



Article

<https://doi.org/10.32523/3136-3385-2026-156-3-254-270>

Development Flow of a Digital Signature Cryptographic Controller with FPGA Verification and MPW Prototyping

I. Seniushin , **N. Glazyrina*** , **R. Ibrayev** , **Y. Atanbayev** 

TSARKA R&D LLP, Astana, Kazakhstan

E mail: sigor@cybersec.kz, glazirinan@yandex.ru, crypto.teacher2020@gmail.com, yernatatanbayev@gmail.com

Abstract. This article proposes and experimentally validates a formalized development flow for a specialized cryptographic controller intended for the hardware implementation of electronic digital signature algorithms. The proposed flow is presented as a methodology covering the full development cycle, from architectural design to preparation for MPW fabrication. The controller architecture is based on a modular principle: the system consists of functionally independent hardware blocks developed and verified as separate modules. For each module, RTL implementation, development of a hardware abstraction layer (HAL), and hardware verification at the FPGA-prototyping stage are performed. After hardware verification, critical modules undergo physical validation within the MPW flow. After modular verification, the blocks are integrated into the system, the final memory map is generated, and system-level hardware verification is performed. The methodology was experimentally validated on a TinyQV demonstration computing module compatible with the RISC-V ISA. Hardware verification was performed on the GOWIN Tang Primer 20K FPGA platform, and physical implementation was carried out in the TinyTapeout MPW flow using the SkyWater SKY130 PDK.

Keywords: cryptographic controller, electronic digital signature, FPGA prototyping, hardware verification, MPW prototyping, information security.

Introduction

Ensuring secure electronic document management and electronic transactions is currently one of the key information-security challenges in government, financial, and corporate information systems. To address this challenge, electronic digital signatures (EDS) are widely used as a cryptographic mechanism that provides authenticity of origin, data integrity, and non-repudiation for electronic documents and transactions [1-4].

An electronic digital signature can be implemented in software or hardware [5]. Software and hardware/software implementations are typically based on general-purpose microcontrollers, where cryptographic operations are executed in software. A hardware implementation involves executing cryptographic operations and storing private keys in a specialized integrated circuit, such as a cryptographic controller, or in other dedicated hardware [6]. This approach increases security by isolating key material and reducing the impact of software vulnerabilities [7-9].

Because hardware implementations of electronic digital signatures provide a higher level of security than purely software-based approaches [7], the development of specialized hardware solutions has become an active research area and is well represented in recent scientific literature. A significant number of studies focus on hardware implementations of cryptographic algorithms for electronic digital signatures. Papers [10-12] present RSA hardware-accelerator architectures implemented on FPGA and ASIC platforms. Studies [13-15] propose hardware-accelerator architectures for elliptic-curve algorithms, including ECDSA and EdDSA. Hardware implementations of national cryptographic standards are also presented in [16, 17], where cryptographic coprocessor architectures and hardware cores for GOST algorithms are proposed.

In parallel with the development of cryptographic accelerators, methods for designing and verifying systems-on-chip (SoCs) that incorporate cryptographic modules are also being actively investigated. Various approaches to functional and formal verification of SoC architectures have been proposed in [18-21], including assertion-based verification, dynamic information-flow analysis, and the Universal Verification Methodology (UVM). FPGA prototyping is widely used for practical verification of architectural solutions, enabling pre-silicon validation and embedded-software bring-up before chip fabrication [22-24]. Open-source ASIC physical-design flows are also maturing, and the feasibility of implementing cryptographic hardware blocks using an open-source RTL-to-GDS flow based on the SkyWater 130 nm process technology has been demonstrated in [25-27].

Despite the substantial body of research on hardware implementations of cryptographic algorithms and methods for designing secure SoCs, the scientific literature still lacks a formalized and reproducible development flow for cryptographic controllers [28]. A consistent approach that covers architectural design, RTL implementation, hardware verification, and subsequent physical implementation as an application-specific integrated circuit (ASIC) has not yet been sufficiently developed [29, 30]. As a result, architectural and interface errors are often detected at late stages of development, including physical prototyping, which increases development time, cost, and technological risk [31, 32].

This paper presents the results of applied research conducted during the development of a specialized cryptographic controller with hardware implementation of digital signature algorithms. Practical validation of the architectural and methodological solutions was carried out within research project AP26103891, "Development of a domestic cryptographic controller and its implementation in a key-storage device," funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan. The cryptographic controller under

development is intended for use in a cryptographic key storage device (USB token) that provides secure authentication and data protection in systems supporting the Fast Identity Online (FIDO) standard [33, 34] and digital signature algorithms.

The cryptographic controller under development imposes strict requirements on the correctness of hardware implementations of cryptographic algorithms and on the reliability of individual modules. Together with the high cost of correcting architectural errors at the physical-prototyping stage, these requirements make it necessary to formalize the development flow.

Accordingly, this paper proposes a formalized development flow for a cryptographic controller. The flow is presented as a methodology based on step-by-step hardware and physical-design verification of individual functional modules, followed by integration testing of the complete system. This approach enables early identification of architectural and interface inconsistencies and mitigates technological risks during the transition to ASIC implementation.

The aim of this work is to develop and experimentally verify a formalized development flow for a specialized cryptographic controller implementing electronic digital signature algorithms, based on step-by-step hardware and physical-design verification of functional modules and of the integrated system as a whole.

To achieve this aim, the following tasks were performed:

1. Develop a modular architecture for a specialized cryptographic controller implementing electronic digital signature algorithms.
2. Define a formalized development flow for a tape-out-ready cryptographic controller.
3. Implement a demonstration processing module at the RTL level as part of the experimental validation of the proposed flow.
4. Conduct FPGA-based hardware verification of the demonstration module.
5. Perform physical design of the demonstration module in the RTL-to-GDS flow, including synthesis, placement, routing, and DRC/LVS verification within the MPW platform.
6. Demonstrate the applicability of the proposed development flow to the integrated cryptographic controller architecture.

The scientific novelty of this paper lies in the creation of a formalized development flow for a specialized cryptographic controller dedicated to digital signature algorithms. This flow is characterized by a step-by-step methodology that combines FPGA-based hardware verification with an RTL-to-GDS pipeline for MPW preparation, first for individual functional modules and then for the integrated system as a whole.

The methodology

This paper proposes a formalized methodology for developing a specialized cryptographic controller that covers the entire development process, from system architecture design to physical implementation as a specialized integrated circuit. The methodology is oriented toward a reproducible open-source tool flow. Its stages are described below.

The first stage involves architectural design of the cryptographic controller, including definition of the system structure and the composition of its functional modules: computational, system, and specialized cryptographic units.

The second stage involves modular development, in which each functional block is implemented at the RTL level and functionally verified. In parallel, a modular hardware abstraction layer (HAL) is developed to provide a software interface to the hardware components. Next, FPGA-based hardware verification of the modules is performed to confirm their correct operation in a

real hardware environment.

