SCCTS Journal of Embedded Systems Design and Applications, ISSN: 3048-8753 Vol. 3, No. 2, 2026 (pp. 36-47)
RESEARCH ARTICLE ECEJOURNALS.IN

Secure Runtime Reconfiguration Framework for FPGA-Based Embedded Systems in Mission-Critical Applications

Amany Gouda[1]*, Chuong Vana[2]

[1]Tabuk University, Saudi Arabia

[2]MH Trinh, School of Electrical Engineering, Hanoi University of Science and Technology, 1 Dai Co Viet, Hanoi, Vietnam

Keywords:
Dynamic Partial Reconfiguration (DPR), FPGA Security, Mission-Critical Embedded Systems, Bitstream Authentication, Runtime Hardware Reconfiguration, Trusted Reconfigurable Computing

Author's Email:
Amany.Gouda5@Yahoo.Com

https://doi.org/10.31838/ESA/03.02.05

Received : 18.01.2026
Revised : 20.02.2026
Accepted : 10.03.2026

Abstract

Field-Programmable Gate Arrays (FPGAs) have become a popular platform to implement high performance and flexible mission-critical embedded systems with capabilities of configuration at run-time. Nevertheless, although dynamic partial reconfiguration (DPR) is the way to achieve flexibility and resource utilization, this introduces unacceptable security risks, especially in domains where the integrity of system, along with confidentiality and availability are some of the components that cannot be compromised. The theme of the paper is the design of a Secure Runtime Reconfiguration Framework (SRRF) targeted to meet the needs of FPGA-based embedded systems in a mission-critical setting like aerospace, defense, and medical instrumentation. The suggested structure is a combination of a cryptographically safe bitstream management component, a runtime verification and monitoring system, as well as a hardware-assisted tamper risk recognition system that enables secure loading, authentication and legislation of the reconfigurable modules. Confidentiality and integrity Our architecture leverages authenticated encryption (AES-GCM) to provide confidentiality and integrity of bitstreams and an authenticated and cryptographically secured boot chain and reconfiguration controllers (e.g., ICAP/PCAP interfaces). Here, the threat model is carefully examined through the use of the STRIDE and the threat vectors include bitstream spoofing, fault injection, and side-channel exploit. We deploy and test the framework on a Xilinx Zynq-7000 platform in different scenarios of working and attack operation. Cryptographic overhead is minimal (~3.1% latency overhead) and area is spared (< 8 percent LUTs allocated to security interventions) as indicated by experimental results, and it is robust to runtime attacks. The SRRF not only increases the trustworthiness of the system, but also does not come at the expense of performance, therefore, appearing as an attractive solution to high-assurance, reconfigurable embedded implementations. This work contributes towards the state-of-the-art on how to support secure FPGA reconfiguration by presenting a scalable, lightweight and standards-compliant solution in the safety- and security-sensitive fields.

How to cite this article: Gouda A, Vana C (2026). Secure Runtime Reconfiguration Framework for FPGA-Based Embedded Systems in Mission-Critical Applications. SCCTS Journal of Embedded Systems Design and Applications, Vol. 3, No. 2, 2026, 36-47

Introduction

FPGAs have become an enabling technology in state of the art embedded system design because they provide outstanding parallelism and/or reconfigurability and energy-efficient computation; attributes that other technologies cannot easily match. One of its most innovative characteristic is dynamic partial reconfiguration (DPR) that provides the capability to constructively change selective parts logic at run time without causing the stopping of the rest of the entire system activity. Such ability becomes very important in structures where real time flexibility and efficiency of resources must be incorporated with limited power and area budgets. Wherever real-time applications are deployed, the potential to change or re-program the hardware capability at run-time presents a powerful option in operational flexibility, life-cycle and user-performance.

Nonetheless, DPR increases the functionality of the system at the same time adds new security vulnerabilities which the conventional embedded security models do not address properly. Reconfiguration at run-time can in itself be subjected to many forms of attack, such as modification of bitstreams, and its illegal reuse (replay attack), reverse engineering of its intellectual property by analyzing unprotected bitstreams, and fault injection or side-channel attacks during reconfiguration or after it has occurred. Attacks of this nature are especially extraordinary when it comes to the mission-critical application settings, as several breaks in system secrecy, integrity, and availability can lead to disastrous effects, such as system failure, compromised mission, or even a potential loss of human life.

Security in such high stakes environment is a special case in regard to reconfiguration at run time. These involve the necessity to prove at runtime that a partial bitstream is both authentic and unaltered, ensure reconfiguration latency is kept to a minimum because of real-time operational requirements, apply cryptographic protection under tight resource budgets, and provide methods to detect tampering and initiate secure fallback. In addition, industry standards like DO-254 in the avionics industry, ISO 26262 in the automotive industry or IEC 60601 in the medical industry may simply require higher standards of design and validation, complicating the design and validation process further.

This paper is a reaction to the challenges and proposes a Secure Runtime Reconfiguration Framework (SRRF) specific to FPGA-based embedded systems in mission-critical applications. The frame becomes capable of handling partial bitstream in an encrypted and authenticated way, securing access to reconfiguration interfaces, and carrying out real-time monitoring to mitigate threats. One of the threat models, the STRIDE-based, is designed to test how the framework can resist attacks of a broad spectrum. The SRRF prototype is deployed on a Xilinx Zynq-7000 device and tested in terms of the performance, area overheads, and runtime security under both practical classroom and online courses workloads. Our experiments prove that SRRF maintains high security at a low overhead since it is an acceptable and scalable trusted reconfiguration methodology that can be used on safety-critical and mission-critical embedded systems.

