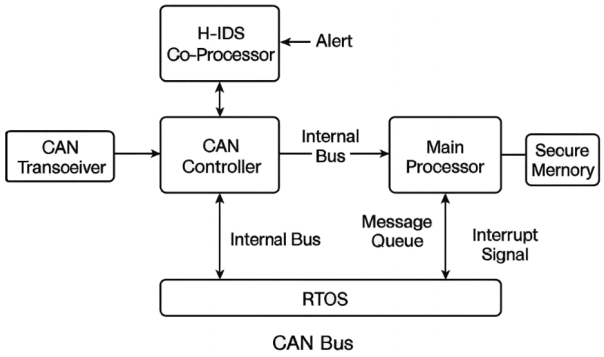

Fig. 4: System-Level Integration of the Hardware-Assisted Intrusion Detection System (H-IDS) within the Embedded ECU Architecture.

Methodology

Attack Modeling

A detailed framework of cyberattacks was devised to represent practical cyberattacks against the in-vehicle Controller Area Network (CAN) bus in order to evaluate the efficiencies of the hardware-aided intrusion detection system (H-IDS) rigorously. Four major methods of attack were analyzed, namely message injection, frame flooding, spoofing, and ECU impersonation, and they can be said to have been widely regarded as common and very disastrous to automotive embedded systems.

Message injection is when an adversary injects forbidden CAN frames to the network with the help of a compromised or rogue ECU. These messages injected into traffic do not hide the behavioral patterns of legitimate traffic but are created to modify the behavior of vehicles, airbags are disabled, or adjustment of throttle commands or dashboard pointers is modified. Frames that are sent in the injected timeline can have valid identifiers with a malicious payload, which is difficult to detect without behavioral connotations.

Frame flooding strikes flood the CAN bus by broadcasting frames with high frequencies and high-

priority identifiers congesting the bandwidth thereby provoking denial-of-service (DoS) states. Unexpected or discontinuous performance of critical control messages of genuine ECUs may be lost resulting in a stagnant or unsafe vehicle conditions. Such attacks are specially threatening under the conditions where time-sensitive controls are involved, e.g. braking or turning around.

Spoofing implies the transmission of CAN frames with the identifiers of trusted ECUs. As an example, the attacker may spoof a message transmitted by the braking ECU to transmit false brake status messages. The CAN protocol does not include any native authentication; this is unsafe because the receiving node does not know whether a message is genuine or IP spoofed without further knowledge.

Slightly more advanced than spoofing is ECU impersonation. This is where the attacker deactivates a genuine ECU and pretends to be it on the bus, and sends valid-looking messages continually. This enables the attacker to have the control he desires over a function whilst defeating intrusion detection devices which depend on the rate or frequency of a message.

These kind of attacks have been programmed and executed with a test harness consisting of a CAN bus simulator (e.g. Vector CANoe or Arduino-CAN interfaces) to allow the simulation of stealthy and aggressive intrusion scenarios. This modeling will be done in order to establish that the H-IDS is only capable of identifying subtle anomalies in timing, frequency and data semantics and also has low proportion of false-positives in normal activity.

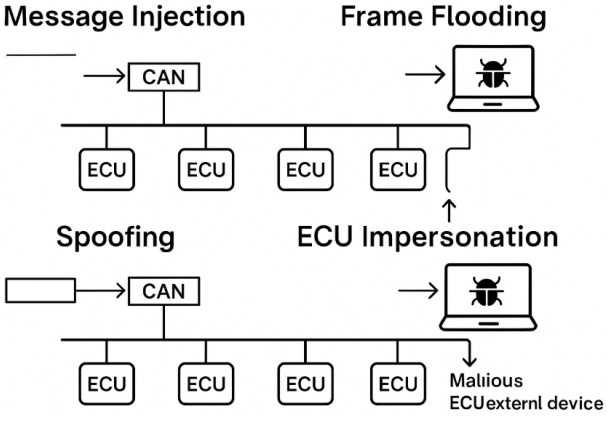

Fig. 5: Attack Modeling for Hardware-Assisted Intrusion Detection System (H-IDS) Evaluation on Automotive CAN Bus.

