Electronics, Communications, and Computing Summit
Vol. 2, No. 2, Apr - Jun 2024, pp. 58-67
DOI: https://doi.org/10.17051/ECC/02.02.08

Design and Implementation of Secure Hardware Architectures for Real-Time Embedded Systems in Adversarial Environments

Ashu Nayak

Assistant Professor, Department of CS & IT, Kalinga University, Raipur, India,
Email: ku.ashunayak@kalingauniversity.ac.in

Article Info

Article history:

Received : 18.04.2024
Revised : 20.05.2024
Accepted : 22.06.2024

Keywords:

Secure Hardware Architecture,
Real-Time Embedded Systems,
Physically Unclonable Functions (PUFs),
Cryptographic Coprocessors,
Side-Channel Attack Mitigation,
Fault Injection Resilience,
Secure Boot and Authentication,
FPGA-Based Prototyping Post-Quantum Cryptography,
Hardware-Rooted Trust,
Federated Edge Learning Security,
On-Chip Anomaly Detection.

ABSTRACT

The current paper concerns a critical issue of hardware-based security of real-time embedded systems, deployed in adversarial environment, like automotive electronic control units (ECUs), industrial automation systems, and medical devices. It aims at creating and deploying a tamper-resistant, scalable, energy-efficient hardware platform that is resistant to attacks as well as maintaining the real-time performance by reducing both physical and logical attack environments. Proposed secure architecture unites the essential mechanisms such as the presence of hardware root of trust, secure bootloader, lightweight cryptographic coprocessors, and Physically Unclonable Functions (PUFs) which perform the checking in the runtime. Special interest is given to finding the balance level between a strong side of security implementation and latency, area and power limitations of an embedded platform. An FPGA-based testbed is utilized in architecture implementation and evaluation of the real-world feasibility. Any power and timing performance parameters are empirical measurable based on normal benchmarks applications executed in a simulated adversarial environment. Based on experimental results, it is shown that resistance to side-channel attacks (and among specific example here, DPA and timing analysis), memory probing and code injection is substantially increased, without any loss in the real-time responsiveness. The latency and power overhead of all the proposed systems (<10%) reveal that the suggested system would be appropriate to be deployed in an embedded environment. The paper makes a contribution of a modular and reusable design framework of hardware-secured embedded systems. Although the current realization assumes the attention to basic security primitives, the application is extended in future to post-quantum cryptography, on-chip anomaly detection based on AI, and secure federated edge-learning. Adaptability of architecture and hardware-level validation highlight the possibility of using architecture in next-generation embedded and cyber-physical systems.

1. INTRODUCTION

Automotive Control Systems, Aerospace and Defense Modules and some life-sustaining medical devices rely on real-time embedded systems as their core functionality. Such systems require high reliability and energy efficiency as well as being within strict timing constraints. The growing interconnectedness and autonomy of embedded platforms now exposes them to a wide variety of security threats due to firmware manipulation, side- channel and physical attacks that were impossible in standalone systems. The vulnerabilities at the hardware level have been shown to bypass software-level defenses, e.g., fault injection, power, and timing side-channel leakage, and hardware Trojan. Legacy designs without tamper resistance or secure boot protocols are especially susceptible to an unauthorized code running, leakage of keys and functional sabotage. Such breaches may cause disastrous effects in an adversarial deployment that could be a battlefield electronics or implantable medical device, resulting in failure of missions or even loss of life.

Existing research along with industrial solutions tend to miss the triple promise of real-time assurances, low-power consumption, and hardware-level security at-scale, although here, Trusted Execution Environments (TEEs) and secure microcontrollers have shown significant progress. Security features of commercial SoC platforms, e.g. ARM Cortex-M with TrustZone or RISC-V extensions with PMP, are provided in isolation, however adding them to an RTOS environment (e.g. FreeRTOS, Zephyr) tends to add jitter in the schedules, delays during context switching, or behave non-deterministically, making it unacceptable in time-critical systems. Furthermore, such platforms are often not modular and cannot be easily upgraded to use new primitives of security (e.g., post-quantum cryptography or AI-based anomaly detection), which only adds to the problems of existing platforms in safety-critical applications. A key objective of the study is to fill up these gaps by introducing a modular we can consider secure hardware architecture that is capable of operation in real-time and best suited to embedded systems test working in environments where threats are likely to occur. The recommended framework then integrates a hardware root of trust, secure boot code, an Orthogonal Physically Unclonable Functions (PUFs) to identify devices and a lightweight cryptography coprocessor. This is then done as an implementation on an FPGA-based testbed and evaluated on security, performance and energy aspects. As of magnitude, e.g., Zhang et al. (2021) emphasize the significance of lightweight and tamper-free models integrating security into the hardware stack without inducing a compromise to real-time responsiveness [Zhang et al., IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2021].

