The Fact of the MATTER: Efficient Hardware Accelerators for Wide-Block Memory Encryption

Shubham Namdeo Shende and Utsav Banerjee
Electronic Systems Engineering, Indian Institute of Science, Bengaluru, India
Email: shubhamn@iisc.ac.in, utsav@iisc.ac.in

***Abstract*—Tweakable block ciphers are important cryptographic primitives for secure memory encryption and partial mitigation of bit flip attacks. Rapid advancements in data-intensive applications such as artificial intelligence and machine learning have motivated the development of new memory technologies with very large data bus widths requiring wide-block memory encryption and its efficient implementation. Recently, the MATTER family of wide-block tweakable ciphers has been proposed for memory encryption in emerging applications. MATTER is a 512-bit balanced Feistel construction which uses the light-weight ASCON permutation as its round function. In this work, we present a comprehensive design space exploration of hardware architectures for accelerating different configurations of MATTER. We provide a detailed comparative analysis of power, performance, area and energy of round-based, unrolled and pipelined hardware implementations of MATTER based on digital synthesis using a 7nm FinFET ASIC standard cell library. We also discuss the suitability of these architectures for different application-specific memory encryption hardware requirements in emerging edge computing systems.**

***Index Terms*—lightweight cryptography, wide-block memory encryption, tweakable block ciphers, MATTER, ASCON, hardware accelerators, ASIC, edge computing, AI, ML, IoT.**

I. Introduction

Modern edge computing systems, such as the Internet of Things (IoT), are witnessing the ubiquitous use of artificial intelligence (AI) and machine learning (ML) technologies [1]. It is well-known that AI/ML applications involve data-intensive computations, thus requiring the movement of large blocks of data between memories and processing units. Traditional hardware architectures for AI/ML are prone to various physical attacks such as side-channel analysis [2], bit-flip faults [3] and model reverse engineering [4]. While various circuit-level, architecture-level, system-level and cryptographic countermeasures have been proposed in recent literature [5], a common denominator in most of these countermeasures is memory encryption. Due to the data-intensive nature of AI/ML computations, these are wide-interface high-throughput memories with 512 bits or 1024 bits being read or written in a single access. Tweakable block ciphers [6] are well-known cryptographic primitives used for memory encryption and decryption using the memory address as the tweak. However, standard tweakable block ciphers do not support such large block sizes, thus leading to inefficient data processing.

Recently, a new tweakable block cipher MATTER with 512-bit block size has been proposed in [7] specifically for such emerging applications. MATTER is a Feistel construction based on the ASCON light-weight permutation function [8], [9]. While there have been several hardware implementation studies for ASCON in recent literature [8], [10]–[13], hardware accelerators for MATTER are yet to be fully explored apart from the preliminary designs presented in [7].

In this work, we present a detailed design space exploration of MATTER. We first present various architectures of the ASCON permutation, and then we use them as building blocks to propose low-area and high-throughput hardware accelerators for different configurations of MATTER. We present detailed analyses of area, power, energy and performance metrics for all the designs of ASCON and MATTER synthesized using the ASAP 7nm standard cell library [14], [15]. Our low-area implementations consume around 100-140 pJ energy to process a 512-bit data block and achieve decent throughputs around 4-5 GBps. Our high-throughput pipelined implementations consume around 70 pJ per 512-bit data block and achieve significantly higher throughputs around 80-100 GBps. Their low energy consumption and wide range of throughputs make them well suited for encryption / decryption of large memory blocks in secure AI / ML applications. Our results indicate that MATTER is a promising candidate for securing wide high-performance memories in edge computing systems.

II. Background

A. Overview of DRAM Technologies

The data-intensive nature of modern edge computing applications such as AI/ML has made efficient on-chip static random access memories (SRAMs) and off-chip dynamic random access memories (DRAMs) critical components of the overall system design. While high-performance SRAMs act as on-chip buffers and caches, high-capacity DRAMs are used as the main working memories in AI/ML edge computing systems [16]. Based on the application requirements, different DRAM technologies such as DDR (Double Data Rate), LPDDR (Low-Power Double Data Rate), GDDR (Graphics Double Data Rate) and HBM (High-Bandwidth Memory) provide trade-offs

© 2025 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 collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This is an extended / revised version of the following paper with the same title published in the proceedings of the 2025 IEEE International Conference on Intelligent Computing and Systems at the Edge (ICEdge) - DOI: 10.1109/ICEdge67252.2025.11412927

TABLE I
COMPARISON OF LATEST DRAM TECHNOLOGIES

	DDR5	LPDDR5X	GDDR7	HBM3E
Maximum Capacity	512 GB per module	64 GB per package	8 GB per chip	48 GB per stack
Interface Width	64 bits	32 bits	32 bits	1024 bits
Peak Bandwidth	$\approx$ 70 GBps	$\approx$ 1 GBps	$\approx$ 192 GBps	$\approx$ 1229 GBps
Access Energy	15-20 pJ per bit	10-15 pJ per bit	$\geq$ 20 pJ per bit	$\leq$ 10 pJ per bit
Packaging Complexity	Standard DIMM	SoC / PoP Integration	Mounted on PCB	2.5D / 3D Stacking
Latency	Medium	Medium	Medium	Low
Cost	Low	Medium	Medium	High

between memory capacity, bandwidth, energy efficiency, design complexity, latency and cost [17]. These modern DRAM technologies offer a wide range of peak memory bandwidths from 1 GBps to 1 TBps with varying interface widths in the range of 32 bits to 1024 bits [17], [18]. DDR is widely used as a cost-effective solution for general-purpose computing and edge servers. Due to its low energy consumption, LPDDR is best suited for battery-powered edge devices with moderate performance needs. The higher bandwidth of GDDR makes it well suited for autonomous systems at the edge. At the cost of complex system-level integration, HBM achieves the lowest energy, lowest latency and highest bandwidth, thus enabling massively parallel computations required by performance-critical edge AI inference and training hardware accelerators. These trade-offs are summarized in Table I [17], [18].

B. DRAM Encryption and Integrity Protection

Practical DRAM hardware is vulnerable to security threats such as data remanence, fault injection, bus snooping and tampering which attempt to compromise confidentiality and/or integrity of the stored data [16]. To counter some of these attacks in general-purpose computing applications, commercial DRAM controllers employ inline encryption and message authentication using standard algorithms XTS (XOR-Encrypt-XOR Tweakable Block Cipher with Ciphertext Stealing) [19] and GMAC (Galois Message Authentication Code) [20] respectively based on the Advanced Encryption Standard (AES) [21]. However, DRAM security has gained renewed interest in the context of AI/ML hardware, edge computing and resource-constrained devices where it is important to maintain the confidentiality of private models and intermediate data stored in memory as well as verify their integrity to prevent misclassification and malfunctioning [5]. This has motivated the design and implementation of advanced cryptographic primitives for authenticated encryption compatible with wide memory block sizes and application-specific power, performance, area and energy requirements of AI/ML computing systems.

