D3.2

Hardware implementations of cryptographic building blocks with formally verifiable protection against physical attacks

Project number:	101070008
Project acronym:	**ORSHIN**
Project title:	Open-source ReSilient Hardware and software for Internet of thiNgs
Project Start Date:	1st October, 2022
Duration:	36 months
Programme:	HORIZON-CL3-2021-CS-01

Deliverable Type:	DEM – Demonstrator
Reference Number:	CL3-2021-CS-01 / D3.2 / 1.0
Workpackage:	WP 3
Due Date:	June 2025 - M33
Actual Submission Date:	30 June 2025

Responsible Organisation:	KUL
Editor:	Benedikt Gierlichs
Dissemination Level:	PU
Revision:	1.0

Abstract:	This deliverable describes hardware implementations of cryptographic building blocks, such as the PRINCE and AES S-boxes, with formally verifiable protection against physical side-channel attacks.
Keywords:	Side-channel attacks, countermeasures, formal verification

Funded by the European Union under grant agreement no. 101070008. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

Editor

Benedikt Gierlichs(KUL)

Contributors (ordered according to beneficiary numbers)

S. V. Dilip Kumar, Ingrid Verbauwhede (KUL)

Reviewers

Maria Chiara Molteni (SEC)
Volodymyr Bezsmertnyi (NXP)

Executive Summary

The ORSHIN project has as a goal to build secure and open-source hardware. In Task 3.2, we evaluated state-of-the-art models for formal verification of security properties of cryptographic hardware implementations, we developed new models, and we designed new countermeasures. This deliverable describes hardware implementations of cryptographic building blocks, such as the PRINCE and AES S-boxes, with formally verifiable protection against physical attacks. More precisely, our implementations are protected against physical side-channel attacks with a novel hardware masking scheme, developed in this task, called Time Sharing Masking (TSM) or its extension to higher protection orders, called Higher-Order Time Sharing Masking (HO-TSM).
The code of the implementations, verification scripts, verification inputs and results are all publicly available under permissive licenses. Our protected implementations have also successfully participated in artifact evaluation. An artifact evaluation committee has determined that our TSM protected implementations satisfy the requirements for “functional” and confirmed reproducibility of synthesis and verification results. Our HO-TSM protected implementations were found to be complete, correct and reproducible.

Table of Content

Chapter 1

Introduction

The ORSHIN project studies open source resilient hardware and software for the Internet of Things. One of the goals of the ORSHIN project, and specifically of WP3, is building prototypes for countermeasures against physical attacks.
Physical side-channels reveal information about what a processor or a circuit is doing. For typical IoT devices which are the focus of the ORSHIN project it is particularly true that an adversary may have physical access to them, and will be able to measure such physical quantities. It is thus naturally important to protect implementations against physical side-channel attacks. Masking has proven to be an effective countermeasure against side-channel attacks. However, it is vital that the implementation of the countermeasure does not introduce flaws that weaken the security. The field of research has therefore evolved toward provable secure masking techniques, automated security verification tools, formal verification tools, and formally verifiable secure implementations. Independent evaluation, verification and re-usability are of course conerstones of such a development, which is why it is important to release artifacts like this one open source.

In Chapter 2, we briefly describe Time Sharing Masking (TSM) and the related implementations. Both the masking technique and the implementations were developed during the project.

In Chapter 3, we introduce Higher-Order Time Sharing Masking (HO-TSM) and the related implementations. HO-TSM is an extension of the TSM idea to achieve higher protection orders. Both the masking technique and the implementations were developed during the project.

Finally, we conclude in Chapter 4.