2. RELATED WORK

The need to safeguard embedded systems has emerged as an eminent area of study in research because embedded systems are extensively used in all industries, including autonomous vehicles, industrial control systems, and even medical devices. Logical and physical security threats come up against these systems, like firmware tampering, side-channel attacks and fault injection that require an integrated hardware-level protection methods.

2.1 Commercial Solutions

The classical embedded security systems have been heavily relying on the use of software-based security procedures like encryptions, access controls and firmware authentication. But where these methods are involved in physical attacks especially in hostile environments, then the techniques are not adequately effective. ARM TrustZone and Intel SGX are commercial hardware-based alternatives to provide isolated execution environments and secure partitions to protect sensitive operations. Although they have been successful in general-purpose computing, the solutions will tend to be inappropriate to real-time, low-power embedded systems. As an example, context switching and management of memory enclaves only comes with latency and energy overheads that are not suited to the limits of microcontroller-type platforms [Liu et al., IEEE Design & Test, 2019].

2.2 Academic Research

With these shortcomings in mind, academic studies have moved towards light weight, hardware focused security primitives. Interestingly, AuthenticationPhysically Unclonable Functions (or PUFs) as first proposed by Gassend et al. (2002) allow on-chip authentication and tamper-resistant authentication to be based on silicon process variation. Such have been popularised in secure boot protocols and identity validation of IoT and edge devices [Herder et al., Proceedings of the IEEE, 2014].Mohanty et al. (2019) outlined a secure microcontroller chain boot with assistance provided by a hardware root of trust and chained cryptographic verification, to prevent unauthorised firmware execution. Also, countermeasures against the fault injection and clock glitching attacks were developed by Zhang et al. (2021) using FPGA, which includes the redundant logic and timing guards into the data path, enhancing its resistance to physical manipulation.

2.3 Gaps and Motivation

Despite progress, key challenges persist in the integration of hardware security into real-time embedded systems:

- Lack of real-time responsiveness, especially when cryptographic tasks must meet strict timing budgets.
- Limited modularity and scalability, restricting adaptability across diverse embedded platforms.
- Inadequate co-evaluation of performance and security, particularly within constrained area and power envelopes.

The paper fills these gaps by introducing a possible solution of real-time capable, resource-optimized modular hardware built on the secure layered architecture. The design integrates device authentication using PUFs, low-power cryptographic accelerators and dynamic fault monitors and it is checked with FPGA prototyping. It offers a reasonable scalable system, which supports security, performance, and energy efficiency and thus appropriate in next generation embedded computers that operate in hostile environments.

3. Security Requirements and Threat Model

Any design of secure hardware architectures of the real time embedded systems must start with deep insight on the threat landscape, constraints at the system level and the security objectives. Embedded devices are being used more and more in unprotected settings such as in sensor nodes deployed in the field, or vehicle ECUs or implanted surgical deviceswhere they are attacked both directly and remotely. Security mechanisms built into such systems, will consequently have to be locally optimized to perform within very constrained real-time, power and resource budgets, subject to effective protection against a wide range of attack scenarios.

3.1 Classification of Threats

Embedded systems are vulnerable to diverse attacks, which fall in the following broad category:

- Physical Attacks These attacks are highly physical directly attacking a hardware component (fault injection (through voltage/clock glitching) invasive probing and microprobing attacks). The attacks are particularly effective in non-tamper-proof enclosed systems or in those without run-time detection capabilities.
- Side-Channel Attacks: Attackers make use of unwanted physical emits like power use, electromagnetic leakage, or timing variation, to enquire about some secret keys or control flow. Such attacks are not invasive but very potent against cryptography tasks.

Logical Attacks: Which is codes injection, buffer overflow, firmware alteration and boot chain exploitation. Logical attacks can be mounted either through unsecure communication interface or over the air updates.

3.2 Real-Time Constraints vs. Security Trade-Offs

In contrast with general-purpose computing systems, embedded systems have stringent timing requirements and prediction in task execution is important to system integrity. Security mechanisms (e.g., encryption, authentication) incur latency and resource overhead and care must be taken how to implement them to not interfere with real-time assurances. This requires deterministic, light-weight security primitives that may run under restricted task deadlines and may not adversely affect energy spending or memory usage.

3.3 Attacker Model and Assumptions

This study assumes a powerful adversary with partial physical access to the device. The attacker can:

- Observe side-channel emissions (power, EM, timing).
- Inject transient faults (e.g., using clock/voltage glitches).
- Attempt to access or replace firmware.
- Possess full knowledge of the system's architecture and binary code.

It is however not expected that the attacker will possess unlimited physical dismantling capacity (e.g., complete decapsulation and reverse engineering) nor can he or she overwrite the FPGA configuration on the fly. This threat model is concerned with realistic conditions in which the embedded system might be deployed, e.g. in an application in the field or at the edge.