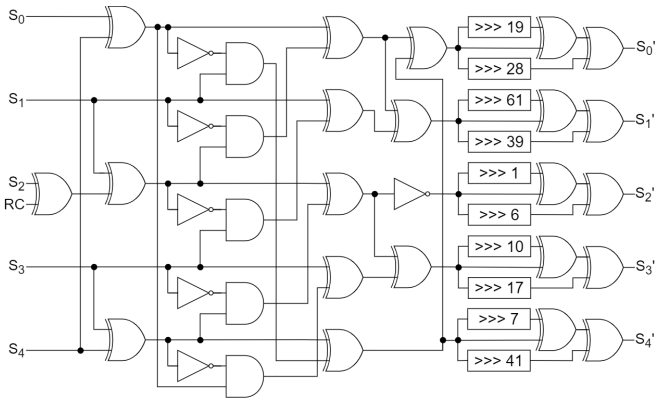


Fig. 1. ASCON-p permutation with 64-bit round constant RC, 64-bit input words S_0, S_1, S_2, S_3, S_4 and 64-bit output words $S'_0, S'_1, S'_2, S'_3, S'_4$.

C. Tweakable Block Ciphers

Tweakable block ciphers [6] are symmetric cryptographic primitives which, apart from the standard inputs of plaintext / ciphertext and secret encryption / decryption key, incorporate an additional "tweak" input. The tweak allows for variability in the encryption process without altering the secret key, thus providing enhanced flexibility and security in various applications requiring unique encryption contexts. The tweak serves a role similar to an initialization vector or nonce but is integrated at the primitive level of the cipher. Tweakable block ciphers are widely used for memory encryption, where the memory address is used as the tweak input to ensure that identical data blocks at different memory locations are encrypted differently [22]. The most common tweakable block cipher used to encrypt commercial memory systems is XTS-AES [19], [21]. Other tweakable block ciphers which have been proposed in recent literature include Deoxys [23], Kiasu [23], Joltik [24], SKINNY [25], MANTIS [25], QARMA [26], [27], CRAFT [28], Lilliput [29], BipBip [30] and MATTER [7]. While the MATTER tweakable block cipher has a 512-bit block size, all others support block sizes of 128 bits or less.

D. ASCON Light-Weight Permutation

ASCON [8] is a family of light-weight cryptographic algorithms (authenticated encryption, message authentication codes, hash function, extendable output function, etc) that has been recently selected as the NIST Light-Weight Cryptography (LWC) standard [9]. ASCON is a sponge construction with a 320-bit state which is divided into 64-bit words and iteratively transformed using 6, 8 or 12 rounds of the permutation function p shown in Figure 1. The permutation involves addition of a round constant followed by a non-linear substitution layer requiring 64 parallel applications of 5-bit S-Boxes derived from the χ function of SHA-3 (Keccak) [31] and finally a linear diffusion layer inspired by the Σ function of SHA-2 [32]. The ASCON-p permutation is designed for efficient software and hardware implementations in constrained environments such as embedded systems and edge computing devices along with improved side-channel resilience [8], [10]–[13].

Algorithm 1 MATTER tweakable block cipher [7]

Require: 320-bit main key K, 320-bit tweak T, 512-bit data input block in, number of Feistel rounds N, number of ASCON key rounds r', number of ASCON main rounds $r_0, r_1, \cdots, r_{N-1}$ and round constants $c_0, c_1, \cdots, c_{N-1}$

Ensure: 512-bit data output block out

1: $t_0 \leftarrow K \oplus T$ ▷ tweak encryption
2: **for** $j = 0; j < r'; j = j + 1$ **do** ▷ ASCON rounds
3: $t_0 \leftarrow$ ASCON-$p(t_0)$
4: **end for**
5: $k \leftarrow t_0 \oplus K$
6: $L_0 \leftarrow$ most significant 256 bits of in
7: $R_0 \leftarrow$ least significant 256 bits of in
8: **for** $i = 0; i < N; i = i + 1$ **do** ▷ Feistel rounds
9: $k_i \leftarrow (2^i \cdot k) \oplus c_i$ ▷ round key derivation
10: $k_i' \leftarrow$ least significant 256 bits of k_i
11: $L_{i+1} \leftarrow R_i$
12: $t_1 \leftarrow$ zero-pad$(R_i) \oplus k_i$ ▷ zero-pad to 320 bits
13: **for** $j = 0; j < r_i; j = j + 1$ **do** ▷ ASCON rounds
14: $t_1 \leftarrow$ ASCON-$p(t_1)$
15: **end for**
16: $R_{i+1} \leftarrow L_i \oplus k_i' \oplus$ truncate(t_1) ▷ truncate to 256 bits
17: **end for**
18: **return** $out = L_N \,\|\, R_N$

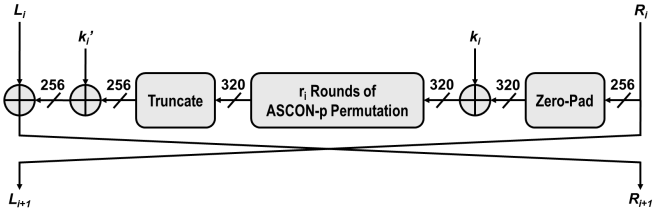


Fig. 2. Structure of the i-th Feistel round function in MATTER-core.

E. MATTER Wide-Block Tweakable Cipher

MATTER [7] is a family of wide-block tweakable ciphers recently proposed by Avanzi, Dunkelman and Minematsu. MATTER has been designed to be a hardware-friendly solution for memory encryption to ensure data confidentiality and partially mitigate data tampering [33]. It is a 512-bit balanced Feistel network constructed using a non-tweakable keyed core function MATTER-core whose key is derived from the main encryption / decryption key and the tweak. The MATTER-core consists of N rounds of the Feistel round function ($3 \leq N \leq 6$) shown in Figure 2. The i-th Feistel round ($0 \leq i < N$) applies r_i rounds of the ASCON-p permutation, where k_i and k_i' are respectively 320-bit and 256-bit round keys, while L_i and R_i are 256-bit half-blocks of the 512-bit input. To balance security and efficiency, it is recommended that $r_i \geq 4$ and $\sum_i r_i \leq 24$. The overall MATTER 512-bit tweakable block cipher is described in Algorithm 1. Derivation of the 320-bit MATTER-core key k from the main key K and tweak T in step 1 involves r' rounds of the ASCON-p permutation (it is recommended to have $r' \geq 6$ for strong security). Derivation of the i-th round key k_i in step 5 involves multiplication of k with 2^i where 2 represents the image of x in $\mathbb{F}_{2^{320}}[x] = \mathbb{F}_2[x]/(x^{320} + x^4 + x^3 + x + 1)$. The round constants c_i are derived from the hexadecimal expansion of the fractional part of π as specified in [7]. A tweakable block cipher instance of the MATTER family with N Feistel rounds, r' ASCON key rounds and $r_0, r_1, \cdots, r_{N-1}$ ASCON main rounds is denoted as MATTER$^{r'}_{r_0,r_1,\cdots,r_{N-1}}$. Some concrete instantiations proposed in [7] which achieve at least 128-bit security level are MATTER$^8_{8,8,8}$ (3 Feistel rounds), MATTER$^6_{6,12,6}$ (3 Feistel rounds), MATTER$^6_{6,6,6,6}$ (4 Feistel rounds), MATTER$^8_{4,8,8,4}$ (4 Feistel rounds), MATTER$^8_{4,4,8,4,4}$ (5 Feistel rounds) and MATTER$^8_{4,4,4,4,4,4}$ (6 Feistel rounds). Note that all these instances ensure $\sum_i r_i = 24$ to have similar latency in the main encryption / decryption path. In this work, we consider the same for our implementations and analyses.

