Towards Zero-Trust Hardware Architectures in Safety and Security Critical System-on-Chips

Francesco Restuccia
University of California San Diego
La Jolla, CA, USA
frestuccia@ucsd.edu

Ryan Kastner
University of California San Diego
La Jolla, CA, USA
kastner@ucsd.edu

***Abstract*—Edge computing applications have strict requirements for latency, throughput, and energy. Increasingly, there are more safety and security requirements due to system-level threats that have been discovered in current SoCs. To address these issues, the research community proposed multiple novel solutions aimed at patching the uncovered threats. Nowadays, open hardware SoC platforms have reached an impressive level of maturity. We believe that such a level of maturity provides interesting opportunities for the research community for the integration, development, and evaluation of innovative methodologies for enhancing the security and safety of next-generation SoCs. In this paper, we describe some of these opportunities we believe are relevant, including system-level architectural extensions, fine-grained timing analysis, and novel methodologies for security and safety verification. Combined, these methodologies can ease the certification process in critical systems and eventually contribute to enhancing security and safety in the next generation of commercial SoCs for critical-edge applications.**

I. Introduction

Modern edge applications have complex requirements at the intersection of high performance, energy efficiency, safety, and security. Safety and security requirements are particularly relevant in modern critical edge applications, such as automotive, medical, and avionics, to name some examples. To meet these complex requirements, current SoCs for the implementation of critical edge applications are complex systems including several heterogeneous components, such as processors, hardware accelerators, hardware root of trusts, shared memories, and IOs, etc. The sample architecture of a modern SoC is represented in Figure 1 – multiple controllers C (e.g., processors, hardware accelerators, DMAs, etc.) are interconnected to multiple shared peripherals P (e.g., DRAM memories, scratchpads, IOs, etc) through a system interconnect implementing a modern standard for on-chip communications. Given their complexity, developing all of the computing components in-house is not typically a convenient choice for an SoC vendor: a modern SoC is generally a composition of multiple third-party IPs sourced from different vendors or open-source projects.

The hardware CWE database enumerating the weaknesses found in commercial systems and maintained by the MITRE consortium witnesses how modern SoCs are unfortunately rife with security and safety weaknesses [1]. Such weaknesses originate from different sources, ranging from bugs in outsourced IPs, optimizations for high-performance generating dangerous conditions, superficial verification, etc. Recent research works demonstrated how the interactions among the integrated IPs are a major source of security and safety concerns – it has been demonstrated how a single misbehaving/malicious IP component can easily break the whole system execution, for instance, taking advantage of a combination of lack of specification in the on-chip communication standard [2] with mechanisms for high performance deployed in modern SoC platforms. This generates particularly critical concerns when the SoC integrates third-party IP modules originating from external sources.

In this paper, we describe some of the opportunities we believe are currently particularly relevant for enhancing the security and safety of SoCs for critical edge applications. We find opportunities at the intersection of three pillars: (i) hardware-enabled security and safety, through the integration of ad-hoc high-performance architectural extensions aimed at supporting proactive supervision of the computing units; (ii) fine-grained worst-case timing analysis, supporting real-time guarantees in timing interactions for critical task execution; and (iii) property-based security and safety verification, supporting the system-level verification process of the SoC, of the functionalities for enhancing its safety and security, and uncovering potential weaknesses. In the combination of these three pillars, we aim at the concept of *zero-trust system architectures* – in contrast with several modern SoCs, in which the computing units are implicitly trusted and assumed to work cooperatively, we support a target specification enforcing supervised access to the system of the computing units, while keeping full compliance with the de-facto on-chip communication standards for supporting best integration compatibility.

We believe that the availability of the full codebase of fully-featured mature open-hardware platforms represents an unprecedented opportunity for the development, integration, and testing in real testbeds of these advanced mechanisms for enhancing the security and safety of the system. We believe that this provides a fundamental platform for the extended research community to have a central role in defining the architecture of the next-generation SoCs for critical edge applications and directly influence the industry.

