POLITECNICO DI TORINO
Repository ISTITUZIONALE

Compact Yet Fast: An Efficient d-Order Masked Implementation of Ascon

Original
Compact Yet Fast: An Efficient d-Order Masked Implementation of Ascon / Mirigaldi, M., Paninforni, N., Martina, M., Masera, G.. - ELETTRONICO. - (2026). (2026 Design, Automation & Test in Europe Conference (DATE) Verona (Ita) 20 - 22 Aprile 2026) [10.23919/date69613.2026.11539534].

Availability:
This version is available at: 11583/3011992 since: 2026-06-16T12:14:48Z

Publisher:
IEEE

Published
DOI:10.23919/date69613.2026.11539534

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2026 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

27 November 2026

Compact Yet Fast: An Efficient d-Order Masked Implementation of Ascon

Mattia Mirigaldi
Politecnico di Torino
Turin, Italy
mattia.mirigaldi@polito.it

Nico Paninforni
Politecnico di Torino
Turin, Italy
nicopaninforni@gmail.com

Maurizio Martina
Politecnico di Torino
Turin, Italy
maurizio.martina@polito.it

Guido Masera
Politecnico di Torino
Turin, Italy
guido.masera@polito.it

***Abstract*—In this work, we present a generic side-channel protected design of Ascon that achieves high efficiency by dynamically reconfiguring the hardware countermeasures during message processing. The resultant implementation is protected and capable of meeting stringent performance requirements whilst minimising resource overhead. The experimental results obtained demonstrate that the implementation meets the required security and achieves superior throughput-to-area ratio across all protection orders. Ascon, recently selected by NIST as the lightweight cryptography standard, is widely deployed in resource-constrained devices that demand both high performance and resistance against threats such as side-channel analysis (SCA). Exploiting Ascon's mode-level structure, which does not require protection against differential power analysis during bulk operations, we introduce a modified masking gadget with dual functionality: serving as a countermeasure during critical operations, and processing multiple data paths in parallel to accelerate bulk computation. Our architecture supports any configurable security order and instantiates only the minimum hardware resources needed to maximize throughput per round. We also evaluate an enhanced Ascon architecture based on the *Changing of the Guards* technique, which eliminates the need for fresh randomness. Security validation is performed using fixed-vs-random t-tests on both first- and second-order masked implementations. Finally, we compare our masked design against state-of-the-art solutions.**

***Index Terms*—Ascon, Hardware, Side-channel attack, Domain-Oriented-Masking, Mode-level implementation**

I. Introduction

Motivation. The increasing demand for resource-constrained devices, prompted by the ever-growing deployment of IoT devices, introduces a range of new security and privacy challenges. To address these challenges, the National Institute of Standards and Technology (NIST) initiated research on lightweight cryptography (LWC), algorithms specifically designed to deliver essential security services with minimal hardware and software overhead. In February 2023, NIST selected the Ascon family as the winner of the LWC competition [1], establishing it as the new standard for lightweight cryptographic applications. Ascon includes authenticated encryption (AE), hashing schemes, and extendable-output functions, ensuring broad applicability across diverse use cases. There are various constraints that an Ascon implementation must satisfy. In the current IoT-to-Cloud paradigm, IoT devices typically perform partial local processing before encrypting data and transmitting it over the network. Consequently, Ascon implementations must strike a balance: they should minimize hardware footprint while simultaneously sustaining high throughput to support the real-time encryption. At the same time, IoT devices often operate in unprotected environments and are vulnerable to side-channel attacks (SCA), which exploit runtime signatures of cryptographic operations to infer secret information. To counter such threats, cryptographic implementations must adopt dedicated countermeasures. Among these, masking stands out as the most effective, thanks to its scalable security levels and proven robustness against side-channel leakage [2].

Related Work. Since its standardization in 2023, several hardware implementations of Ascon have been presented. Owing to its permutation-based design, Ascon architecture can be tailored to meet the application constraints. Design space explorations conducted in [3], [4], [5], [6] have investigated the different architectural strategies that can be adopted: serial, processing one sub-state per cycle; round-based, executing one permutation per cycle; and unrolled, computing multiple permutations per cycle. As expected, the serial design achieves the smallest area but, due to control logic overhead, it becomes inefficient once more than five bits per cycle are processed. Beyond unrolling, higher throughput can be achieved reducing the critical path through state partitioning (how shown in [7]). Another line of work integrates Ascon as a tightly-coupled accelerator in RISC-V processors [8], [9]. In particular in [8] the processor's internal register file is re-used hence limiting area overhead to only 9% of the processor baseline. However, these explorations focus only on performance optimization, whereas a real-world Ascon deployment must also be secure against side-channel attacks. Several works use Ascon as a test case for novel masking countermeasures [10], [11], [12], [13], but the combination of high-performance techniques and side-channel protection remains largely unexplored.