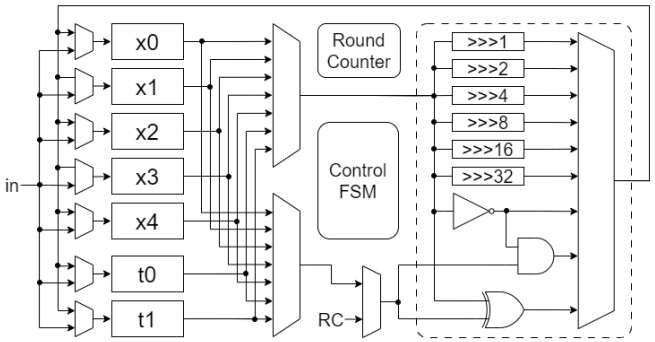
Various hardware architectures of ASCON were proposed by [8], [10]–[13]. Preliminary hardware implementations of MATTER$^8_{8,8,8}$ and MATTER$^6_{6,6,6,6}$ were presented by [7]. In this work, we present a comprehensive design space exploration of hardware architectures of different MATTER instantiations and their detailed comparative analysis. We use Verilog HDL for all our designs, Cadence Incisive for register transfer level simulations and Cadence Genus for synthesis using the ASAP7 predictive 7nm FinFET process design kit (PDK) and 7.5-track ASIC standard cell library [14], [15]. All our implementations are tuned for high performance during synthesis by aggressively constraining the clock period to obtain the best critical path optimization [27].

III. Design Space Exploration of ASCON Permutation Hardware Architectures

As evident from Algorithm 1, the ASCON-p permutation is the most important building block of MATTER. Therefore, we first present various hardware architectures with design trade-offs which can compute p^4, p^6, p^8 and p^{12}, that is, 4, 6, 8 and 12 rounds respectively of ASCON-p.

A. ALU-Based Implementations

A compact 64-bit data-path micro-controller-like architecture of ASCON was proposed in [10] with five 64-bit state registers x_0, x_1, x_2, x_3, x_4, two 64-bit temporary registers t_0, t_1, a 64-bit ASCON-specific *arithmetic logic unit* (ALU), a round counter and a controller *finite state machine* (FSM) which executes a sequence of instructions over 59 clock cycles to perform a single ASCON-p round. This architecture, which we denote as A_0, is shown in Figure 3 along with the ALU design and the instructions executed. The ALU takes two input operands selected from the registers using two 7-to-1 multiplexers, and supports the following instructions: ROTATE by 1 / 2 / 4 / 8 / 16 / 32, NOT, NOT-AND and XOR. We propose an improved ALU-based architecture which requires only 28 clock cycles while having similar area as the ALU-based implementation from [10]. Our proposed design, which we denote as A_1, is shown in Figure 4. This ALU supports only two instructions: ROTATE-XOR and NOT-AND. Instead



Sequence of ALU Instructions for ASCON-p Round Computation:

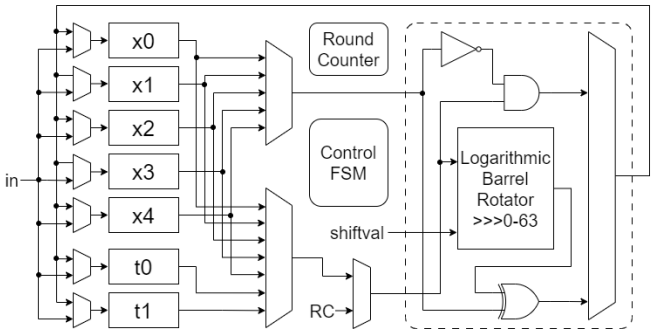
1. $x_2 \leftarrow x_2 \oplus RC$	21. $x_2 \leftarrow \neg x_2 \wedge x_3$	41. $x_1 \leftarrow x_1 \oplus t_1$
2. $t_1 \leftarrow x_0 \oplus x_4$	22. $x_1 \leftarrow x_1 \oplus x_2$	42. $t_0 \leftarrow x_2 \ggg 1$
3. $t_0 \leftarrow x_3 \oplus x_4$	23. $x_1 \leftarrow x_0 \oplus x_1$	43. $t_1 \leftarrow x_2 \ggg 2$
4. $x_2 \leftarrow x_1 \oplus x_2$	24. $x_0 \leftarrow x_0 \oplus x_4$	44. $t_1 \leftarrow t_1 \ggg 4$
5. $x_0 \leftarrow \neg x_1 \wedge x_2$	25. $x_3 \leftarrow t_0 \oplus t_1$	45. $x_2 \leftarrow x_2 \oplus t_0$
6. $x_0 \leftarrow x_0 \oplus t_1$	26. $x_2 \leftarrow \neg t_0$	46. $x_2 \leftarrow x_2 \oplus t_1$
7. $x_4 \leftarrow \neg x_1 \wedge x_2$	27. $t_0 \leftarrow x_0 \ggg 16$	47. $t_0 \leftarrow x_3 \ggg 2$
8. $x_4 \leftarrow x_0 \oplus x_4$	28. $t_1 \leftarrow t_0 \ggg 8$	48. $t_0 \leftarrow t_0 \ggg 8$
9. $x_4 \leftarrow \neg x_4 \wedge x_1$	29. $t_1 \leftarrow t_1 \ggg 4$	49. $t_1 \leftarrow x_3 \ggg 1$
10. $x_4 \leftarrow t_0 \oplus x_4$	30. $t_0 \leftarrow t_0 \ggg 2$	50. $t_1 \leftarrow t_1 \ggg 16$
11. $t_1 \leftarrow \neg x_1 \wedge x_2$	31. $t_0 \leftarrow t_0 \ggg 1$	51. $x_3 \leftarrow x_3 \oplus t_0$
12. $t_1 \leftarrow x_0 \oplus t_1$	32. $x_0 \leftarrow x_0 \oplus t_0$	52. $x_3 \leftarrow x_3 \oplus t_1$
13. $t_1 \leftarrow \neg t_0 \wedge t_1$	33. $x_0 \leftarrow x_0 \oplus t_1$	53. $t_0 \leftarrow x_4 \ggg 1$
14. $t_1 \leftarrow x_3 \oplus t_1$	34. $t_0 \leftarrow x_1 \ggg 32$	54. $t_1 \leftarrow t_0 \ggg 8$
15. $t_0 \leftarrow \neg x_1 \wedge x_2$	35. $t_0 \leftarrow t_0 \ggg 1$	55. $t_1 \leftarrow t_1 \ggg 32$
16. $t_0 \leftarrow x_0 \oplus t_0$	36. $t_0 \leftarrow t_0 \ggg 4$	56. $t_0 \leftarrow t_0 \ggg 2$
17. $t_0 \leftarrow \neg t_0 \wedge x_1$	37. $t_1 \leftarrow t_0 \ggg 16$	57. $t_0 \leftarrow t_0 \ggg 4$
18. $t_0 \leftarrow x_4 \oplus t_0$	38. $t_1 \leftarrow t_1 \ggg 8$	58. $x_4 \leftarrow x_4 \oplus t_0$
19. $t_0 \leftarrow \neg x_3 \wedge t_0$	39. $t_0 \leftarrow t_0 \ggg 2$	59. $x_4 \leftarrow x_4 \oplus t_1$
20. $t_0 \leftarrow x_2 \oplus t_0$	40. $x_1 \leftarrow x_1 \oplus t_0$	