For physical validation of critical modules, the MPW approach is used together with an automated RTL-to-GDS flow that includes logic synthesis, placement, routing, and DRC/LVS checks, thereby verifying the physical implementation at the physical-design stage.

After modular verification is complete, the blocks are integrated step by step into the overall architecture, the system memory map is generated, and the HAL is finalized.

The integrated architecture then undergoes system-level FPGA-based hardware verification, including verification of inter-module interactions and firmware execution.

At the final stage, the integrated system is implemented through the RTL-to-GDS flow and submitted to an MPW shuttle for fabrication of a prototype integrated circuit.

The general flowchart for developing a cryptographic controller within the proposed methodology is shown in Figure 1.

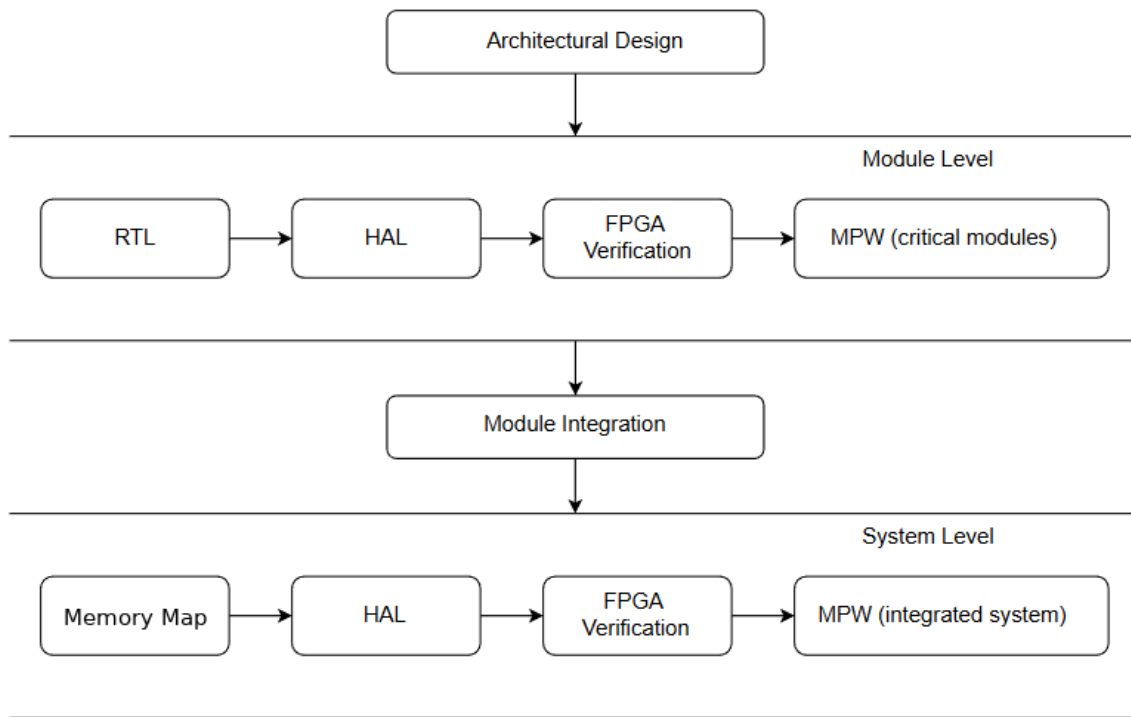


Figure 1. Formalized development flow for a cryptographic controller with step-by-step hardware and physical-design verification

For physical design, the proposed methodology uses the MPW approach, which enables prototype integrated circuits to be fabricated at significantly lower cost and with shorter turnaround times than dedicated ASIC fabrication.

The MPW approach requires a platform that provides a reproducible and technologically reliable prototyping flow, an open PDK, and automated design tools.

To this end, a comparative analysis of available MPW platforms was conducted with regard to supported process technology, openness of the tool environment, and degree of design-flow automation (Table 1).

The comparison shows that, among the platforms considered, TinyTapeout combines support for the open-source SkyWater SKY130 PDK, a fully automated design flow, and open accessibility for research and small-scale projects. These factors determined its use in the proposed methodology.

Table 1. Comparison of MPW prototyping platforms by process and tool-flow characteristics

Platform	Process	Participation cost	Availability	Open-source PDK (SkyWater)	Automated design flow	Tool flow
Global Foundries MPW [35]	GF130, GF180, 65 nm, 22 nm	Upon request	Limited (corporate access, NDA)	No	No	Commercial EDA
MOSIS [36]	SkyWater SKY130, TSMC, GF, OnSemi	~10 000–12 000 USD	Limited (through broker)	Limited (through broker)	No	Commercial EDA
Muse Semiconductor [37]	TSMC 180 nm, 65 nm	1 250–14 000 USD	Open	No	No	Commercial EDA
Cadence MPW (SkyWater)[38]	SkyWater SKY130	~10 000–12 000 USD	Limited (corporate programs)	No	No	Cadence EDA
TinyTapeout [39]	SkyWater SKY130	~385 €	Open	Yes	Yes	Yosys, OpenLane, Magic, KLayout

Findings/Discussion

Architecture of the Dedicated Cryptographic Controller

The architecture of the dedicated cryptographic controller is modular and includes a base processor module, a system bus, a memory subsystem, peripheral interfaces, and specialized hardware blocks designed to implement cryptographic functions.

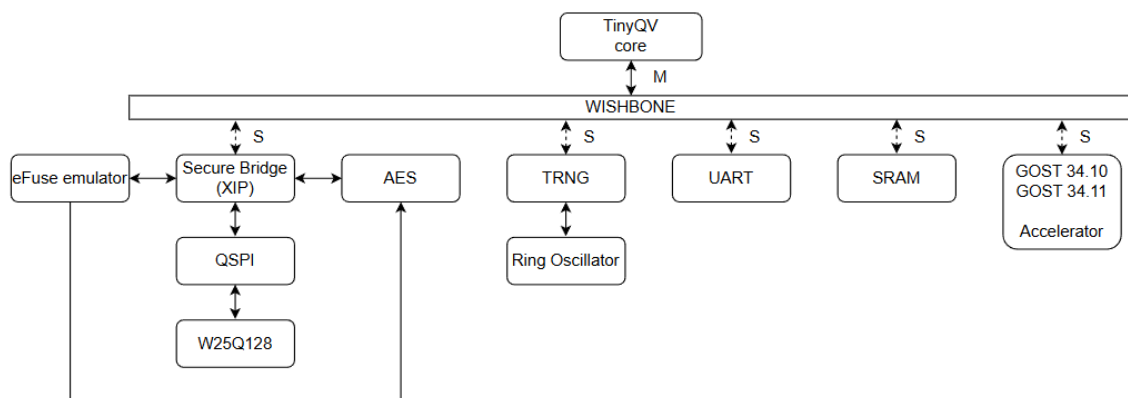


Figure 2. Architecture of a dedicated cryptographic controller

Architecture of the Demonstration Processing Module

In this paper, the demonstration module is the base processing unit of the cryptographic controller, namely the TinyQV core. For this module, the complete development and MPW-prototyping cycle was carried out, from RTL description to preparation for fabrication. Figure 3 shows the architecture of the demonstration core module.