Our contribution. In this paper, we show how secure strategies such as leveled implementations and masking can be combined with high-performance techniques like unfolding to obtain a side-channel resistant design that achieves high throughput with minimal area overhead, significantly more compact than state-of-the-art masked implementations. In the case of Ascon, security proofs show that only the initialization and finalization phases, where the long-term key is mixed, must be strongly protected against side-channel attacks [14]. During bulk message processing, leakage does not threaten key recovery thanks to the mode-level design. This enables reusing hardware that operates in masked mode for sensitive phases to

also accelerate bulk encryption. The resulting architecture can be configured for side-channel protection at any security order while still meeting the stringent performance requirements of IoT applications. Moreover, because our solution builds on the general principles of mode-level implementations and domain-oriented masking (DOM), it is not limited to Ascon. The same approach can be extended to other authenticated encryption schemes and crypto primitives with similar security proofs, such as ISAP [15], Xoodyak [16] or Keccak-based designs.

Open Source. Our hardware design is public on Github[1].

Outline. The remainder of this article is organized as follows. In Section II, we introduce the required background: (A) SCA, (B) DOM as a countermeasure, (C) an overview of Ascon, and (D) the Mode-level operation used in Ascon. In Section III, we describe the design of an implementation of Ascon with d-order security against SCA. Section IV presents our high-throughput Ascon design with configurable security order. Section V report results and comparisons against the state-of-the-art, while Section VI concludes the paper.

II. Background

A. Side-Channel Analysis

Breaking cryptographic algorithms by exploiting mathematical weaknesses is often infeasible. However, once implemented in hardware or software and deployed in unprotected environments, these algorithms become vulnerable to side-channel attacks. By monitoring physical leakages (such as power consumption, execution time, or electromagnetic radiation) an attacker can infer sensitive data being processed. The first demonstrations of such attacks appeared in [17], and later [18] introduced Differential Power Analysis (DPA), showing that secret keys can be recovered by exploiting correlations across multiple power traces. Subsequent works [19], [20], as well as Ascon-specific attacks [21], [22], [23], [24], have confirmed how inexpensive yet highly effective these techniques are at compromising cryptographic implementations. To mitigate these threats, various countermeasures have been proposed, including masking [25], shuffling [26], and random delay insertion [27], each aiming to decorrelate or obscure the side-channel information available to an adversary.

B. Domain Oriented Masking

Masking is an algorithm countermeasure against DPA that decorrelates intermediate data from the secret key. The sensitive intermediate variable, x, is split into shares, and computations are performed on these shares rather than on the secret directly. Observing fewer than all shares reveals no information about the secret. In Boolean masking, d-th order security is achieved by splitting the secret into $s = d + 1$ shares over $\mathbb{F}_2$:

$$x = x_1 \oplus x_2 \oplus \cdots \oplus x_{d+1}, \tag{1}$$

The masked computation exploits the fact that the function $F(\cdot)$ can be decomposed into share-wise computations, producing output shares $F_1, F_2, \ldots, F_{d+1}$ that, when recombined, yield the same result as the unmasked computation.

$$F(x) = F_1(x_1) \oplus F_2(x_2) \oplus \cdots \oplus F_{d+1}(x_{d+1}), \tag{2}$$

According to the d-probe model [28], an implementation is d-th order secure if an attacker probing up to d internal signals gains no advantage in recovering secret variables. To build efficient hardware circuits under this model, Domain-Oriented Masking (DOM) [29] was introduced. The masked computation is partitioned into domains, each operating on a single share. Since each sub-circuit processes only one share, the domain remains secure against probing attacks. Linear operations are trivially implemented by duplicating the logic across domains, while non-linear operations (such as AND) require cross-domain communication, which exposes the implementation to the risk of leakage. To prevent this, DOM applies resharing to cross-domain terms with additional randomness and inserts register stages to block glitches from propagating across domains. The original work introduced two types of multipliers: i) DOM-indep, which requires independently shared inputs and has the advantage of lower randomness consumption and smaller size, and ii) DOM-dep, which relaxes the independence requirement but it is more costly and insecure for masking orders of two or higher. Figure 1 illustrates a one-bit DOM-indep multiplier implementing a masked AND gate. It is composed of a calculation phase, a resharing phase, and an integration phase, producing $d + 1$ output shares. The resharing step requires $d \cdot (d + 1)/2$ fresh random bits Z and at least $d \cdot (d + 1)$ registers, increasing both latency and overhead. Overall, masking significantly increases hardware utilization, with the area overhead growing quadratically with the protection order. For simplicity, in the remainder of this paper we will use the term DOM-AND gate to refer to the DOM-indep AND gate.

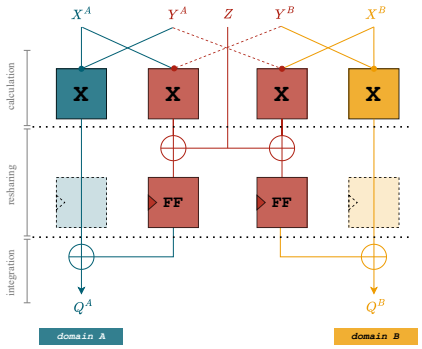


Fig. 1: First-order DOM-indep $GF(2^n)$ multiplier. The inputs X and Y are shared across domains A and B, and the resulting output Q remains in shared form.