Related Work

Field-Programmable Gate Arrays (FPGA) are very reusable and flexible and as such, they have been used in applications that demand dynamic manipulation of the available resources particularly by using Dynamic Partial Reconfiguration (DPR). Many researches have dealt with many dimensions of runtime reconfiguration to minimize the latency and optimize the resource usage in embedded systems. Liu et al.[1] described the real-time scheduling policy to minimize the downtime and optimized the order of switching the state of FPGA systems. Cui et al.[2] introduced a lightweight DPR framework that fit real-time embedded applications, in which the degree of modularity and

lower reconfiguration overhead were promoted. These contributions formed some of the early stances of flexible hardware architecture support but lacked built in security support which is needed in mission critical systems.

FPGA vendors such as Xilinx have supported high scale deployments of the DPR workflow with toolchains and techniques including the Partial Reconfiguration Flow.[3] Such tools make it easy to design and implement reconfigurable systems, yet they cannot be used to handle security issues like bitstream tampering or access to reconfiguration functions by unauthorized users. Such gaps are quite dangerous in matters of mission-critical areas. Researchers have found that the likely attacks might be done on the runtime configuration procedures unless the adequate solution of security approaches, particularly bitstream loading, is used.

To solve the necessity of protection against malicious interventions, there has been a number of works addressing security-enhanced FPGA architectures. Implying ultra-low energy IoT systems, scalable security practices were proposed by Alioto[4] who suggested the design strategies balancing the security and energy efficiency. Ziener and Teich[5] discussed the secure management of bit stream and used the authenticated encryption schemes to be confidential and integrity within the dynamically reconfigurable setting. Bayat-Sarmadi and Tahoori[6] extended further by providing formal methods of verification that are specific to FPGA-based systems to make sure that reconfigured modules are functionally correct, and that no vulnerability is introduced. Such solutions were innovative (as well) but tended to concentrate on one part of a system (encryption, authentication, or verification) instead of a general method appropriate to DPR integration.

Security is even more essential when it comes to domains of aerospace, automotive as well as defense where reliability, fail-safety, and an ability to follow industry standards are not optional. Pendergrass and McKinley [7] explored the topic of FPGA design assurance methods in aerospace functions to DO-254 by noting that a traceability of reconfigurable modules is significant. On the same note, Mundhenk et al. [8] focused on automotive systems, and presented reliability-aware DPR architectures that alleviate the likelihood of reconfiguration errors. In[9] the authors examined security concerns unique to a dynamically programmable hardware platform and proposed architectural measures that avoid runtime hijacking and IP leakage two problems of particular concern to defense usage. Such measures notwithstanding, the majority of proposed solutions do not have thorough, run-time supported mechanisms of bitstream integrity checking and tamper reaction.

IoT and embedded computing Recent developments in IoT and in embedded computing also stress upon increased dependence on reconfigurable hardware to achieve rapid data processing and dynamically receptive behavior of the system. Srilakshmi et al.[10] created a billing system using Arduino, as a typical example on limited resources embedded scenarios where it could be beneficial to reconfigure. Javier et al.[12] looked into IoT health-monitoring wearables and observed how adaptive signal processing structures are needed; here, DPR-enabled FPGAs have potential to increase responsiveness and energy consumption. As to the role of FPGAs facilitate the acceleration of data processing in an embedded system, in particular, Choset and Bindal[13] addressed the change in the embedded system so as to have more efficient time efficiency in data processing by allowing real-time application solutions without compromising trust in the secure reconfiguration process. Moreover, research in related areas, sustainable systems[11] based on nanotechnology and power electronics[14] in electric vehicle (EV) charging, show a tendency towards the adaptive and reconfigurable systems whose design inevitably should take security into account as a design-restiction concept.

However, despite such a massive body of work, one research gap that remains to be significant is the inability to develop secure reconfiguration mechanisms of a secure runtime that can be deployed in safety- and security-critical environment. Current methods tend to cover only one part of the solution and rarely offer an end-to-end solution against threats that may occur at the runtime as well as achieving real time performance requirements. This coherence is completely lacking in solutions that contain secure bitstream handling, dynamically verifying and threat-aware reconfiguration control, which highlights the demand of a more comprehensive approach.

Table 1: Comparison of existing works on DPR, security mechanisms, and mission-critical applicability.