979-8-3503-6334-0/24/$31.00 ©2024 IEEE
DOI 10.1109/RAGE62451.2024.00014

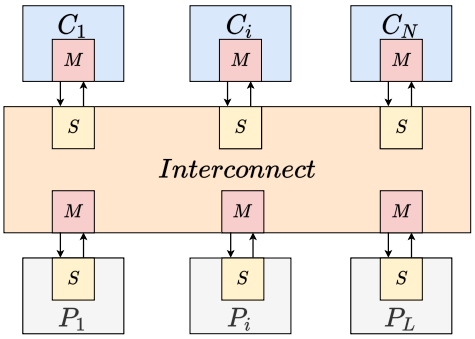

Fig. 1. The sample architecture of a modern crossbar-based SoC: controllers C are interconnected to shared peripherals P through a system interconnect.

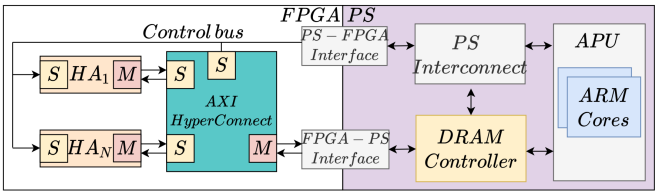

Fig. 2. A sample architecture featuring an AXI HyperConnect in a hardware-accelerated application deployed on a modern FPGA SoC platform. In this case, the control port of the AXI HyperConnect is connected to the Processing System (PS) providing security and safety functionalities to a Hypervisor.

II. First Pillar: Architectural Hardware Extensions for Safety and Security

As introduced in Section I, the architecture of a modern SoC is typically composed of a collection of third-party hardware IPs. Integration of multiple IPs in a system creates system-level threats for safety and security, in particular when integrating third-party controller IPs: it has been demonstrated, for instance, how a single misbehaving or malicious controller IP can easily exploit lack of specification in the de-facto standard for on-chip communications [2] to generate conditions ranging from bandwidth-stealing [3] to Denial-of-Service in accessing shared resources (e.g., a shared DRAM memory) [4]. Such conditions have been demonstrated to be capable of affecting the whole system execution and generating system failures. Clearly, a system failure is a critical condition to avoid in the majority of edge critical systems. From the previous considerations, third-party IP integration can be challenging in critical systems – given the complexity of current IPs, verifying the absence of bugs, weaknesses, or undercover malicious behaviors possibly triggered by the integrated IPs is not straightforward, also considering that controllers can be provided by the vendor as encrypted IPs (in these cases, no direct RTL analysis is possible) and that generally modern controllers IPs are configurable by software.

An approach to tackle these challenges is the deployment of ad-hoc hardware defensive mechanisms, specifically aimed at actively monitoring and supervising the execution of each computing unit IP. Some examples of such functionalities involve enforcing an active supervision and control of: (i) the structure of the transactions issued by each controller IP. This enables the shielding of the system against bandwidth-stealing-related threats and enforcing a fair and predictable bandwidth distribution among the controllers [3]; (ii) the timing in data provisioning after write requests are generated by the controller IPs [4] or validating write data before the propagation of a write request [5]. These two solutions shield the system from potential attempts of Denial-of-Service attacks of the shared resources; (iii) the address space that can be accessed by each controller at runtime [6], [7], enforcing a target access control policy specification supporting the confidentiality and integrity of the shared resources and the controllers.

In an SoC deploying a crossbar-based system interconnect, such as the one represented in Figure 1, the previously introduced solutions can be seamlessly deployed leveraging a custom system interconnect enhanced with advanced safety and security features [8] or enhancing the functionalities of a standard system crossbar with ad-hoc IPs [9]. The AXI HyperConnect [8] is an example of an enhanced AXI interconnect featuring advanced functionalities for system-level safety and security. AXI HyperConnect is a one-to-one substitute for AXI interconnects deploying all of the standard routing functionalities of a standard AXI interconnect, enhanced with advanced features for the active monitoring and supervising of the computing units and aimed at enforcing secure and safe system interactions. The AXI HyperConnect works in synergy with a trusted entity (e.g., a Hardware Root of Trust, a hypervisor, etc.), enabling dynamic supervising, runtime management, and proactive reaction to misbehaviors and security attacks attempted by the controllers. Such functionalities provide a crucial feature for critical systems: keeping the system operational even when faults, misbehavior, or security attacks are attempted on the system by one or multiple misbehaving/malicious computing unit controller(s). A sample architecture deploying the AXI HyperConnect is reported in Figure 2.