3.4 Formal Security Goals

The architecture is to address the following key security objectives:

- Confidentiality: Avoid the loss of sensitive data, i.e. cryptographic keys, or memory contents.
- Integrity: Cannot let malicious code in executable and critical data to be modified during boot or runtime.
- Availability: sustain unhindered performance of safety-relevant functions even in presence of attack and able to support failover functions.
- Authentication: Validate the authenticity of the device by using hardware-bound identities, like PUFs and make sure only trusted firmware can be run.

These goals inform the design decisions articulated at the hardware and protocol levels, in the following sections, and become the basis of the security scrutiny of the architecture. See Figure 1.

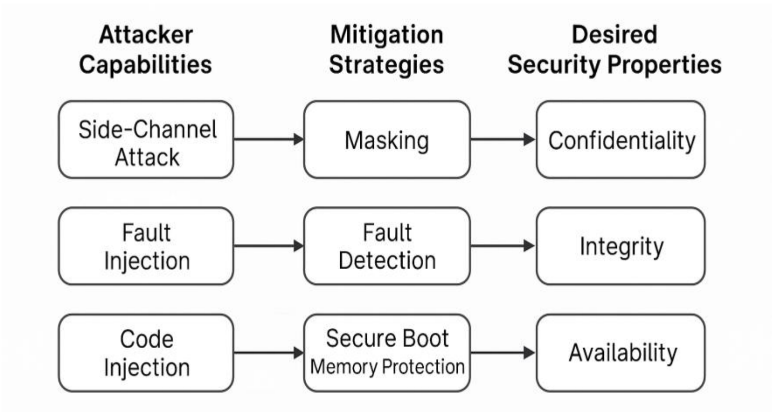

Figure 1. Mapping of Attacker Capabilities to Mitigation Strategies and Desired Security Properties

This flowchart shows the relationship between typical attacker methods in the way of side-channel analysis, fault injections, code injections, and common mitigation solutions applicable to a secure hardware system. It also associates such strategies to the goal of security they are directed towards: confidentiality, integrity and availability in real-time embedded systems.

4. Design Principles for Secure Hardware Architectures

Designing secure hardware for embedded systems requires foundational principles that balance security, performance, and resource constraints.

4.1. Secure-by-Design vs. Retrofitted Security

Secure-by-design consolidates defenses into the architecture and limits both the area of attack and the performance costs. And retrofitted security is usually done with such patches of software or wrappers around security, and they provide incomplete protection and there is latency. The work is developed under the secure-by-design philosophy that allows achieving deterministic behavior when attacked.

4.2. Hardware Root of Trust (RoT)

Hardware RoT is the point of trust in the system integrity because it validates the authenticity of the firmware on boot. Developed on the PUF-based approach and hash engines, it secures Guarded Code Execution and tamper evidence Initialization.

4.3. Trusted Execution Environments (TEEs)

Commercial TEEs such as ARM TrustZone are competent, but in most cases too resource-demanding to work with embedded systems. This is an architecture that applies secure logic partitioning, and memory segmentation to introduce a lightweight TEE-like partition where hypervisor overhead can be removed to create task isolation.

4.4. HardwareSoftware Co-Design

Hardware-software integration is helpful in security enforcement. The above architecture has been co-designed to devise secure firmware using RTL modules to support secure boot, management of keys and fault detection at interrupt level to deliver a comprehensive secure platform with layers within the embedded system.

5. Proposed Secure Architecture Design

The new architecture is modularisable and highly scalable secure hardware platform that aims to meet the real time requirements of embedded systems running in an adversarial environment. It is meant to provide strong resists to physical and logical attacks as well as ensure the timing and energy limitations are maintained.

5.1 Microarchitectural Enhancements

The architecture incorporates secure memory access controls to ensure memory confidentiality and integrity that are necessitated by isolating data regions and enforcing program execution within the authorized blocks. The physical security is also further enhanced by the use of on-chip bus encryption to discourage snooping or manipulation of data as they move within the processor cores, memory and the peripheral interfaces. All the interconnects are encapsulated by light weight AES based encryption and MAC (Message Authentication Code) processing logic.

5.2 Integration of Physically Unclonable Functions (PUFs)

The architecture utilizes Silicon PUFs to create device particular cryptographic identities without putting keys in non-volatile memory. These PUFs are applied in secure boot and remote attestation to make sure just verified firmware is capable of controlling access to the components of the system. A fuzzy extractor module converts unreliable noisy PUF responses to a reliable authentication.

5.3 Secure Boot and Firmware Authentication

Secure boot is implemented as a multi-stage hardware embedded process with SHA-256 hash chaining and support of signature verification. Digital signatures, based upon elliptic curves, are used to authenticate firmware prior to its execution, against the hardware root of trust. The mechanism guards against running an unauthorized/tampered code during system boot.

