engineering proceedings

MDPI

Proceeding Paper

FPGA Prototyping of Heterogeneous Security Architecture for Educational Purposes †

Stefan Stoyanov, Nikolay Kakanakov * and Maria Marinova

Department Computer Systems and Technologies, Faculty of Electronics and Automation, Technical University of Sofia Branch Plovdiv, 4000 Plovdiv, Bulgaria; s.stoyanov@std.tu-plovdiv.bg (S.S.); m_marinova@tu-plovdiv.bg (M.M.)

* Correspondence: kakanak@tu-plovdiv.bg

† Presented at the 14th International Scientific Conference TechSys 2025—Engineering, Technology and Systems, Plovdiv, Bulgaria, 15–17 May 2025.

Abstract

Modern day hardware design is heavily focused on software simulations and verification. ASIC design is complex, and its abstraction may lead to losing motivation to develop knowledge and skills in the field. It is also very costly and inaccessible outside the industry, especially on a larger scale. FPGAs are devices which are affordable and allow unexperienced people, like students, to have an accessible and observable starting point in hardware development. On the other hand, protocol standards and devices nowadays are so complex that one will need to spend years to understand and feel comfortable dealing with corporate methodologies, environments and development process. One of the still evolving, recently developed ISA for example, RISC-V, is a broad area to learn. Often the fastest and in-depth way to learn is by observation, getting familiar with accessible yet challenging design and making modifications to and experimenting with such platforms.

Keywords: FPGA; security; education; hardware design; computer architectures; RISC-V

check for updates

Academic Editors: Nikola G. Shakev, Valyo N. Nikolov and Sevil A. Ahmed-Shieva

Published: 7 July 2025

Citation: Stoyanov, S.; Kakanakov, N.; Marinova, M. FPGA Prototyping of Heterogeneous Security Architecture for Educational Purposes. *Eng. Proc.* **2025**, *100*, 18. https://doi.org/10.3390/engproc2025100018

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Hardware development is demanding because it requires broad knowledge of aspects from electronics to programming and at the same time in-depth understanding of architecture, hardware algorithms and hardware development principles, using specific CAD tools and many more. Unless working in a bring-up lab, however, often the final product remains disconnected from developers somewhere at the end of a long and complex process. This process consists of many steps and layers of engineers specialized in every part of it. So, the question is how a student can be exposed to an overview of that long development cycle in a limited time course and at the same time find hardware design appealing and draw their attention to education further.

In an incredibly fast-evolving device world, where standards, protocols and architectural solutions are constantly introduced at a very fast rate, the next objective is to figure out what are the key design elements that are always present in a device and what modern protocols and problems are there so that not only knowledge of the process but also understanding of what this knowledge is useful for can be acquired [1].

And finally, the complexity of modern-day circuits is so great that not only a single man but also a single team is insufficient to create a device from A to Z. Instead, it requires the industry to use global collaboration efforts with teams spread all around the world where qualities like the ability to work within a team and global culture environment,

understanding how complex schematics work and how to develop as part of a larger scale unit, even reverse-engineering, should be established [2].

2. Benefits from Using FPGA

2.1. FPGA Specifics

FPGAs or Field Programmable Gate Arrays are chips that have the capability to operate as a hardware logic device and to be reprogrammed. Unlike ASIC chips, the logic is not etched but they consist of cells (CLBs) that can be programmed to operate as logic gates. While ASIC designs require a significant amount of time and money to be fabricated, creating designs for FPGAs is easier, cheaper and most importantly does not carry the burden of utterly costly design or verification errors [3].

Compared to ASICs, however, FPGAs have different design considerations in terms of operating frequency, number of logic gates, power consumption and wires, and this primarily means less resources, slower frequencies, a limited number of gates and memories, etc. [4].

There is a large variety of FPGA devices in the market, and they can be used to address various applications. High-capacity devices or high-end FPGAs are used for industrial prototyping where large SoCs functionality is tested before becoming an ASIC and costs tens of thousands of dollars. There are also low-cost small devices that can be used for small controllers, training purposes or even as hobby kits [5].