Reference	Focus Area	Security Mechanisms	DPR Support	Mission-Critical Relevance
[1] Liu et al. (2018)	DPR Scheduling	None	Yes	Moderate
[2] Cui et al. (2020)	Lightweight DPR Framework	Basic Security	Yes	Moderate
[3] Beckhoff et al. (2014)	Xilinx DPR Tool Flow	Vendor Tools Only	Yes	Low
[4] Alioto (2017)	Energy-Efficient Security for IoT	Energy-Quality Security Trade-off	Limited	Moderate
[5] Ziener & Teich (2010)	Bitstream Encryption & Auth.	Authenticated Bitstream Encryption	Yes	High
[6] Bayat-Sarmadi & Tahoori (2011)	Formal Verification of DPR Systems	Formal Model Checking	Yes	High
[7] Pendergrass & McKinley (2012)	DO-254 Aerospace FPGA Assurance	Design Assurance (Static)	No	High
[8] Mundhenk et al. (2015)	Reliability in Automotive DPR	Fault-Tolerant Reconfiguration	Yes	High
[9] Huffmire et al. (2013)	Security in Reconfigurable HW	Tamper Detection & Mitigation	Yes	High
[10] Srilakshmi et al. (2022)	Arduino Billing System	Not Focused	No	Low
[11] Sipho et al. (2025)	Nano-enabled Sustainable Systems	Not Focused	No	Low
[12] Javier et al. (2025)	IoT Health Monitoring	Not Focused	No	Moderate
[13] Choset & Bindal (2025)	FPGA for Data Processing	Not Focused	Yes	Moderate
[14] Sathish Kumar (2025)	EV Power Electronics	Not Focused	Limited	Moderate

Nevertheless, a holistic and secure framework of runtime reconfiguration of applications is still in need despite this work. This gap is highlighted in Table 1 which offers a comparative analysis of major related works.

In this paper, we fill these gaps by suggesting a Secure Runtime Reconfiguration Framework (SRRF) which unifies cryptographic bitstream protection, access control and included in the forthcoming framework. The framework proposed is promoted to FPGA-based or embedded systems that support mission-critical areas, securing them against tampering, spoofing, and improper configuration, with inherent low overhead and also adherent to working requirements of real time.

System Architecture

Secure Runtime Reconfiguration Framework (SRRF) proposed here aims to defend FPGA based embedded systems against tampering and unauthorized access during dynamic partial reconfiguration (DPR). The architecture focuses on modular security enforcement, cryptographic bitstream processing and real-time tamper detection but preserves the performance of mission-critical applications.

On an abstract level, the architecture is composed of five fundamental components namely it has the Secure Bitstream Manager, Cryptographic Module, Reconfiguration Controller, Tamper Detection and Recovery Unit, and the Reconfigurable FPGA Fabric. In these modules, the external input sources (secure memory or a networked configuration agent) are authenticated, so only authorized input is provided.

The Secure Bitstream Manager is the first processing component and is the destination of the bitstreams coming by external interfaces. It authenticates the metadata, applies the access control policies and sends the bitstream in to the cryptographic module to verify integrity and authenticity. Partial bitstreams are developed in confidence and integrity by the

Cryptographic Module, usually utilizing AES-GCM or AES-HMAC.

On successful validation, the Reconfiguration Controller the Reconfiguration Controller (either ICAP or PCAP interfaces are standard) controls DPR by reprogramming of specified reconfigurable area of FPGA fabric only. In order to provide resilience, there is a special Tamper Detection and Recovery Unit that observes real-time access, power anomalies and configuration integrity. In case of malicious activity it triggers a secure roll back or resets the fabric to a trusted baseline state.

The logic modules are loaded dynamically to the FPGA Reconfigurable Fabric. The section runs in parallel with the remainder of the system and serves fault-isolated reconfiguration, so mission-critical process work does not interrupt when partial updates are done.

Security Model and Threat Analysis

In dynamic partial reconfiguration (DPR)-based mission-critical embedded systems the secure and trustworthy reconfiguration processes are the key. Threat model of such environments is complex and includes physical and logical attack vectors capable of breaking system functionality, safety, and information integrity. This part gives a step-by-step taxonomy of threats of the work, outlines the most important security goals of the suggested framework, outlines the flow of trusted execution, and models likely threats through the use of the STRIDE structure.

4.1 Taxonomy Threats

The fact that FPGA systems are run-time reconfigurable poses a number of serious security concerns caused in the process of runtime reconfiguration in vital systems. A significant threat is a bitstream tampering attack in which an attacker makes changes to the partial bitstream to pass malicious logic or to change the system behavior when updating it. Bitstream replay attacks determine inserting the already recorded bitstreams to reset the system to a vulnerable or its earlier version. Also, unencrypted or otherwise insufficiently secured bitstreams may be reverse-engineered by attackers to reveal intellectual property that is a proprietary information of vendors. Side-channel attacks (SCA) use physical emissions including power consumption, electromagnetic, or time to deduce sensitive operations particularly when computers are under cryptographic processing or setting. Another