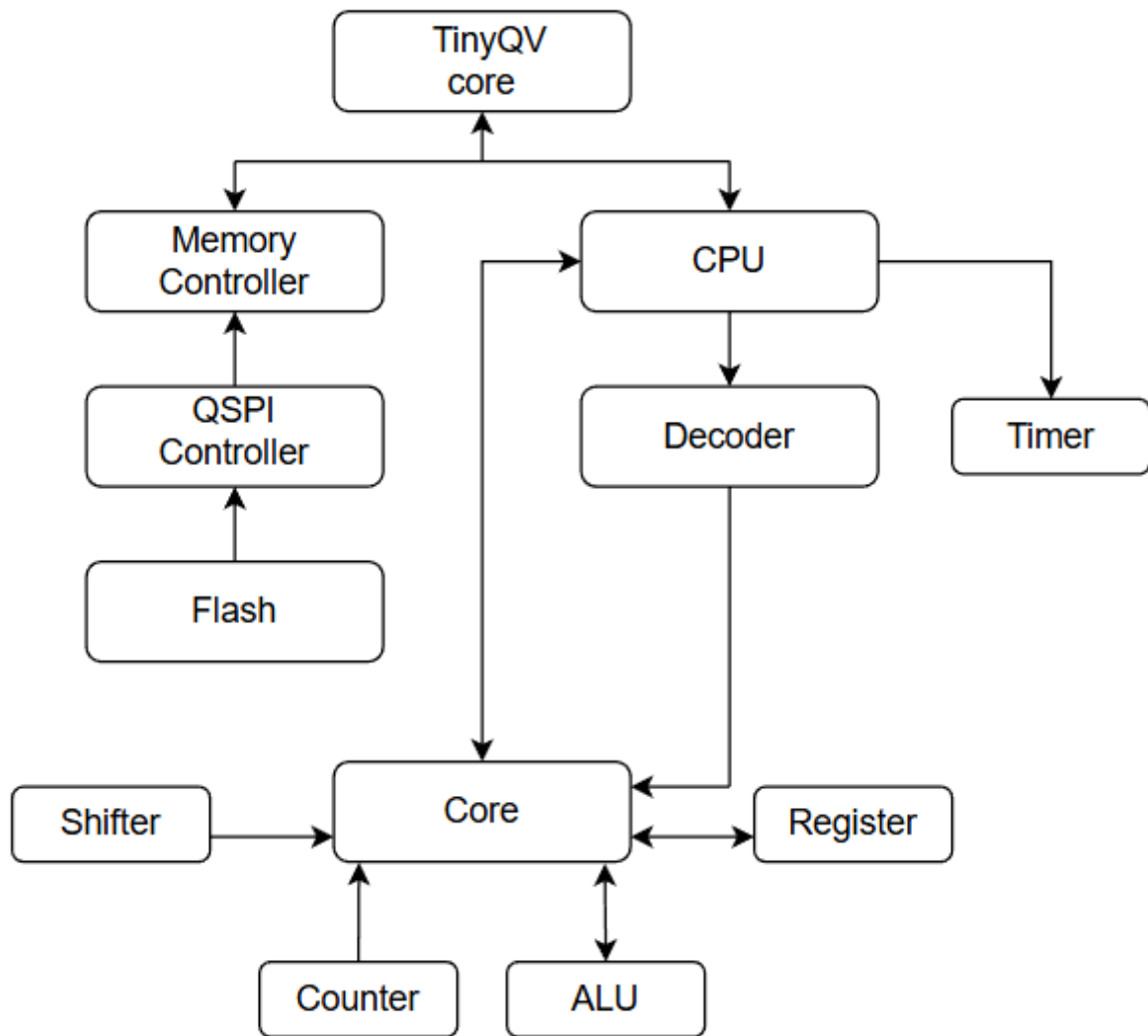


Figure 3. Architecture of the TinyQV demonstration core module

The architecture of the TinyQV demonstration processing unit includes a processor core, an instruction fetch and decode unit, a register file, an arithmetic logic unit, a timer, a counter, and a memory-access subsystem with a QSPI interface for external flash memory.

RTL Implementation of the Demonstration Core Module

The RTL implementation of the TinyQV processing core is written in Verilog HDL and organized as a module hierarchy with the top-level module `tinyqv.v`. The processor subsystem includes an instruction decoder for the RV32EC ISA, a register file, an execution unit with 4-bit nibble-serial data processing, an arithmetic logic unit (ALU), a shift unit, and a timer. The memory

subsystem is implemented using a memory-access controller and a QSPI controller. The RTL module hierarchy of the TinyQV processing core is shown in Figure 4.