An FPGA development kit is a board that has an FPGA chip, typically external programmable memory to store the programmable image and wide variety of combinations of various peripherals—RAMs, communication interfaces like USB, SPI, Ethernet, storage SD Card slots, output displays of many types including LEDs, control switches and more. When choosing a kit, it is very important to consider what exactly will be required, although some FPGA boards even have ports for expansion boards that allow upgrading [6].

Modern FPGA devices also have hybrid functionality in terms of two options:

- Additional controllers or processors on board, that can work in parallel with the programmed logic in the FPGA chip;
- Microprocessors, communication IPs or various other blocks that are part of the FPGA silicon.

Another very important feature related to the designing logic for FPGAs is software. Besides tools that are mandatory to design and verification in terms of logical simulations, a device-targeted software is required to be able to synthesize, place and route, create an image and program that specific FPGA. These come either with a development board or have to be bought separately.

Such software often comes with an example of IPs and IPs that can operate with the various already mentioned peripheral devices [7].

2.2. Educational Benefits

Outlining that some of the marketed devices are advertised as hobby kits, it is easy to assume that low-end and even mid-level FPGAs are an affordable and easy to learn and use platforms that one can exercise and experiment with developing a variety of hardware architectures. The learning process has three aspects that suit the FPGA features very well:

- Learning is an incremental process—one cannot start with designing very complex blocks. Instead, it will take many steps to build knowledge and expertise while progressing. This will be achieved by creating different logic blocks with increasing complexity. The reprogrammability of FPGAs allows these steps to be tracked and brought to completeness seeing every design operational.

- Learning is error-prone—as mentioned earlier, ASIC design faults are very costly—the cost is in terms of time to debug, find software workaround or worst-case large amount of money when it requires respin [8]. FPGA design errors require only time to debug and reprogram.

Learning requires attention—it is very difficult to doubt that the excitement of a student seeing their device make an LED blink is much greater than running a logic simulation on a computer. From a motivational point of view, being able to observe and interact with the product of one's effort is much more satisfying than merely checking a log or waveforms.

From a practical point of view, it is very useful that some of the FPGA platforms come with software that is sufficient to cover the entire design process—from coding IDE, running simulations and everything else that will result in a programmed FPGA image. As this is not a single step process, using a single tool will keep the focus on the design rather than learning how to use various programs for the individual steps.

Being able to create a block that interacts with an already existing device in terms of vendor-provided IPs is another useful aspect, which will provide knowledge and an understanding of integration, communication protocols and allow higher scale final design.

3. Key Architectural Elements to Gain Practical Development Knowledge

Traversing through any common university textbook [9], one can easily outline the big areas that have stayed present in architecture and which a future computer expert needs to have profound knowledge of:

- Pipelines and processing—this comes down to many aspects like data path and control path, instruction sets, parallelism, data and control hazards, RISC and CISC architectures but, in general, it is a specific ISA processor and its organization and architecture.
- Memories and memory hierarchy—this includes various protocols and device types—DRAMs, caches, flash memories, permanent storage devices, as well as the organization of storage in general, addressing, data storage and retrieval, etc.
- Intra- and inter-chip communication, inter-system communication (networking)—various bus protocols within the chip, modern high-speed serial communication between chips, network communication protocols—all these are a way for data to travel from one storage or processing point to another.

Along with this, modern day computing is exposed to numerous security hazards and issues, which adds one bigger field to this list, which is security. It spans over the already mentioned three. Hardware devices have many architectural elements to prevent different threats like data theft, system hijacking, communication interception, etc. As all architecture elements evolve to seek better performance, security solutions have to adapt and outsmart all arising means of attack.

4. Heterogeneous Security Architecture for Educational Purposes

4.1. Architecture Overview

Figure 1 shows the top-level block diagram of the Heterogeneous Security Architecture [10]. This architecture consists of two cores, a CPU and a tightly coupled small security processor core, meant to serve security purposes like encryption, protected key generation and storage as well as active monitoring of the central processor operation [11–13]. The security core is isolated [14] from the CPU but the two can communicate actively and this communication can be completely controlled and secured by the protected module. All system resources are accessible by the protected core with hardware-implemented higher

privileges and still some of the resources, related specifically to data protection, can be only accessed by it (marked in dark blue on the diagram) [15].

