Journal of
Low Power Electronics
Vol. 15, 104–114, 2019

Low Area and Low Power Implementation for Competition for Authenticated Encryption, Security, Applicability, and Robustness Authenticated Ciphers

Amr Abbas[1,*], Hassan Mostafa[2,3], and Ahmed Nader Mohieldin[3]

[1] *IC Verification Solutions, Mentor Graphics, A Siemens Business, Cairo, 11843, Egypt*
[2] *Nanotechnology Program at Zewail City of Science and Technology, Cairo, 14021, Egypt*
[3] *Electronics and Communications Engineering Department, Cairo University, Giza, 12613, Egypt*

(Received: 3 January 2019; Accepted: 10 February 2019)

Authenticated Encryption (AE) and Authenticated Encryption with Associated Data (AEAD) play a significant role in cryptography as they simultaneously provide confidentiality, integrity, and authenticity assurances on the data. The Competition for Authenticated Encryption, Security, Applicability, and Robustness (CAESAR) seeks optimal authenticated ciphers based on multiple criteria, including security, performance, area, and energy-efficiency. Low power consumption is one of the main requirements for any chip design targeting the Internet of Things (IoT) applications. In this research paper, low area and low power implementations of selected ciphers from the CAESAR candidates namely NORX, Tiaoxin, SILC, COLM, and JAMBU are provided and Implemented in both Application Specific Integrated Circuits (ASIC) and Field Programmable Gate Arrays (FPGA). For FPGA Implementations a reduction in area with an average of 32% and a reduction in dynamic power with an average of 56% are achieved compared to their corresponding high-speed architectures. While for ASIC Implementations a reduction in area with an average of 36% and a reduction in dynamic power with an average of 43% are achieved compared to their corresponding high-speed architectures. Moreover, throughput (TP) decreases by an average of 70%.

Keywords: Authenticated Cipher, CAESAR, FPGA, Lightweight, Power, Energy.

1. INTRODUCTION

IoT makes use of data collected from IoT devices to optimize the observation and control of the world in domains such as logistics, retail, military, and healthcare. This huge and continuously increasing number of devices is leading to more attack vectors by hackers. The emergence of the Internet of Things (IoT) applications has made the security issue more critical and complicated As a result, the security becomes one of the main challenges required by IoT stakeholders to deploy the IoT applications in the market.

A decision that system designers face in IoT field is deciding between software-based or hardware-based security solutions. The first solution to show up was software-based security which is relatively inexpensive as it shares resources with other programs to secure the data. The software-based implementation is capable of being revisited and upgraded as threats and vulnerabilities evolve. The software approach is the weak link within systems-security architecture because secrets remain vulnerable to discovery and the algorithms typically run on general-purpose non-secure hardware and are an attack risk.

Hardware security is achieved through a dedicated integrated circuit (IC), or a processor with specialized security hardware, specifically designed to provide cryptographic functions. Security operations, such as encryption/decryption and authentication, take place at the IC hardware level where crypto algorithm performance is optimized.

Lack of sufficient resources in terms of computing ability is one of the characteristics for Majority of the loT devices.[1,2] Also, Form factor and cost play an important role, further limiting the overall capability of the IoT devices. Recent advances in ultra-low-power technology enabled the development of smaller, more mobile, autonomous devices. Examples of this trend are smart cards, Radio Frequency IDentification (RFID), and wearables. The power available to these devices is less than what common battery powered devices consume. Batteries for these devices are tiny and can supply 10 W for

*Author to whom correspondence should be addressed.
Email: amr_abbas@mentor.com

 1546-1998/2019/15/104/011 doi:10.1166/jolpe.2019.1593

only one day. Moreover, some of these technologies collect energy from environmental sources, such as light, heat, noise, or vibration using power scavengers which produce between 1 W and 500 W.

Conventional approaches such as advanced encryption standard (AES), though secure and robust, are not suitable for ensuring the integrity of data traveling among resource-constrained devices.[3] This raised the need for Authenticated ciphers which combine the cryptographic services of confidentiality, integrity, and authentication into one algorithm, they can potentially replace distinct block ciphers and hash functions that are required to work together, which both reduces resources and eliminates potential security vulnerabilities.

The Competition for Authenticated Encryption Security, Applicability, and Robustness (CAESAR), now entering its final stages, evaluates candidates based on several criteria, including performance in hardware, to choose a portfolio of authenticated ciphers that offer advantages over AES-GCM, and are suitable for widespread adoption. The majority of these implementations were optimized for high speed (HS), in that they employed either basic iterative or unrolled architectures, and used full-width datapaths and large I/O bus widths. Such design choices are not surprising, in that HW submissions are historically evaluated based on best throughput-to-area (TP/A) ratios.

The objective of this work is to provide a low area low power optimized implementation for cryptography algorithms to match the power constraints imposed by the low power IoT applications. The addressed algorithms are selected from algorithms that have participated in CAESAR. The selected algorithms are NORX, Tiaoxin, SILC, COLM, and JAMBU. The algorithms are implemented using the Field Programmable Gate Array (FPGA) flow and Application Specific Integrated Circuits (ASIC) flow. The Optimized implementations are benchmarked against the high-speed implementations.[5]

The paper is organized as follows. Section 2 discuss the previous research. Section 3 discusses Authenticated Encryption. Section 4 explains the CAESAR competition background and the hardware Application Programming Interface (API) of the different algorithms used in this paper. Section 5 provides brief descriptions of the selected algorithms and the proposed optimization. Section 6 provides the ASIC and FPGA Implementations. Section 7 provides the results. Section 8 concludes the work of this research.

2. PREVIOUS RESEARCH

Reduction in the area and power of ciphers can be achieved through algorithmic or architectural choices which may result in security and performance losses.[4] Our research use architectural choices to achieve a reduction in the area and power consumption.

There were attempts to provide dedicated lightweight authenticated encryption schemes. An example Hummingbird-2, which required 2.2 kGE in ASIC.[6] Later, AES-Based LW Authenticated Encryption was presented which require an area of 2.5 kGE and use the standard AES cryptographic primitive.[7]

The Competition for Authenticated Encryption, Security, Applicability, and Robustness (CAESAR) intends to select a portfolio of authenticated ciphers that are optimized for certain criteria, including performance in hardware. Certain CAESAR candidates can be realized using low area implementations. An example in Ref. [8] where low area implementation of Ascon is presented which uses 2.57 Kilo-Gate Equivalent (KGE) in 90 nm ASIC technology, however, this version is not compliant with the CAESAR Hardware Application Programming Interface (HW API). In Ref. [9], the authors proposed a low area implementation of AEGIS-128 by sharing of resources which requires 18 KGE. Other LW implementations include an 8-bit ACORN implementation was proposed in Ref. [10], and several versions of NORX are available at Ref. [11].