III. Second Pillar: Fine-grained timing analysis for timing predictability

Timing predictability is another crucial requirement in critical systems. Guaranteeing that critical tasks executing on top of the hardware computing units of the SoC can execute within predefined time deadlines is fundamental to ensuring the safety of the system and avoiding dangerous conditions [10]. This process is particularly relevant when a certification process is required. As also introduced in Section II, challenges arise in ensuring the timing predictability of the system interactions when multiple heterogeneous controllers compete to access a shared resource, for instance, the central DRAM memory controller. The architectural hardware extensions described in previous sections provide a framework for enforcing nominal and predictable system interactions. However, another important matter must be addressed – bounding the legal interference that the computing units generate on each other during execution, considering also that critical and non-critical tasks can

compete in the same system in accessing a shared resource. Challenges arise when considering modern SoCs, specifically considering that advanced features such as multiple outstanding transactions and burst transactions are fundamental to provide the high performance in system interactions required in modern applications, but also introduce a relevant level of unpredictability.

To tackle this challenge, recently, multiple system-level worst-case timing analyses have been proposed. These analyses are capable of providing mathematical upper bounds on system interactions, considering the features for high performance deployed in modern AXI-based SoCs and the possible hierarchical structure of the architecture [11], [12]. The solutions follow a mathematical approach aimed first at modeling and analyzing the maximum interference suffered by each computing unit during execution and follow computing the overall cycle-accurate response time analysis considering the specific architecture of the target system to provide a safe upper bound of the system interactions. This approach has been demonstrated to be effective in real designs and applications – an example is the application of this approach to upper-bound the execution time of the AMD Deep learning Processing Unit (DPU) (part of the AMD Vitis AI framework) executing Deep Neural Network (DNN) algorithms for autonomous driving applications on a modern commercial FPGA SoC [13]. A similar approach has also been applied to upper bound the response time of a Pointer Authentication (PAC) module extension for ARMv7 architectures on reconfigurable SoCs platforms [14].

IV. THIRD PILLAR: SECURITY/SAFETY VERIFICATION

As described, the solutions proposed in Section II aim at enhancing the system-level safety and security of the system. Given their criticality, these functionalities should be carefully verified – the verification process provides fundamental support for ensuring the safe and secure operations of the introduced hardware extensions. Recently, property-based security verification processes have been demonstrated to be effective in discovering weaknesses and vulnerabilities when the RTL is available, even in complex scenarios such as when applied to the OpenTitan open-source hardware root of trust [15]. As of today, security verification is mainly a manual process involving, among other steps, the definition of the threat model, the identification of the security assets, and the definition and verification of the security properties – some works have been currently proposed in the direction of partially or completely automating this process [16]–[18].

Property-based security verification can have a fundamental contribution in ensuring the security and safety of newly proposed hardware extensions, providing support with verifiable hardware properties to the aimed functionalities. To make an example, property-based security verification has been effectively applied, for instance, for the verification of the secure operation of a newly proposed access control system framework for modern SoCs [6], [7]. Access control systems are sadly known to be one of the major sources of security issues – the MITRE consortium identifies 5 out of 12 of the most relevant hardware CWE to be related to access control [19]. In this context, the property-based security verification enables the verification of the system against the common bugs and weaknesses by defining properties covering the common hardware weaknesses enumerated by MITRE and verifying them with ad-hoc tools [20].

Similar methodologies apply to other hardware extensions for improving the security of the system, such as the one summarized in this paper in Section II and future solutions. Analogous property-based verification methodologies have also been recently demonstrated for covering safety issues, for instance, in demonstrating the absence of IPs introducing the issue of denial-of-service of shared resources introduced in Section II [21].

V. PUTTING IT ALL TOGETHER IN OPEN PLATFORMS

Open-hardware SoC platforms have reached an impressive level of maturity, providing rich functionalities for the deployment of fully functional SoCs. As mentioned in Section I, we believe that the availability of such mature open-hardware platforms provides unprecedented opportunities for the development, integration, analysis, and testing of the methodologies discussed in the previous sections, such as: (i) the integration and demonstration on a comprehensive testbed of the effectiveness of the proposed architectural extensions (Section II); (ii) the application of timing analysis methodologies and improvement of the methodologies thanks to the whole availability of the RTL of the platform (Section III); and (iii) enhancements in the security and safety verification process, by stimulating knowledge and developments in property definition and automatic generation on the RTL of complete platforms (Section IV). Following, some exciting developments we are actively participating in are briefly summarized.