Chapter 2

Time Sharing Masking

This chapter briefly introduces Time Sharing Masking (TSM) and describes the demonstrator, also referred to as artifact. The TSM demonstrator delivers hardware implementations of cryptographic building blocks that are formally verified and provide first-order security against side-channel attacks.
Conceived for latency-critical cryptographic applications, TSM was introduced in the paper *"Time Sharing – A Novel Approach to Low-Latency Masking"* (TCHES 2024, Issue 3) [3]. This artifact translates that research into a reproducible, open-source package containing masked PRINCE and AES S-Boxes and a full PRINCE core, each synthesizable with commonly used electronic design automation (EDA) tools such as Synopsys Design Compiler and verifiable using formal verification tools such as SILVER.
TSM processes different input shares across distinct clock cycles, introducing a single register stage that separates their computation in the time domain. This scheduling strategy enables low-latency operation while preserving share separation and maintaining side-channel security.
Each design included in the artifact satisfies first-order side-channel resistance under the robust probing model, which accounts for hardware effects such as glitches. Additionally, the masking construction is composable secure, meaning several secure modules can be securely composed into a larger circuit without introducing new vulnerabilities. For detailed formal definitions and security proofs, please refer to the associated TCHES paper or ORSHIN deliverable D3.3.

2.1 Objectives and Key Contributions

The primary goal of the artifact is to provide a deployable reference implementation of the TSM methodology. It allows hardware designers to (i) inspect the Register Transfer Level code (RTL), (ii) reproduce the synthesis and verification flow, and (iii) incorporate the masked components into their own cryptographic implementations.

Contributions. TSM offers a low-latency masking solution with significantly improved efficiency over prior approaches.
Gate-level masking techniques often suffer from increased latency due to the need to insert registers after each non-linear gate. One such solution, GLM [1], attempts to address this by removing these register stages entirely. However, eliminating registers leads to exponential growth in the number of shares and substantial increases in area and randomness usage.
In contrast, algorithm-level methods such as GHPC [2] ensure composability, but also exhibit large area and randomness requirements. While GHPC-style techniques allow secure composition,

their implementation cost remains high, particularly for larger S-Boxes.
TSM improves these trade-offs. It achieves single-cycle masked evaluation, maintains composable first-order security guarantees, and reduces both area and randomness consumption, as demonstrated in the AES and PRINCE case studies provided in the artifact.

2.2 Artifact Package Description

The top-level directory `./` contains a general `README.md` file that outlines prerequisites and usage instructions. The artifact is divided into two subdirectories:

- `./AES_SBox/` — includes source code, netlists, and documentation for the masked AES S-Box.
- `./PRINCE_SBox/` — includes source code, netlists, and documentation for the masked PRINCE S-Box.

Each subdirectory contains its own `README.md` file that describes module-specific build and verification steps in detail.
For each module, the package includes the following:

- Verilog RTL written using standard synthesizable constructs.
- Synopsys Design Compiler TCL scripts targeting the Nangate 45nm open cell library. These scripts support two flows:
  - `compile -exact_map`: Retains the RTL structure in the final netlist to simplify analysis.
  - `compile_ultra -no_autoungroup -no_boundary_optimization`: Performs aggressive logic optimization while preserving register boundaries to maintain security properties.
- SILVER, a formal verification tool, is used to validate security properties. Input and output files are included to demonstrate conformance to various probing security as well as composability notions (NI, SNI, and PINI) for the synthesized netlists.

2.3 Addressed Challenges and Comparative Advantages

Latency. TSM achieves single-cycle latency irrespective of the degree of nonlinearity. This approach results in single-cycle S-Box evaluations and allows a full PRINCE cipher encryption to complete precisely within twelve cycles, directly corresponding to its twelve encryption rounds.

Area and Randomness. For an AES S-Box, TSM achieves roughly one-fourth of the area and uses approximately one-eighth of the randomness compared to masking schemes such as GLM and GHPC. This efficiency primarily comes from the systematic reuse of intermediate computations, substantially reducing redundancy compared to existing techniques.

Formal Verifiability. TSM methodology ensures formal security by processing each masking share independently in separate clock cycles. This temporal separation allows the resulting constructions to meet formal security notions such as the glitch-extended Probe-Isolating Non-Interference (PINI). The netlists included in the artifact successfully pass verification using the formal verification tool SILVER.