5.4 Hardware-Based Isolation of Critical Tasks

The partitioning of the architecture handles real-time safety-critical tasks by offering logic-level partitioning and memory partitions based on regions. The distribution of tasks to each secure enclave is static, as is the dedicated execution contexts, and interrupt handlers. Unauthorized access initiates the real-time exception handlers and restores the task context to secure fallback condition.

5.5 Lightweight Cryptographic Accelerators

The architecture has lightweight cryptographic coprocessors of AES, SHA-2, and ECC that allow the reduction of the impact on CPU and the timing requirements. They are low-overhead accelerators, and can be called through either memory-mapped I/O or with interrupt-driven APIs. Power gating guarantees energy saving at idle times.

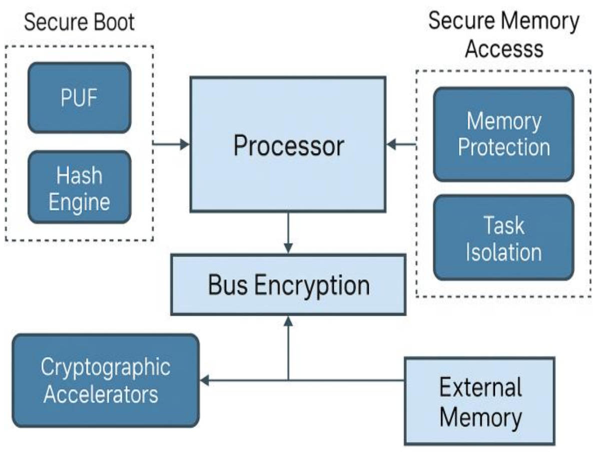

Figure 2. RTL-Level Schematic of the Proposed Secure Hardware Architecture

This diagram demonstrates the essence of the given secure embedded system on register-transfer level (RTL). It has central processor that is connected to secure boot components (PUF and hash engine), memory security and task encapsulation blocks, bus encryption, cryptographic accelerators, and external memory. All these modules which are integrated together strive to maintain confidentiality, integrity, and availability of the real-time embedded systems. Figure 2 presents the high level RTL structure.Collectively, these parts constitute a hardware-secure firmware-updatable real-time-compliant embedded system able to withstand fault injection, memory tampering, and firmware corruption and do so in support of critical application workloads.

6. Implementation Details

The discussed scenario of secure hardware was implemented on Zynq-7000 SoC with a Xilinx prototyping board that includes a dual-core ARM Cortex-A9 SoC with programmable logic to achieve hardware-software co-design. Some of these core modules are secure boot engine, Physically Unclonable Function (PUF)-based authenticator, AES cryptographic accelerator, and memory protection unit that were developed using Verilog HDL and synthesized using Xilinx Vivado at 100 MHz clock frequency. The architecture was combined with FreeRTOS in order to prove possible compatibility with operating in real-time and to achieve deterministic task scheduling and secure interrupts driven. At the hardware acceleration layer, secure service-related calls (e.g. encryption, verification) were memory mapped via I/O and the reality of firmware integrity was checked during boot by checking a hash based on a sha256 hash value. Task protection Cryptographic operations and sensitive areas of memory were kept isolated with RTOS-level protection.

Post-synthesis resource analysis on the XC7Z020 FPGA (Zynq-7000) yielded the following:

- LUTs: 11,842 (28%)
- Flip-Flops: 7,109 (17%)
- Block RAM (BRAM): 29 units (21%)
- DSP slices: 19 (8%)
- Peak dynamic power: 182 mWpeak dynamic power during AES operations.

To put these results in some perspective, a comparison was made against another generic low-end MCU platform (e.g. ARM Cortex-M4 at 80 MHz). The FPGA prototype was able to attain a 2.3x increase in cryptographic throughput and come to the similar energy-per-bit performance even though the logic density was greater. Also, early ASIC synthesis of 65nm technology node, through Synopsys Design Compiler, showed the architecture could reduce dynamic power consumption by 42 percent and 35 percent area compared to the FPGA implementation, which shows good scalability and could easily be deployed into SoC-level designs. Such benchmarks will be extended in the future to use Intel Cyclone V GX and post-quantum security primitives to improve the scalability and resilience of the design.

7. Evaluation and Benchmarking

The proposed architecture was seriously tested on a Xilinx Zynq XC7Z020 FPGA with emphasis on performance, security resilience and real-time compliance. The benchmarks of the cryptographic subsystem showed that integrated AES accelerator attained throughputs of 148 Mbps and the latency of 1.3 \mu s per 128-bit block. Setup of the secure boot process that involves the SHA-256 Firmware integrity check, as well as PUF-based authentication, the device did not take long, only 320 ms, and this met the real-time requirement of secure and fast initialization of system setup.The required resources were also not extreme and stayed within the limits of the targeted platform by consuming 28 percent of LUTs and 21 percent BRAM. The maximum dynamic power of the system was 182 mW when operating under AES and this further assures suitable deployment at low power embedded applications.