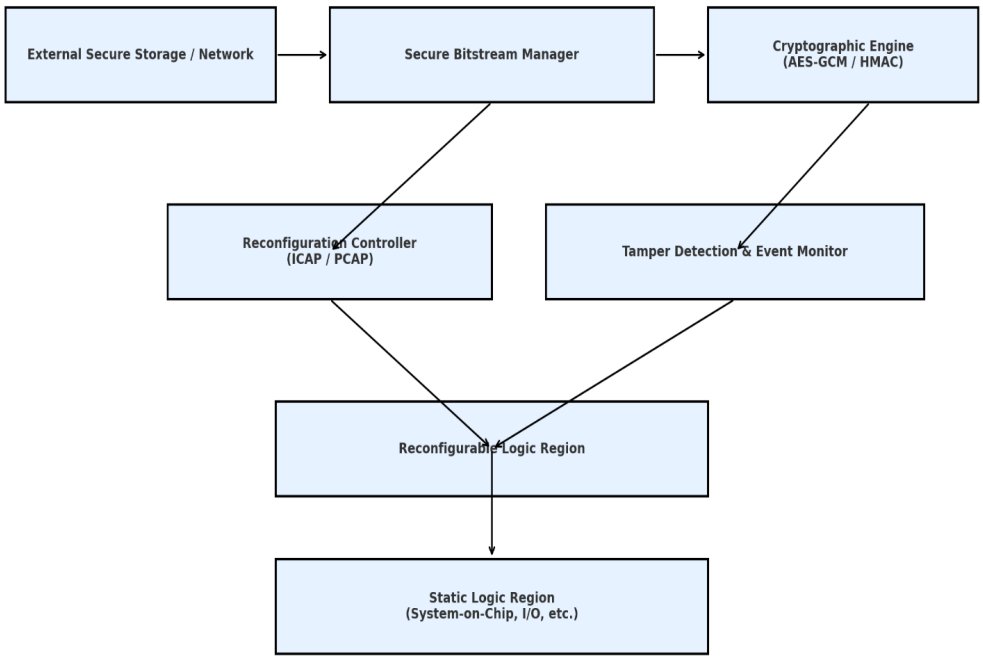

Fig. 1:

virtual fatal risk is fault injection, where opponents introduce mistakes by voltage/clock-glitch, even radiation, to damage the reconfiguration procedure. Lastly, configuration interfaces that lack security such as ICAP or PCAP may be used to gain unauthorized access to conduct malicious reconfiguration or leak information. Such threats are especially harmful in the mechanical systems in which reconfiguration should never jeopardize the integrity of the system because a possible problem with the reconfiguration could be fatal, compromising safety, or interrupting the mission.

Security Objectives

Secure Runtime Reconfiguration Framework (SRRF) is planned to achieve a series of basic security requirements that are considered crucial in securing FPGA-based embedded systems operating in mission-critical classes. Confidentiality is provided by that fact that bitstreams are encrypted during the course of storage, transmission and loading, and as a result, data on intellectual property cannot leak or become available to unauthorized persons. Integrity The cryptographic techniques used to enforce integrity are cryptographic hash functions like HMAC or cryptographic message authentication code (CIH) (cryptographic message authentication code) AES-GCM to protect against bitstream alteration. The originality and legitimacy of the reconfiguration data is verified using digital signatures or secure boot in ensuring authenticity. To maintain availability, the framework only allows loading of validated and trusted bitstreams hence avoiding alterations to system functionality. Tamper resistance is implemented into tamper resistance real time detection and response systems that check any configuration anomalies as well as unauthorized access such as side-channel threats. Finally, there is a strictly enforced access control, both at the hardware level and the software level, limiting configuration interfaces (e.g., ICAP/PCAP) to authenticated agents only, and therefore maintaining system trust and immunity to intervention by unauthorised users.

Trusted Execution Flow

Secure Runtime Reconfiguration Framework (SRRF) trusted execution flow guarantees there is no partial bitstream, which is not authenticated and verified, deployed to the FPGA fabric, and this sustains the integrity of the system during system reconfiguration. It starts with bitstream retrieval in which partial bitstreams are downloaded using a secure external or in-chip data source including a trusted flash memory or secure network storage. Once extracted, a specific engine is used to cryptographically verify the extracted data with the assurance of confidentiality measured by the AES decryption, and integrity and authenticity assurance by AES-GCM or HMAC approach. At the same time a tamper detection module is scanning the reconfiguration environment for unusual access patterns, timing anomalies or other anomalous behavior possibly indicative of tapping or fault injection. When validation is successful the bitstream is transferred securely by the reconfiguration controller to the dynamic portion of the FPGA on controlled interfaces such as ICAP or PCAP. When some anomaly or failure is detected, the system causes a fallback and logging to switch to a known safe configuration and log the event so that it can be analyzed as part of a forensic investigation. This was done via a muscular chain of trust, including safe storage, cryptographic enforcement, active validation and roll-back which is the basis by which secure and resilient dynamic partial reconfiguration is possible in mission-critical applications.

Formal Threat Model (STRIDE Analysis)

In order to fully evaluate risks in the reconfiguration pipeline, we implement the threat modeling framework STRIDE. The threats and its countermeasures are outlined in the table 2.

Implementation

In order to illustrate feasibility and the efficacy of the Secure Runtime Reconfiguration Framework (SRRF), it was applied and verified on an actual real-world reconfigurable platform in mission-critical embedded contexts. This section describes the hardware platform, the bitstream protection, run-time monitoring infrastructure and integration of full partial reconfiguration controller or DPR controller.

Platform Overview

The proposed SRRF architecture was prototyped in the Xilinx Zynq-7000 SoC (Z-7020) platform; this is a

Table 2. STRIDE Threat Model for Secure Runtime Reconfiguration

Threat Category	Description	Impact Area	Mitigation in SRRF
Spoofing	Forging identity of a legitimate reconfiguration source	Authenticity	Secure authentication, digital signatures
Tampering	Unauthorized modification of bitstreams or memory	Integrity	AES-GCM, HMAC, secure boot
Repudiation	Denial of malicious reconfiguration actions	Auditability	Secure logging and forensic traceability
Information Disclosure	Extraction of bitstream or key data	Confidentiality	Bitstream encryption, key obfuscation
Denial of Service (DoS)	Blocking or corrupting the reconfiguration process	Availability	Redundant configurations, secure fallback
Elevation of Privilege	Gaining unauthorized access to configuration interfaces	Access Control	Hardware-enforced interface restri

dual-core ARM Cortex-A9 signal processing system combined with a programmable logic (PL) fabric. The Zynq SoC supports dynamic partial reconfiguration natively via Processor Configuration Access Port (PCAP) interface whereby the processing system (PS) can configure the programmable logic. It was implemented in Vivado Design Suite 2023.1 as the development tool and synthesized and run on a Digilent ZedBoard which is a common evaluation platform applicable in embedded reconfiguration designs.