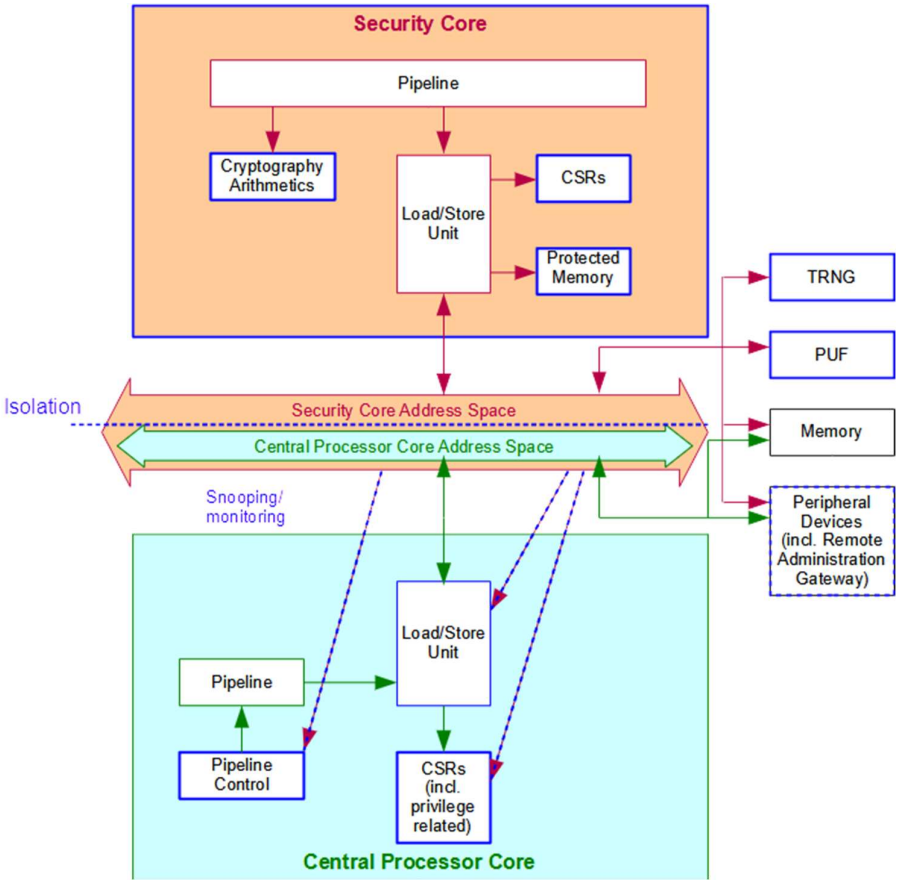

Figure 1. Heterogeneous Security Architecture.

4.2. Implementation Notes

All components are specially designed for this system. The security core is an M-mode only RV64I five-stage pipeline [16]. For simplicity at this stage, the CPU is a subversion of the same pipeline with additional hardware security features controlled by the other core. The idea is that by reusing these security features, this CPU can be easily replaceable by any other processor or having more than one CPU all controlled in the same way. In case of more than one CPU it is possible to also have different processor types.

The architecture uses a standard RISC-V tool chain embedded in a simulation and programming environment [17]. RTL is implemented in SystemVerilog [18].

4.3. State of Completeness

The Heterogeneous Security Architecture is designed and implemented as a PhD dissertation project with the goal to have a small-sized but scalable IoT-applicable device that provides efficient information security [19–22]. Although it has reached a certain state of completeness many features can be thoroughly examined, analyzed and enhanced—both in terms of hardware architecture and components and in terms of software running on the two cores.

It has only passed logical simulations, and the next step is FPGA prototyping, so performance is yet to be examined.

Additionally, the complete development of a project of such scale requires much greater human resources.