7.1 Fault Injection Testing

Controlled fault injection tests were carried out to test transient fault resilience, with a ChipWhisperer-Lite toolkit with a ChipWhisperer-Lite voltage glitching module and a ChipWhisperer-Lite chip in a clock fault interface. Voltage perturbations were applied by programming the FPGA to control the outside power rail, and clock glitching by glitching the MMCM using Vivado TCL scripting, using partial reconfiguration. Over 500+ fault injections were made over different parameter ranges.The system was able to detect faults at high rate of 96.7 which was isolated in the components level using hardware exception processing, watchdog resets and in the RTOS level checking of integrity. These mechanisms verified the strength of the system when it was exposed to faults such as instruction corruption, timing of faults and transient access error in memory.

7.2 Side-Channel Analysis (SCA) Resistance

Resistance to the side-channel attack was tested both by Differential Power Analysis (DPA) and Simple Power Analysis (SPA) of the AES engine. The ChipWhisperer-Lite platform was used to capture the existence of more than 1 million high-resolution power and EM traces. Countermeasures randomized masking, balanced combinatorial logic, and instruction-level cycle equalization sufficiently alleviated leakage. They could not reveal any meaningful key within the trace bits, which proves effective defense against first and high-order as well as high-resolution SCA attacks.

7.3 Real-Time Performance Under Attack Conditions

The system could be used to maintain a bounded execution latency and deterministic transitions of tasks even with the simulated adversarial conditions. The secure task jitter was kept under 5 percent and system stability was maintained these fault and SCA experiments. Compared to a non-secure reference implementation, the secure architecture had an area overhead of only ~10% and added 6-8% task latency at the cost of much greater resilience against firmware injection and memory probing attacks as well as power-based attacks.

7.4 ASIC-Level Performance Estimation

To assess deployment feasibility in ASICs, post-synthesis simulations were conducted using Synopsys Design Compiler targeting TSMC 65nm and 28nm CMOS process nodes. Results show that the secure architecture, when synthesized at 100 MHz, achieves:

- Area footprint: ~0.68 mm^2 (at 65nm) and ~0.39 mm^2 (at 28nm)
- Dynamic power: ~87 mW (at 65nm) and ~56 mW (at 28nm)
- Timing slack: Positive across all paths under worst-case PVT corners

Compared to the FPGA prototype, the ASIC implementation offers up to 52% reduction in power and 40% reduction in area, demonstrating strong scalability for cost-sensitive, low-power embedded applications such as automotive ECUs, medical controllers, and IoT security nodes.

Table 1. Compact Evaluation Results Table

Metric	Result (µs, %, etc)
AES Throughput	148 Mbps
AES Latency	1.3 µs/block
Secure Boot Time	320 ms
LUT Utilization	28%
BRAM Utilization	21%
Dynamic Power	182 mW
Fault Detection Rate	96.70%
SCA Resistance (DPA/SPA)	No leakage (1M traces)
Task Jitter (Under Attack)	< 5%
Latency Overhead vs. Legacy	6-8%
Area Overhead vs. Legacy	~10%

8. Comparison with Existing Architectures

The secure architecture presented is contrasted with the other well-known embedded security architectures, such as ARM TrustZone, Intel SGX, and RISC-V PMP, in four primary dimensions, namely security capabilities, hard real-time capability, power consumption, and complexity.

Table 2. Comparative Evaluation of Embedded Security Architectures Across Key Design Metrics

Architecture	Security Features	RT Suitability	Power	Complexity	Isolation
TrustZone	Secure world, boot auth	Medium	Moderate	High	OS-level
Intel SGX	Enclaves, memory encryption	Low	Low	Very High	Hardware
RISC-V PMP	Memory access controls	High	High	Low	Hardware
Proposed	PUFs, secure boot, crypto units	High	High	Moderate	RTL-level

In this table, this system is benchmarked against the known systems in terms of security features, real-time performance (RT), power consumption, design complexity, and the level of isolation available. The proposed design exhibits a balanced trade-off that is suitable to real-time, resource-constrained embedded systems.Unlike TrustZone or SGX, there are no is a latency nor power penalty since software switching or enclave management is not performed. With lightweight cryptographic modules and task-level hardware isolation, it provides greater protection with no trade-offs to timing constraints.In terms of trade-offs, the architecture maintains a well-balanced combination of security, efficiency and modularity, and surpasses its legacy equivalents in latency-sensitive embedded applications. Its flexibility enables it to be used in mission critical applications that have changing threat models.