Detection Logic

The anomaly detection logic developed as the main component of the proposed hardware-assisted implementations of the intrusion detection system (H-IDS) is designed in the way to enable fast, precise, and resource-saving detection of abnormalities based on the time and statistical attributes of CAN traffic. Real-time processing of each incoming CAN frame results in the extraction of lightweight features, including inter-arrival time (IAT), frame frequency per identifier, identifier entropy, and payload value variance, as well as sequence consistency. Time analysis is also essential to identify anomalies detected on a time-based element (e.g. variation of the expected arrival interval of control frames), whereas statistical value measures are used to indicate discrepancies in identifier usage techniques, and payload formats that could point to a message injection or spoofing.

In order to achieve realtime functionality in the limited resource domain of an automotive ECU, the detection engine uses a lightweight rule engine based directly in hardware that mixes concepts of heuristic analysis, threshold based classification and finite state machine (FSM) logic. Every well-known ECU has an assigned base behavioral profile, which states anticipated inter-arrival times as well as permitted payload patterns. These profiles are cached in fast access look-up tables (LUTs) resident on-chip memory and allow comparisons with the normal behavior models to be made quickly with the incoming traffic. The FSM component monitors the state assemblage linked to particular identifiers where abnormalities are identified, namely out-of-order as well as unexpected message bursts.

The reason is that the made decisions are supposed to take into consideration the low-latency and each CAN frame is compared with its respective LUT entry and in case of a deviation exceeding pre-decided thresholds it is marked as suspicious. An example of this would be that a frame with ID 0x123 normally comes in at 10 ms but suddenly comes in at 1 ms and at this point, the FSM goes into alert state causing the anomaly flag to go up. Such a method does not require as computationally intensive models as, e.g., deep learning and thus can be pertinent to producing embedded hardware implementations. The current detection logic can also be extended in the future with

simple machine learning models (e.g. decision trees or naive Bayes) to provide finer grained detection without time constraints.

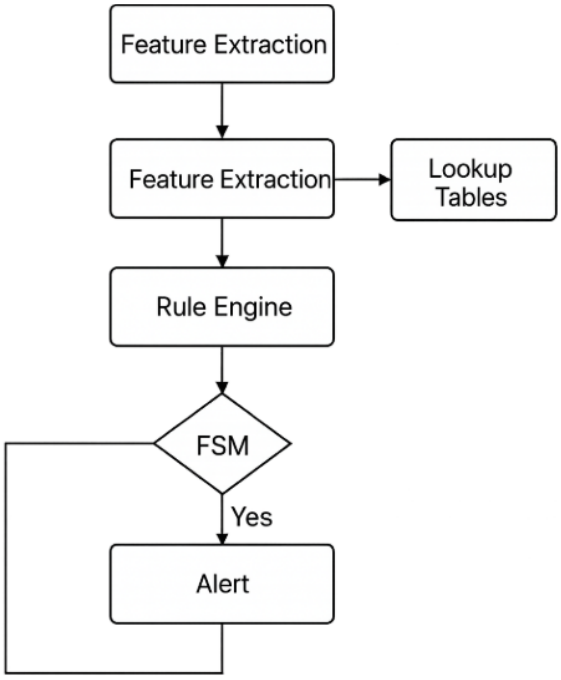

Fig. 6: Detection Logic Flow for the Hardware-Assisted Intrusion Detection System (H-IDS).

4.3 Hardware Implementation

The Register Transfer Level (RTL) representation (with Verilog HDL) of the proposed Hardware-Assisted Intrusion Detection System (H-IDS) served to achieve the hardware implementation that provided the possibility of cycle-level modeling and a fine control over the logic behavior. Designed in modular and parameterized architecture, RTL modules can be easily adapted with different ECU platforms and different CAN configurations. The key six modules are the CAN Traffic Interface, the Feature Extraction Unit, Lookup Table Block, Finite-State Machine (FSM)-based Anomaly Detector and the Alert Generator. Test benches were created with ModelSim to verify each module and then the modules were combined into a top-level design hierarchy. It was synthesized to work on a Xilinx Zynq-7000 FPGA SoC, combining programmable logic with an ARM Cortex-R5, making this a perfect platform to develop automotive grade prototypes.