The Carfield open-hardware project [22] ambitiously aims at providing a fully open-source platform for automotive-driven computing architectures closing the gap between RISC-V and ARM-powered solutions. Carfield is a fully-featured crossbar-based SoC integrating a safety island embedding three 32-bit RISC-V cores executing in lockstep, a security island embedding an OpenTitan hardware root of trust [23], a floating point vector cluster, two RISC-V CVA6 cores [24], and a full set of peripherals such as an HyperBus memory controller [25], [26], scratchpad, and IOs. The resources are interconnected through the Carfield predictable AXI interconnect, which, seamlessly with the AXI HyperConnect introduced in Section II, is in charge not only of implementing the standard routing functionalities, but also of providing active support to the system-level safety and security of the platform, implementing an ad-hoc module [9] deploying a stack of the solutions introduced in Section II. The Carfield project provides an example of how the system architecture of a modern SoC can be enhanced with architectural extensions aimed at enforcing safe and secure system interactions.

Besides architectural extensions, open-hardware platforms enable unprecedented opportunities for real-time analysis – the

full availability of the RTL code of open-hardware platforms enables fine-grained analysis directly at the RTL level, thus enabling computing tighter bounds, and thus reducing the pessimism of the analysis. This has been demonstrated in a recent research project in which the analysis methodologies proposed in Section III have been applied to a fully open-hardware PULP-based platform [27]. The project demonstrates how the full availability of the RTL code can support tighter bounds, and therefore higher timing predictability and less performance waste.

Regarding verification technologies, the availability of the whole RTL design in open-hardware platforms allows the development of hardware properties and enhancements in the verification processes directly on full SoC platforms, without requiring an industrial partner or hitting NDA walls.

The methodologies discussed in this paper are mainly focused on crossbar-based SoC. However, the proposed hardware extensions, timing analysis, and verification methodologies can be extended to other popular SoC architectures – currently, we are working on extending them to Network-on-Chip-based architectures. In this context, our reference is the ESP platforms [28], one of the most mature NoC-based open hardware platforms currently available.

VI. Conclusions

This paper describes recently proposed methodologies for improving the system-level safety and security of SoC platforms for critical applications. The introduced methodologies are based on three pillars: (i) ad-hoc high-performance architectural hardware extensions; (ii) fine-grained timing analysis, and (iii) security and safety verification. The combination of these three pillars aims at the concept of zero-trust hardware architectures, aiming at removing the trust implicitly provided to the computing units while keeping full compliance with the de facto standard for on-chip communications in modern SoCs to guarantee compatibility and performance.

The presented methodologies have been applied and are under application in some of the most relevant open-hardware SoC platforms. We believe that the availability of such mature open hardware platforms provides an unprecedented opportunity for the research community to enhance the security and safety of the next generation of SoC for critical applications. The described methodologies have been applied to crossbar-based SoCs and are under application to NoC-based SoCs. We believe that similar methodologies will be relevant also in enhancing the safety and security of future chiplet-based architectures.

References

[1] *The CWE Official Webpage*, MITRE, https://cwe.mitre.org/.

[2] *AMBA® AXI™ and ACE™ Protocol Specification*, ARM, iHI 0022D.

[3] F. Restuccia, M. Pagani, A. Biondi, M. Marinoni, and G. Buttazzo, "Is your bus arbiter really fair? restoring fairness in AXI interconnects for FPGA SoCs," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 5s, p. 51, 2019.

[4] F. Restuccia, A. Biondi, M. Marinoni, and G. Buttazzo, "Safely Preventing Unbounded Delays During Bus Transactions in FPGA-based SoC," in *2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, 2020.

[5] F. Restuccia and R. Kastner, "Cut and forward: Safe and secure communication for fpga system on chips," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2022.

[6] F. Restuccia, A. Meza, and R. Kastner, "Aker: A design and verification framework for safe and secure soc access control," in *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 2021, pp. 1–9.