Fig. 3. Original ALU-based architecture of ASCON-p from [10] and sequence of instructions executed by its FSM over 59 cycles per round.

of decomposing bit-wise rotations into power-of-2 shifts over multiple clock cycles, it implements a *logarithmic barrel rotator* [34] which can rotate its input operand by any value in the range 0-63 specified by an immediate operand *shiftval* in a single clock cycle. Inspired by the software optimizations from [8], we reduce the number of instructions required in A_1 by $\approx 2\times$ compared to A_0. We also re-order the input operands in all instructions such that the first operand is always chosen from x_0-x_4, thus replacing one of the 7-to-1 multiplexers with a smaller 5-to-1 multiplexer. The post-synthesis area, power, energy and performance metrics of ALU-based ASCON-p architectures A_0 and A_1 are summarized in Table II.

B. Round-Based Implementation

Round-based architectures are commonly used in hardware implementations of iterative ciphers [10], and are known to be more energy-efficient compared to serialized designs [35]. Figure 5 shows a generic architecture for ASCON-p where the round-based implementation has a 320-bit state register, 1 instance of the p permutation module and control logic to perform 1 permutation round per clock cycle. It computes p^r, that is, r rounds of the ASCON permutation, over r clock cycles. We denote this architecture as R_1, and its post-synthesis area, power, energy and performance metrics are summarized in Table II.



Sequence of ALU Instructions for ASCON-p Round Computation:

1. $x_2 \leftarrow x_2 \oplus RC$	15. $x_1 \leftarrow x_0 \oplus x_1$
2. $x_0 \leftarrow x_0 \oplus x_4$	16. $x_3 \leftarrow x_2 \oplus x_3$
3. $x_4 \leftarrow x_3 \oplus x_4$	17. $x_0 \leftarrow x_0 \oplus x_4$
4. $x_2 \leftarrow x_1 \oplus x_2$	18. $x_2 \leftarrow x_2 \oplus$ `64'hFFFFFFFFFFFFFFFF` †
5. $t_0 \leftarrow \neg x_4 \wedge x_0$	19. $t_0 \leftarrow x_0 \oplus (x_0 \ggg 9)$
6. $t_1 \leftarrow \neg x_1 \wedge x_2$	20. $x_0 \leftarrow x_0 \oplus (t_0 \ggg 19)$
7. $x_0 \leftarrow x_0 \oplus t_1$	21. $t_0 \leftarrow x_1 \oplus (x_1 \ggg 22)$
8. $t_1 \leftarrow \neg x_3 \wedge x_4$	22. $x_1 \leftarrow x_1 \oplus (t_0 \ggg 39)$
9. $x_2 \leftarrow x_2 \oplus t_1$	23. $t_0 \leftarrow x_2 \oplus (x_2 \ggg 5)$
10. $t_1 \leftarrow \neg x_0 \wedge x_1$	24. $x_2 \leftarrow x_2 \oplus (t_0 \ggg 1)$
11. $x_4 \leftarrow x_4 \oplus t_1$	25. $t_0 \leftarrow x_3 \oplus (x_3 \ggg 7)$
12. $t_1 \leftarrow \neg x_2 \wedge x_3$	26. $x_3 \leftarrow x_3 \oplus (t_0 \ggg 10)$
13. $x_1 \leftarrow x_1 \oplus t_1$	27. $t_0 \leftarrow x_4 \oplus (x_4 \ggg 34)$
14. $x_3 \leftarrow x_3 \oplus t_0$	28. $x_4 \leftarrow x_4 \oplus (t_0 \ggg 7)$

† Performs bit-wise NOT by XOR with `64'hFFFFFFFFFFFFFFFF`

Fig. 4. Our proposed improved ALU-based architecture of ASCON-p and sequence of instructions executed by its FSM over 28 cycles per round.

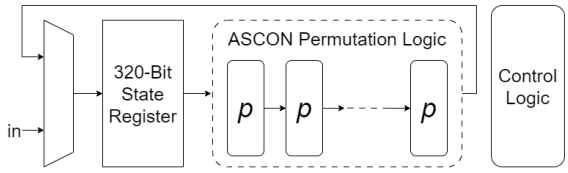


Fig. 5. Architecture of ASCON-p for round-based (1 p instance), partially-unrolled (2 p instances) and fully-unrolled designs (4 / 6 / 8 / 12 p instances).

C. Partially-Unrolled Implementation

For the configurations of MATTER mentioned in Section II, we need to support p^4, p^6, p^8 and p^{12}. A partially-unrolled design based on the architecture from Figure 5 with 2 instances of the p permutation module can perform 2 permutation rounds per clock cycle. Therefore, it computes p^4, p^6, p^8 and p^{12} in 2, 3, 4 and 6 cycles respectively. We denote this architecture as R_2, and it is faster than R_1 at the cost of larger area. Its post-synthesis area, power, energy and performance metrics are summarized in Table II.

D. Fully-Unrolled Implementations

Fully-unrolled designs for p^4, p^6, p^8 and p^{12} can perform respectively 4, 6, 8 and 12 permutation rounds per clock cycle. Based on the architecture from Figure 5, these designs have 4, 6, 8 and 12 instances of the p permutation module respectively. We denote these architectures as U_4, U_6, U_8 and U_{12} respectively, and they are faster than R_1 and R_2 at the cost of much larger area. Their post-synthesis area, power, energy and performance metrics are summarized in Table III.