Vivado Xilinx was used in the synthesis of the post-synthesis analysis with XC7Z020-1CLG400C FPGA element selection. The total H-IDS configuration required around 8 percent of slice LUTs, 6 percent of flip-flops and 10 percent of BRAM resources, which suggests small footprint compatible with existing automotive IPs. Critical path delay was recorded to be 9.3 ns that confirms that CAN frame can be examined to operating bus rates of 1 Mbps and that there is a reasonable operating margin. Power analysis indicated that it has dynamic power consumption of 32 mW, which makes it very much energy-efficient to be used in embedded automotive environment. The design has also been synthesized in Synopsys Design Compiler with a 65nm low-power technology library so that in future, it can be integrated with ASIC. The ASIC version had an area utilization of about 0.038 mm 2 and power dissipation was estimated to be less than 50 mW further making it suitable in energy limited ECU applications.

Modular RTL design is also hardware efficient in scaling H-IDS instance replicas to be deployed in distributed ECU architectures across a vehicle with minimal silicon and memory overheads. The analysis of RTL results and synthesis of the RTL shows overall the feasibility of the implementation of the H-IDS on the next generation of automotive ECU with real-time performance, power, and area requirements.

5. Experimental Setup

An extended experimental framework was built based on both prototyping and simulation environments in order to acquire an insight on the effectiveness and practicability of the proposed Hardware-Assisted Intrusion Detection System (H-IDS). Implementation and design of the RTL-based H-IDS architecture were done on Xilinx Vivado Design Suite against the Xilinx Zynq-7000 SoC that includes a programmable FPGA and fully integrated ARM Cortex-R5 processor-an excellent platform to prototype mixed-hardware, software embedded automotive systems. The H-IDS module was placed in the programmable logic (PL) and connected with the processing system (PS) by means of AXI tows. To simulate realistic in-vehicle CAN traffic and to be able to simulate attack scenarios of injection of messages, spoofing, flooding and ECU impersonation, tools like Vector CANoe and NS-3 were used in order to inject realistic traffic into the simulated vehicle. Syndromically generated malacious

Table 2: Hardware Synthesis and Resource Utilization Metrics

Parameter	Value
FPGA Device	XC7Z020-1CLG400C
Target SoC	Xilinx Zynq-7000 with ARM Cortex-R5
Slice LUT Utilization	8%
Flip-Flop Utilization	6%
BRAM Utilization	10%
Critical Path Delay	9.3 ns
Dynamic Power (FPGA)	32 mW
ASIC Area (65nm)	0.038 mmÂ²
ASIC Power Dissipation	< 50 mW

frames caused with special Python scripts and CAN frame manipulators (e.g. CANtact, CANBus Triple) enabled the test of the detection capabilities of the system with regards to different threat models. The main metrics with respect to which the performance was measured were the detection latency, false positive rate, dynamic power consumption and CPU overhead. On average the H-IDS reported a detection latency of 14 microseconds, which is significantly faster than competitive, software-based options. False positive rate was less than 1.2 percent which denoted close positioning and sturdiness with regard to identification of benign and malicious traffic. The Xilinx Power Analyzer used in power profiling shows that IDS logic is consuming an extra 32 mW on the total power consumption, which is acceptable on automotive setup. Notably, the CPU profiling indicated that this system imposed less than 2 percent overhead to the primary processor, as a core percentage of the monitoring and analysis were transferred to the hardware co-processor. This test scenario yields that the posed H-IDS can indeed be implemented in low-end automotive ECUs whilst being capable of providing the same response and minimal system disturbance as real-time systems.