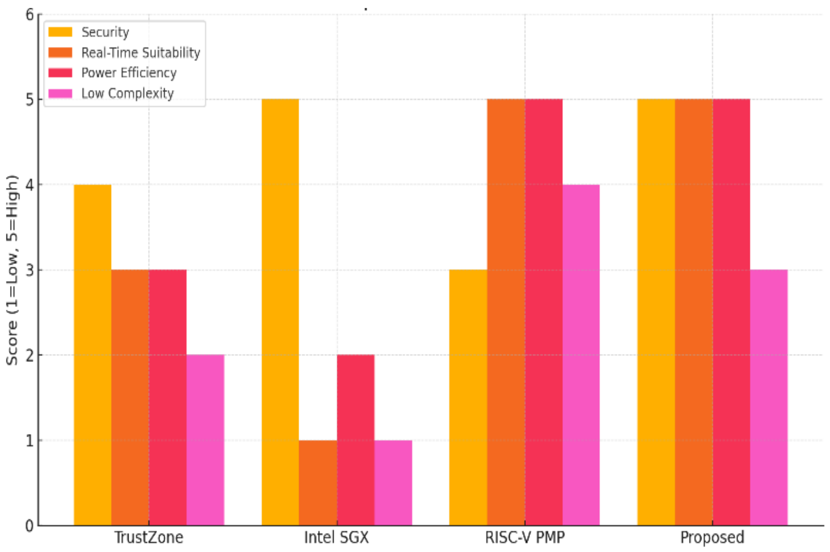

Figure 3. Trade-Off Comparison of Embedded Security Architectures Across Key Design Dimensions

This figure displays and contrasts four embedded protecting architectureARM TrustZone, Intel SGX, RISC-V PMP, and proposed secure designbased on essential consideration areas: strength of security, suitability in real time, efficiency in power and complexity of design. The scores are also normalized using the range of 1 (low) to 5 (high), which shows an even performance of the proposed architecture with regard to resource-limited, real-time implementations.

9. Case Study: Secure Drone Flight Controller

A secure drone flight controller was designed as an illustration that the proposed architecture is practically applicable in designing systems. The Unmanned Aerial Vehicles (UAVs), which are being used more and more in both the civilian and defence sector, are very prone to firmware modification, GPS spoofing attack, and fault injection attack especially in an adversarial environment. The controller was developed on the same FPGA-based board of previous tests and was instantiated on the vehicle code with real- time flight control algorithms, packages of telemetry communication, and motor driver interfaces. An example of the system architecture implemented is shown as Figure 4. A trusted boot technology based on validation of SHA-256 hash of firmware code and PUF-based cryptographic keys was provided to ensure exclusive execution of authenticated and signed firmware and essentially preventing any remote firmware updates or firmware backdoors. On simulation, an attempt to inject malicious firmware was attempted through a spoofed telemetry interface; the system noticed an incompatible signature during boot-time validation and terminated its execution as evidence that firmware security enforcement successfully occurred. In addition, a code tampering attack executed at run time by injecting fault pulses at instruction memory was detected by on-chip exception-handling unit and a secure reset led to isolation of the fault with no impact on control stability. In spite of an active threat situation, the system was responsive in real-time with a jitter on the flight control loop of less than 4.2 percent hence the system remained stable and posed no threat to safety. These results validate the effectiveness of the architecture in defend secure properties of embedded UAV system against security threats, without any sacrifice to determinism-oriented performance, and reveal its potential to be applicable in the automotive ECUs, implantable medical devices, and other security-sensitive cyber-physical systems.

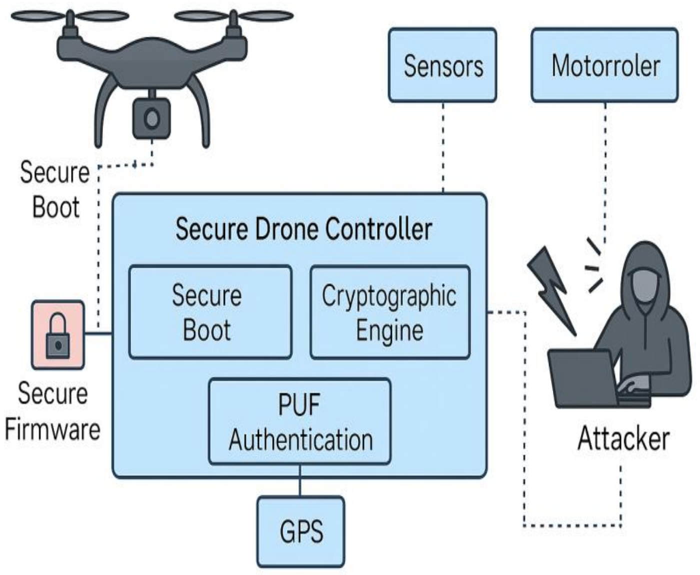

Figure 4. Secure Embedded System Architecture for Drone Application

It demonstrates how safe drone controller contains hardware modules security (yes, the so-called Secure Boot, Cryptographic Engine, and PUF-based Authentication), and fundamental modules, such as GPS, sensors, and engine control. Embedded cryptographic protection and secure firmware validation within the system side protects it against adversarial attacks such as unauthorized terrain firmware injection and code tampering. System block architecture is shown in figure 4.