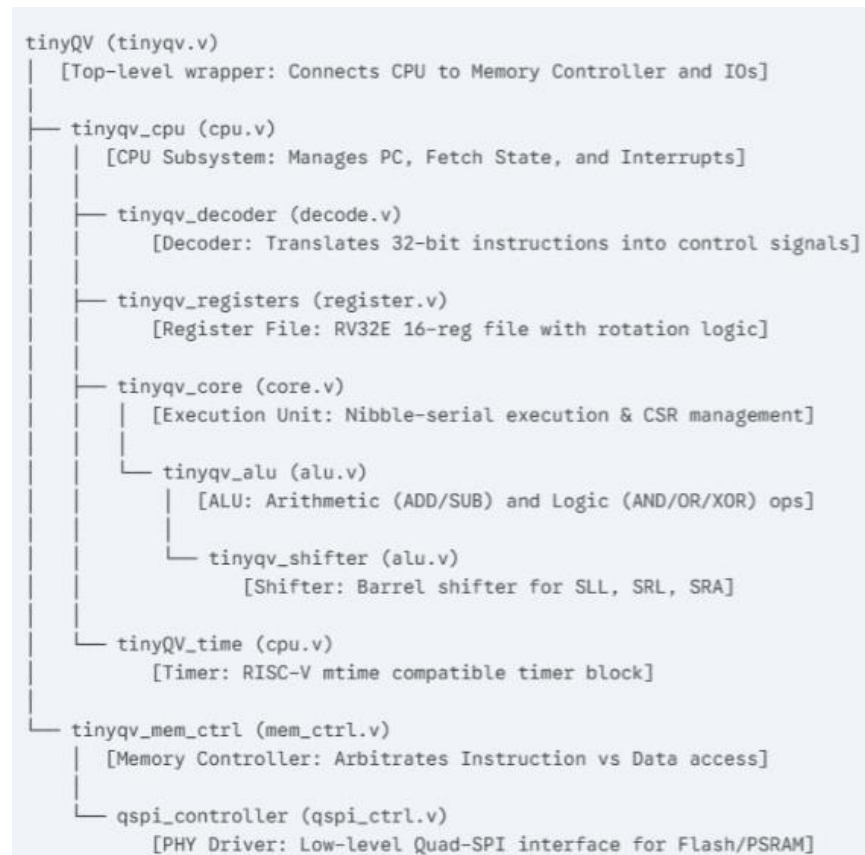


Figure 4. Hierarchy of RTL modules of the TinyQV processing core

Hardware Verification of the Demonstration Processing Module

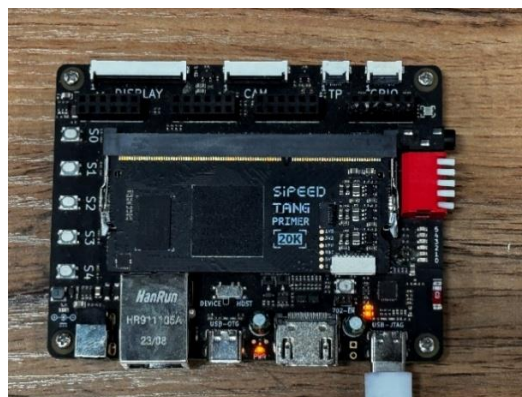


Figure 5. GOWIN Tang Primer 20K FPGA platform

Hardware verification of the RISC-V-compatible TinyQV processing core was performed on the GOWIN Tang Primer 20K FPGA platform (Figure 5).

The memory subsystem in the FPGA design was implemented using GOWIN FPGA BRAM primitives in single-port mode. To implement the memory-access scheme, the `sim_qspi.v` module was developed. This module contains one read-only memory (ROM) block for storing program code and two random-access memory (RAM) blocks for storing data. It enables interaction between the TinyQV computational core and the memory subsystem during FPGA verification. The structural diagram of the interaction between the computational core and the memory subsystem is shown in Figure 6.

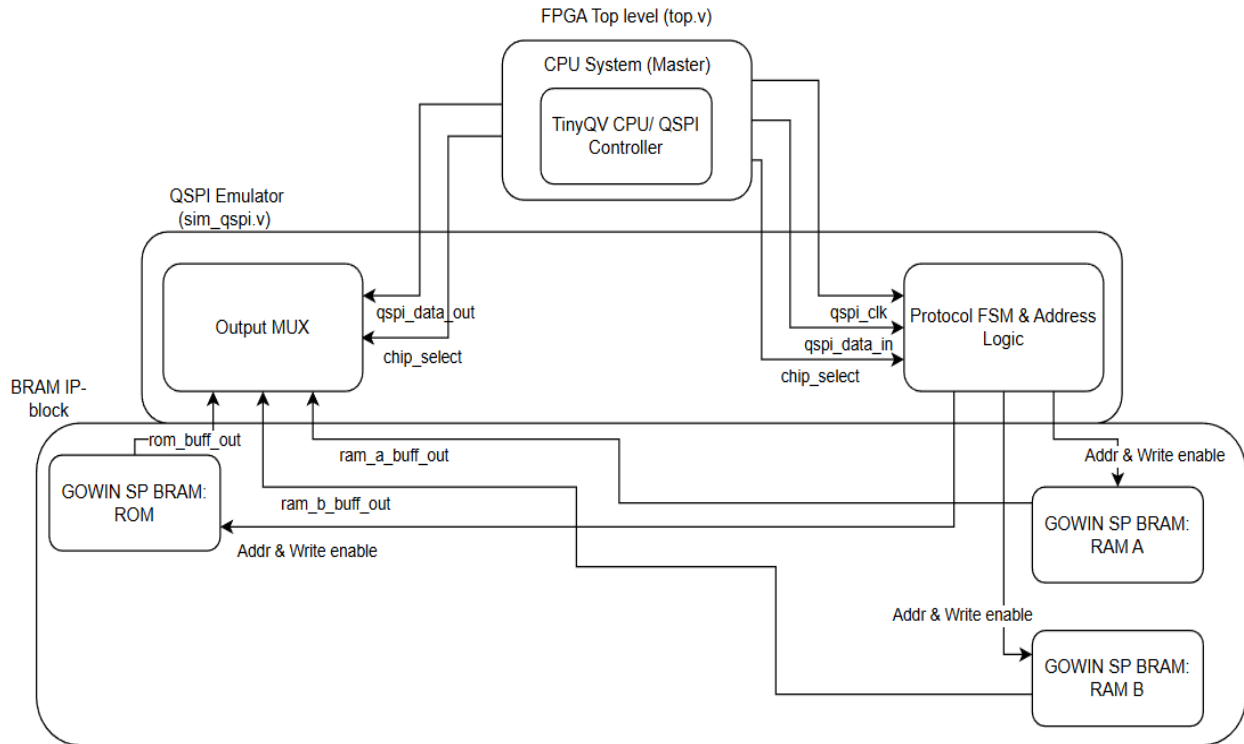


Figure 6. Structural diagram of the interaction between the TinyQV processing core and the memory subsystem during FPGA verification

A fragment of the RTL description of the `sim_qspi.v` module, illustrating the instantiation and connection of GOWIN_SinglePort BRAM primitives, is shown in Figure 7.

Memory addressing is byte-based. Instructions are fetched as 32-bit words formed by sequentially concatenating four bytes in little-endian order.

During the FPGA verification phase, correct operation of the memory subsystem was verified. Byte addressing, data integrity during read and write operations, and the absence of access conflicts in single-port mode were monitored. The results confirmed correct implementation of the memory-access scheme.

After correct operation of the memory subsystem had been confirmed, a test program in HEX format, generated in little-endian order, was loaded into BRAM. Memory initialization was performed byte by byte in accordance with the instruction-fetch order of the TinyQV processing core.

During verification, correct formation of the 32-bit instruction word, decoding accuracy, data integrity during BRAM reads, and correct write-back of the result to the register file were checked.