TABLE II
PERFORMANCE COMPARISON OF ALU-BASED, ROUND-BASED AND PARTIALLY-UNROLLED ASCON-p ARCHITECTURES IN ASAP7 PDK

	A_0	A_1	R_1	R_2
Area (kGE)	3.91	3.83	4.19	7.08
Maximum Frequency (GHz)	2.03	1.76	3.39	2.17
Clock Cycles per ASCON-p Round	59	28	1	0.5
Time (ns) per ASCON-p Round	29.02	15.87	0.29	0.23
Power (mW) at 1 GHz	0.59	0.80	1.75	3.22
Energy (pJ) per ASCON-p Round	34.99	22.40	1.75	1.61

TABLE III
PERFORMANCE COMPARISON OF FULLY-UNROLLED ASCON-p ARCHITECTURES IN ASAP7 PDK

	U_4	U_6	U_8	U_{12}
ASCON Permutation	p^4	p^6	p^8	p^{12}
Area (kGE)	13.54	19.74	25.99	38.36
Maximum Frequency (GHz)	1.32	0.91	0.70	0.48
Clock Cycles per Permutation	1	1	1	1
Time per (ns) Permutation	0.76	1.09	1.43	2.10
Power (mW) at 0.25 GHz	1.62	2.47	3.37	5.09
Energy (pJ) per Permutation	6.48	9.88	13.47	20.37

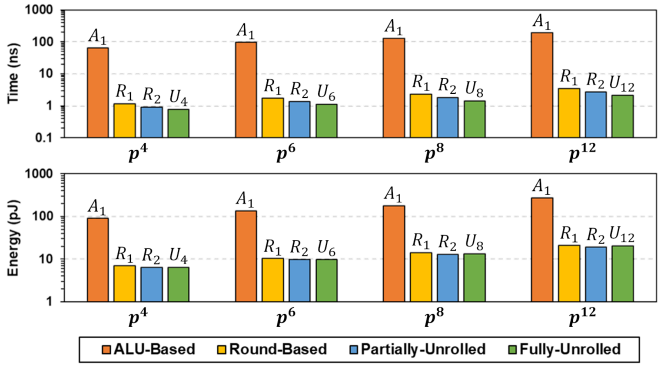


Fig. 6. Comparison of time and energy required to compute p^4, p^6, p^8 and p^{12} by ALU-based (A_1), round-based (R_1), partially-unrolled (R_2) and fully-unrolled (U_4 / U_6 / U_8 / U_{12}) architectures of the ASCON-p permutation.

E. Comparative Study

While A_1, R_1 and R_2 are all flexible to compute p^4, p^6, p^8 and p^{12} over different number of clock cycles, U_4, U_6, U_8 and U_{12} are specifically designed to compute only p^4, p^6, p^8 and p^{12} respectively in single clock cycle. The critical path delay increases with more cascaded instances of the p permutation module in unrolled designs, thus reducing the maximum clock frequency. We observe that the ALU-based implementation A_1 is only marginally smaller than the round-based design R_1, but is an order of magnitude slower. The partially-unrolled design R_2 is marginally faster than the round-based design R_1, but is significantly larger. The fully-unrolled designs are all slightly faster than the round-based and partially-unrolled designs, but are an order of magnitude larger. Figure 6 compares the time and energy required by these different ASCON-p hardware architectures to compute p^4, p^6, p^8 and p^{12}.

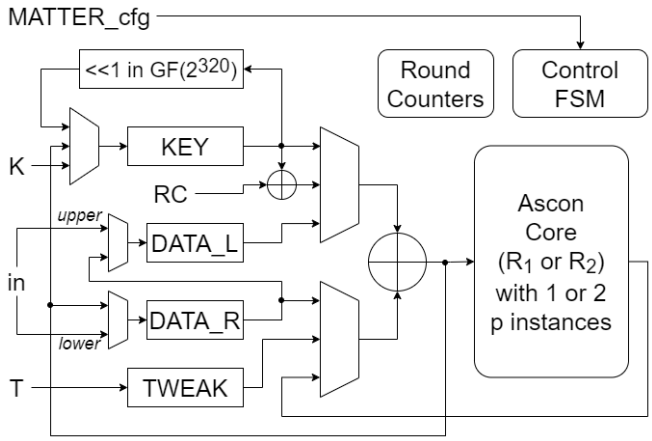


Fig. 7. Low-area configurable MATTER accelerator using round-based and partially-unrolled ASCON cores (R_1 and R_2 respectively).

IV. HARDWARE ACCELERATORS FOR MATTER

With the ASCON architectures presented in Section III as building blocks, we now present ASIC implementations of various configurations of MATTER targeting low area and high throughput while maintaining low energy consumption.

A. Low-Area Implementations

Our low-area MATTER designs are based on round-based and partially-unrolled ASCON, as shown in Figure 7. Apart from the ASCON core, these implementations contain input / output registers, 64-bit XOR logic for round key generation, 320-bit XOR logic for encryption / decryption computation, several multiplexers, round counters (for both the MATTER Feistel rounds and the ASCON permutation rounds) and control logic. We denote the accelerators with round-based and partially-unrolled ASCON as L_1 and L_2 respectively. Both these designs can accelerate any instantiation $\text{MATTER}^{r'}_{r_0,r_1,\cdots,r_{N-1}}$ by configuring the parameters $N, r', r_0, r_1, \cdots, r_{N-1}$, thus making them flexible and efficient. Since design L_1 performs one ASCON-p round per cycle, its total latency is $(2 + 3N + r' + \sum_{i=0}^{N-1} r_i)$ cycles. Since design L_2 performs two ASCON-p rounds per cycle, its total latency is $(2 + 3N + \frac{1}{2}(r' + \sum_{i=0}^{N-1} r_i))$ cycles. These cycle counts include both the tweak and data processing. The post-synthesis area, power and performance metrics of low-area configurable MATTER accelerators L_1 and L_2 are summarized in Table IV. The cycle counts required to encrypt / decrypt one 512-bit block using $\text{MATTER}^{r'}_{r_0,r_1,\cdots,r_{N-1}}$ are also provided (include both tweak and data processing). Clearly, L_1 is smaller, has lower power consumption and operates at a higher clock frequency, while L_2 requires smaller number of clock cycles to process a 512-bit input block.