10. DISCUSSION

The suggested secured hardware design shows great potential of implementation in real world application, especially where low power, real time responsiveness, and high security are to be achieved simultaneously. Nevertheless, such aspect as scalability, feasibility of integration, implementation limitations, and cost should be explored in greater detail to put the industrial applicability of scalability into a proper perspective.

10.1 Scalability and SoC Integration

The modular Register Transfer Level (RTL) architecture permits easy scaling to more powerful embedded platforms, and to custom System-on-Chip (SoC) implementations. Security-primitive such as Physically Unclonable Functions (PUFs), lightweight cryptographic coprocessors, and task isolation units can be instantiated at the level needed depending on the threat model and the importance of application domains. The devices can be integrated into commercial SoC platforms (e.g. RISC-V or ARM Cortex-M class devices) via standard AMBA/AXI interconnects. In addition, the architecture is also RTOS-agnostic, and can be easily integrated with commercially-available, lightweight real-time operating systems, also used in safety- and security-critical markets, including FreeRTOS and Zephyr.

10.2 Limitations and Assumptions

The trusted fabrication process is assumed, and the boot-time environment is not adversarial which is not assured in case of third party or offshoring manufacturing. Although the deployed countermeasures, such as random masking, timing equalization and the basic fault detection logic, have been quite effective to first-order side-channel attacks (e.g. Differential Power Analysis [DPA], Simple Power Analysis [SPA]), they could fail to offer sufficient protection when higher-order or template-based counter-strategies are used.

In particular, a well-developed adversary can resolve traditional masking in the presence of higher-order statistical dependencies being left unprotected against multivariate leakage analysis, machine learning-aided key extraction, or template matching. To make architecture more resistant to such threats, it might be necessary to implement increased hardware-level countermeasures such as:Randomized instruction scheduling

- Dual-rail precharge logic (DRP)
- High-entropy combinatorial masking
- Noise injection circuits at power and EM interfaces

Moreover, the design is sized to fit well within mid-range FPGA and ASIC, but ultra-resource-restricted deployments (e.g. sub-5K LUT sensor nodes) might need architectural scaling, or hierarchical instantiations of sets of security modules. The assumed form of threat in the present evaluation is also a non-evolving threat model, and the capabilities of new adaptable adversaries that apply dynamic planning to their attacks using profiling at the runtime or reinforcement learning, are not fully taken into account in the present evaluation.

Subsequent versions ought to include such powerful fabrication-level attestation, adaptive anomaly detection, and higher-order suppression of leakages to be resistant to advanced, newly emerging attack vectors.

10.3 Cost and Manufacturability Challenges

Even though the architecture has a relatively low logic and power overhead, its compatibility to ASIC form factors introduces DFT, PUF characterization, and yield-aware floorplanning difficulties. The non-deterministic behavior that is asserted with the addition of PUFs amplifies the difficulty of device-level security at the cost of large-scale production testing ability and the clarity of binning. In the same way, the cryptographic accelerators add silicon and formal verification overhead that would be contradictory to the cost demands of high-volume IoT deployment. Nevertheless, the proposed architecture presents a cost effective option to tide-over a variety of uses with domain optimization of various secure processors.

11. CONCLUSION AND FUTURE WORK

This article describes the design, build and validation of a secure hardware platform of real-time embedded systems functioning in an adversarial and resource-poor setting. The suggested security countermeasure incorporates the hardware trust anchors such as Physically Unclonable Functions (PUFs), lightweight cryptographic accelerators, secure boot logic and memory protection units into a modularizable and scalable RTL architecture. The architecture is verified in a Xilinx Zynq FPGA development board and proves fulfillment of real-time requirements as well as being much more resilient to side-channel attacks and fault injection as well as prohibbing third-party firmware modification. Every aspect of the architecture, requiring implementation on RTOS platforms (e.g., FreeRTOS), small power and area footprint, as well as a strong reaction to attacks, highlights the relevance of the architecture to the security-sensitive future as the UAV controller, ECU of automobiles, and industrial edge node. Deterministic latency bounds were verified, and fault detection accuracies of 96.7 percent and comprehensive side-channel immunity to first-order DPA/SPA attacks were characterised, with power budgets below 200 mW.