The verification results confirmed that instructions are correctly read from memory and executed by the processing core.

```

        wire wre3;
        assign wre3 = reading ? (1'b0) : (!addr[0] & !qspi_flash_select);

    Gowin_SP rom(
        .dout(rom_buff_out),
        .clk(qspi_clk),
        .oce(1'b1),
        .ce(1'b1),
        .reset(!rst_n),
        .wre(1'b0),
        .ad(addr[ROM_BITS:1]),
        .din({data_buff_in, qspi_data_in})
    );

    wire wrel;
    assign wrel = writing & addr[0] & ~qspi_ram_a_select;

    Gowin_SP ram_a(
        .dout(ram_a_buff_out),
        .clk(qspi_clk),
        .oce(1'b1),
        .ce(1'b1),
        .reset(!rst_n),
        .wre(1'b0),
        .ad(addr[RAM_A_BITS:1]),
        .din({data_buff_in, qspi_data_in})
    );

    wire wre2;
    assign wre2 = writing & addr[0] & ~qspi_ram_b_select;

    Gowin_SP ram_b(
        .dout(ram_b_buff_out),
        .clk(qspi_clk),
        .oce(1'b1),
        .ce(1'b1),
        .reset(!rst_n),
        .wre(1'b0),
        .ad(addr[RAM_B_BITS:1]),
        .din({data_buff_in, qspi_data_in})
    );

```

Figure 7. Fragment of the RTL description of the sim_qspi.v module: instantiation and connection of GOWIN_SinglePort memory primitives (BRAM)

The `addi x10, x0, 1` instruction was used as a test instruction to verify correct execution of an arithmetic operation and write-back of the result to the `a0` register. The representation of the instruction in assembly, binary, and hexadecimal formats is shown in Figure 8.

```

Assembly =   addi x10, x0, 1

Binary  =   0000 0000 0001 0000 0000 0101 0001 0011

Hexadecimal = 0x00100513

```

Figure 8. Assembly, binary, and hexadecimal representation of the `addi x10, x0, 1` instruction

To verify correct operation of the arithmetic logic unit and the write-back mechanism, a functional increment test was implemented. The `addi x0, x0, 1` instruction was executed in an infinite loop together with an unconditional jump instruction, `jal x0, -4`, to ensure cyclic program execution. The representation of the unconditional jump instruction is shown in Figure 9.

```
Assembly =    jal x0, -4
Binary  =    1111 1111 1101 1111 1111 0000 0110 1111
Hexadecimal = 0xffdff06f
```

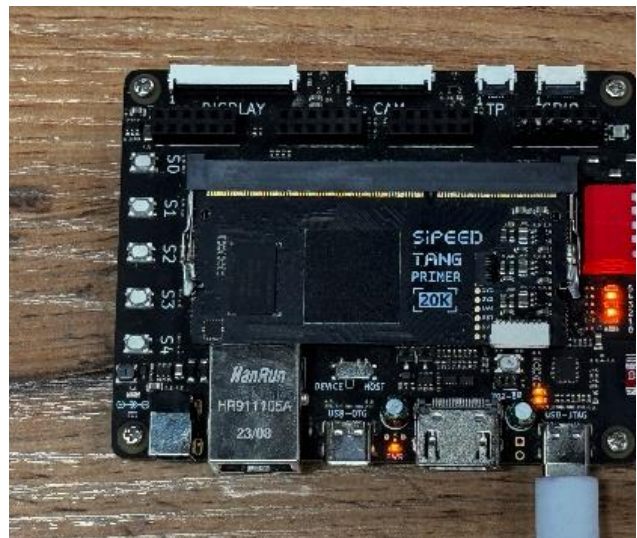
Figure 9. Representation of the unconditional jump instruction used to organize cyclic program execution

This test verified execution-flow stability, correct execution of sequential arithmetic operations, and the absence of dropped cycles.

At a clock frequency of 27 MHz, changes in the least significant bits of the `a0` register occurred too quickly to be observed visually. Therefore, the most significant bits of the `a0` register (bits 24-26) were displayed on the LED indicator. A uniform change in the LED pattern (Figure 10) indicates stable execution of the cyclic program and correct write-back of computation results.



a) LED indication state, where `a0` = 100



b) LED indication state after the register is incremented, resulting in `a0` = 101

Figure 10. LED indication of the counter state in register `a0` during hardware verification of the TinyQV processing core

The built-in GOWIN Analyzer Oscilloscope logic analyzer was used to confirm that the hardware behavior matched the expected logic. The timing diagrams for the internal `led_r` (`a0`) register, shown in Figure 11, demonstrate a consistent increase in value without gaps or deviations,

confirming correct implementation of the increment mechanism at the hardware level.

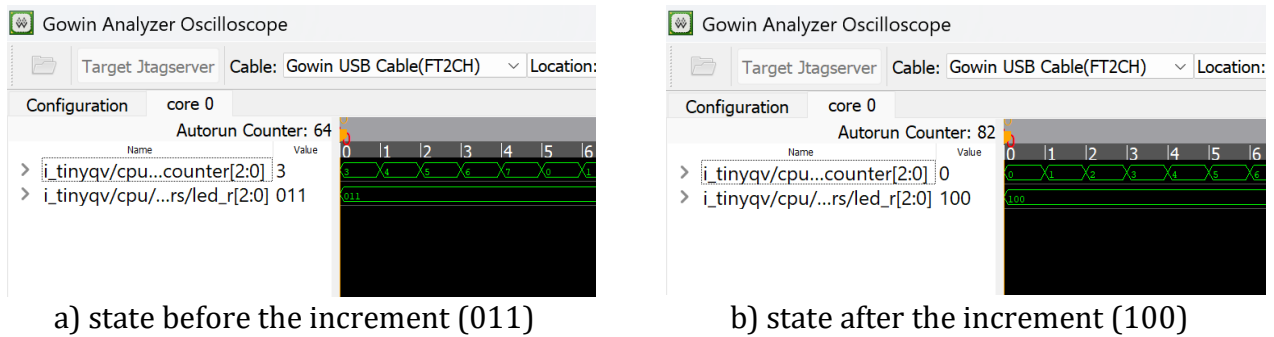


Figure 11. Timing diagrams of changes in the led_r (a0) register state during execution of the increment instruction

HAL of the Demonstration Processing Module

In the demonstration processing module, hardware-abstraction functions are provided through the native TinyQV toolchain and are not separated into a standalone HAL software layer. The native RISC-V GNU toolchain and accompanying SDK take into account the architectural features of the processing core and its fixed memory map, ensuring correct compilation and execution of software on the MPW prototype.

Preparing the Demonstration Processing Module for MPW Prototyping

When preparing the demonstration processing module for prototyping in the TinyTapeout MPW pipeline, MPW platform constraints were taken into account, including the limited footprint of the 3-by-2 tile configuration, a fixed number of I/O lines (8 input lines, 8 output lines, and 8 general-purpose bidirectional lines), and the maximum operating frequency of 64 MHz.

The RTL description of the processing core was adapted to the requirements of the TinyTapeout platform template [40]. Functional verification was performed using a cocotb-based test environment [41].

Physical implementation of the design was performed using the automated OpenLane flow, including logic synthesis, standard-cell placement, routing, and GDSII file generation. Successful completion of DRC and LVS checks was confirmed, and timing closure against the specified constraints was achieved. After completion of the automated OpenLane flow, the demonstration processing-unit design was submitted to the TinyTapeout MPW shuttle for prototype fabrication.

Application of the Formalized Development Flow at the Cryptographic-Controller Level

The proposed development flow and MPW-prototyping approach were applied not only to the demonstration processing module but also to other functional modules of the cryptographic controller, including specialized cryptographic modules that implement digital signature algorithms within a single architectural model.

After development and hardware verification of the individual modules were completed, the modules were integrated step by step into the cryptographic controller, the final system memory map was generated, and development of the hardware abstraction layer (HAL) was completed.