Some of the hardware aspects that can be updated include the following:

- Integration of more advanced memory management system—the current Load/Store Unit is very basic, and memory management is crucial to making the system more practical but it also is a component that improves security allowing address virtualization [23].
- True Random Number Generator (TRNG) [24] and Physically Unclonable Function (PUF) [25]—these two security components are each an aspect of advanced research especially in the scope of FPGAs. As of now, pseudo-random number generator and fixed PUF signature placeholders are used which allows working of the architecture itself. Having them in the future, when the system targets real applications, is mandatory.
- Adding different peripheral devices—all sorts of peripheral devices can be added to the system and test their compatibility and reliability.
- Multi-CPU system.
- Modified RISC-V core—the RV64I pipeline can be enhanced to support additional RISC-V extensions based on performance analysis. Additionally custom instructions can be added, for example, to optimize the performance of security algorithms.

Currently developed software is intended to demonstrate some of the security characteristics of architecture. This platform can also be used for the following:

- Testing the reliability of hardware security;
- Testing different security algorithms in the scope of a real system;
- Development of multi-processor communication protocols;
- Development of real-time operation software.

5. Conclusions

The target of this article is to suggest an accessible way to expose students to hardware design, modern day architecture and gain practical skills in development in Computer Systems Design, Computer Architectures and various other hardware related university courses.

Although the architecture is already implemented and was intended as a small-area and power-consumption device, it is not yet FPGA tested and can be enhanced and upgraded in many ways in terms of processing, memory system, communication and security, which are all important fields of study. Building different components of a SoC system can be used to promote teamwork among students.

RISC-V is a modern open-source instruction-set architecture that has been adopted and improved by the biggest companies in the hardware industry in the recent years. Still, it is developed and widely used by academia [26]. The Heterogeneous Security Architecture was developed after an analysis of modern hardware solutions provided information security [1].

SystemVerilog is the most widely used hardware definition language and an industry standard. FPGA prototyping on a larger scale is a common way to test pre-silicon designs to avoid costly errors, and smaller-sized FPGAs are widely used in cheaper small-sized logic solutions. This will allow not only having huge possibilities for development and experiments but also getting in touch with "how things are done", exposing students to good practices or even how to overcome the difficulties of understanding design basics.

Author Contributions: Conceptualization, S.S., N.K. and M.M.; methodology, M.M. and N.K.; software, S.S.; validation, N.K. and M.M.; writing—original draft preparation, S.S.; writing—review and editing, N.K.; visualization, S.S.; supervision, M.M.; funding acquisition, N.K. All authors have read and agreed to the published version of the manuscript.

Funding: The authors gratefully acknowledge the financial support provided within the Technical University of Sofia, Research and Development Sector, Project for PhD student helping, contract № 251ПД0046-19 "Platform for investigation of heterogenous processor architectures for information protection".

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are included in the article.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

ASIC	Application Specific Integrated Circuit
CLB	Cell Logic Block
CSR	Control and/or Status Register
FPGA	Field-Programmable Gate Array
IDE	Integrated Development Environment
ISA	Instruction Set Architecture
PUF	Physically Unclonable Function
RTL	Register Transfer Logic
TRNG	True Random Number Generator