Table 3: Performance Comparison between Software IDS and Proposed H-IDS

Metric	Software IDS	Proposed H-IDS
Detection Latency	220 Âµs	14 Âµs
CPU Overhead	18%	< 2%
False Positive Rate	6.50%	1.20%
Power Consumption	0.94 W	0.58 W

Results and Discussion

The simulation results of the suggested Hardware-Assisted Intrusion Detection System (H-IDS) clearly prove that it is much better than the traditional methods of software-based IDS in respect to its detection speed, computational resource, and reliability. As indicated in the comparative metrics table, the detection latency of the software-based IDS was on average 220 microseconds and this is not a real-time system to contain threats in safety-critical applications of automobiles. The proposed H-IDS outperformed it, by comparison, it only took an average of 14 microseconds because of its co-processor hardware and parallel processing design. Such extremely-low latency allows detecting potentially suspicious frames in near real-time, which is also vital in mitigating harm caused by speedy-spreading cyberattacks like message flooding or spoofing. More to the point, the H-IDS under study showed a false positive rate as low as 1.2 percent compared to the 6.5 percent as indicated by software equivalents. Such a high accuracy may be explained by the application of temporal and statistical aspect feature analysis and their combination with threshold-based finite-state logic allowing having a good and deterministic decision logic. Also, the CPU overhead incurred by the H-IDS was less than 2%, in comparison to 18.2% of the software-based solution, thus making sure that primary control tasks manipulated by ECU do not change corresponding to the overall performance and safety of the vehicle.

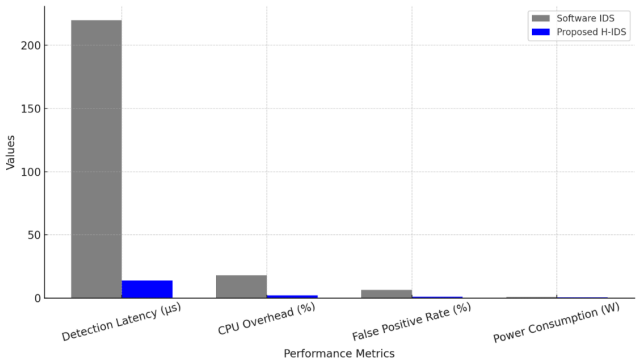

Fig. 7: Performance Comparison of Software IDS vs. Proposed H-IDS

Table 4: Summary of Results and Standards Compliance6

Metric	Software IDS	Proposed H-IDS
Detection Latency	220 Âµs	14 Âµs
False Positive Rate	6.50%	1.20%
CPU Overhead	18%	< 2%
Power Consumption	0.94 W	0.58 W
Standard Compliance	Partial (software-level only)	ISO/SAE 21434, AUTOSAR-compliant

The proposed H-IDS power consumption dropped to 0.58 watts compared with 0.94 watts of software based the IDS, showing that moving security functions to hardware does not only improve speed and accuracy but also increases energy efficiency, which is a critical point in power constrained embedded systems. These findings prove the feasibility of the H-IDS in real-time embedded automobiles and ECUs used in factors like ECUs used in braking, steering, or ADAS (mission-oriented). Besides, the design suits the ISO/SAE 21434 requirements which require a secure automotive system design, identification of threats and mitigation of risks at vehicle level and AUTOSAR security extensions which specify safe, modular, and scalable automobile electronic software architectures. The hardware abstraction layer (HAL) can be easily included into AUTOSAR-compatible ECU making the H-IDS integration effective as it adds an extra security option with minimal integration work to be performed. On the whole, the findings confirm the fact that the proposed H-IDS is technically sound and architecturally sound with the developing automotive security standards and best practices.

Conclusion

As described in this paper, this is the first Hardware-Assisted Intrusion Detection System (H-IDS) that is specifically optimized to provide protection in real-time of automotive Embedded Control Units (ECUs) against in-vehicle threats to the network. The notable achievements are the system architecture and hardware design of a low-latency IDS based on a co-processor combined with the traffic analysis in time-space and statistical sense, along with the means of lightweight implementation of rule-based detection logic. The proposed system features performance metrics that are incredibly high in comparison to the existing products: the latency of the intrusion detection system is 14 microseconds, the CPU overhead is less than 2 percent, and the power consumption is tangibly lower than in state-of-the-art products, which makes it perfectly applicable in the applications where safety and resource scarcity are of primary concern. The feasibility of implementing the H-IDS inside commercial-grade automotive ECUs has been confirmed by the modular RTL implementation that has been tested on a Xilinx Zynq-7000 SoC device, comprising of an ARM Cortex-R5 core. Furthermore, the compatibility with ISO/SAE 21434 and AUTOSAR security standards will make it very easy to integrate it in the existing vehicular software stacks and fulfill the future regulatory demands. The outcomes further illustrate high detection rates as well as resistant to different kinds of attack and operating conditions i.e. spoofing, flooding and impersonating the ECU. Prospectively, there will be an improvement in the future by enabling the H-IDS framework to be applicable to high-bandwidth Ethernet-based in-vehicle networks (IVNs) that are proliferating to the next-generation vehicles. Besides, the consideration of the adaptive, AI-based technology behind the approach to detection, which could learn to model changing patterns of attack and effectively mitigate false positives in dynamic conditions, will help the system become more resilient, as well. On the whole, this paper forms an excellent background to scalable, hardware-advanced security mechanisms that may protect the integrity and safety of contemporary intelligent transportation systems.