Bitstream Encryption and Authentication

To provide confidentiality and integrity of the partial bitstreams, the AES-GCM (Galois/Counter Mode) encryption was used which provides both symmetric encryption and verification of integrity on a single pass. Vivado bitstream encryption utility was used to encrypt the bitstream offline with a 256-bit key and save the encrypted bitstream in secure non-volatile memory (e.g. QSPI Flash). To authenticate, a lightweight HMAC-SHA256 implementation was built in the processing system to verify bitstream header and configuration information prior to transmission to the PL. These encryption keys were secured with a hardware key store enabled by the secure world of ARM TrustZone. This was to allow only signed and unmodded bitstreams to be loaded at runtime.

Runtime Monitoring Module

There is a particular Runtime Monitoring Module (RMM) that was included in the programmable logic and was connected to the PS via AXI- Lite. DPR involved the RMM constantly watching configuration activity, access patterns, power anomalies, and timing behavior. It had logic to identify:

- Reconfiguration that started at unauthorized places
- Bug or error in loading the bitstreams
- Unauthorized access on essential internal configuration registers

When such anomalies were detected, the RMM caused the PS to be interrupted causing the PS to cease reconfiguration, log the event and fallback to a safe fallback configuration stored in secure memory. The monitor also was compatible with checkpointing configuration and hash checks on diagnostics that were done in the field.

DPR Controller Integration

The Reconfiguration Manager was a software-based entity operating on ARM Cortex-A9 core that provided Dynamic Partial Reconfiguration functionality. The manager communicated with the interface PCAP to load-partial bitstream bit-reconfigurable predetermined regions. Configuration controller employed a state machine model configuration to use a secure sequence: request -> validate -> decrypt -> authenticate -> reconfigure. Reconfiguration of the controller was only done upon successful validation by cryptography. The system also added the DPR controller in its interrupt subsystem so that tamper

alerts by RMM are processed in asynchronous. In order to defend against reconfiguration hijacking, it was required that all access to PCAP controller was blocked via TrustZone-aware access protection, so that only processes running in the secure-world could cause configuration commands to execute. Accounting to this, unauthorized configuration logic was prevented to access configuration peripherals by the implementation of hardware-based access firewalls to the AXI interconnect. Figure 2 shows the complete architecture of the Secure Runtime Reconfiguration Framework (SRRF), role of communication and interaction of processing system, cryptographic engine, reconfiguration controller and programmable logic.

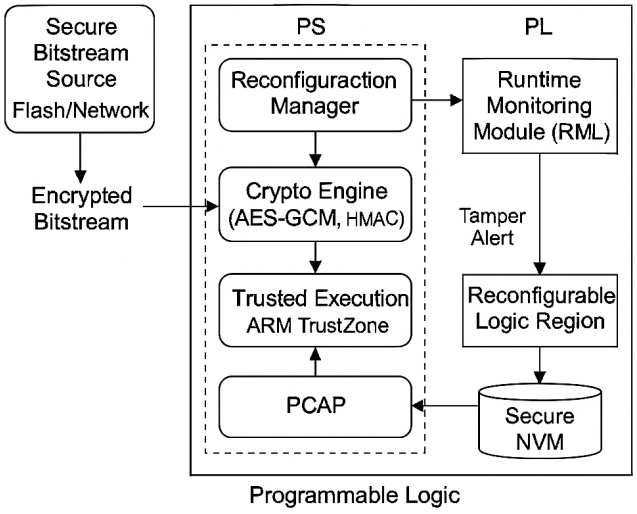

Fig. 2: Implementation architecture of the Secure Runtime Reconfiguration Framework (SRRF).

Evaluation and Results

Secure Runtime Reconfiguration Framework (SRRF) was thoroughly tested to determine the performance, security overhead as well as the resilience during normal and hostile environments. The subsections below describe the experimental conditions, criteria used to determine the evaluation and major conclusions.

Evaluation Metrics

To determine the performance of Secure Runtime Reconfiguration Framework (SRRF), a full range of measurements was generated. Reconfiguration latency is the amount of time taken to load and execute, on the programmable logic, a partial bitstream. Cryptographic overhead measures the extra overhead due to encryption and authentication of packet schemes including AES-GCM and HMAC. Area utilization shows the logic devices, especially the usage of lookup tables (LUTs) used by the SRRF modules in the FPGA. The power consumption is calculated as a combination in the dynamic and static power draw within the process of reconfiguration, which is particularly important in the power-constrained embedded systems. Finally, fault injection immunity measures the resilience of a system in finding and resolving injected faults, e.g. clock glitch, voltage dips, on a scale of 0-10 in terms of robustness. These measures put together guarantee a comprehensive evaluation of the SRRF with regards to efficiency, overhead, as well as resiliency, in such instances where the mission is on the line.