References

1. Stoyanov, S.; Kakanakov, N.; Marinova, M. Survey on Hardware Components Providing Information Security. *AIP Conf. Proc.* **2025**, *3274*, 040007. [CrossRef]
2. Peters, M.S.; Eppinger, S.D. Faster and Faster: Automating the Hardware Engineering Process. In *DS 130: Proceedings of NordDesign 2024, Reykjavik, Iceland, 12–14 August 2024*; Massachusetts Institute of Technology, United States of America: Cambridge, MA, USA, 2024. [CrossRef]
3. Benefits of FPGAs over Other Options for AI and Machine Learning. Available online: https://www.sundance.com/fpga-benefits/ (accessed on 14 May 2025).
4. Papamichael, M.K.; Hoe, J.C. Connect: Re-examining conventional wisdom for designing NOCs in the context of FPGAs. In Proceedings of the International Symposium on Field Programmable Gate Arrays, Monterey, CA, USA, 22–24 February 2012. [CrossRef]
5. AMD FPGAs. Available online: https://www.amd.com/en/products/adaptive-socs-and-fpgas/fpga.html (accessed on 15 May 2025).
6. FPGA Boards—A Smart Start to Powerful Solutions. Available online: https://digilent.com/shop/fpga-boards (accessed on 15 May 2025).
7. AMD Vivado™ Design Suite. Available online: https://www.amd.com/en/products/software/adaptive-socs-and-fpgas/vivado.html (accessed on 15 May 2025).
8. ASIC Design Flow in VLSI Engineering Services—A Quick Guide. Available online: https://www.einfochips.com/blog/asic-design-flow-in-vlsi-engineering-services-a-quick-guide/ (accessed on 15 May 2025).
9. Patterson, D.A.; Hennessy, J.L. *Computer Organization and Design—The Hardware/Software Interface: RISC-V Edition*; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 2017.
10. Stoyanov, S.; Kakanakov, N.; Marinova, M. Secure Heterogeneous Architecture Based on RISC-V and Root-of-Trust. In Proceedings of the 24th International Conference on Computer Systems and Technologies, Ruse, Bulgaria, 16–17 June 2023; ACM International Conference Proceeding Series. pp. 19–23. [CrossRef]
11. Lestari, D.; Tama, A.; Karlina, S.; Sultan, A.; Tarwoto, T. Factors Affecting Security Information Systems: Information Security, Threats and Cyber Attack, Physical Security, and Information Technology. *Int. J. Inform. Inf. Syst.* **2024**, *7*, 16–21. [CrossRef]
12. Anoop, M.S.; Security Needs in Embedded Systems. Cryptology ePrint Archive 2008. Available online: https://eprint.iacr.org/2008/198 (accessed on 3 July 2025).

13. Mukhopadhyay, D.; Chakraborty, R.S. *Hardware Security: Design, Threats and Safeguards*; Chapman and Hall/CRC Press: Boca Raton, FL, USA, 2014.
14. Chen, Y.; Chen, Y.; Zhu, M.; Guo, N. Data Security Isolation and Exchange for Smart Grid. *IOP Conf. Ser. Earth Environ. Sci.* **2019**, *242*, 022039. [CrossRef]
15. Privilege Level. Available online: https://www.sciencedirect.com/topics/computer-science/privilege-level (accessed on 26 January 2024).
16. Waterman, A.; Asanovic, K.; Hauser, J. *The RISC-V Instruction Set Manual Volume II: Privileged Architecture, Document Version 1.12-Draft*; RISC-V Foundation: San Francisco, CA, USA, 2022.
17. RISC-V Software Ecosystem. Available online: https://tech.riscv.org/ecosystem/public/dashboard/9f6bf4ae-99bd-4717-b351-03465e8cf5f6 (accessed on 3 July 2025).
18. *IEEE Std 1800-2017*; IEEE Standard for SystemVerilog—Unified Hardware Design, Specification, and Verification Language. IEEE Standards Association Corporate Advisory Group: New York, NY, USA, 2017.
19. 10 IoT Security Challenges and Solutions to Protect Your Devices. Available online: https://www.linkedin.com/pulse/10-iot-security-challenges-solutions-protect-your-2023-jansasoy-/ (accessed on 24 July 2023).
20. Chen, C.-Y.; Hasan, M.; Mohan, S. Securing Real-Time Internet-of-Things. *Sensors* **2018**, *18*, 4356. [CrossRef] [PubMed]
21. Dhanda, S.S.; Singh, B.; Jindal, P. Lightweight Cryptography: A Solution to Secure IoT. *Wirel. Pers. Commun.* **2020**, *112*, 1947–1980. [CrossRef]
22. Mishra, P.; Bhunia, S.; Tehranipoor, M. (Eds.) *Hardware IP Security and Trust*; Springer: Cham, Switzerland, 2017.
23. Heninger, N. *Introduction to Security: Isolation and Secure Design*; University of California: San Diego, CA, USA, 2021.
24. Rozic, V.; Yang, B.; Dehaene, W.; Verbauwhede, I. Highly Efficient Entropy Extraction for True Random Number Generators on FPGAs. In Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, 7–11 June 2015. [CrossRef]
25. Cirne, A.; Sousa, P.R.; Resende, J.S.; Antunes, L. Hardware Security for Internet of Things Identity Assurance. *IEEE Commun. Surv. Tutor.* **2024**, *26*, 1041–1079. [CrossRef]
26. About RISC-V. Available online: https://riscv.org/ (accessed on 10 May 2025).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.