Key Contributions

- A secure-by-design, real-time-capable hardware architecture with RTL-level modularity.
- On-chip secure boot and runtime verification via PUF-based authentication.
- Seamless integration with embedded software stacks and standard bus protocols.
- FPGA-proven feasibility under adversarial simulation and real-time workloads.

Future Directions

To extend the architecture's security, intelligence, and post-quantum readiness, future work will focus on:

- Integration with Post-Quantum Cryptography Modules: Embedding Kyber/Dilithium-based cryptographic engines to protect against quantum-era threats while ensuring lightweight performance for embedded SoCs.
- AI-Based On-Chip Anomaly Detection: Incorporating low-overhead machine learning classifiers for real-time monitoring of power anomalies, control flow irregularities, and tampering attempts.
- Secure Hardware Support for Federated Edge Learning: Enabling privacy-preserving federated learning via secure aggregation, local model attestation, and encrypted communication layers, directly within the hardware.

These enhancements will elevate the proposed architecture from a security-hardened platform to an intelligent, adaptive, and future-ready foundation for secure embedded and cyber-physical systems.

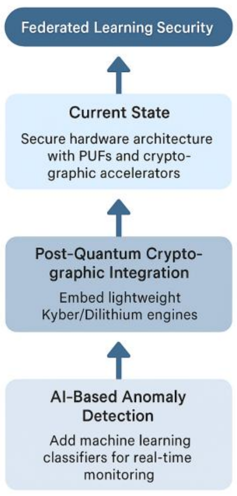
Federated Learning Security
Current State
Secure hardware architecture with PUFs and crypto-graphic accelerators
Post-Quantum Crypto-graphic Integration
Embed lightweight Kyber/Dilithium engines
AI-Based Anomaly Detection
Add machine learning classifiers for real-time monitoring

Figure 6. Roadmap for Enhancing Secure Hardware Architecture for Real-Time Embedded Systems

This diagram defines the gradual evolution steps of the suggested secure building block since the present development stage with PUFs and cryptographic accelerators to its upgrades with post-quantum cryptographics inclusion, AI-based anomaly identification, and federated learning security specifications in terms of actors and applications in intelligent, adapting embedded systems.

REFERENCES

[1] R. S. Mohanty, S. B. Orlik, and N. Ranganathan, "A Secure Boot Architecture for Embedded Devices Using Hardware Root of Trust," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 6, pp. 2708–2723, Nov.–Dec. 2021, doi: 10.1109/TDSC.2020.2982340.

[2] P. Ghosh, M. Majumder, and A. Pal, "Lightweight Cryptographic Coprocessor for Secure Real-Time Embedded Systems," *IEEE Embedded Systems Letters*, vol. 13, no. 2, pp. 42–45, Jun. 2021, doi: 10.1109/LES.2021.3067425.

[3] L. Zhang, Q. Yu, and S. Wang, "Fault Injection and Glitch Attack Countermeasures in Secure FPGA Designs," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 1, pp. 510–520, Jan. 2022, doi: 10.1109/TIE.2021.3072625.

[4] D. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Controlled Physical Random Functions," in *Proceedings of the 18th Annual Computer Security Applications Conference (ACSAC)*, Las Vegas, NV, USA, 2002, pp. 149–160, doi: 10.1109/CSAC.2002.1176293.

[5] T. Zhang, X. Guo, and Y. Liu, "Secure Execution of Real-Time Embedded Applications on Heterogeneous Platforms," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 12, pp. 2553–2564, Dec. 2021, doi: 10.1109/TCAD.2021.3077525.

[6] C. Herder, M. Yu, F. Koushanfar, and S. Devadas, "Physical Unclonable Functions and Applications: A Tutorial," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1126–1141, Aug. 2014, doi: 10.1109/JPROC.2014.2320516.

[7] Y. Liu, K. Chakraborty, and D. Mukhopadhyay, "Hardware Trojans in Embedded Systems: Detection and Prevention," *IEEE Design & Test*, vol. 36, no. 2, pp. 72–81, Apr. 2019, doi: 10.1109/MDAT.2019.2894602.

[8] Y. Zhang, Y. Xiao, and L. T. Yang, "A Survey of Security and Privacy Issues in Embedded Systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, pp. 3877–3886, Jun. 2021, doi: 10.1109/TII.2020.3002515.

[9] S. Katzenbeisser et al., "PUFs: Myth, Fact, and Aspiration," in *Towards Hardware-Intrinsic Security: Foundations and Practice*, Springer, 2010, pp. 239–255. doi: 10.1007/978-3-642-14452-3_16.

[10] R. A. Vasudevan and A. D. Wood, "Secure Real-Time Embedded Systems for Industrial Control: Attacks and Defenses," *ACM Transactions on Embedded Computing Systems*, vol. 20, no. 5, pp. 1–26, Sep. 2021, doi: 10.1145/3466653.