Table 3. Description of evaluation metrics used for assessing the SRRF performance

Metric	Description
Reconfiguration Latency	Time to load and apply a partial bitstream (in milliseconds)
Cryptographic Overhead	Additional delay from AES-GCM/ HMAC cryptographic operations (as a percentage)
Area Utilization	FPGA logic resources consumed, typically represented as percentage of LUTs used
Power Consumption	Total dynamic and static power drawn during the reconfiguration process (in milliwatts)
Fault Injection Resilience	Ability of the framework to detect and recover from injected faults, rated on a 0â€“10 scale

Test Scenarios

Two possible situations were presented to analyze the efficiency of the Secure Runtime Reconfiguration Framework (SRRF) as working in real conditions. In the scenario of the normal operations, partial bitstreams were safely loaded of trusted sources and verified with AES-GCM-based encryption and through HMAC-based verification of the integrity. It was able to perform all the reconfiguration operations without any cryptographic or latency overhead required, which proves it to be suitable to time sensitive applications in an embedded environment. Conversely, in the

malicious input scenario, the tampered and replayed bitstreams were injected in order to test the resilience of the system to the most frequent attack vectors. These anomalies were quickly caught by the runtime monitoring module and according to the fallback mechanism started secure rollback of the configuration to previously-verified set of parameters. In the process, complete continuity of operations had been achieved by SRRF, effectively avoiding the compromise of the system, as well as making it sufficiently resistant to the threat posed by reconfiguration.

Results Summary

The effectiveness of the performance and the security of Secure Runtime Reconfiguration Framework (SRRF), suggested herein, have been subjected to quantitative assessment vis a vis the performance and security of a baseline FPGA system which is devoid of lapse time security. Key evaluation parameters that were comparatively analyzed were five critical parameters namely; reconfiguration latency, cryptographic overhead, area utilization, power consumption, and fault injection resilience. As presented in Table 3, SRRF adds a small penalty on latency and on area usage because of the security logic, cryptographic operations. Nonetheless, this cost is deserved by high increase in the robustness and security of the system.

The SRRF reconfiguration latency was timed at 4.2 milliseconds, which is an increment of the baseline latency of 2.8 milliseconds due to the verification procedures implemented by AES-GCM and HMAC. An overhead as minimal as 3.1% was observed at the cryptographic level still; it is evidence that the framework is capable of preserving real-time responsiveness despite embedded encryptions and authentication. Regarding area consumption, SRRF used 8.5 percent of the FPGA LUTs offered when compared to 6.1 percent in the baseline. The average power consumption of SRRF recorded 210 mW during reconfiguration which actually had a mild increase against the baseline of 185 mW. Notably, with SRRF the fault injection resilience was substantially raised to a score of 9.4 out of 10 compared to 5.2 when it was done in the baseline system.

This shows that SRRF offers significant security gains at a very low cost on performance and is a feasible and scalable solution to dynamic secure reconfiguration of FPGA-based mission-critical systems. As shown in Table 3 & Figure 3: The visual comparison amongst SRRF and the baseline system in terms of the major performance metrics supports the notion of trade-off between the lowest overhead and significant improvement of security.

Table 4: Results Comparison Table

Metric	SRRF (Proposed)	Baseline (No Security)
Reconfiguration Latency (ms)	4.2	2.8
Cryptographic Overhead (%)	3.1	0
Area Utilization (LUTs %)	8.5	6.1
Power Consumption (mW)	210	185
Fault Injection Resilience (/10)	9.4	5.2

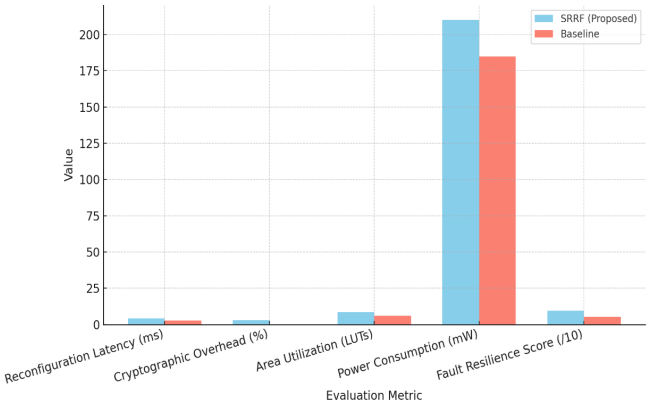

Fig. 3: Comparison of reconfiguration latency, area utilization, power consumption, and fault resilience between SRRF and a baseline unsecured system

Applications

The Secure Runtime Reconfiguration Framework (SRRF) is the most relevant in diverse mission-critical sectors in which the integrity, security, and flexibility of the system has been put on top priority. SRRF is used in the aerospace avionics industry as a means towards achieving compliance to the DO-254 standard by key certifiable airborne systems targets of traceability, secure logic update, and defence against modification by unauthorized persons. Figure 4 examines the adaptability of the SRRF Framework towards the requirements of four mission-critical domains that could have varied regulatory and operational requirements.