The majority of HW submissions of CAESAR are implemented using the CAESAR HW Development Package v1.0[12] then a new version of the CAESAR HW Development Package v2.0 supporting lightweight (LW) implementations[13] was released. In Ref. [14] authors present LW implementations of CAESAR candidates Ketje Sr, Ascon-128, and Ascon-128a. They demonstrate that the use of a prototype version of the LW Development Package v2.0 significantly reduces the overhead of interface modules compared to the previous CAESAR HW Development Package v1.0. In Ref. [15] authors improved upon the HS implementations of ACORN, NORX, CLOC, and SILC ciphers by designing true LW implementations. Their design methodology consists of two aspects:

- Use of the LW CAESAR HW Development Package v2.0, with I/O bus widths of 8, 16, or 32 bits.
- Use of internal data paths for cryptographic primitives and authenticated cipher layer operations, which are matched to their corresponding I/O bus widths.

In Ref. [16] we proposed a low area and low power implementations of selected ciphers from the CAESAR candidates namely NORX, Tiaoxin, SILC, and COLM. The optimized implementations were implemented on Virtex-7 FPGA and benchmark against high-speed implementations.[5] This research is an extension to the work done in Ref. [16] where more details of the optimized implementations are provided. Additionally, the optimized implementations are implemented in ASIC flow and benchmarked against high-speed implementations.[5] Moreover a low are and low power implementation is proposed for JAMBU-AES algorithm in ASIC and FPGA flows. For FPGA Implementations a reduction in area with an average of 32% and a reduction in dynamic power with

an average of 56% are achieved compared to their corresponding high-speed architectures. While for ASIC Implementations a reduction in area with an average of 36% and a reduction in dynamic power with an average of 43% are achieved compared to their corresponding high-speed architectures. Moreover, throughput (TP) decreases by an average of 70%.

To the best of our knowledge, the implementations proposed for COLM, JAMBU, and Tiaoxin ciphers are the smallest implementations as the only available implementations are the HS implementations. For SILC and NORX the proposed optimized implementations are compared to work proposed in Ref. [15]. For NORX proposed implementation has lower area reduction with 31% compared to 53.3% in Ref. [15]. For SILC proposed implementation has lower area reduction with 33% compared to 69% in Ref. [15] while proposed implementation has less reduction in throughput-to-area (TP/A) with 25% compared to 65% in Ref. [15]. The Optimization technique proposed in this work which depends on resource sharing is different from that proposed in Ref. [15] which depends on using of reduced internal data path widths, and the LW CAESAR Development Package. The two optimization techniques could be combined together to achieve more area reduction.

3. AUTHENTICATED ENCRYPTION

The need for Authenticated encryption emerged from the observation that securely combining separate confidentiality and authentication block cipher operation modes could be error-prone and difficult. Authenticated encryption was designed as a single primitive that is easy for developers to use. It provides all the necessary cryptographic services of confidentiality, integrity, and authentication.

Authenticated encryption ciphers take a message (M), an associated data (AD), a public message number (Npub), and an optional secret message number (Nsec) as an input and generate resulting ciphertext (C), Tag (Tag) and optional encrypted (Nsec). Integrity of data and authenticity of sender are ensured by a keyed-hash computation which occurs on all blocks of (Npub), (AD) and (M). The result of these computations is forwarded to the recipient as a Tag, as shown in Figure 1. In authenticated decryption, the recipient receives original (AD) and (Npub), along with (C) and (Tag), and uses Key to decrypt (C) to (M).

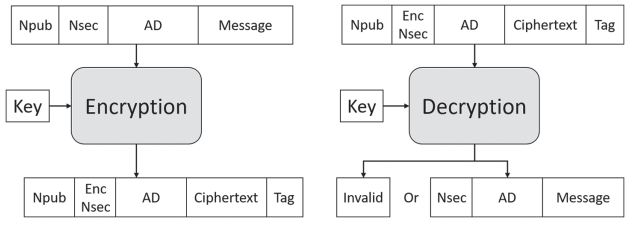

Fig. 1. Input and output of an authenticated cipher.[17]

The authenticated decryption recreates a Tag (Tag′), and releases the ciphertext if and only if Tag = Tag′, then authentication and integrity of the transaction are assured, otherwise the decrypted ciphertext is not released.

4. CASEAR COMPETITION

4.1. Cryptographic Competitions

Many cryptographic competitions are held to gather the cryptanalysts and cipher designers from all over the world to share their knowledge and designs. Following each competition, a final portfolio is announced. These competitions provide a great boost to the cryptographic research community understanding of block ciphers and a tremendous increase in confidence in the security of block ciphers. The first competition was held in 1997 when the United States National Institute of Standards and Technology (NIST) announced an open competition for a new AES. Eventually, NIST selected Rijndael as the standard AES.[18] During the Early Symmetric Crypto workshop in Mondorfles-Bains in 2013, Competition for Authenticated Encryption: Security, Applicability, and Robustness (CAESAR) was announced.[19]

The CAESAR Competition for Authenticated Encryption Security, Applicability, and Robustness was announced in order to encourage the design of AE algorithms. The contest started off with 57 candidates in round 1, then only 29 candidates qualified to round 2, and finally, in round 3, 15 candidates were selected. The Cryptographic Engineering Research Group (CERG) at George Mason University (GMU), USA, runs and maintains the online platform ATHENa[20] aimed at automated evaluation of hardware cryptographic cores targeting Field Programmable Gate Arrays (FPGAs), Systems on Chip, and Application Specific Integrated Circuits (ASICs). One of their on-going projects is the comparison of FPGA implementations of the CAESAR competition candidates. They have also provided high-speed round-based implementations of round 2 and round 3 candidates. The most recent benchmarking results are published in Ref. [21], where the authors provided a summary of available implementations for round 3 candidates that are either designed by the CERG research group or other members of the cryptographic community.

4.2. Hardware Application Programming Interface (API) for Authentication Ciphers

The Hardware Application Programming Interface (API) for authenticated ciphers has been developed to meet all the requirements of all algorithms that have been submitted to the CAESAR competition. The top level of the API is the Authenticated Encryption with Associated Data (AEAD) core. The architecture of the AEAD core consists of three main blocks: pre-processor, cipher core, and post-processor, as shown in Figure 2. The main difference between the different algorithms is in the cipher core