C. Ascon

Ascon is a family of authenticated encryption and hashing schemes [30]. All Ascon ciphers are built on the sponge construction and use a 320-bit permutation as their core component. Two variants of the permutation are defined: p_a, used

[1] https://github.com/NicoPaninforni/myAscon128-A

during initialization and finalization, and p_b, used during data processing . The only difference between them is the number of iterations of the round function *Ascon-p*. In Ascon, the encryption process is divided into four phases: initialization, processing of associated data, processing of plaintext, and finalization (see Figure 2). The permutation operates on a 320-bit state S, bit-sliced into five 64-bit words in big-endian order:

$$S = x_0 \,||\, x_1 \,||\, x_2 \,||\, x_3 \,||\, x_4 \,||\, x_5 \tag{3}$$

Each round transformation is composed of three sequential layers:

$$\text{Ascon-}p \;=\; p_c \circ p_s \circ p_l \tag{4}$$

where p_c is the round constant addition (dependent on the round index), p_s is the substitution layer (S-box) which operates by applying a non-linear 5-bit substitution to each state column, and p_l is the linear diffusion layer, which combines bitwise rotations ($\ggg$) and XOR operations ($\oplus$) as:

$$\begin{aligned}
x_0 &= \Sigma_0(x_0) = x_0 \oplus (x_0 \ggg 19) \oplus (x_0 \ggg 28)\\
x_1 &= \Sigma_1(x_1) = x_1 \oplus (x_1 \ggg 61) \oplus (x_1 \ggg 39)\\
x_2 &= \Sigma_2(x_2) = x_2 \oplus (x_2 \ggg 1) \oplus (x_2 \ggg 6)\\
x_3 &= \Sigma_3(x_3) = x_3 \oplus (x_3 \ggg 10) \oplus (x_3 \ggg 17)\\
x_4 &= \Sigma_4(x_4) = x_4 \oplus (x_4 \ggg 7) \oplus (x_4 \ggg 41)
\end{aligned} \tag{5}$$

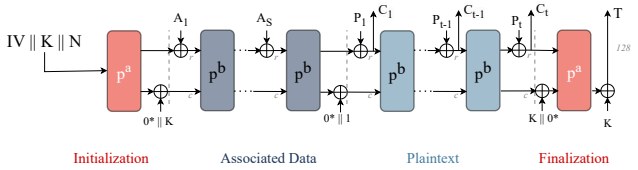


Fig. 2: Ascon encryption scheme. The mode-level implementation determines that only the red blocks require protection against DPA attacks.

D. Mode-Level Implementation

Hardware-level countermeasures against SCAs (e.g., masking) are effective but costly, hence an alternative line of research explores cryptographic primitives and modes of operation with inherent resistance to leakage (known as leakage-resilient cryptography [31]). In fact, it has been shown that integrity and confidentiality can still be ensured under weak physical assumptions for the bulk of the computation. For instance, ciphertext integrity can be maintained even if all intermediate states leak, as long as the few manipulations involving the long-term key are strongly protected. This observation enables leveled implementations, introduced by Pereira et al. [32]. Instead of uniformly applying costly countermeasures to the entire design, protection is only required in specific phases, while the rest of the computation can tolerate leakage. The effectiveness of leveled implemention in hardware is still unexplored, though initial results [14] indicate that can lead to substantial energy savings for long messages. The leveled implementation of Ascon translates to only DPA-level security to initialization and finalization phases (Red blocks in Figure 2).

III. Ascon Generic d-Order Secure Design

In Ascon during initialization, two of the five inputs to each S-box are secret key bits, while the remaining three are known and user-controlled. This creates a vulnerability to SCAs, since the statistical dependency between these controllable intermediate states and a subkey, over which exhaustive search is feasible, enables recovery of the entire secret key [33].

A. Masking the Ascon S-box

The DOM masked Ascon design considered in this work follows [13]. The operation to be masked is the S-box , which is composed of a linear layer, the χ mapping derived from Keccak [34], and an affine layer. The non-linear χ mapping updates each bit as an AND-XOR with one inverted input:

$$x_i \oplus \big((x_{i+1} \oplus 1) \cdot x_{i+2}\big). \tag{6}$$

The linear and affine layers are trivially masked by duplicating the operations across all shares. The non-linear layer requires special care (DOM masked construction is shown in Fig. 3). The shares of x_{i+1} and x_{i+2} are multiplied, with cross-domain terms re-shared using random bit Z_0. The masked S-box adds two register stages: one required by the masking scheme and one before each DOM-AND gate to ensure input independence. In order to reduce logic depth and minimize glitch propagation, the XOR from the integration layer is moved to the resharing layer, x_i addition before the DOM register stage.