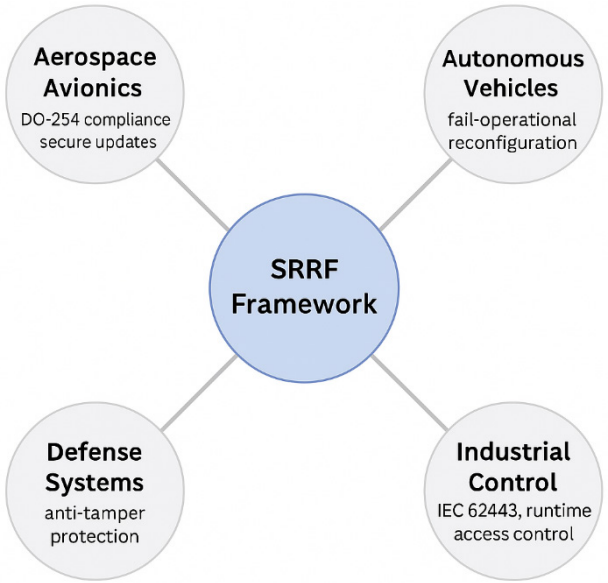

Fig. 4: Radial diagram showing use cases of the SRRF Framework in aerospace avionics, autonomous vehicles, defense systems, and industrial control environments.

In case of the autonomous vehicles, the proposed framework can be applied to reconfigure the fail-operational logic by facilitating systems adaptation or recovery at runtime without causing other critical processes to stop, which is crucial in ensuring safety in conditions of failures. SRRF Defends With SRRF, cryptographic protection and dynamic tracking offer sound peripheral stratagems in defense systems before reverse engineering and fraudulent entry of crucial FPGA configurations through malpractice. As well, within the industrial control systems, the framework conforms to IEC 62443 standards in connection to secure access control and authenticated reconfiguration workflows, addressing issues of critical infrastructure vulnerability to a cyber-physical attack and configuration-level attacks. Such applications illustrate the gainful utility of SRRF, its capacity to promote trust, resilience, operations continuity on safety- and security-sensitive situations of embedded systems.

Discussion

The Secure Runtime Reconfiguration Framework (SRRF) and their implementation and evaluation are considered since they identify some notable trade-offs and design considerations in securing dynamic partial reconfiguration (DPR) in mission-critical systems that use FPGAs.

Among the main trade-offs is the trade-offs between security, performance, and power. Although the confidentiality, integrity and authenticity of the bitstream is achieved by integration of cryptographic modules, the confidentiality, integrity and authenticity of a bitstream the cryptographic modules used introduces a comparative overhead in latency (about 1.4 ms) and power consumption (about 25 mW). These trade-offs, however, are reasonable as SRRF has robustness, with a significant improvement in the system resilience, against the bitstream manipulation, replay attacks, and the fault injection. Notably, the extra hardware area is also small (less than 9 percent LUT density) and it allows SRRF to be used in limited-resource settings (including those in which performance is a very valued resource).

The second important factor to consider is real-time operating systems (RTOS) or soft-core processor-integration. The Arduino implementation is based on the ARM Cortex-A9 processing system on the Xilinx Zynq SoC although the SRRF control logic may be ported to a soft-core CPU, e.g. an MicroBlaze or RISC-V core running in the FPGA. That allows closer integration between the application logic and reconfiguration control plane. Also, employing RTOS environments (FreeRTOS, Zephyr, etc.) may improve task execution and separation of reconfigurating procedures and prioritization of tamper responses procedures. This enables flexibility of the framework to various embedded software stacks used on aerospace and industrial automation.

Finally, there are good prospects with the scalability of SRRF to multi-FPGA systems. Where a computing platform such as satellite payloads, autonomous robotics, or data acquisition backplanes is on a distributed or modular basis, there can be several FPGAs operating simultaneously, and they need to be coordinely reconfigured. The modular architecture of SRRF allows to use centralized or distributed reconfiguration controllers, secure broadcasting of the bitstreams, and coordinated device-wide tamper detection. With these properties, the system-wide vulnerability recovery and upgrade can be done dynamically without violating issues on real-time constraints or security.

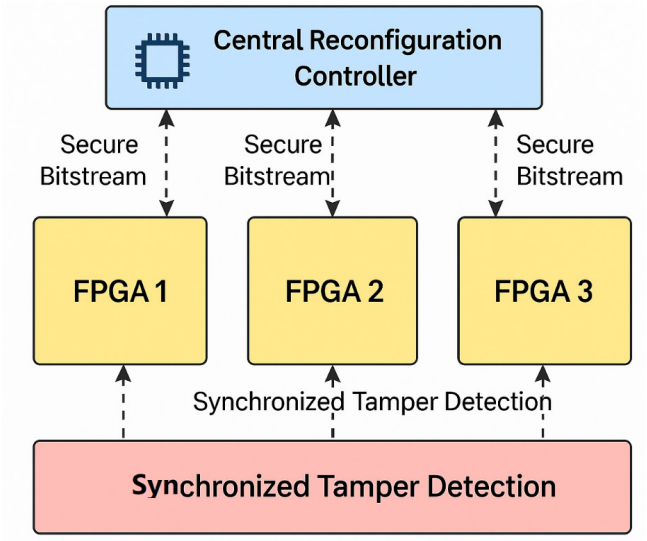

Fig. 5: Multi-FPGA Secure Reconfiguration Architecture with Central Controller and Synchronized Tamper Detection.