Next, system-level hardware verification of the integrated FPGA architecture was performed, including verification of inter-module communication, operation of the memory subsystem and peripheral interfaces, and execution of test firmware.

Based on the FPGA verification results, the cryptographic-controller architecture was adapted to the MPW platform requirements and prepared for submission to an MPW shuttle for prototype integrated-circuit fabrication.

The following discussion evaluates these results in terms of reproducibility, verification coverage, and readiness for MPW-based physical prototyping.

The results obtained in this study confirm the practical feasibility of the proposed formalized development flow for a specialized cryptographic controller, in which architectural design, RTL implementation, FPGA-based hardware verification, an automated RTL-to-GDS flow, and subsequent submission of the design for MPW fabrication are combined into a single reproducible process. Unlike publications focused on microarchitectural optimization of individual cryptographic accelerators [10-17], studies examining particular aspects of SoC verification [18-24], or works on open-source RTL-to-GDS design [25-27, 39-41], this study demonstrates the integration of these stages into an end-to-end methodology for developing a specialized integrated circuit. The key result is that the modular architecture of the cryptographic controller enables early design verification before both system integration and physical implementation. At the modular level, RTL implementation and FPGA verification of individual functional blocks were performed. Critical modules, including the TinyQV processing core and specialized cryptographic blocks, were subjected to physical-design validation using an automated RTL-to-GDS flow, followed by tape-out preparation for MPW fabrication. At the system level, FPGA verification of the unified architecture was conducted, including verification of inter-module communication, operation of the memory subsystem and peripheral components, and execution of test firmware. The integrated architecture was prepared for physical implementation within the MPW platform. This development framework enables architectural and interface inconsistencies to be identified before the tape-out stage and reduces the risk of discovering critical errors at later design stages.

Experimental verification of the methodology on the TinyQV demonstration processing unit confirms the applicability of the proposed approach to a functionally significant cryptographic-controller core. FPGA verification on the GOWIN Tang Primer 20K platform demonstrated correct instruction fetching and execution, correct operation of the write-back mechanism, and correct operation of the memory subsystem at a clock frequency of 27 MHz. Instrumental testing using the GOWIN Analyzer Oscilloscope showed stable execution of the test program without dropped cycles or observable timing anomalies, confirming the consistency of the microarchitectural solutions and the correct transition from RTL description to hardware implementation.

An important result is the confirmation that the demonstration module can be transferred to the physical implementation level under MPW-prototyping conditions. The project was adapted to the limitations of the TinyTapeout platform, including area, I/O line count, and frequency constraints, and the automated RTL-to-GDS flow in the OpenLane environment was successfully completed. The design passed DRC and LVS checks and met the specified timing constraints. These results demonstrate the reproducibility of the physical-design stage of the methodology within an open-source toolchain and the feasibility of preparing the design for MPW fabrication without commercial EDA tools.

In addition to the demonstration computing module, the same flow was also applied to specialized cryptographic blocks. Preliminary FPGA resource estimates for hardware accelerators implementing electronic digital signature algorithms under ST RK GOST R 34.10-2015 are approximately 14 602 LUTs, 6 702 FFs, and an operating frequency of up to 23.9 MHz. These results confirm that such modules can be implemented within the proposed development flow.

This study primarily focuses on formalizing the development flow and experimentally validating it. The quantitative estimates obtained demonstrate the potential of this approach and provide a basis for further optimization of cryptographic modules in terms of performance, area, and power consumption.

Overall, the results confirm that the proposed approach provides a reproducible and technologically sound transition from an architectural model to hardware verification and physical implementation for MPW prototyping. It can therefore serve as a practical basis for creating specialized cryptographic controllers and other classes of secure SoC solutions.

Conclusion

This paper presents and experimentally verifies a formalized development flow for a specialized cryptographic controller intended for the hardware implementation of electronic digital signature algorithms. The proposed flow is implemented as a methodology based on modular development and step-by-step hardware verification of individual functional blocks, followed by physical-design verification using the MPW approach and by integration testing and physical prototyping of the system as a whole. This study established a modular architecture for a specialized cryptographic controller, including a computing core, system bus, memory subsystem, peripheral interfaces, and specialized cryptographic hardware blocks. A formalized development flow for the cryptographic controller was also formulated. This flow integrates architectural design, RTL implementation, FPGA-based hardware verification, and physical implementation in an automated RTL-to-GDS workflow. As part of the experimental verification, a demonstration computing module based on the TinyQV processor core was implemented, and a complete development cycle was carried out for this module, from RTL description to preparation for MPW prototyping.

Hardware verification of the demonstration module was conducted on the GOWIN Tang Primer 20K FPGA platform. FPGA verification confirmed correct instruction execution, operation of the memory subsystem, and operation of the write-back mechanism. Physical-design verification was performed using the automated RTL-to-GDS flow in the open-source OpenLane toolchain. Successful completion of DRC and LVS checks was confirmed, and timing closure against the specified design constraints was achieved. The results demonstrate the applicability of the proposed development flow to the integrated cryptographic-controller architecture.

The measured FPGA implementation metrics (2 460 LUTs, 964 FFs, and an operating frequency of up to 27 MHz) confirm the applicability of this approach with constrained hardware resources and its suitability for prototyping specialized cryptographic devices. Future research will include a more detailed evaluation of the developed cryptographic modules in terms of performance, area, and power consumption. Another important research direction is the analysis of the resilience of the hardware implementation to side-channel attacks.

Acknowledgment

This research was funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (grant no. AP26103891, "Development of a domestic cryptographic controller and its implementation in a key-storage device").

The contribution of the authors

I. Seniushin: conceptualization of the study, development of the cryptographic controller

architecture and the main stages of the development flow, participation in planning the experimental verification, analysis of the results, and critical revision of the manuscript.

N. Glazyrina: development and formalization of the research methodology, analysis of the scientific literature, analysis and interpretation of the results, preparation of the original draft, scientific editing, and revision of the manuscript.

R. Ibrayev: formulation of requirements for the cryptographic modules, participation in the development and analysis of hardware implementations of electronic digital signature algorithms, analysis of FPGA verification results, and critical revision of the manuscript.

Y. Atanbayev: participation in the development of the architecture and experimental verification strategy, analysis of the TinyQV FPGA verification results and the RTL-to-GDS/MPW prototyping stages, generalization of the experimental results, and critical revision of the manuscript.

All authors discussed the research results, approved the final version of the manuscript for publication, and agreed to be accountable for the accuracy and integrity of the work.