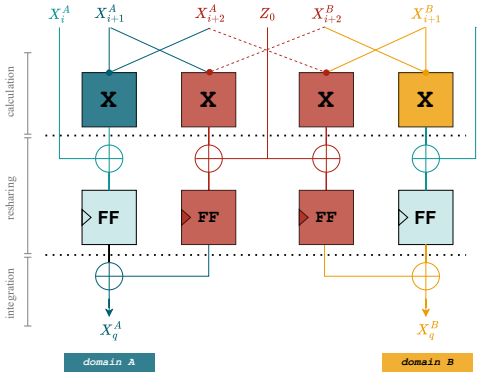


Fig. 3: First order DOM-AND structure to mask the AND-XOR operation.

B. Masking Ascon without Fresh Randomness

The generation of random bits for resharing in DOM-AND requires dedicated hardware, which increases the area and energy overhead of the final masked implementation. To mitigate this cost, the *Changing of the Guards* technique [35] proposes reusing shares of neighboring S-boxes as randomness, since these shares are (by definition) uniformly distributed random variables. However, not all neighbours can be selected as *Guards*, as independence must be preserved to avoid compromising security. In the first-order masked Ascon implementation, each S-box requires five *Guards* (one per DOM-AND gate). The selection of guard bits is straightforward, as it only

requires avoiding those that interact within the linear diffusion and S-box linear layers. At higher orders, however, the challenge increases: each DOM-AND requires $d\cdot(d+1)/2$ random bits (e.g., three for second-order), and independence must be ensured both with respect to the masked variable and across *Guards*. To address this complexity, Prasad et al. [13] formulate the second-order *Guard* selection as a constraint satisfaction problem and employ a SAT solver (boolean satisfiability solver) to automatically identify valid guards.

IV. Compact d-order Secure Ascon Design

In typical use cases, the message processing phase (associated data and plaintext) dominates the overall execution time. Owing to Ascon's mode-level design, no fresh secret-dependent computations occurs during message processing, which ensures that confidentiality and integrity remain preserved even in the presence of leakage. Consequently, masking countermeasures can be selectively simplified or disabled in this phase without compromising security. In the proposed design, the hardware resources originally allocated for masking are repurposed to enhance data-path parallelism and increase throughput. In particular, this reallocation enables multiple state columns to be processed in parallel during message processing, thereby accelerating the permutation phase. This adaptive reuse of resources yields a significant performance improvement while preserving full d-order security during the critical safety initialization and finalization phases. Our design does not adopt a fully parallel architecture with 64 S-boxes per round. This choice is motivated by the structure of Ascon's sponge permutation, where a new round can only begin once the previous round has been fully completed. Repurposing the masking hardware to process multiple rounds per cycle would lengthen the critical path with the unrolling degree, lowering the maximum frequency and requiring extra control logic with consequent area/performance overhead. Therefore, we avoid full parallelization and instead instantiate a number of masked S-boxes according to the masking order. When repurposed during the bulk data processing phase, this configuration achieves exactly one permutation per cycle. This solution minimizes the critical path and, importantly, maintains a uniform critical path across all phases. This solution achieves a uniform critical path across all phases. Table I reports the number of S-boxes that can be processed in parallel for different masking orders, which, when repurposed, enable maximum data parallelism during the Ascon permutation. This constraint makes our solution well suited to unprotected, resource-constrained devices. It substantially improves throughput without increasing area, maximizing performance-to-area efficiency.

A. Reconfigurable DOM-AND Gadget for Parallel Processing

In the context of Ascon message-related operations, where masking is not required, we propose a reconfigurable DOM-AND gadget that enables parallel processing of multiple data paths. The central idea is to reuse the hardware to perform several independent AND operations in parallel. Instead of connecting the $d+1$ shares of a single variable to the DOM-AND, as in the classical scheme, the $d+1$ columns of the state are directly mapped to the domains during message processing. In this way, the columns are treated as independent variables, enabling efficient multi-column computation.

TABLE I: Maximum usable S-box parallelism for different masking degrees, enabling masking logic reuse.

Masking Degree d	**Max Parallel S-Boxes** $\left\lceil \frac{64}{d+1} \right\rceil$
1	32
2	22
3	16
5	11

For the first-order case, the modified architecture (Fig. 4) reuses resources to process two columns simultaneously, doubling throughput during message phases. In principle, the hardware could be extended to support up to $(d+1)^2$ columns in parallel by fully exploiting reuse, but this would significantly increase control complexity and interconnect overhead. Instead, the solution we adopted achieves meaningful throughput improvements at minimal additional cost. The proposed resource-reuse strategy generalizes across protection orders d, yielding throughput gains that scale linearly with d. For instance, the $d = 2$ configuration (three shares) maintains the same architectural structure while achieving approximately $3\times$ higher throughput compared to a strictly serialized masked-AND baseline. These results highlight that the proposed reconfigurable DOM-AND gadget not only ensures higher-order security, even under stronger threat models, but also delivers substantial efficiency improvements, making it a practical solution for protected cryptographic implementations.