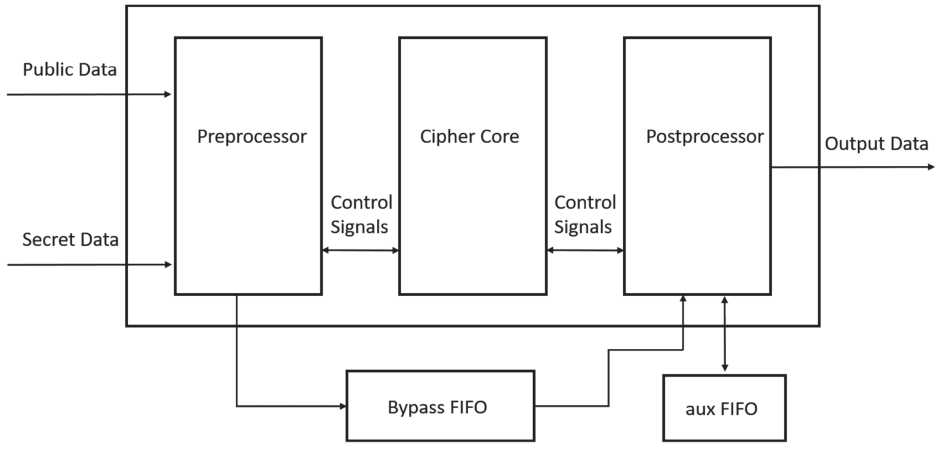

Fig. 2. Hardware application programming interface (API) for authentication ciphers.

implementation, as it contains the hardware blocks that perform either encryption or decryption and authentication algorithm steps. The George Mason University Application Programming Interface (GMU-API)[12] blocks are described as follows:

4.2.1. Pre-Processor

Pre-processor is the first block of the AEAD core which receives public and secret data and start processing them.

4.2.2. Post-Processor

The post-processor is the output stage of the API.

4.2.3. Cipher Core

The cipher core is divided into two blocks: the core data path and the core controller. The core data path contains the hardware which is responsible for encryption or decryption and processing the associative data to perform tag generation, in addition to the hardware which is responsible for the key scheduling and the generation of round keys. The cipher core controller is an algorithmic state machine that takes some information signals from the pre-processor and generates control signals to the core data path.

4.2.4. Bypass First-In-First-Out (FIFO)

Small FIFO which bypasses the tags, header, associated data and any data blocks that are used in the authentication process and will not be encrypted.

4.2.5. Auxiliary FIFO

The memory used by the post-processor to temporarily store the decrypted message till the result of authentication is ready.

4.3. Common Features for CASEAR Candidates

The following selected algorithms: SILC, Tiaoxin, COLM, and JAMBU-AES are based on Advanced Encryption Standard (AES) to perform the encryption and the decryption processes. AES is a symmetric block cipher that uses several key sizes. AES has various standard versions: AES-128, AES-192, and AES-25611. The number of rounds for each version depends on the key size. It uses 10, 12, and 14 rounds for a key size of 128, 192, and 256 respectively. Figure 3 shows a flowchart for the AES encryption algorithm. AES operates on a 4×4 column-major order array of bytes, termed the state and applies four permutation functions in each round which are

- Substitute bytes: Uses an S-box to perform a byte-by-byte substitution of the block.
- ShiftRows: A simple permutation that rotates the state rows right with a different number of positions.
- MixColumns: A substitution that combines the four bytes of each column of the state using an invertible linear transformation.
- AddRoundKey: A simple bitwise XOR of the current block with a portion of the round key.

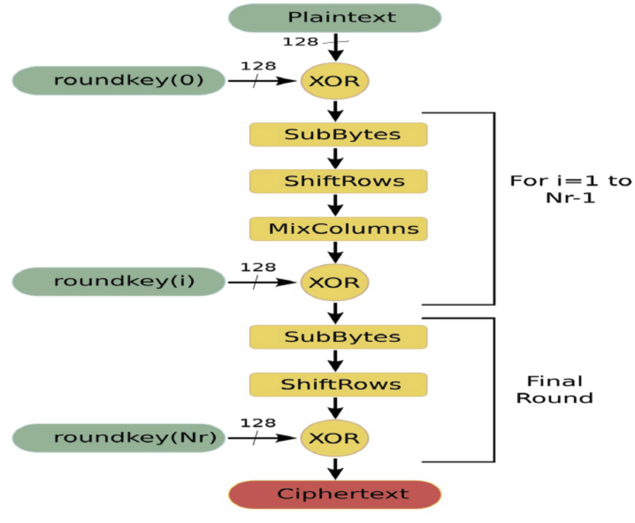

Fig. 3. AES encryption algorithm.

5. LOW POWER AND LOW AREA IMPLEMENTATIONS

5.1. NORX

NORX[22] has a unique parallel architecture based on monkey duplex construction, where the degree of parallelism and tag size can be changed arbitrarily. The scheme is based on Addition-Rotation-XOR (ARX) instead of modular addition. This cipher was optimized to be efficient in both software as well as hardware with a single-instruction multiple-data (SIMD)-friendly core and no secret-dependent memory lockups.

The underlying permutation F is designed by referencing ChaCha stream cipher where the integer addition is replaced by a simple bit-wise XOR operation i.e., $(a \oplus b) \oplus (a \wedge b) << 1$ which leads to improve its hardware efficiency. The permutation function is NORXs core which is applied on NORX internal state S. The state is a concatenation of 16 w-bit words in the form $S = s_0 \| \ldots \| s_{15}$, where the words $s_0, \ldots, s_{11}$ are called the rate words, where data is injected and extracted from, and the remaining words $s_{12}, \ldots, s_{15}$ are called capacity words. Conceptually, the state can be viewed as a 4×4 matrix:

$$S = \begin{bmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{bmatrix}$$

The pseudo code for the NORX core permutation F is given in Figure 4. A single NORX round F processes the state S by first transforming its columns with the function G using function Col(S), and then transforming its diagonals using function Diag(S). The G function uses cyclic rotations and non-linear operation interchangeably to update its four input words.

5.1.1. Optimization

The high-speed NORX hardware implementation (see Fig. 5) duplicates the G function 8 times. The round operation is done in 2 steps, at the first step, 4 G functions operate on the columns, and at the second step, the other 4 G functions operate on the diagonals so the same process is done on columns and diagonals sequentially.

Algorithm: $\mathsf{F}^l(S)$
1. **for** $i \in \{0, \ldots, l-1\}$ **do**
2. $\quad S \leftarrow \mathsf{diag}(\mathsf{col}(S))$
3. **end**
4. **return** S

Algorithm: $\mathsf{G}(a, b, c, d)$
1. $a \leftarrow \mathsf{H}(a, b)$
2. $d \leftarrow (a \oplus d) \ggg r_0$
3. $c \leftarrow \mathsf{H}(c, d)$
4. $b \leftarrow (b \oplus c) \ggg r_1$
5. $a \leftarrow \mathsf{H}(a, b)$
6. $d \leftarrow (a \oplus d) \ggg r_2$
7. $c \leftarrow \mathsf{H}(c, d)$
8. $b \leftarrow (b \oplus c) \ggg r_3$
9. **return** a, b, c, d

Algorithm: $\mathsf{col}(S)$
1. $(s_0, s_4, s_8, s_{12}) \leftarrow \mathsf{G}(s_0, s_4, s_8, s_{12})$
2. $(s_1, s_5, s_9, s_{13}) \leftarrow \mathsf{G}(s_1, s_5, s_9, s_{13})$
3. $(s_2, s_6, s_{10}, s_{14}) \leftarrow \mathsf{G}(s_2, s_6, s_{10}, s_{14})$
4. $(s_3, s_7, s_{11}, s_{15}) \leftarrow \mathsf{G}(s_3, s_7, s_{11}, s_{15})$
5. **return** S