B. High-Throughput Implementations

Our high-throughput MATTER designs are based on fully-unrolled and pipelined ASCON, as shown in Figure 8. While a naive approach would be to have the tweak encryption and each Feistel round as an individual pipeline stage, our unrolled

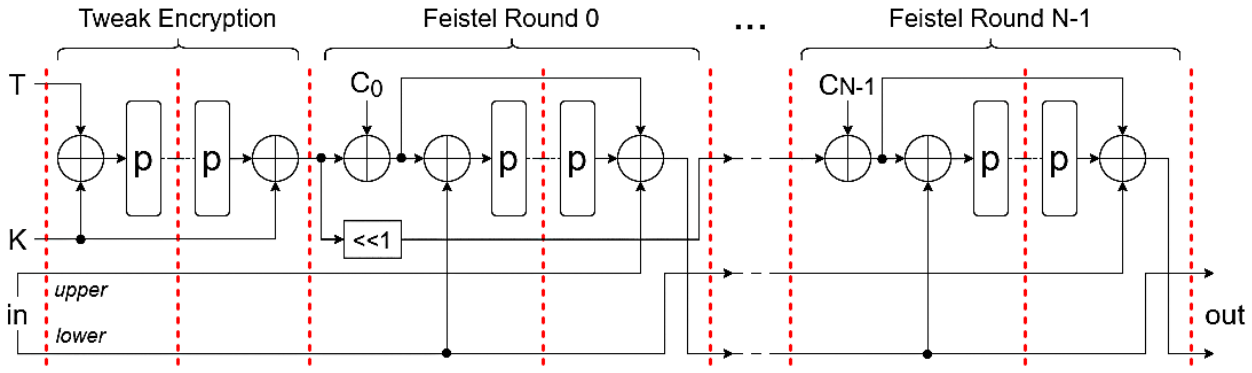


Fig. 8. Generic architecture of high-throughput MATTER accelerator with fully-unrolled and pipelined ASCON rounds (pipeline stages shown in red).

TABLE IV
PERFORMANCE OF PROPOSED LOW-AREA CONFIGURABLE MATTER ACCELERATORS IN ASAP7 PDK

	L_1	L_2
Area (kGE)	12.68	15.91
Maximum Frequency (GHz)	3.34	2.28
Power (mW) at 1 GHz	2.28	3.92
Clock Cycles per 512b Block with MATTER$^{r'}_{r_0,r_1,\cdots,r_{N-1}}$	$2+3N+r'$ $+\sum_{i=0}^{N-1} r_i$	$2+3N+\frac{1}{2}r'$ $+\frac{1}{2}\sum_{i=0}^{N-1} r_i$

TABLE V
PERFORMANCE OF PROPOSED HIGH-THROUGHPUT PIPELINED MATTER ACCELERATORS IN ASAP7 PDK

Design	**MATTER Configuration**	**Area (kGE)**	**Max Freq (GHz)**	**Power (mW) at 1 GHz**
H_1	MATTER$^8_{8,8,8}$	131.05	1.28	67.85
H_2	MATTER$^8_{4,8,8,4}$	130.67	1.27	67.22
H_3	MATTER$^8_{4,4,8,4,4}$	130.65	1.26	67.38
H_4	MATTER$^8_{4,4,4,4,4,4}$	130.86	1.25	67.44
H_5	MATTER$^6_{6,12,6}$	132.48	1.62	67.22
H_6	MATTER$^6_{6,6,6,6}$	133.30	1.64	67.94

ASCON implementation results from Section III indicate that this can lead to significant critical path delay in MATTER configurations which require p^6, p^8 and p^{12}. Based on the performance metrics of the unrolled ASCON architectures in Table III, we decide to implement p^4, p^6, p^8 and p^{12} as 1, 2, 2 and 4 pipeline stages respectively. This ensures that each pipeline stage involves 3 or 4 cascaded ASCON-p instances along with XOR gates required for the Feistel round computations. Therefore, we have 8-stage pipelined architectures for MATTER$^8_{8,8,8}$, MATTER$^8_{4,8,8,4}$, MATTER$^8_{4,4,8,4,4}$ and MATTER$^8_{4,4,4,4,4,4}$ (4 cascaded ASCON-p instances per stage), and 10-stage pipelined architectures for MATTER$^6_{6,12,6}$ and MATTER$^6_{6,6,6,6}$ (3 cascaded ASCON-p instances per stage). Apart from permutation logic, each pipeline stage also contains several XOR gates required for tweak encryption and Feistel round computations. We denote these designs as H_1, H_2, H_3, H_4, H_5 and H_6 respectively. These implementations

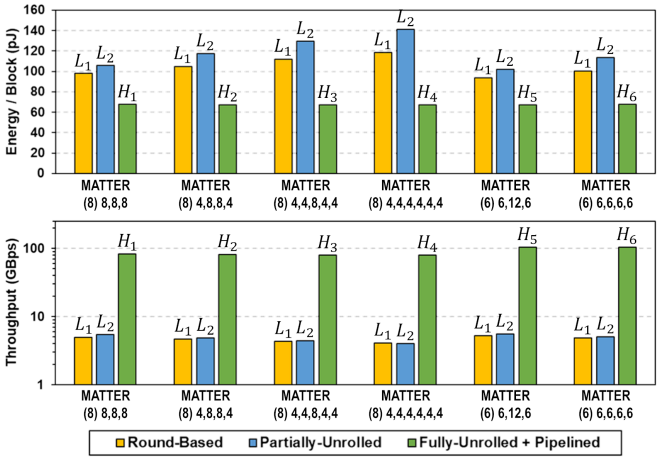


Fig. 9. Comparison of energy consumption and throughput achieved by our proposed low-area and high-throughput MATTER accelerators.

are fixed for specific MATTER configurations and their post-synthesis area, power, energy and performance metrics are summarized in Table V. Clearly, these pipelined implementations are an order of magnitude larger than the low-area designs and have significantly higher power consumption. However, they are also able to achieve low energy consumption and very high throughput, as discussed next.

C. Comparison and Analysis

Figure 9 compares the energy consumption and throughput achieved by our proposed MATTER accelerators. The low-area designs L_1-L_2 require around 100-140 pJ to process a 512-bit input block and achieve decent throughputs around 4-5 GBps. They are well suited for low-power edge computing systems requiring DDR and LPDDR encryption / decryption. The high-throughput designs H_1-H_6 have a lower energy consumption of around 70 pJ per block and achieve significantly higher throughputs around 80-100 GBps. They can be used in high-performance edge computing systems requiring GDDR and HBM encryption / decryption. We note that the throughputs of our pipelined implementations can be further increased through deeper pipelining, but at the expense of higher sequential circuit area and power consumption.

TABLE VI
COMPARISON OF PROPOSED HARDWARE ACCELERATORS FOR MATTER$^{8}_{8,8,8}$ AND MATTER$^{6}_{6,6,6,6}$ WITH STATE-OF-THE-ART

Configuration	Design	Technology Library	Frequency (GHz)	Area (kGE)	Latency (ns)	Throughput (GBps)	FoM (GBps / kGE)
MATTER$^{8}_{8,8,8}$	Fully-Unrolled [7]	TSMC 5nm	0.69	97.06	1.45	44.14	0.45
	Round-Based (This work)	ASAP 7nm	3.34	**12.68**	12.90	4.96	0.39
	Partially-Unrolled (This work)	ASAP 7nm	2.28	**15.91**	11.88	5.39	0.34
	Fully-Unrolled + Pipelined (This work)	ASAP 7nm	1.28	131.05	6.23	**82.24**	**0.63**
MATTER$^{6}_{6,6,6,6}$	Fully-Unrolled [7]	TSMC 5nm	0.78	92.44	1.29	49.61	0.54
	Round-Based (This work)	ASAP 7nm	3.34	**12.68**	13.2	4.85	0.38
	Partially-Unrolled (This work)	ASAP 7nm	2.28	**15.91**	12.76	5.02	0.32
	Fully-Unrolled + Pipelined (This work)	ASAP 7nm	1.64	133.30	6.11	**104.77**	**0.79**