B. Ascon d-order Masked with Reconfigurable DOM-AND

Building on the principles introduced in the previous section, we now present the generic d-order masked architecture of Ascon. The design ensures full protection during the initialization and finalization phases, while achieving high performance during message processing. The control of the Ascon core is managed by a synchronous finite state machine (FSM), which orchestrates the datapath through the different phases: initial-

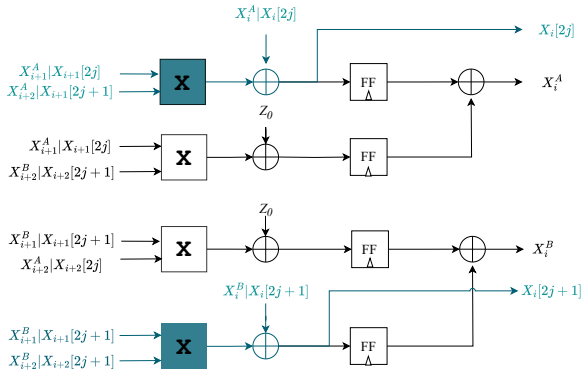


Fig. 4: Reconfigurable DOM-AND gadget

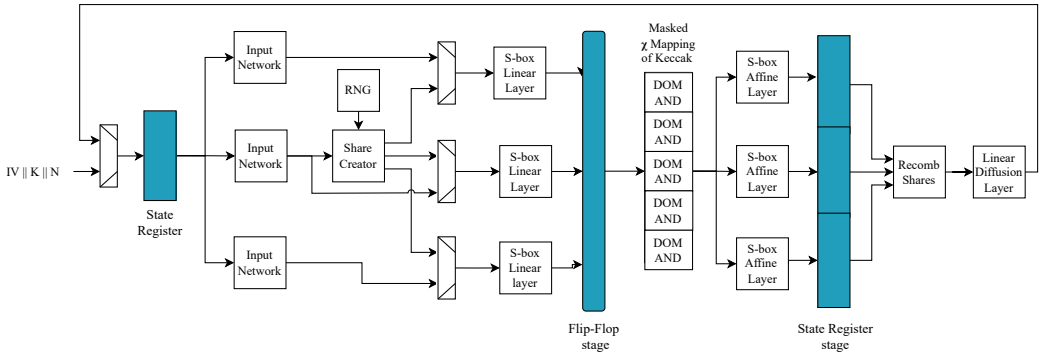


Fig. 5: Modified Ascon architecture with 2nd-order masking degree

ization, associated data and message absorption, permutations, finalization, and tag generation. The architecture operates in two modes: *masked* and *high-throughput*. It consists of several main blocks: Round Constant Addition, Share Creation, the modified S-box layer followed by a register stage, and the Linear Diffusion Layer. The register stage is required to maintain domain separation across all shares (refer to Sec. III) and has a width of $(d+1) \times 320$ bits. In masked mode, the state is first processed by applying the S-box layer across all columns. The number of columns processed per cycle depends on the selected degree of parallelism (PAR). The intermediate results are stored in the shift register, shifted by PAR columns per cycle. Once all 64 columns have been processed and stored, the shares are recombined and the Linear Diffusion Layer is applied. During message processing, the architecture is reconfigured for parallel processing. In this mode, the inputs of the S-box are generalized: instead of receiving the $d+1$ shares of a single column, the S-box processes $d+1$ independent columns of the state. As a result, each cycle handles PAR $\times\ (d+1)$ columns, effectively increasing throughput. This requires adapting the S-box logic to interpret columns as shares.

The modified Ascon second-order architecture is shown in Fig. 5. The main architectural costs introduced by this generalization can be attributed to the insertion of additional multiplexers, which dynamically select whether the S-box operates on shares (for initialization/finalization) or on columns (during message processing), and to the $d+1$ parallel Round Constant Addition blocks, one per domain/column, which are needed to preserve masking consistency in all configurations. This architecture supports secure and high-throughput modes within a single design, with dynamic reconfiguration according to the current phase of Ascon and the security order d. A critical requirement in both share creation and DOM-AND computation is the availability of fresh randomness at each round to guarantee side-channel resistance. For security orders up to 2, however, *Changing of the Guards* technique (see Sec. III) removes the need for fresh random bits within masked AND computations by reusing input shares from neighbouring S-boxes—referred to as *Guards*—as an implicit randomness source. The *Guard* selection must guarantee that all interactions remain independent of sensitive intermediate values, which is particularly challenging in Ascon due to its linear diffusion layer that couples the outputs of different S-boxes [13]. In our second-order masked design, we adopt this technique, requiring three *Guards* per DOM-AND instance. Importantly, fresh randomness is still required for share creation, since the initial mask generation cannot rely on *Guards*. Extending *Changing of the Guards* beyond second order remains an open challenge, as the number and complexity of required *Guard* interactions grow rapidly with the security order d.