Algorithm: $\mathsf{diag}(S)$
1. $(s_0, s_5, s_{10}, s_{15}) \leftarrow \mathsf{G}(s_0, s_5, s_{10}, s_{15})$
2. $(s_1, s_6, s_{11}, s_{12}) \leftarrow \mathsf{G}(s_1, s_6, s_{11}, s_{12})$
3. $(s_2, s_7, s_8, s_{13}) \leftarrow \mathsf{G}(s_2, s_7, s_8, s_{13})$
4. $(s_3, s_4, s_9, s_{14}) \leftarrow \mathsf{G}(s_3, s_4, s_9, s_{14})$
5. **return** S

Algorithm: $\mathsf{H}(x, y)$
1. **return** $(x \oplus y) \oplus ((x \wedge y) \ll 1)$

Fig. 4. The NORX permutation function.[22]

In order to optimize NORX for low area, only one G function is used so that the Round operation is processed in 8 cycles instead of 1 cycle. In Ref. [16] A register is added which is shifted every clock cycle from the 8 cycles to prepare the data for the G function and another one is added to store the output from the G function. In this research, a mux is added which select the input to G function and the state register is used to store the output of the G function, so no more sequential elements are added to reduce the switching power. A small FSM is added to control the flow of data, and in order to account for the additional delay due to the insertion of the pipe stage, few counters are added. The optimization removes 7 instances of the G function by converting the implementation to be sequential.

5.2. Tiaoxin-346

Tiaoxin-346[23] is a nonce-based authenticated encryption scheme. It is analyzed against various types of attacks. The design decisions (choice of state sizes, output function, etc.) were made in order to make the cipher secure. It provides full security for nonce-respecting adversaries. The internal state consists of 13 words of 16 bytes each. The 13 words are divided into three groups of 3, 4 and 6 words each. The state update function for Tiaoxin-346 absorbs a message block of 32 bytes and produces a new internal state, as illustrated in Figure 6.

The state updated of Tiaoxin is based on a round transformation opearation $R(T_s, M)$ with state T_s and word M as inputs. The output T_s^{new} of the $R(T_s, M)$ is the new state and is given by:

$$T_s^{\text{new}}[0] = AES(T_s[s-1], T_s[0]) \oplus M$$
$$T_s^{\text{new}}[1] = AES(T_s[0], Z_0)$$
$$T_s^{\text{new}}[2] = T_s[1]$$

The Update operation (round function), based on the above $R(T_s, M)$, is used to compute the new value of the states (in the different phases). As inputs, beside the three states, Update takes three additional words M_0, M_1, M_2

$$\text{Update: } T_3 \times T_4 \times T_6 \times M_0 \times M_1 \times M_2 \rightarrow T_3 \times T_4 \times T_6$$

$$T_3^{\text{new}} = R(T_3, M);\ T_3 = T_3^{\text{new}}$$
$$T_4^{\text{new}} = R(T_4, M);\ T_3 = T_4^{\text{new}}$$
$$T_6^{\text{new}} = R(T_6, M);\ T_3 = T_6^{\text{new}}$$

Tiaoxin-346 is a stream cipher based design and as such it works in four phases: Initialization, Processing associated data, Encryption, and Finalization.

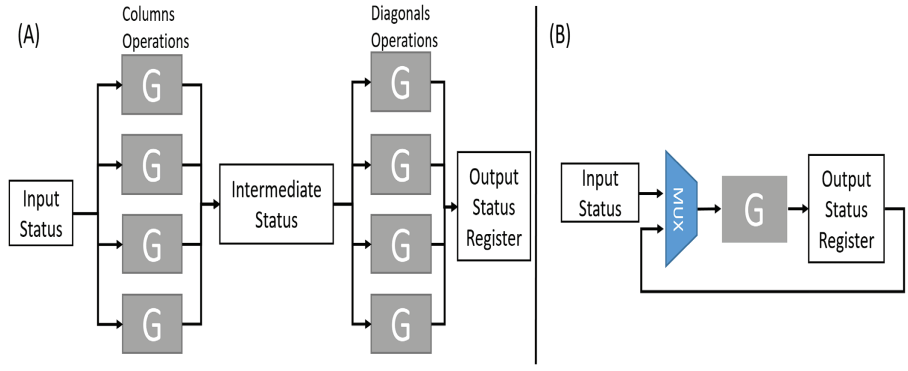

Fig. 5. NORX hardware architecture (A) before optimization (B) after the proposed optimization.

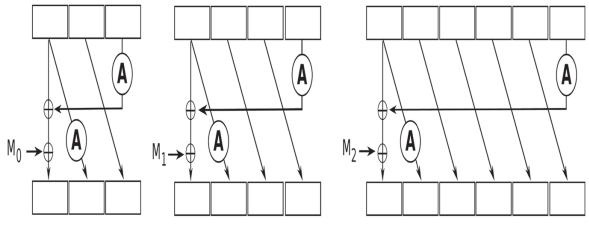

Fig. 6. The round function in Tiaoxin 346. Circled A stands for one AES round.[23]

5.2.1. Optimization

The high-speed Tiaoxin-346 hardware implementation duplicates AES 6 times. In order to optimize Tiaoxin-346 for low area, only one AES is used. The round operation is processed in 6 cycles instead of 1 cycle. A small FSM is added to control the flow of data, Multiplexes are added to control data to the AES, counters are added to account for the additional delay in the round operation.

The block diagram in Figure 7 shows the optimized implementation, as it shown only one AES round is used. The state updated takes 6 clock cycles. A MUX controls the input to AES and select one of T3(2), T3(0), T4(3), T4(0), T6(5) then T6(0) to computes T3(0), T3(1), T4(0), T4(1), T6(0) then T6(1) respectively. The optimization removes 5 instances of AES.

5.3. COLM

COLM[24] is a block cipher based on Encrypt-Linear mix-Encrypt mode, designed with the goal to achieve online misuse resistance, to be fully parallelizable, and to be secure against blockwise adaptive adversaries.

The authenticated encryption for complete message block is shown in Figure 8 where E is an n-bit block cipher, K denotes the key, N the nonce, A associated data, M the message, C the ciphertext, and T the tag. COLM consists of two-layer parallelizable encryption. COLM mixes the output of the first encryption layer to generate the input to the second encryption layer, using linear mixing function.