2.4 Integration and Reproducibility Guidelines

Synthesis Flow. The artifact provides a synthesis script, `compile.tcl`, which generates the reported area, timing numbers, and corresponding netlists. The script is self-contained and specifies library paths and output formats clearly, making it straightforward for designers to adapt the artifact to different technology libraries.

Verification Flow. The artifact includes verification scripts to run formal verification using SILVER on the generated netlists. The PRINCE S-Box is sufficiently small to be verified directly within SILVER, whereas the AES S-Box exceeds the tool constraints, thus necessitating FPGA-based testing as an alternative.

2.5 Open Access and Independent Evaluation

Availability. The complete package is publicly available under the MIT license on github at: https://github.com/KULeuven-COSIC/TSM/tree/main/TSM.

Independent Validation. The artifact underwent independent validation as part of the CHES 2024 Artifact Evaluation, confirming reproducibility of synthesis and verification results. This validation supports the artifact's credibility and suitability for practical use.

Chapter 3

Higher-Order Time Sharing Masking

Higher-Order Time Sharing Masking (HO-TSM) generalizes the core principles of Time Sharing Masking (TSM) to formally ensure higher-order security against side-channel attacks. Specifically, HO-TSM addresses environments where protection against first-order leakage alone is inadequate, such as when adversaries can exploit higher-order statistical relationships from physical measurements.
Introduced in *"Higher-Order Time Sharing Masking – Low-Latency Masking for Higher-Order Security"* (TCHES 2025, Issue 2) [4], HO-TSM extends TSM's underlying approach (processing masking shares across distinct clock cycles using dedicated register stages) to achieve scalable higher-order security. The artifact associated with this methodology provides reproducible, formally verified, and open-source implementations of cryptographic primitives, notably including first- and second-order masked AES S-Boxes and a complete first-order masked round-based AES-128 encryption core. All implementations are synthesizable using electronic design automation (EDA) tools such as Synopsys Design Compiler, and their security properties are validated using formal verification tools such as SILVER, PROLEAD, and maskVerif.
Additionally, this artifact presents a secondary contribution through an area-latency trade-off construction. This variant introduces one additional clock cycle to the base HO-TSM latency in exchange for significantly reduced area and randomness requirements, thus making higher-order masking viable for resource-constrained applications.
Each provided implementation within the artifact satisfies the glitch-extended Probe-Isolating Non-Interference (PINI) and Strong Non-Interference (SNI) security models. For detailed formal definitions and security proofs, please refer to the associated TCHES paper or ORSHIN deliverable D3.3.

3.1 Objectives and Key Contributions

The artifact demonstrates the scalability of HO-TSM to higher-order security through verified reference designs and introduces a complementary construction that reduces area and randomness by adding a single cycle of latency. Together, these contributions establish HO-TSM as both a generalization of TSM to higher orders and a practical design choice for constrained hardware settings.

Contributions. This artifact addresses two distinct research contributions:
(1) Higher-order extension: HO-TSM extends TSM to arbitrary security orders. A dth-order secure implementation is realized using $d+1$ shares over precisely d cycles. The latency remains fixed at

d cycles regardless of the algebraic degree of the function, which makes this approach particularly suitable for masking components like AES S-Boxes.
(2) Area-latency trade-off: This variant construction increases latency by exactly one clock cycle, significantly improving resource utilization by reducing area and randomness requirements. For instance, the provided two-cycle, first-order secure AES S-Box implementation demonstrates notable efficiency advantages compared to single-cycle constructions. This construction is further employed to implement a complete round-based AES-128 core in 20 clock cycles.