V. Evaluation Results and Discussion

A. Experimental leakage assessment

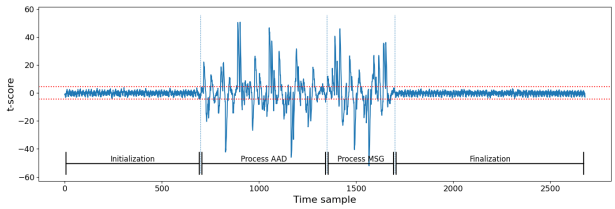
To assess whether a cryptographic implementation leaks sensitive information, the most widely adopted methodology is the Test Vector Leakage Assessment (TVLA) [36] (based on Welch's *t*-test). Two independent sets of side-channel traces are collected: one using a fixed key K_A, and the other using random keys K_B. The Welch's *t*-statistic for the first-order case is computed as:

$$t = \frac{\mu_A - \mu_B}{\sqrt{\frac{\sigma_A^2}{N_A} + \frac{\sigma_B^2}{N_B}}}, \tag{7}$$

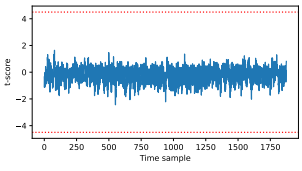
where μ_A and μ_B denote the sample means, σ_A and σ_B the sample standard deviations, and N_A and N_B the number of traces in groups A and B, respectively. For higher-order designs, TVLA extends naturally by considering higher statistical moments. Following the established convention, $|t| > 4.5$ strongly indicates the presence of leakage. We conducted Test Vector Leakage Assessment (TVLA) on the masked S-box operation using 100k traces, divided equally between fixed and random inputs (50k each). The evaluation was performed for both first- and second-order masked implementations (see Fig. 6), with focus on the initialization phase, since it operates directly on the secret key. Measurements were carried out on the CW305 Artix-7 board[2] with the FPGA was clocked at 10 MHz. For the initialization-phase TVLA, the PicoScope 5000-series oscilloscope sampled at 100 MS/s, corresponding to 10 samples per FPGA clock cycle. The *t*-scores were computed

[2]https://www.newae.com/products/nae-cw305

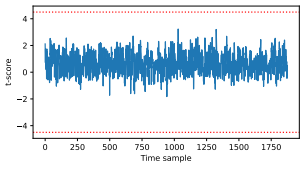
online using SCALIB [37]. For both the first- and second-order implementations (Fig. 6) the t-scores remained below the accepted ± 4.5 threshold.



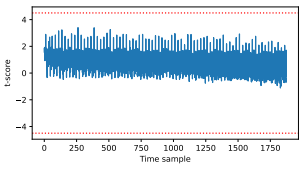
(a) 1st-order t-test over the full Ascon encryption (first-order masking).



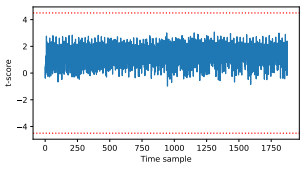
(b) 1st-order t-test.



(c) 2nd-order t-test.



(d) 1st-order t-test.

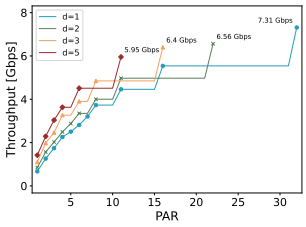


(e) 2nd-order t-test.

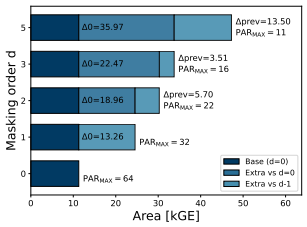
Fig. 6: TVLA results for masked Ascon: the top plot (a) shows the 1st-order t-test over the complete encryption (first-order masking), while the remaining subfigures report 1st- and 2nd-order t-tests restricted to the initialization phase for first-order masking (b,c) and second-order masking (d,e); no significant leakage is observed.

B. Design exploration and comparison with SoA.

The Ascon architecture proposed in this work is fully configurable with respect to both the degree of parallelism (PAR)—that is, the number of S-boxes instantiated in parallel—and the masking order d. These two parameters are inherently interconnected: during message processing, the masked S-boxes are reconfigured to operate in an unfolded fashion. As a result, the degree of parallelism increases from PAR to PAR$\times(d+1)$, since the hardware initially reserved for masking is repurposed to accelerate data processing. Figure 7a shows the throughput as a function of parallelism for different security orders ($d \in 1, 2, 3, 5$). We limit our evaluation to $d = 5$, as higher masking orders are generally unnecessary in practical applications. Due to the sponge structure of Ascon, a new round starts only after the previous one completes, limiting the maximum throughput at one round per cycle. The number of parallel S-box instances required to reach this throughput therefore plateau at a value that depends on the masking order, as summarized in Table I. In parallel, Figure 7b illustrates how the area grows with the masking order d when configured for maximum throughput (one round per cycle). Notably, the incremental overhead of moving from one masking order to the next remains moderate.



(a) Throughput vs. parallel S-boxes.



(b) Area vs. security order

Fig. 7: Area vs. masking order at PAR_MAX.

A comparison with state-of-the-art implementations is in Table II. The encryption throughput is computed as :

$$\mathrm{TP} = \frac{f_{ck} \cdot size(P_i)}{\mathrm{Latency}_{round}} \tag{8}$$

where $size(P_i)$ is the size of the plaintext block processed in each cycle, f_{ck} represents the clock frequency, $\mathrm{Latency}_{round}$ is the latency per round. The throughput in the table differs from Fig. 7a due to the FSM critical path. Experimental results demonstrate that our design achieves superior throughput to area ratios (TP/A) across all protection orders. This is due to the re-purpose of the masking hardware during message processing.