References

1. H. Meng, Z. Sang. "Research and Development of Digital Signatures." *Journal of Computing and Electronic Information Management*, 16(3), 2025, DOI: <https://doi.org/10.54097/2g9b0j24>, pp. 23–28, Panama.
2. K. Gashi, B. Hyseni. "Digital Signatures in the Modern Era: A Review of Cryptographic Progress and Post-Quantum Challenges." *International Journal of Innovative Technology and Interdisciplinary Sciences*, 8(4), 2025, DOI: <https://doi.org/10.15157/ijitis.2025.8.4.1082-1112>, pp. 1082–1112, Estonia.
3. W. Lin. "Digital Signature." In: V. Mulder, A. Mermoud, V. Lenders, B. Tellenbach (eds.), *Trends in Data Protection and Encryption Technologies*, Springer, Cham, Switzerland, 2023, DOI: https://doi.org/10.1007/978-3-031-33386-6_15, pp. 77–81.
4. K. Somsuk. "The development of signing and verification methods for high-speed digital signatures on electronic official documents by using RSA cryptography." *Cogent Engineering*, 11(1), 2024, DOI: <https://doi.org/10.1080/23311916.2024.2432513>, article no. 2432513, United Kingdom.
5. T.-T. Nguyen, D.-D. Nguyen, T.-T. Dao, N.-Q. Luc. "Implementation Efficiency of Falcon Digital Signature Scheme on Arty-7 XC7A35T Board." *Electronics*, 14(22), 2025, DOI: <https://doi.org/10.3390/electronics14224504>, article no. 4504, Switzerland.
6. R. Focardi, F. L. Luccio. "A Formally Verified Configuration for Hardware Security Modules in the Cloud." In: CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15–19, 2021, DOI: <https://doi.org/10.1145/3460120.3484785>, pp. 412–428.
7. B. Kieu-Do-Nguyen, K.-D. Nguyen, T.-K. Dang, N. The Binh, C. Pham-Quoc, N.-T. Tran, C.-K. Pham, T.-T. Hoang. "A Trusted Execution Environment RISC-V System-on-Chip Compatible with Transport Layer Security 1.3." *Electronics*, 13(13), 2024, DOI: <https://doi.org/10.3390/electronics13132508>, article no. 2508, Switzerland.
- A. Barengi, D. Carrera, S. Mella, A. Pace, G. Pelosi, R. Susella. "Profiled side-channel attacks against the RSA cryptosystem using neural networks." *Journal of Information Security and*

- Applications*, 66, 2022, DOI: <https://doi.org/10.1016/j.jisa.2022.103122>, article no. 103122, Netherlands.
8. S. Jin, S. M. Cho, H. Kim, S. Hong. "Enhanced Side-Channel Analysis on ECDSA Employing Fixed-Base Comb Method." *IEEE Transactions on Computers*, 71(9), 2022, DOI: <https://doi.org/10.1109/TC.2022.3191736>, pp. 2341–2350, United States.
 9. V. R. Kolagatla, A. Raveendran, V. Desalpine. "A Novel and Efficient SPI-enabled RSA Crypto Accelerator for Real-Time Applications." In: 2024 28th International Symposium on VLSI Design and Test (VDATE), Vellore, India, September 1–3, 2024, DOI: <https://doi.org/10.1109/VDATE63601.2024.10705738>, pp. 1–6.
 10. G. Leelavathi, K. Shaila, K. R. Venugopal. "Hardware performance analysis of RSA cryptosystems on FPGA for wireless sensor nodes." *International Journal of Intelligent Networks*, 2, 2021, DOI: <https://doi.org/10.1016/j.ijin.2021.09.008>, pp. 184–194, China.
 11. R. Ghayoula, S. Bahroun, W. Amara, J. Fattahi, A. Smida, I. E. Gmati, W. Mohamed. "Optimized RSA-1024 FPGA Implementation on Artix-7." In: J. Blackledge, A.F.M. Shahen Shah, eds., *Coding Theory - Advances and Applications in Informatics, Data Analysis, and Cryptography*, IntechOpen: London, United Kingdom, 2025, DOI: <https://doi.org/10.5772/intechopen.1011874>
 12. R. Agrawal, J. Yang, H. Javaid. "Efficient FPGA-based ECDSA Verification Engine for Permissioned Blockchains." In: 2022 IEEE 33rd International Conference on Application-Specific Systems, Architectures and Processors (ASAP), Gothenburg, Sweden, July 12–14, 2022, DOI: <https://doi.org/10.1109/ASAP54787.2022.00032>, pp. 148–155.
 13. M. Bisheh-Niasar, R. Azarderakhsh, M. Mozaffari-Kermani. "Cryptographic Accelerators for Digital Signature Based on Ed25519." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 29(7), 2021, DOI: <https://doi.org/10.1109/TVLSI.2021.3077885>, pp. 1297–1305, United States.
 14. M. Kashif, I. Cicek, M. Imran. "A Hardware Efficient Elliptic Curve Accelerator for FPGA-Based Cryptographic Applications." In: 2019 11th International Conference on Electrical and Electronics Engineering (ELECO), Bursa, Turkey, November 28–30, 2019, DOI: <https://doi.org/10.23919/ELECO47770.2019.8990437>, pp. 362–366.
 15. Родионов А. Ю. "Архитектура криптографического сопроцессора на ПЛИС". *Вопросы защиты информации: научно-практический журнал*, 3(114), 2016, С. 16–19.
 - I. A. Kashirin, D. R. Usmanov, I. L. Reva, K. V. Zaharov. "Development of optical transceivers SFF standard with support for cryptographic kernel." In: 2016 13th International Scientific-Technical Conference on Actual Problems of Electronics Instrument Engineering (APEIE), Novosibirsk, Russia, October 3–6, 2016, DOI: <https://doi.org/10.1109/APEIE.2016.7806437>, pp. 159–162.
 - J. A. Campbell, L. He, L. Yang, S. T. Gurumani, K. Rupnow, D. Chen. "Debugging and verifying SoC designs through effective cross-layer hardware-software co-simulation." In: DAC '16: The 53rd Annual Design Automation Conference, Austin, Texas, USA, June 5–9, 2016, DOI: <https://doi.org/10.1145/2897937.2898002>, pp. 1–6.
 - K. Chen, L. Sun, Q. Deng. "Hardware and software co-verification from a security perspective in SoC platforms." *Journal of Systems Architecture*, 122, 2022, DOI: <https://doi.org/10.1016/j.sysarc.2021.102355>, article no. 102355, Netherlands.
 16. P. Pamula, D. P. Gorthy, P. S. Ngangbam, A. Alagarsamy. "Verification of SoC Using Advanced Verification Methodology." In: International Conference on "Holography Meets Advanced