[7] F. Restuccia, A. Meza, R. Kastner, and J. Oberg, "A framework for design, verification, and management of soc access control systems," *IEEE Transactions on Computers*, vol. 72, no. 2, pp. 386–400, 2022.

[8] F. Restuccia, A. Biondi, M. Marinoni, G. Cicero, and G. Buttazzo, "Axi hyperconnect: A predictable, hypervisor-level interconnect for hardware accelerators in fpga soc," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2020, pp. 1–6.

[9] B. Thomas, O. Alessandro, B. Robert, G. Angelo, R. Francesco, B. Alessandro, and B. Luca, "AXI-REALM: A Lightweight and Modular Interconnect Extension for Traffic Regulation and Monitoring of Heterogeneous Real-Time SoCs," in *2024 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2024.

[10] A. Biondi, F. Nesti, G. Cicero, D. Casini, and G. Buttazzo, "A safe, secure, and predictable software architecture for deep learning in safety-critical systems," *IEEE Embedded Systems Letters*, vol. 12, no. 3, pp. 78–82, 2019.

[11] F. Restuccia, M. Pagani, A. Biondi, M. Marinoni, and G. Buttazzo, "Bounding memory access times in multi-accelerator architectures on fpga socs," *IEEE Transactions on Computers*, vol. 72, no. 1, pp. 154–167, 2022.

[12] ——, "Modeling and analysis of bus contention for hardware accelerators in fpga socs," in *32nd Euromicro Conference on Real-Time Systems (ECRTS 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[13] F. Restuccia and A. Biondi, "Time-predictable acceleration of deep neural networks on fpga soc platforms," in *2021 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 2021, pp. 441–454.

[14] G. Serra, P. Fara, G. Cicero, F. Restuccia, and A. Biondi, "Pac-pl: Enabling control-flow integrity with pointer authentication in fpga soc platforms," in *2022 IEEE 28th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2022, pp. 241–253.

[15] A. Meza, F. Restuccia, J. Oberg, D. Rizzo, and R. Kastner, "Security verification of the opentitan hardware root of trust," *IEEE Security & Privacy*, 2023.

[16] C. Deutschbein, A. Meza, F. Restuccia, M. Gregoire, R. Kastner, and C. Sturton, "Toward hardware security property generation at scale," *IEEE Security & Privacy*, vol. 20, no. 3, pp. 43–51, 2022.

[17] C. Deutschbein, A. Meza, F. Restuccia, R. Kastner, and C. Sturton, "Isadora: Automated information flow property generation for hardware designs," in *Proceedings of the 5th Workshop on Attacks and Solutions in Hardware Security*, 2021, pp. 5–15.

[18] R. Kastner, F. Restuccia, A. Meza, S. Ray, J. Fung, and C. Sturton, "Automating hardware security property generation," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, 2022, pp. 1384–1387.

[19] *The 2021 MITRE CWE Most Important Hardware Weaknesses*, MITRE, https://cwe.mitre.org/scoring/lists/2021_CWE_MIHW.html.

[20] *The Cycuity Radix-S offical website*, Cycuity, https://cycuity.com/solutions/.

[21] A. Meza, F. Restuccia, and R. Kastner, "Safety verification of third-party hardware modules via information flow tracking," in *1st Real-time And intelliGent Edge computing workshop (RAGE) co-located with the 2022 59th Design Automation Conference (DAC)*.

[22] *The Carfield Github Repository*, PULP platform - University of Bologna and ETH Zurich, https://github.com/pulp-platform/carfield.

[23] *The OpenTitan official website*, https://opentitan.org/.

[24] *The CVA6 official Github*, Open Hardware Group, https://github.com/openhwgroup/cva6.

[25] PULP, "HyperRAM Controller RTL," 2022, https://github.com/pulp-platform/hyperbus.

[26] Infineon, "HyperBUS specifications," 2022.

[27] L. Valente, F. Restuccia, D. Rossi, R. Kastner, and L. Benini, "Top: Towards open & predictable heterogeneous socs," *arXiv preprint arXiv:2401.15639*, 2024.

[28] *The ESP open-hardware platform official website*, Columbia University, https://www.esp.cs.columbia.edu/.