5.3.1. Optimization

The high-speed COLM implementation instantiates two instances of AES to implement the two layers of encryption. In order to optimize COLM for low area, only one instance of AES is used to perform the two encryption layers. A Finite state machine and Multiplexers are added to

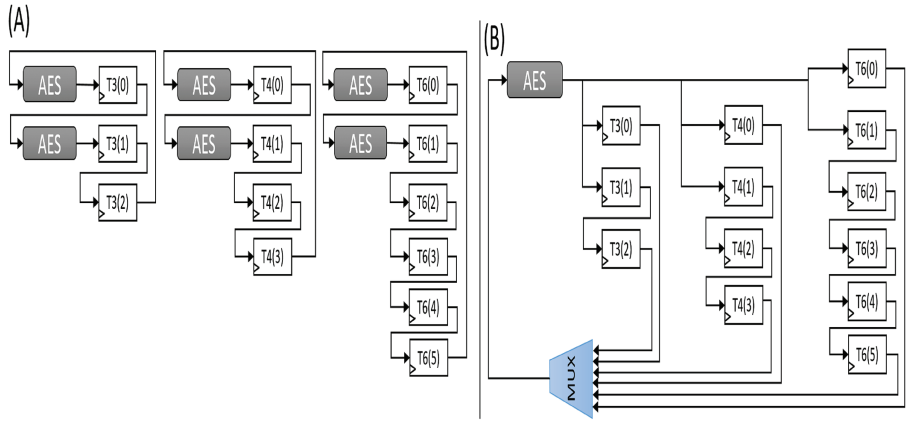

Fig. 7. Tiaoxin-346 hardware architecture (A) before optimization (B) after the proposed optimization.

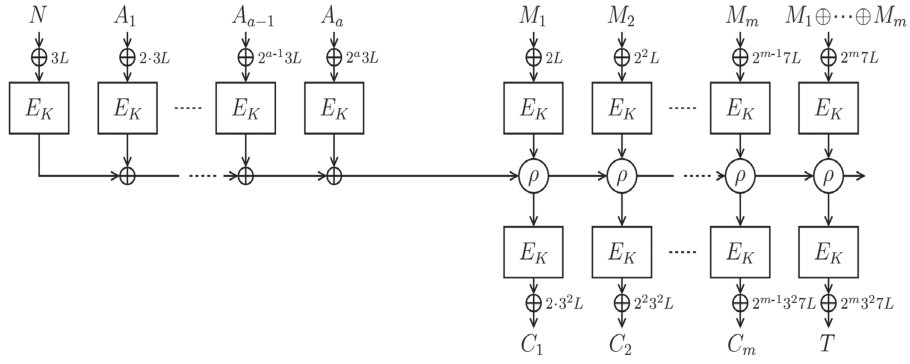

Fig. 8. COLM authenticated encryption for complete message block. E_K denotes the block cipher AES-128.[24]

Algorithm SILC-$\mathcal{E}_K(N,A,M)$	Algorithm SILC-$\mathcal{D}_K(N,A,C,T)$
1. $V \leftarrow \mathsf{HASH}_K(N,A)$	1. $V \leftarrow \mathsf{HASH}_K(N,A)$
2. $C \leftarrow \mathsf{ENC}_K(V,M)$	2. $T^* \leftarrow \mathsf{PRF}_K(V,C)$
3. $T \leftarrow \mathsf{PRF}_K(V,C)$	3. if $T \neq T^*$ then return $\perp$
4. return (C,T)	4. $M \leftarrow \mathsf{DEC}_K(V,C)$
	5. return M

Fig. 9. The encryption and the decryption algorithms of SILC.[25]

control the data flow to the AES. The optimized encryption operation is processed in twice the clock cycles of the non-optimized one and the same applies for the decryption operation.

5.4. SILC

SImple Lightweight CFB (SILC)[25] is an authenticated cipher. SILC uses cipher feed back (CFB) and cipher block chaining (CBC) Message Authentication Code (MAC) modes of operation. CBC-MAC is used for processing associated data and ciphertext, and CFB is used for generating ciphertext. SILC uses AES-128 block cipher which improves latency and memory utilization more than LED and PRESENT block ciphers. The encryption and decryption operations are done using only the encryption function. Both encryption and decryption are online operations which means that every output block depends on all the previous input blocks. The only pre-computation in SILC is the round keys for key scheduling of the block cipher.

The four functions used in SILC are shown in Figure 9, HASH, ENC, DEC, and HASH, are all sequential. However, the block cipher calls in ENC and PRF can be done in parallel. The order of these subroutines is based on whether the operation is encryption or decryption as shown in Figure 9.

5.4.1. Optimization

The high-speed implementation (see Fig. 10) uses two AES cores to achieve the highest possible throughput exploiting the parallelism of ENC and PRF subroutines, which in turn increased the area massively. In order to optimize SILC for low area, the block cipher calls in ENC and PRF are done sequentially (see Fig. 10). Each function of ENC and PRF use One AES core, so calling the ENC and PRF sequentially will require only one AES core to perform both functions. As a result, one round operation is done in 2 cycles instead of 1 cycle.

5.5. JAMBU

JAMBU is a nonce-based authenticated encryption operating mode proposed by Wu and Huang,[26] that can be

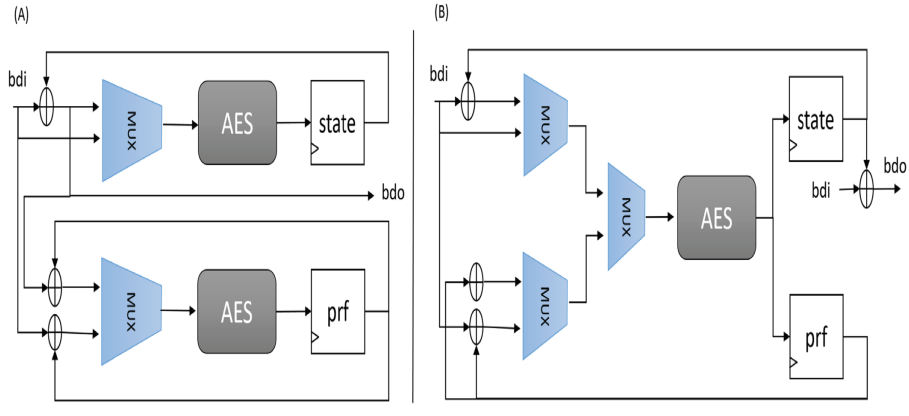

Fig. 10. SILC hardware architecture (A) before optimization (B) after the proposed optimization.