- Manufacturing”, Online, February 20–22, 2023, DOI: <https://doi.org/10.3390/HMAM2-14160>, article no. 12.
- A. Hegazy, S. El-Ashry, M. W. El-Kharashi, M. Taher. “Verification of an ARM Cortex-M3-based SoC using UVM.” In: 2023 10th International Conference on Signal Processing and Integrated Networks (SPIN 2023), Noida, India, March 23–24, 2023, DOI: <https://doi.org/10.1109/SPIN57001.2023.10116300>, pp. 778–783.
17. U. Farooq, H. Mehrez. “Pre-Silicon Verification Using Multi-FPGA Platforms: A Review.” *Journal of Electronic Testing*, 37(1), 2021, DOI: <https://doi.org/10.1007/s10836-021-05929-1>, pp. 7–24, United States.
18. E. Perdomo, A. Kropotov, F. K. Cano Ladino, S. Zafar, T. Cervero, X. M. Bofill, B. Salami. “Makinote: An FPGA-Based HW/SW Platform for Pre-Silicon Emulation of RISC-V Designs.” In: RAPIDO '24: Proceedings of the 16th Workshop on Rapid Simulation and Performance Evaluation for Design, Munich, Germany, January 18, 2024, DOI: <https://doi.org/10.1145/3642921.3642928>, pp. 29–34.
19. S. Stoyanov, N. Kakanakov, M. Marinova. “FPGA Prototyping of Heterogeneous Security Architecture for Educational Purposes.” In: The 14th International Scientific Conference TechSys 2025 – Engineering, Technology and Systems, Plovdiv, Bulgaria, May 15–17, 2025, DOI: <https://doi.org/10.3390/engproc2025100018>, article no. 18.
20. P. Fath, M. Moser, G. Zachl, H. Pretl. “Open-source design of integrated circuits.” *e+i Elektrotechnik und Informationstechnik*, 141, 2024, DOI: <https://doi.org/10.1007/s00502-023-01195-5>, pp. 76–87, Austria.
21. Chew, Ch.-Y. Ooi, Ch. N. Chye. “Integrating Design for Testability Technique into OpenLane with Skywater 130-Nanometer Process Design Kit.” *Semarak Engineering Journal*, 3(1), 2023, DOI: <https://doi.org/10.37934/sej.3.1.1421b>, pp. 14–21, Malaysia.
22. D. Franck, G. A. Ginja, J. P. Carmo, J. A. Afonso, M. Luppe. “Custom ASIC Design for SHA-256 Using Open-Source Tools.” *Computers*, 13(1), 2024, DOI: <https://doi.org/10.3390/computers13010009>, article no. 9, Switzerland.
23. H. Witharana, Y. Lyu, S. Charles, P. Mishra. “A Survey on Assertion-based Hardware Verification.” *ACM Computing Surveys*, 54(11s), 2022, DOI: <https://doi.org/10.1145/3510578>, pp. 1–33, United States.
24. S. Canakci, L. Delshadtehrani, F. Eris, M. B. Taylor, M. Egele, A. Joshi. “DirectFuzz: Automated Test Generation for RTL Designs using Directed Graybox Fuzzing.” In: 58th ACM/IEEE Design Automation Conference (DAC 2021), San Francisco, California, USA, December 5–9, 2021, DOI: <https://doi.org/10.1109/DAC18074.2021.9586289>, pp. 529–534.
25. M. Hossain, A. Vafaei, K. Z. Azar, F. Rahman, F. Farahmandi, M. Tehranipoor. “SoCFuzzer: SoC Vulnerability Detection using Cost Function enabled Fuzz Testing.” In: 2023 Design, Automation & Test in Europe Conference & Exhibition (DATE), Antwerp, Belgium, April 17–19, 2023, DOI: <https://doi.org/10.23919/DATE56975.2023.10137024>, pp. 1–6.
26. J. Hur, S. Song, D. Kwon, E. Baek, J. Kim, B. Lee. “DifuzzRTL: Differential Fuzz Testing to Find CPU Bugs.” In: 2021 IEEE Symposium on Security and Privacy (SP), Virtual, San Francisco, California, USA, May 24–27, 2021, DOI: <https://doi.org/10.1109/SP40001.2021.00103>, pp. 1286–1303.
27. H. Al-Shaikh, A. Vafaei, M. M. Rahman, K. Z. Azar, F. Rahman, F. Farahmandi, M. Tehranipoor. “SHarPen: SoC Security Verification by Hardware Penetration Test.” In: ASPDAC '23: 28th Asia

- and South Pacific Design Automation Conference, Tokyo, Japan, January 16–19, 2023, DOI: <https://doi.org/10.1145/3566097.3567918>, pp. 579–584.
28. Wagner, K. Heid, J. Heider. “Remote WebAuthn: FIDO2 Authentication for Less Accessible Devices.” In: 7th International Conference on Information Systems Security and Privacy (ICISSP 2021), Virtual Event, February 11–13, 2021, DOI: <https://doi.org/10.5220/0010192703680375>, pp. 368–375.
29. M. Rivera-Dourado, M. Gestal, A. Pazos, J. A. Vázquez-Naya. “A Novel Protocol Using Captive Portals for FIDO2 Network Authentication.” *Applied Sciences*, 14(9), 2024, DOI: <https://doi.org/10.3390/app14093610>, article no. 3610, Switzerland.
30. *GlobalShuttle™ multi-project wafer program*. Available online: <https://gf.com/manufacturing-services/multi-project-wafer-program/> (accessed on 27 May 2026).
31. *Offered MPW Services – Si CMOS, Compound Semiconductor*. Available online: <https://www.mosis2.com/mpw-services> (accessed on 27 May 2026).
32. *TSMC MPW Shared Tapeouts*. Available online: <https://www.musesemi.com/shared-block-tapeout-pricing> (accessed on 27 May 2026).
33. *Cadence and SkyWater: Fostering the Next Generation of Innovators*. Available online: https://community.cadence.com/cadence_blogs_8/b/can/posts/cadence-and-skywater-fostering-the-next-generation-of-innovators (accessed on 27 May 2026).
34. *Tiny Tapeout*. Available online: <https://tinytapeout.com/chips/> (accessed on 27 May 2026).
35. M. Venn. “Tiny Tapeout: A shared silicon tape out platform accessible to everyone.” *IEEE Solid-State Circuits Magazine*, 16(2), 2024, DOI: <https://doi.org/10.1109/MSSC.2024.3381097>, pp. 20–29, United States.
36. *Cocotb*. Available online: <https://www.cocotb.org/> (accessed on 27 May 2026).

Information about the authors:

Igor Seniushin – TSARKA R&D LLP, 51/1 Kabanbai Batyr Avenue, 010000, Astana, Republic of Kazakhstan, project manager

Natalya Glazyrina – corresponding author, TSARKA R&D LLP, 51/1 Kabanbai Batyr Avenue, 010000, Astana, Republic of Kazakhstan, research project coordinator, PhD, associate professor,

Renat Ibrayev – TSARKA R&D LLP, 51/1 Kabanbai Batyr Avenue, 010000, Astana, Republic of Kazakhstan, cryptographic security specialist

Yernat Atanbayev – TSARKA R&D LLP, 51/1 Kabanbai Batyr Avenue, 010000, Astana, Republic of Kazakhstan, head of the Scientific Department



Copyright: © 2026 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY NC) license (<https://creativecommons.org/licenses/by-nc/4.0/>)