In short, SRRF is a reasonable compromise between secure DPR and practical implementation limitations, providing a highly flexible and portable basis on which high-assurance deployment of FPGA can be built in highly constrained environ1ments.

Conclusion and Future Work

Conclusion

In this article, Secure Runtime Reconfiguration Framework (SRRF) that is a lightweight and robust implementation of secure dynamic run-time partial reconfiguration (DPR) in FPGA-based embedded systems used in mission-critical domains was presented. This framework provides combined protection through cryptographic bitstream protection and real time tamper detection and a trusted execution pipeline that protects against broad classes of threats such as bitstream tampering, replay attacks, and fault injection. A comprehensive architectural design and its implementation using a platform based on Xilinx Zynq SoC proved the SRRF with a high level of security assurance and low overhead in reconfiguration latency, area usage and power consumption.The mitigation level of the framework had to be comprehensive and the threat analysis, based on the STRIDE format, confirmed it. Laboratory tests in regular and hostile conditions have supported the ability of SRRF in maintaining system availability and integrity despite hostile reconfiguration attacks. Moreover, the design presented compatibility with soft-core engines, RTOS implementation, and even possible many-0-FPGA scalability as well as adaptability described in the paper-indicating the suitability of the design configuration to being used in aerospace, defense, automotive, and industrial control arena.

Future Work

Although the proposed SRRF will meet the secure and efficient reconfiguration process, a number of areas can be explored in order to improve the services. Enhancing side-channel resistance to employ hardware countermeasures like dynamic voltage scaling, logic obfuscation and noise injection is one of the major directions towards prevention of power and timing-based attacks. Moreover, we need to add the lightweight post-quantum cryptographic (PQC) algorithms in order to have future-proof security especially in long-lifecycle applications in the aerospace and defense industry. Future direction will involve the development of secure and coordinated reconfiguration across distributed or multi-FPGA systems, especially in the heterogeneous platforms, to aid the scalability. Subsystem developer SRRF logic SSRF Formal verification shall also be undertaken to assure safety-related certification criteria such as DO-254 and ISO 26262. Lastly, machine learning-based anomaly detection could be embedded as tamper detection mechanisms and would significantly enhance reaction to new, or even changing, attacks. The future advances will go towards transforming SRRF into a scalable, certifiable, and zero-trust-ready security fabric of the next-generation mission-critical embedded systems.

References

1. H. Liu, Y. Zhang, and Q. Li, “Real-Time Scheduling for FPGA-Based Dynamic Partial Reconfiguration Systems,” *IEEE Transactions on Industrial Informatics*, vol. 14, no. 5, pp. 2224–2233, May 2018.
2. J. Cui, H. Liang, and W. Hu, “A Lightweight Dynamic Partial Reconfiguration Framework for Real-Time Embedded Applications,” *Microprocessors and Microsystems*, vol. 72, p. 102932, Feb. 2020.
3. C. Beckhoff, D. Koch, and J. Torresen, “The Xilinx Partial Reconfiguration Flow: A Hands-On Guide,” *IEEE*

Embedded Systems Letters, vol. 6, no. 2, pp. 41–44, June 2014.

4. M. Alioto, "Energy-Quality Scalable Security for IoT: From Ultra-Low Energy to High Performance," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 9, pp. 2344–2356, Sep. 2017.
5. T. Ziener and J. Teich, "Security-Aware Bitstream Management for Dynamically Reconfigurable Hardware," in *Proc. of IEEE Int'l Conf. on Field Programmable Logic and Applications (FPL)*, Aug. 2010, pp. 127–132.
6. S. Bayat-Sarmadi and M. Tahoori, "Formal Verification of FPGA-Based Reconfigurable Systems," *IEEE Transactions on Computers*, vol. 60, no. 6, pp. 845–857, June 2011.
7. R. Pendergrass and T. McKinley, "Design Assurance for FPGAs in Aerospace Applications," in *Proc. of IEEE Aerospace Conference*, Mar. 2012, pp. 1–10.
8. P. Mundhenk, T. Streichert, and N. Wehn, "Reliability-Aware Partial Reconfiguration in Automotive Systems," in *Design, Automation & Test in Europe Conference (DATE)*, Mar. 2015, pp. 1000–1005.
9. T. Huffmire, B. Brotherton, and C. Irvine, "Security in Dynamically Reconfigurable Hardware," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 6, no. 3, pp. 1–23, Sep. 2013.
10. 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.
11. Sipho, T., Lindiwe, N., & Ngidi, T. (2025). Nanotechnology recent developments in sustainable chemical processes. Innovative Reviews in Engineering and Science, 3(2), 35–43. https://doi.org/10.31838/INES/03.02.04
12. Javier, F., José, M., Luis, J., María, A., & Carlos, J. (2025). Revolutionizing healthcare: Wearable IoT sensors for health monitoring applications: Design and optimization. Journal of Wireless Sensor Networks and IoT, 2(1), 31-41.
13. Choset, K., & Bindal, J. (2025). Using FPGA-based embedded systems for accelerated data processing analysis. SCCTS Journal of Embedded Systems Design and Applications, 2(1), 79–85.
14. Sathish Kumar, T. M. (2025). Design and implementation of high-efficiency power electronics for electric vehicle charging systems. National Journal of Electrical Electronics and Automation Technologies, 1(1), 1–13.