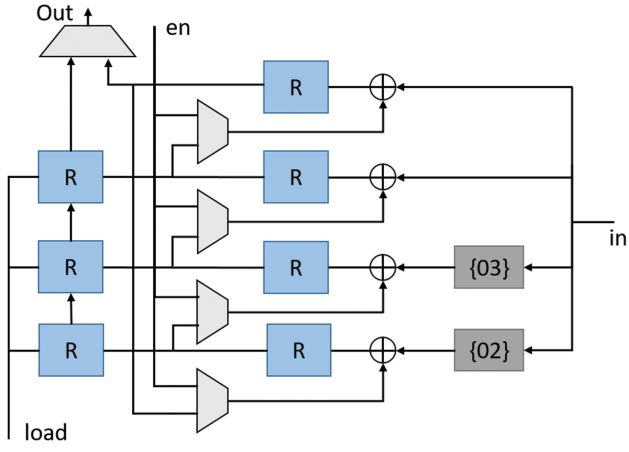

Fig. 11. MixColumn construction.[30]

instantiated with any block cipher. Yet, the submission AES-JAMBU to the CAESAR competition uses AES-128[27] as the internal block cipher. The main advantage of JAMBU mode is its low memory requirement, which places it in the group of lightweight authenticated encryption modes. It is not as fast as the parallelizable schemes such as OCB[28] and OTR,[29] but it is inverse-free, using only XOR operations, and has a lower state size in the cost of a shorter nonce and tag length.[26] In the encryption of JAMBU, the plaintext is divided into blocks of *n*-bit. In each step of the encryption, a plaintext block *Pi* is encrypted to a ciphertext block *Ci*.

5.5.1. Optimization

In order to optimize JAMBU for low area, The AES core adopts an iterative architecture with an 8-bit data path.[30] The high-level architecture of our AES encryption core is shown in Figure 12. The core supports 128-bit keys and computes one round at a time in 5 clock cycles. It consists of 4 Pipelined MixColumns multiplier shown in Figure 11, 5 S-box (4 used in state substitution and one used for the key substitution), and AddRoundKey module.

The State register is shifted every clock cycle from the 5 cycles to process each row of the state matrix iteratively and the data from sbox is stored in a register to be used with the output of Mixcolumns when it is ready after 4 clock cycles then it is introduced to the AddRoundKey module with the round key. A small FSM is added to control the flow of data, and in order to account for the additional delay due to the insertion of the pipe stage, few counters are added. As a result of the proposed optimization, one round operation is done in 5 cycles instead of 1 cycle.

6. ASIC/FPGA IMPLEMENTATIONS

Very high speed integrated circuit Hardware Description Language (VHDL) is the used hardware description language (HDL) to implement algorithms in register transfer level (RTL). The operating frequency is chosen to be 100 MHz for all algorithms. The implementation is done for the CipherCore only, the Postprocessor, preprocessor, and FIFOs are excluded from implementation as the optimization is done for CipherCore only.

6.1. Implementation on Field Programmable Gate Array (FPGA)

The FPGA implementation of the candidates is performed using the Xilinx Vivado 2016.2 design suite. The algorithms are synthesized using the Virtex-7 FPGA device. Vivado tool is used to perform the logic synthesis, mapping, placing, and routing. Vivado results report the area and power consumption of the algorithms. For power consumption the Inputs/Outputs (IOs) power is excluded from the total dynamic power as in real case the CipherCore IOs will be connected to internal signals not primary IOs of the FPGA.

6.2. Implementation on Application Specific Integrated Circuit (ASIC)

Synthesis step is done using Synopsys Design Compiler (DC) B-2008.09 for Linux. CMOS UMC 130 nm technology is the used technology for synthesis and place and

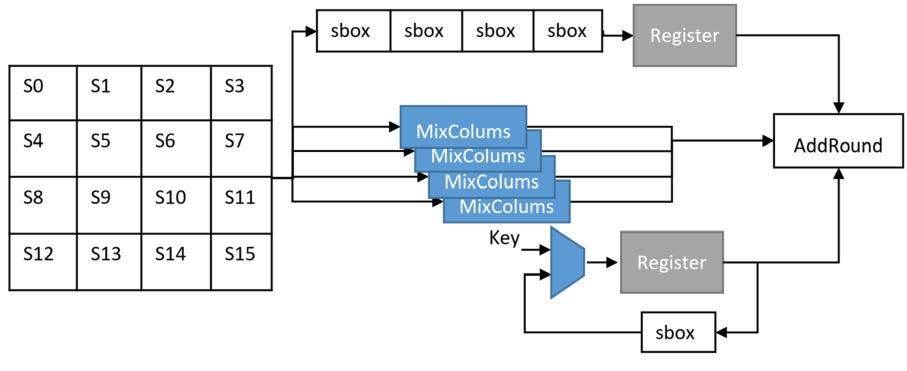

Fig. 12. Optimized AES core.

Table I. Results of implementations of ciphers in virtex-7 FPGA.

Algorithms	Area [slices]	Reduction [%]	Dynamic power [mW]	Reduction [%]	Freq [MHz]	TP [Gb/sec]	Reduction [%]	TP/area [Mbps/slices]	Reduction [%]
High-speed implementations									
NORX	1367	–	416	–	100	19.2	–	14.04	–
Tiaoxin	2030	–	527	–	100	25.6	–	12.6	–
SILC	984	–	230	–	100	1.28	–	1.3	–
COLM	2149	–	149	–	100	1.16	–	0.54	–
JAMBU	511	–	106	–	100	0.64	–	1.25	–
Optimized implementations									
NORX	949	31	127	70	100	2.4	87.5	2.53	82
Tiaoxin	1250	38	183	65	100	4.27	83	3.42	73
SILC	662	33	140	39	100	0.64	50	0.97	25
COLM	1543	28	73	51	100	0.58	50	0.38	30
JAMBU	357	30	46	57	100	0.128	80	0.36	71

Table II. Results of implementations of ciphers in ASIC.

Algorithms	Area [μm^2]	Reduction [%]	Dynamic power [mW]	Reduction [%]	Freq [MHz]	TP [Gb/sec]	Reduction [%]	TP/area [Gbps mm^2]	Reduction [%]
High-speed implementations									
NORX	187266	–	9.47	–	100	19.2	–	102.5	–
Tiaoxin	402603	–	21.84	–	100	25.6	–	63.6	–
SILC	139225	–	11.53	–	100	1.28	–	9.2	–
COLM	529777	–	19.15	–	100	1.16	–	2.2	–
JAMBU	80924	–	4.79	–	100	0.64	–	7.9	–
Optimized implementations									
NORX	103992	44	4.37	54	100	2.4	87.5	23.1	77
Tiaoxin	228271	43	11.68	46	100	4.27	83	18.7	70
SILC	103109	26	7	39	100	0.64	50	6.2	33
COLM	327476	38	12.08	48	100	0.58	50	1.8	19
JAMBU	58073	28	3.39	30	100	0.128	80	2.2	72

route steps. DC takes RTL codes, technology libraries, and constraints file as an input and produced the gate level netlist as an output. The switching activity file generated from modelsim is included for accurate power consumption results.