3.2 Artifact Package Description

The artifact includes a detailed `README.md` file at the top-level directory that outlines the directory structure, tool dependencies, and usage instructions for synthesis and verification.
The artifact provides higher-order masked implementations of AES S-Boxes and a first-order secure round-based AES-128 encryption core. Both the standard HO-TSM and the area-latency trade-off variant are included. These implementations serve as practical demonstrations of the proposed design methodology and allow comparative analysis against prior low-latency masking approaches.

3.3 Addressed Challenges and Comparative Advantages

HO-TSM addresses several critical challenges inherent in the deployment of masked hardware cryptographic implementations:

Efficient Scaling to Higher Orders. Previous low-latency masking techniques typically restrict their applicability to first-order security, as extending to higher orders significantly escalates complexity and resource requirements. HO-TSM mitigates these issues by precisely structuring masking computations across $d + 1$ shares and d cycles for dth-order security. This structured scalability ensures predictable latency and manageable hardware overhead even at higher security orders.

Improved Efficiency via Area-Latency Trade-Off. Implementations rapidly become resource-intensive with increasing security order. HO-TSM's trade-off variant addresses this challenge by partitioning the input space into smaller domains, which are processed independently and recombined securely, incurring only one additional cycle of latency. This approach significantly reduces the required area and randomness making it highly attractive for resource-limited hardware platforms.

Formal Verification Assurance. HO-TSM implementations satisfy formal security properties defined by the glitch-extended PINI and SNI security models. The verification process relies on automated formal tools, specifically SILVER for smaller functional blocks, and PROLEAD and maskVerif for larger-scale components. Formal verification confirms the adherence of HO-TSM implementations to security requirements, ensuring protection against side-channel attacks and composability in larger cryptographic circuits.

3.4 Integration and Reproducibility Guidelines

Synthesis Methodology. We include a `compile.tcl` script for synthesis using Synopsys Design Compiler with the Nangate 45nm library. The script supports different compiler options.

Formal Verification Approach. Verification strategies differ based on the complexity of the implemented circuits. Small building blocks undergo formal verification using SILVER, confirming their security properties directly. More complex designs, such as the AES S-Box, are formally validated using PROLEAD and maskVerif. Comprehensive instructions and configuration details for reproducibility of verification results are documented in the top-level `README.md`.

3.5 Open Access and Independent Evaluation

Availability. The HO-TSM artifact is released under the MIT license on github at: `https://github.com/KULeuven-COSIC/TSM/tree/main/HO_TSM`.

Independent validation. The CHES 2025 Artifact Evaluation Committee confirmed the artifact's completeness, correctness, and reproducibility.

Chapter 4

Summary, conclusion and outlook

This deliverable briefly described TSM and HO-TSM, two masking schemes developed as part of the ORSHIN project to protect hardware implementations of cryptographic building blocks against physical attacks. This deliverable also introduced several such protected hardware implementations that demonstrate the approaches and that were developed during the project.
The code of the implementations, verification scripts, verification inputs and results are all publicly available under permissive licenses. Our protected implementations have also successfully participated in artifact evaluation. An artifact evaluation committee has determined that our TSM protected implementations satisfy the requirements for “functional” and confirmed reproducibility of synthesis and verification results. Our HO-TSM protected implementations were found to be complete, correct and reproducible.

Bibliography

[1] Hannes Groß, Rinat Iusupov, and Roderick Bloem. Generic low-latency masking in hardware. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(2):1–21, 2018.

[2] David Knichel, Pascal Sasdrich, and Amir Moradi. Generic hardware private circuits towards automated generation of composable secure gadgets. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(1):323–344, 2022.

[3] Dilip Kumar S. V., Siemen Dhooghe, Josep Balasch, Benedikt Gierlichs, and Ingrid Verbauwhede. Time sharing - A novel approach to low-latency masking. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2024(3):249–272, 2024.

[4] Dilip Kumar S. V., Siemen Dhooghe, Josep Balasch, Benedikt Gierlichs, and Ingrid Verbauwhede. Higher-order time sharing masking. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2025(2):235–267, 2025.