Table VI compares our hardware accelerators for MATTER$^{8}_{8,8,8}$ and MATTER$^{6}_{6,6,6,6}$ with state-of-the-art fully-unrolled implementations from [7] in terms of area and throughput. Our round-based and partially-unrolled designs L_1 and L_2 are significantly smaller. Our fully-unrolled pipelined designs H_1 and H_6 are somewhat larger but achieve significantly higher throughputs. For fair comparison, we use throughput per unit area as the figure-of-merit (FoM). Our designs H_1 and H_6 achieve a higher FoM compared to those from [7] despite being implemented in an older technology node with a generic standard cell library (the foundry-provided library used in [7] is expected to have more optimized logic circuits). This demonstrates the superiority of our proposed hardware accelerators and validates the design space exploration analyses presented in this work.

V. CONCLUSION

In this work, we have presented a detailed design space exploration of the recently proposed wide-block tweakable cipher MATTER. First, we have presented ALU-based, round-based, partially-unrolled and fully-unrolled architectures of the ASCON light-weight permutation which is the most important building block of MATTER. Next, we have proposed various low-area and high-throughput hardware accelerators for different configurations of MATTER. Our implementations are able to support a wide range of throughputs up to 100 GBps suitable for wide-block memory encryption / decryption in secure AI / ML applications. For all the designs of ASCON and MATTER, we have presented detailed analyses of area, power, energy and performance metrics obtained by synthesizing our designs using a 7nm ASIC standard cell library. Our results indicate that MATTER is a promising candidate for securing wide high-performance memories in edge computing systems.

APPENDIX: COMPARISON WITH XTS-AES

As mentioned in Section II, most commercial memory systems use AES [21] in XTS mode [19] for encryption / decryption of large blocks of data using memory addresses as tweaks. The block size of XTS-AES is limited to 128 bits, which is the block size of AES. Figure 10 shows how a 512-bit input can be split into four 128-bit blocks and processed

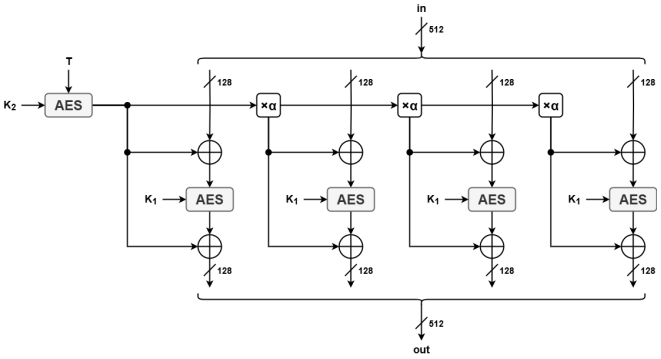


Fig. 10. Processing of 512-bit data as four 128-bit blocks using XTS-AES.

TABLE VII
PERFORMANCE OF XTS-AES-128 ACCELERATORS IN ASAP7 PDK

	Round-Based	Fully-Unrolled + Pipelined
Area (kGE)	14.08	348.04
Maximum Frequency (GHz)	1.37	1.39
Power (mW) at 1 GHz	3.93	168.55
Latency (ns) per 512b Block	47.45	15.83
Throughput (GBps)	1.35	88.96
FoM (GBps / kGE)	0.096	0.256

using XTS-AES to generate the output. Here, T is the 128-bit tweak and $\times\alpha$ denotes multiplication by the element $[x]$ in $\mathbb{F}_2[x]/(x^{128}+x^7+x^2+x+1)$. At the 128-bit security level, we use two 128-bit keys K_1 and K_2 which together form the 256-bit key $K = K_1 \,\|\, K_2$. Overall, this requires 5 invocations of AES-128 (1 for tweak encryption and 4 for encryption / decryption of the 4 data blocks).

Following the same ASIC synthesis methodology from Section II, we implement two XTS-AES-128 accelerators for 512-bit data encryption / decryption – (1) a low-area design with a single instance of round-based AES, and (2) a high-throughput design with five instances of fully-unrolled and pipelined AES. The round-based architecture requires total 65 cycles to process 512-bit data (with 11 cycles per AES-128 invocation). The high-throughput architecture has total 22 pipeline stages (11 for AES-128 tweak encryption followed by 11 for the

4 parallel AES-128 invocations with shared key expansion). Table VII shows that the fully-unrolled pipelined XTS-AES achieves throughput comparable to our high-throughput MATTER accelerators, but has $\approx 2.6\times$ larger area in the same 7nm technology. Therefore, MATTER achieves significantly better FoM in hardware compared to XTS-AES for tweakable encryption of 512-bit blocks at the same 128-bit security level.

ACKNOWLEDGMENT

This work was supported by the Anusandhan National Research Foundation (erstwhile Science and Engineering Research Board), Department of Science and Technology, Government of India under Grant SRG/2023/001633.

REFERENCES

[1] Z. Chang, S. Liu, X. Xiong, Z. Cai, and G. Tu, "A Survey of Recent Advances in Edge-Computing-Powered Artificial Intelligence of Things," *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 13 849–13 875, 2021.

[2] S. Maji, U. Banerjee, and A. P. Chandrakasan, "Leaky Nets: Recovering Embedded Neural Network Models and Inputs Through Simple Power and Timing Side-Channels—Attacks and Defenses," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12 079–12 092, 2021.

[3] C. Qian, M. Zhang, Y. Nie, S. Lu, and H. Cao, "A Survey of Bit-Flip Attacks on Deep Neural Network and Corresponding Defense Methods," *Electronics*, vol. 12, no. 4, p. 853, 2023.

[4] S. Potluri and F. Koushanfar, "SoK: Model Reverse Engineering Threats for Neural Network Hardware," Cryptology ePrint Archive, Paper 2024/913, 2024. [Online]. Available: https://eprint.iacr.org/2024/913

[5] K. Lee, M. Ashok, S. Maji, R. Agrawal, A. Joshi, M. Yan, J. S. Emer, and A. P. Chandrakasan, "Secure Machine Learning Hardware: Challenges and Progress [Feature]," *IEEE Circuits and Systems Magazine*, vol. 25, no. 1, pp. 8–34, 2025.

[6] M. Liskov, R. L. Rivest, and D. Wagner, "Tweakable Block Ciphers," in *Annual International Cryptology Conference (CRYPTO)*, 2002, pp. 31–46.

[7] R. Avanzi, O. Dunkelman, and K. Minematsu, "MATTER: A Wide-Block Tweakable Block Cipher," Cryptology ePrint Archive, Paper 2024/1186, 2024. [Online]. Available: https://eprint.iacr.org/2024/1186

[8] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "Ascon v1.2: Lightweight Authenticated Encryption and Hashing," *Journal of Cryptology*, vol. 34, pp. 1–42, 2021.