TABLE II: Comparison with State-of-the-Art Ascon masked implementations.

Work	Protection Order	Area [kGE]	Randomness [bit/cycle]	Latency [cycles/round]	TP [Gbps]	TP/A [Mbps/kGE]
Our Design	1-order	24.5	0*	2	5.28	215.5
	2-order	30.2	0*	2	5.21	172.5
	5-order	47.3	4800	2	5.21	110.1
[10]	1-order	30.42	4	2	3.77	124
[38]	1-order	50.4	320	1	4.35	79.8
	2-order	102.4	960	1	4.02	39.3
	5-order	3557.7	4800	1	3.34	9.3
[12]	1-order	42.8	2048	1	2.77	64.8
	2-order	90.9	4608	1	3.34	52.2
	5-order	339.8	18432	1	2.99	8.8
[13]	1-order	26.1	0*	2	n.d.	n.d.
	2-order	52.6	0*	2	n.d.	n.d.

* Obtained with *changing of the guards* technique

VI. CONCLUSION

The masked Ascon design presented in this work achieves a superior throughput-to-area ratio compared to existing designs by reconfiguring hardware originally used for side-channel protection to boost parallelism during message processing. This reconfigurable architecture combines higher-order security with efficiency, making it well-suited for constrained IoT devices. Security was validated on hardware through statistical t-tests, confirming the absence of leakage. Since the approach relies on mode-level implementations and domain-oriented masking, it generalizes beyond Ascon to schemes such as ISAP and Xoodyak. Future work will focus on evaluating this approach with additional algorithms and extending it to support multiple rounds per cycle.

ACKNOWLEDGMENT

Put sponsor acknowledgments in the unnumbered footnote on the first page.

REFERENCES

[1] National Institute of Standards and Technology. *Lightweight Cryptography*. Accessed: 2025-08-22. 2023. URL: https://csrc.nist.gov/projects/lightweight-cryptography.

[2] M. Mirigaldi, V. Piscopo, M. Martina, and G. Masera. "The Quest for Efficient ASCON Implementations: A Comprehensive Review of Implementation Strategies and Challenges". In: *Chips* 4.2 (2025), p. 15.

[3] S. Khan, W.-K. Lee, and S. O. Hwang. "Scalable and efficient hardware architectures for authenticated encryption in IoT applications". In: *IEEE Internet of Things Journal* 8.14 (2021), pp. 11260–11275.

[4] S. Khan et al. "Securing the IoT ecosystem: ASIC-based hardware realization of Ascon lightweight cipher". In: *International Journal of Information Security* (2024), pp. 1–12.

[5] S. Khan, W.-K. Lee, A. Karmakar, J. M. B. Mera, A. Majeed, and S. O. Hwang. "Area–time efficient implementation of nist lightweight hash functions targeting IoT applications". In: *IEEE Internet of Things Journal* 10.9 (2022), pp. 8083–8095.

[6] S.-N. Tran, V.-T. Hoang, and D.-H. Bui. "A hardware architecture of NIST lightweight cryptography applied in IPSec to secure high-throughput low-latency IoT networks". In: *IEEE Access* 11 (2023), pp. 89240–89248.

[7] L. Pallavi, P. Singh, B. Patnaik, and B. Acharya. "High frequency architecture of lightweight authenticated cipher ASCON-128 for resource-constrained IoT devices". In: *2023 OITS International Conference on Information Technology (OCIT)*. IEEE. 2023, pp. 405–410.

[8] S. Steinegger and R. Primas. "A fast and compact RISC-V accelerator for Ascon and friends". In: *International Conference on Smart Card Research and Advanced Applications*. Springer. 2020, pp. 53–67.

[9] A. Dolmeta, V. Piscopo, M. Martina, G. Masera, et al. "CHIMERA: Cryptographic Hardware for Integrated Multipurpose Engine on RISC-V with ASCON". In: *2025 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. IEEE. 2025.

[10] H. Groß, E. Wenger, C. Dobraunig, and C. Ehrenhöfer. "Suit up!–made-to-measure hardware implementations of Ascon". In: *2015 Euromicro Conference on Digital System Design*. IEEE. 2015, pp. 645–652.

[11] H. Groß and S. Mangard. "Reconciling masking in hardware and software". In: *International Conference on Cryptographic Hardware and Embedded Systems*. Springer. 2017, pp. 115–136.

[12] H. Groß, R. Iusupov, and R. Bloem. "Generic low-latency masking in hardware". In: *IACR transactions on cryptographic hardware and embedded systems* (2018), pp. 1–21.

[13] S. H. Prasad, F. Mendel, M. Schläffer, and R. Nagpal. "Efficient low-latency masking of ascon without fresh randomness". In: *Cryptology ePrint Archive* (2023).

[14] D. Bellizia et al. "Mode-level vs. implementation-level physical security in symmetric cryptography: A practical guide through the leakage-resistance jungle". In: *Annual International Cryptology Conference*. Springer. 2020, pp. 369–400.