7. RESULTS

7.1. FPGA Results

Results for benchmarking the proposed optimized implementations (denoted by Optimized Implementations) and the corresponding publicly-available HS implementations[5] (donated by High-Speed Implementations) are shown in Table I. The results show that the proposed optimized implementations achieve an area reduction for NORX, Tiaoxin, SILC, COLM, and JAMBU with 31%, 38%, 33%, 28% and 30% respectively, and a Dynamic Power consumption reduction by 70%, 65%, 39%, 51%, 57% respectively. As a cost, throughput (TP) decreases for NORX, Tiaoxin, SILC, COLM and JAMBU by 87.5%, 83%, 50%, 50%, and 80% respectively, and throughput-to-area (TP/A) decreases by 82%, 73%, 25%, 30% and 71% respectively.

Table III. Comparison of results to work proposed in Ref. [15].

Algorithms	Area reduction [%]	Dynamic power reduction [%]	TP/area change [%]
Work proposed in Ref. [15]			
NORX	53.3	82	+25.5
SILC	69.1	29	−65
Optimized implementation			
NORX	31	70	−82
SILC	33	39	−25

For NORX and SILC the proposed optimized implementations are compared to work proposed in Ref. [15]. In Ref. [15] virtex-6 FPGA is used for implementation, while virtex-7 FPGA is used in this research so a comparison is done between Area reduction, Dynamic Power reduction and throughput-to-area change achieved by proposed work and the work in Ref. [15]. The comparison is summarized in Table III. For NORX proposed implementation has lower area reduction with 31% compared to 53.3% in Ref. [15] and the throughput-to-area (TP/A) has decreased with 82% while it increased with 25.5% in Ref. [15]. For SILC proposed implementation has lower area reduction with 33% compared to 69% in Ref. [15] while proposed implementation has less reduction in throughput-to-area (TP/A) with 25% compared to 65% in Ref. [15].

7.2. ASIC Results

Results for benchmarking the proposed optimized implementations and the corresponding publicly-available HS implementations[5] are shown in Table II. The results show that the proposed optimized implementations achieve an area reduction for NORX, Tiaoxin, SILC, COLM, and JAMBU with 44%, 43%, 26%, 38% and 28% respectively, and a Dynamic Power consumption reduction by 54%, 46%, 39%, 48%, 30% respectively. As a cost, throughput (TP) decreases for NORX, Tiaoxin, SILC, COLM and JAMBU by 87.5%, 83%, 50%, 50%, 48%, and 80% respectively, and throughput-to-area (TP/A) decreases by 77%, 70%, 33%, 19% and 72% respectively.

8. CONCLUSIONS

In this paper, low area and low power implementations for five candidates (NORX, Tiaoxin-346, SILC, COLM, JAMBU) of CAESAR Round 3 are proposed. The optimized implementations and the corresponding high-speed implementations are benchmarked in the Virtex-7 FPGA flow and ASIC flow. For FPGA flow a reduction in area with an average of 32% and a reduction in dynamic power with an average of 56% are achieved compared to their corresponding high-speed architectures. Moreover, throughput (TP) in (Mbps) decreases by an average of 70% and throughput-to-area (TP/A) in (Mbps/Slices) decreases by an average of 56%. For ASIC flow a reduction in area with an average of 36% and a reduction in dynamic power with an average of 43% are achieved compared to their corresponding high-speed architectures. Moreover, throughput (TP) in (Mbps) decreases by an average of 70% and throughput-to-area (TP/A) in (Gbps/mm^2) decreases by an average of 54%. The reduction in TP and TP/A ratio is expected as latency and throughput are sacrificed for area reduction.

References

1. T. Wan, Y. Karimi, M. Stanacevic, and E. Salman, Perspective paper: Can AC computing be an alternative for wirelessly powered loT devices? *IEEE Embedded Systems Letters* 9, 13 **(2017)**.
2. L. Atzori, A. Lera, and G. Morabito, The Internet ofthings: A survey. *Computer Networks* 54, 2787 **(2010)**.
3. D. Bailey, D. Coffin, A. Elbirt, J. Silverman, and A. Woodbury, NTRU in constrained devices. *Workshop on Cryptographic Hardware and Embedded Systems CHES 2001, Volume 2162 of Lecture Notes in Computer Science (LNCS)*, edited by C. Koc, D. Naccache, and C. Paar, Springer-Verlag, Berlin, May **(2001)**.
4. A. Biryukov and L. Perrin, State of the Art in Lightweight Symmetric Cryptography, Cryptology ePrint Archive, Report 2017/511 **(2017)**, https://eprint.iacr.org/2017/511.
5. CERG, Hardware Benchmarking of CAESAR Candidates, August **(2017)**, https://cryptography.gmu.edu/athena/index.php?id=CAESAR.
6. D. Engels, O. Saarinen, P. Schweitzer, and E. Smith, The hummingbird-2 lightweight authenticated encryption algorithm, RFID, *Security and Privacy: 7th International Workshop, RFIDSec'11*, Amherst, Massachusetts, USA, June **(2011)**.
7. A. Bogdanov, F. Mendel, F. Regazzoni, V. Rijmen, and E. Tischhauser, ALE: AES-based lightweight authenticated encryption, *20th International Workshop, FSE 2013*, Singapore, March **(2013)**.
8. H. Gro, E. Wenger, C. Dobraunig, and C. Ehrenhofer, Suit up! made-to-measure hardware implementations of ASCON, *2015 Euromicro Conference on Digital System Design*, Funchal, Portugal, August **(2015)**.
9. D. Bhattacharjee and A. Chattopadhyay, Efficient hardware accelerator for AEGIS-128 authenticated encryption, *10th International Conference, Inscrypt 2014*, Beijing, China, December **(2014)**.
10. T. Huang, Round 3 hardware submission: ACORN. https://groups.google.com/forum/#!forum/crypto-competitions.
11. S. Schelle and M. Tempelmeier. NORX Implementations. https://gitlab.lrz.de/tueisec/crypto-implementations.
12. E. Homsirikamol, W. Diehl, A. Ferozpuri, F. Farahmand, and K. Gaj, Implementers Guide to Hardware Implementations Compliant with the CAESAR Hardware API, v1.0, https://cryptography.gmu.edu/athena/CAESAR_HW_API/CAESAR_HW_Implementers_Guide_v1.0.pdf.
13. E. Homsirikamol, W. Diehl, A. Ferozpuri, F. Farahmand, and K. Gaj, Implementers guide to hardware implementations compliant with the CAESAR hardware API, v2.0, https://cryptography.gmu.edu/athena/CAESAR_HW_API/CAESAR_HW_Implementers_Guide_v2.0.pdf.
14. P. Yalla and J. P. Kaps, Evaluation of the CAESAR hardware api for lightweight implementations, *2017 International Conference on ReConFigurable Computing and FPGAs, ReConFig 2017*, Cancun, Mexico, December **(2017)**.
15. F. Farahmand, W. Diehl, A. Abdulgadir, J. Kaps, and K. Gaj, Improved lightweight implementations of CAESAR authenticated ciphers, *Proceedings of the 26th IEEE International Symposium on Field-Programmable Custom Computing Machines, FCCM 2018*, Boulder, CO, USA, June **(2018)**.
16. A. Abbas, H. Mostafa, and A. Nader Mohieldin, Low area and low power implementation for CAESAR authenticated ciphers, *NGCAS* **(2018)**.
17. E. Homsirikamol, W. Diehl, A. Ferozpuri, F. Farahmand, M. U. Sharif, and K. Gaj, A universal hardware API for authenticated ciphers, *2015 International Conference on Reconfigurable Computing and FPGAs, ReConFig 2015*, Mayan Riviera, Mexico, December **(2015)**.
18. Introduction to Cryptographic Competitions, https://competitions.cr.yp.to/index.html.
19. CAESAR competition for authenticated encryption, http://competitions.cr.yp.to/caesar.html.
20. Automated Tool for Hardware Evaluation (ATHENa), https://cryptography.gmu.edu/athena.
21. E. Homsirikamo, E. Farahmand, F. Diehl, and W. Gaj, Benchmarking of round 3 CAESAR candidates in hardware: Methodology, designs and results, https://cryptography.gmu.edu/athena/presentations/CAESAR_R3_HW_Benchmarking.pdf.
22. J. Aumasson, P. Jovanovic, and S. Neves, NORX, https://competitions.cr.yp.to/round3/norxv30.pdf.
23. I. Nikolic, TIAOXIN, https://competitions.cr.yp.to/round1/tiaoxinv1.pdf.
24. E. Andreevam, A. Bogdanov, N. Datta, A. Luykx, B. Mennink, M. Nandi, and E. Tischhauser, CLOM, https://competitions.cr.yp.to/round2/colm.pdf.
25. T. Iwata, K. Minematsu, J. Guo, S. Morioka, and E. Kobayashi, CLOC and SILC, https://competitions.cr.yp.to/round3/clocsilcv3.pdf.
26. H. Wu and T. Huang, The JAMBU Lightweight Authentication Encryption Mode (v2.1), CAESAR Competition Proposal **(2016)**.
27. J. Daemen and V. Rijmen, The design of rijndael: AES—The advanced encryption standard. Information Security and Cryptography, Springer, New York, United States **(2002)**.