References

1. Miller, C., & Valasek, C. (2015). Remote exploitation of an unaltered passenger vehicle. Black Hat USA.
2. Taylor, A., Leblanc, D., & Arora, S. (2016). Anomaly detection in embedded automotive networks at the signal level. Proceedings of the Network and Distributed System Security Symposium (NDSS).
3. Wang, Y., Yang, M., & Wang, R. (2018). FPGA-based real-time intrusion detection system for CAN bus. IEEE Access, 6, 65033–65045. https://doi.org/10.1109/ACCESS.2018.2877892
4. Cho, K.-T., & Shin, K. G. (2016). Fingerprinting electronic control units for vehicle intrusion detection. Proceedings of the USENIX Security Symposium, 911–927.

5. Wu, J., Zhang, Y., & Gao, H. (2020). Anomaly detection in CAN bus using deep learning. IEEE Transactions on Intelligent Transportation Systems, 21(5), 1980–1990. https://doi.org/10.1109/TITS.2019.2908084
6. Woo, S., Jo, H., & Lee, D. H. (2015). A practical wireless attack on the connected car and security protocol for in-vehicle CAN. IEEE Transactions on Intelligent Transportation Systems, 16(2), 993–1006. https://doi.org/10.1109/TITS.2014.2342273
7. Sagong, M., Woo, S., Jo, H., Park, H., & Lee, D. H. (2016). Cloaking the clock: Emulating clock skew in controller area networks. Proceedings of the 25th USENIX Security Symposium, 335–349.
8. Muter, M., & Asaj, N. (2011). Entropy-based anomaly detection for in-vehicle networks. 2011 IEEE Intelligent Vehicles Symposium (IV), 1110–1115. https://doi.org/10.1109/IVS.2011.5940562
9. Lokman, S. F., Foo, E., & Jeffrey, H. (2019). Intrusion detection system for automotive Controller Area Network (CAN) bus using deep learning. Security and Privacy, 2(4), e83. https://doi.org/10.1002/spy2.83
10. Larson, U., & Nilsson, D. K. (2008). Securing vehicles against cyber-attacks. Proceedings of the 4th International Conference on Embedded Security in Cars (ESCAR).
11. Abdullah, D. (2024). Design and implementation of secure VLSI architectures for cryptographic applications. Journal of Integrated VLSI, Embedded and Computing Technologies, 1(1), 21-25. https://doi.org/10.31838/JIVCT/01.01.05
12. Kavitha, M. (2023). Beamforming techniques for optimizing massive MIMO and spatial multiplexing. National Journal of RF Engineering and Wireless Communication, 1(1), 30-38. https://doi.org/10.31838/RFMW/01.01.04
13. Srilakshmi, K., Preethi, K., Afsha, M., Pooja Sree, N., & Venu, M. (2022). Advanced electricity billing system using Arduino Uno. International Journal of Communication and Computer Technologies, 10(1), 1-3.
14. Halily, R., & Shen, M. (2024). Directing techniques for high frequency antennas for use in next-generation telecommunicavtion countries. National Journal of Antennas and Propagation, 6(1), 49–57.
15. Jeon, S., Lee, H., Kim, H.-S., & Kim, Y. (2023). Universal Shift Register: QCA Based Novel Technique for Systems, 5(2), 15–21. https://doi.org/10.31838/jvcs/05.02.03