[15] C. Dobraunig et al. "Isap v2. 0". In: *IACR Transactions on Symmetric Cryptology* (2020), pp. 390–416.

[16] J. Daemen, S. Hoffert, M. Peeters, G. Van Assche, and R. Van Keer. "Xoodyak, a lightweight cryptographic scheme". In: *IACR Transactions on Symmetric Cryptology* (2020), pp. 60–87.

[17] R. Diffie-Hellman. "Timing Attacks on Implementations of". In: *Advances in Cryptology—CRYPTO'96: 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18–22, 1996, Proceedings*. Vol. 1109. Springer. 1996, p. 104.

[18] P. Kocher, J. Jaffe, and B. Jun. "Differential power analysis". In: *Annual international cryptology conference*. Springer. 1999, pp. 388–397.

[19] E. Oswald, S. Mangard, C. Herbst, and S. Tillich. "Practical second-order DPA attacks for masked smart card implementations of block ciphers". In: *Cryptographers' Track at the RSA Conference*. Springer. 2006, pp. 192–207.

[20] H. Yu, Z. Xuecheng, L. Zhenglin, and C. Yicheng. "Efficient DPA attacks on AES hardware implementations". In: *International Journal of Communications, Network and System Sciences* 1.1 (2008), pp. 68–73.

[21] L. Weissbart and S. Picek. "Lightweight but not easy: Side-channel analysis of the ascon authenticated cipher on a 32-bit microcontroller". In: *Cryptology ePrint Archive* (2023).

[22] A. Rezaeezade, A. Basurto-Becerra, L. Weissbart, and G. Perin. "One for all, all for ascon: Ensemble-based deep learning side-channel analysis". In: *International Conference on Applied Cryptography and Network Security*. Springer. 2024, pp. 139–157.

[23] S.-C. You, M. G. Kuhn, S. Sarkar, and F. Hao. "Low trace-count template attacks on 32-bit implementations of ASCON AEAD". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2023.4 (2023), pp. 344–366.

[24] K. Ramezanpour, P. Ampadu, and W. Diehl. "SCARL: side-channel analysis with reinforcement learning on the ascon authenticated cipher". In: *arXiv preprint arXiv:2006.03995* (2020).

[25] L. Goubin and J. Patarin. "DES and differential power analysis the "Duplication" method". In: *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer. 1999, pp. 158–172.

[26] C. Herbst, E. Oswald, and S. Mangard. "An AES smart card implementation resistant to power analysis attacks". In: *International conference on applied cryptography and network security*. Springer. 2006, pp. 239–252.

[27] C. Clavier, J.-S. Coron, and N. Dabbous. "Differential power analysis in the presence of hardware countermeasures". In: *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer. 2000, pp. 252–263.

[28] Y. Ishai, A. Sahai, and D. Wagner. "Private circuits: Securing hardware against probing attacks". In: *Annual International Cryptology Conference*. Springer. 2003, pp. 463–481.

[29] H. Groß, S. Mangard, and T. Korak. "Domain-oriented masking: Compact masked hardware implementations with arbitrary protection order". In: *Cryptology ePrint Archive* (2016).

[30] M. S. Turan, K. A. McKay, D. Chang, J. Kang, and J. Kelsey. *Ascon-Based Lightweight Cryptography Standards for Constrained Devices: Authenticated Encryption, Hash, and Extendable Output Functions*. NIST Special Publication 800-232 (Initial Public Draft). NIST SP 800-232 ipd. Gaithersburg, MD: National Institute of Standards and Technology, 2024.

[31] S. Dziembowski and K. Pietrzak. "Leakage-resilient cryptography". In: *2008 49th Annual IEEE Symposium on Foundations of Computer Science*. IEEE. 2008, pp. 293–302.

[32] O. Pereira, F.-X. Standaert, and S. Vivek. "Leakage-resilient authentication and encryption from symmetric cryptographic primitives". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. 2015, pp. 96–108.

[33] N. Samwel and J. Daemen. "DPA on hardware implementations of Ascon and Keyak". In: *Proceedings of the Computing Frontiers conference*. 2017, pp. 415–424.

[34] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. "Keccak". In: *Annual international conference on the theory and applications of cryptographic techniques*. Springer. 2013, pp. 313–314.

[35] J. Daemen. "Changing of the guards: A simple and efficient method for achieving uniformity in threshold sharing". In: *International conference on cryptographic hardware and embedded systems*. Springer. 2017, pp. 137–153.

[36] T. Schneider and A. Moradi. "Leakage assessment methodology: Extended version". In: *Journal of Cryptographic Engineering* 6.2 (2016), pp. 85–99.

[37] G. Cassiers and O. Bronchain. "Scalib: A side-channel analysis library". In: *Journal of Open Source Software* 8.86 (2023), p. 5196.

[38] R. Nagpal, B. Gigerl, R. Primas, and S. Mangard. "Riding the waves towards generic single-cycle masking in hardware". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* (2022), pp. 693–717.