[9] M. Sönmez Turan, K. McKay, D. Chang, J. Kang, and J. Kelsey, "Ascon-Based Lightweight Cryptography Standards for Constrained Devices: Authenticated Encryption, Hash, and Extendable Output Functions," National Institute of Standards and Technology, Special Publication 800, NIST SP 800-232 Initial Public Draft, 2024.

[10] H. Groß, E. Wenger, C. Dobraunig, and C. Ehrenhöfer, "Suit up! – Made-to-Measure Hardware Implementations of ASCON," in *Euromicro Conference on Digital System Design (DSD)*, 2015, pp. 645–652.

[11] I. Elsadek, S. Aftabjahani, D. Gardner, E. MacLean, J. R. Wallrabenstein, and E. Y. Tawfik, "Hardware and Energy Efficiency Evaluation of NIST Lightweight Cryptography Standardization Finalists," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2022, pp. 133–137.

[12] A. Koppuravuri, H. Pasupuleti, S. Gvk, and J. Bapat, "A High Throughput ASCON Architecture for Secure Edge IoT Devices," in *International Conference on VLSI Design (VLSID)*, 2024, pp. 486–491.

[13] K.-D. Nguyen, T.-K. Dang, B. Kieu-Do-Nguyen, D.-H. Le, C.-K. Pham, and T.-T. Hoang, "ASIC Implementation of ASCON Lightweight Cryptography for IoT Applications," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 72, no. 1, pp. 278–282, 2025.

[14] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthy, and G. Yeric, "ASAP7: A 7-nm finFET Predictive Process Design Kit," *Microelectronics Journal*, vol. 53, pp. 105–115, 2016.

[15] V. Vashishtha, M. Vangala, and L. T. Clark, "ASAP7 Predictive Design Kit Development and Cell Design Technology Co-Optimization: Invited Paper," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2017, pp. 992–998.

[16] M. Shafique, A. Marchisio, R. V. Wicaksana Putra, and M. A. Hanif, "Towards Energy-Efficient and Secure Edge AI: A Cross-Layer Framework ICCAD Special Session Paper," in *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2021, pp. 1–9.

[17] S. Li, D. Reddy, and B. Jacob, "A Performance & Power Comparison of Modern High-Speed DRAM Architectures," in *Proceedings of the International Symposium on Memory Systems (MEMSYS)*, 2018, p. 341–353.

[18] D. Patel, J. Koch, T. Bennett, W. Chu, and A. Ahmad, "The Memory Wall: Past, Present, and Future of DRAM – Winners & Losers in the 3D DRAM Revolution," SemiAnalysis, 2024. [Online]. Available: https://semianalysis.com/2024/09/03/the-memory-wall

[19] M. Dworkin, "Recommendation for Block Cipher Modes of Operation: the XTS-AES Mode for Confidentiality on Storage Devices," National Institute of Standards and Technology, Special Publication 800, NIST SP 800-38E, 2010.

[20] ——, "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC," National Institute of Standards and Technology, Special Publication 800, NIST SP 800-38D, 2007.

[21] NIST, "Advanced Encryption Standard (AES)," National Institute of Standards and Technology, Federal Information Processing Standards Publication, FIPS 197, 2001.

[22] M. Henson and S. Taylor, "Memory Encryption: A Survey of Existing Techniques," *ACM Computing Surveys*, vol. 46, no. 4, pp. 1–26, 2014.

[23] J. Jean, I. Nikolić, and T. Peyrin, "Tweaks and Keys for Block Ciphers: The TWEAKEY Framework," in *International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT)*, 2014, pp. 274–288.

[24] ——, "Joltik v1. 3," CAESAR Competition Round 2, 2015. [Online]. Available: https://competitions.cr.yp.to/round2/joltikv13.pdf

[25] C. Beierle, J. Jean, S. Kölbl, G. Leander, A. Moradi, T. Peyrin, Y. Sasaki, P. Sasdrich, and S. M. Sim, "The SKINNY Family of Block Ciphers and its Low-Latency Variant MANTIS," in *Annual International Cryptology Conference (CRYPTO)*, 2016, pp. 123–153.

[26] R. Avanzi, "The QARMA Block Cipher Family. Almost MDS Matrices Over Rings With Zero Divisors, Nearly Symmetric Even-Mansour Constructions With Non-Involutory Central Rounds, and Search Heuristics for Low-Latency S-Boxes," *IACR Transactions on Symmetric Cryptology*, vol. 2017, no. 1, pp. 4–44, 2017.

[27] R. Avanzi, S. Banik, O. Dunkelman, M. Eichlseder, S. Ghosh, M. Nageler, and F. Regazzoni, "The QARMAv2 Family of Tweakable Block Ciphers," *IACR Transactions on Symmetric Cryptology*, vol. 2023, no. 3, pp. 25–73, 2023.

[28] C. Beierle, G. Leander, A. Moradi, and S. Rasoolzadeh, "CRAFT: Lightweight Tweakable Block Cipher with Efficient Protection Against DFA Attacks ," *IACR Transactions on Symmetric Cryptology*, vol. 2019, no. 1, pp. 5–45, 2019.

[29] A. Adomnicai, T. P. Berger, C. Clavier, J. Francq, P. Huynh, V. Lallemand, K. Le Gouguec, M. Minier, L. Reynaud, and G. Thomas, "Lilliput-AE: A New Lightweight Tweakable Block Cipher for Authenticated Encryption with Associated Data," NIST Lightweight Cryptography Project, 2019. [Online]. Available: http://rfc.nop.hu/nist/LILLIPUT-AE-spec.pdf

[30] Y. Belkheyar, J. Daemen, C. Dobraunig, S. Ghosh, and S. Rasoolzadeh, "BipBip: A Low-Latency Tweakable Block Cipher with Small Dimensions ," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2023, no. 1, pp. 326–368, 2023.

[31] NIST, "SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions," National Institute of Standards and Technology, Federal Information Processing Standards Publication, FIPS 202, 2015.

[32] ——, "Secure Hash Standard (SHS)," National Institute of Standards and Technology, Federal Information Processing Standards Publication, FIPS 180-4, 2015.

[33] R. Avanzi, I. Mihalcea, D. Schall, H. Montaner, and A. Sandberg, "Hardware-Supported Cryptographic Protection of Random Access Memory," Cryptology ePrint Archive, Paper 2022/1472, 2022. [Online]. Available: https://eprint.iacr.org/2022/1472

[34] J. M. Rabaey, A. P. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*. Prentice-Hall, 2002.

[35] U. Banerjee, A. Wright, C. Juvekar, M. Waller, Arvind, and A. P. Chandrakasan, "An Energy-Efficient Reconfigurable DTLS Cryptographic Engine for Securing Internet-of-Things Applications," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 8, pp. 2339–2352, 2019.