28. P. Rogaway, M. Bellare, and J. Black, OCB: A block-cipher mode of operation for efficient authenticated encryption. *ACM Transactions on Information and System Security (TISSEC)* 6, 365 **(2003)**.
29. K. Minematsu, AES-OTR v3, CAESAR Competion Proposal **(2016)**.
30. P. Hämäläinen, T. Alho, M. Hännikäinen, and T.D. Hämäläinen, Design and implementation of low-area and low-power aes encryption hardware core, *9th EUROMICRO Conference on Digital System Design (DSD'06)* **(2006)**, pp. 577–583.
31. K. Khateb, M. Ahmed, A. K. ELdin, M. AbdelGhany, and H. Mostafa, Dynamically recongurable power efficient security for internet of things devices, *IEEE International Conference on Modern Circuits and Systems Technologies (MOCAST2018)*, Thessaloniki, Greece **(2018)**, pp. 1–4.
32. A. M. Bahnasawi, K. Ibrahim, A. Mohamed, M. Khalifa, A. Moustafa, K. Abelmonim, Y. Ismail, and H. Mostafa, ASIC-oriented comparative review of hardware security algorithms for the internet of things applications, *IEEE International Conference on Microelectronics (ICM 2016)*, Cairo, Egypt **(2016)**, pp. 285–288.
33. N. Samir, A. Hussein, M. Hussein, A. N. El-Zeiny, M. Osama, H. Yassin, A. Abdelbaky, O. Mahmoud, A. Shawky, and H. Mostafa, ASIC and FPGA comparative study for IoT lightweight hardware security algorithms. *Journal of Circuits, Systems and Computers* **(2018)**.

Amr Abbas

Amr Abbas *received the B.Sc. in Electronics Engineering from Cairo University, Cairo, Egypt, in 2012. He has been working as Quality Assurance Engineer in mentor graphics for Digital Design and Verification CAD Tools since 2012. Currently he is a master student in the Electronics and Communications Department, Cairo University.*

Hassan Mostafa

Hassan Mostafa *received the B.Sc. and M.Sc. degrees (with honors) in Electronics Engineering from Cairo University, Cairo, Egypt, in 2001 and 2005, respectively, and the PhD. degree in Electrical and Computer Engineering in the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada in 2011. He is currently an Assistant Professor at the Electronics and Communications Department, Cairo University, Cairo, Egypt and also an Assistant Professor at the Nanotechnology Program at Zewail City of Science and Technology, Cairo, Egypt Dr. Mostafa is collaborating with several sponsors across the world such as Intel (USA), Fujitsu (Japan), National Telecommunications Regulation Agency <NTRA> (Egypt), University of Toronto (Canada), University of Waterloo (Canada), Academy of Scientific Research and Technology <ASRT> (Egypt), Natural Sciences and Engineering Research Council of Canada <NSERC>, Cairo University (Egypt), Blackberry (Canada), Qatar National Research Fund <QNRF> (Qatar), Information Technology Industry Development Agency <ITIDA> (Egypt). He has authored/coauthored over 140 papers in international journals and conferences and is the author/co-author of 5 published books. His research interests include Neuromorphic computing, IoT hardware security, Software defined radio, reconfigurable low power systems, Analog-to-Digital Converters (ADCs), low-power circuits, Subthreshold logic, variation-tolerant design, soft error tolerant design, statistical design methodologies, next generation FPGA, spintronics, Memristors, energy harvesting, MEMS/NEMS, Power management, and Optoelectronics.*

Ahmed Nader Mohieldin

Ahmed Nader Mohieldin *was born in Cairo, Egypt in 1974. He received the B.Sc. and M.Sc. degrees from the Electronics and Communications Department, Cairo University, Egypt, and the Ph.D. degree from Texas A&M University, College Station in 1996, 1998, and 2003, respectively. From 1996 until 1998, he worked as a teaching assistant at the Electronics and Communications Department, Cairo University. He has been a research assistant in the Electrical Engineering Department of Texas A&M University in the Analog and Mixed-Signal Center from 1999 until 2003. In Summer of 2000 he worked in the RFIC Design Group of Texas Instruments in Dallas for his internship. From 2003 until 2008, he has been working as an analog and mixed-signal design engineer with Texas Instruments Inc. at the same group. He has been a technical design manager at SysDsoft Inc., RF/AMS Business Unit from 2008 to 2011. From 2011 to 2013, he has been a project manager at Intel Mobile Communications, Wireless R&D Department. Currently, he is an Associate Professor at Electronics and Communications Department, Cairo University, Egypt. Dr. Mohieldin has published more than 40 journal and conference papers. He has 3 US patents and 1 Egyptian patent issued. His research interests include analog and mixed-signal circuit design, wireless communication, energy harvesting, and mm-Wave for biomedical applications. He is an IEEE Senior Member since 2009.*