

Automatika

Journal for Control, Measurement, Electronics, Computing and Communications

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/taut20

Performance-efficient flexible architecture of m-Crypton cipher for resource-constrained applications

Pulkit Singh, S. V. S. Prasad, Shipra Upadhyay & Rajan Singh

To cite this article: Pulkit Singh, S. V. S. Prasad, Shipra Upadhyay & Rajan Singh (2024) Performance-efficient flexible architecture of m-Crypton cipher for resource-constrained applications, *Automatika*, 65:4, 1447-1457, DOI: [10.1080/00051144.2024.2395617](https://doi.org/10.1080/00051144.2024.2395617)

To link to this article: <https://doi.org/10.1080/00051144.2024.2395617>



© 2024 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.



Published online: 08 Sep 2024.



Submit your article to this journal [↗](#)



Article views: 204



View related articles [↗](#)



View Crossmark data [↗](#)



Performance-efficient flexible architecture of m-Crypton cipher for resource-constrained applications

Pulkit Singh ^a, S. V. S. Prasad ^b, Shipra Upadhyay ^c and Rajan Singh ^b

^aDepartment of Electronics and Communication Engineering, SRM Institute of Science and Technology, Kattankulathur, India;

^bDepartment of Electronics and Communication Engineering, MLR Institute of Technology, Hyderabad, India; ^cDepartment of Electronics and Communication Engineering, Ramaiah Institute of Technology, Bengaluru, India

ABSTRACT

Some traditional cryptographic techniques, like the Secure Hash Algorithm (SHA-256 for hashing), Rivest-Shamir-Adleman (RSA/Elliptic Curve for signing) and Advanced Encryption Standard (AES for encryption), perform well on systems with good hardware memory and processing capabilities. However, these techniques engage in conflict to keep up with the world of sensor networks and embedded systems. Lightweight cryptography plays a major role in security constraints, especially in resource-limited devices such as RFID tags, smart cards, sensor nodes and IoT. This paper proposes a flexible hardware architecture of lightweight m-Crypton block cipher for high-speed resource-constrained applications. The proposed architecture enables a single architecture appropriate for the many encryptions' key sizes. Therefore, the proposed architecture changes the security level in resource-constrained applications by integrating several key sizes into a single design. Furthermore, this architecture outperformed the conventional block ciphers in terms of throughput-to-area ratio achieving a 10.37 throughput-to-area ratio better than other lightweight block ciphers. The proposed design can be used in high bandwidth applications, high-end RFID and IoT smart devices. Hence, the proposed design demonstrates that increasing the speed of cipher implementation results in more plaintext transformations into ciphertext. All results have been verified and simulated for several Xilinx design suite families.

ARTICLE HISTORY

Received 4 January 2024
Accepted 17 August 2024

KEYWORDS

FPGA; hardware implementation; lightweight cryptography; resource-constrained device; throughput


1. Introduction

Nowadays, the usage of RFID tags is aggressively increasing, but there is a need to secure the information on smart cards or tags. Furthermore, the Internet transmission of information is crucial for transactions using cards or tags and protecting such information is essential [1]. Encrypting the data is the best technique for protecting the information. Data can be encrypted in a variety of ways. One of the encryption methods involves incorporating stream and block ciphers. The block ciphers are used for security purposes, but they require high computational power and time. Hence, the use of ciphers in RFID tags is not convenient. Therefore, the researchers came up with the idea of lightweight block ciphers for secure implementation [2,3].

Cryptography plays a significant role in making secure communication. It is used to send information or messages in a secure way to protect it from unauthorized access. Cryptography is the study of statistical procedures inter-related to information security characteristics such as confidentiality, non-repudiation, data integrity and authentication. It involves encryption and decryption to secure the data while transmission through the channels. Ciphers are data encrypting

technique that involves several processes to protect hidden information during data transmission. Using these methodologies, data can be barred from meticulous attacks. Conventional cryptographic algorithms are not suitable for devices with small available computing power. These algorithms such as AES are suitable for securing high-security applications like banking and social networking where a user shares critical information through laptops and desktops [4,5]. On the other hand, resource-constrained devices require lightweightness and moderate security with reduced energy requirements. Moreover, resource-bound applications require less hardware and power requirement. Therefore, many previous methods have attempted to modify the conventional ciphers for resource-constrained applications but those methods have not fitted well to provide multiple security levels in such applications. Lightweight cipher techniques are preferable for low-resource devices because of their limited computing power. Moreover, block cipher techniques play a very important role in the security of such devices [6,7].

Lightweight block ciphers are generally based on substitution, permutation and key addition. These are the cipher's core building blocks and their processing

CONTACT Shipra Upadhyay  shipra@msrit.edu  Department of Electronics and Communication Engineering, Ramaiah Institute of Technology, Bengaluru, Karnataka 560054, India

© 2024 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.

This is an Open Access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. The terms on which this article has been published allow the posting of the Accepted Manuscript in a repository by the author(s) or with their consent.

varies depending on the substitution blocks, permutation method and key scheduling. These blocks provide security with low computational power and less required time for the encryption operation. Different types of lightweight block ciphers have been used for encryption such as PRESENT [8], LED [9], SIMON [10], m-Crypton [11], MIDORI [12], etc. These ciphers take plaintext blocks as inputs and encrypt the ciphertext as an output. Moreover, these ciphers operate on different keys and block sizes; for example, the key sizes of the PRESENT cipher are 64 bits and 128 bits; for the LED cipher, it is 64 bits and 128 bits and for the m-Crypton, it is 64 bits, 96 bits and 128 bits, whereas plaintext block size is 64 bits for all aforementioned ciphers [13].

Lightweight cryptography plays a major role in security constraints, especially in resource-limited devices such as RFID tags, smart cards, sensor nodes and IoT. In lightweight cryptography, there must be a trade-off between complexity and security. Conventional cryptographic algorithms are generally used on desktops, servers, smartphones and tablets, whereas lightweight cryptography algorithms are used in embedded systems, RFID and sensor networks. Keeping these factors in mind, an efficient hardware implementation of cryptographic algorithms is required so that it can use fewer components and take up less space, while meeting timing and power requirements [14].

In short, the major contributions of this paper are as follows:

- Proposed a flexible hardware architecture of the m-Crypton lightweight block cipher available for various key sizes.
- Evaluated the performance of the proposed architecture in hardware metrics such as frequency, area and speed.
- Compared the proposed implementation to state-of-the-art lightweight block ciphers and analysed the results in terms of hardware metrics.

The rest of the paper is organized as follows. Section 2 summarizes the state-of-the-art works of lightweight block ciphers. Section 3 presents a detailed overview of the m-Crypton algorithm. The proposed hardware architecture of m-Crypton lightweight block cipher is designed in Section 4. Implementation results and comparison are discussed in Section 5. Finally, Section 6 summarizes the concluding remarks.

2. Related work

An Advanced Encryption Standard (AES) has garnered more attention to alternative ciphers. AES cipher works on 128-bit word length and key sizes vary from 256, 192 and 128 bits. AES has a Substitution Permutation Network (SPN) and performs operations by

taking input messages or keys in the form of matrices [15]. Different lightweight block ciphers have been designed for resource-constrained applications. Over the years, various hardware implementations of FPGA have been developed using different design methodologies. The m-Crypton algorithm was first proposed in 2005 [11]. This cipher was designed to provide the required security in resource-constrained devices. m-Crypton algorithm has an SPN network with 13 rounds and is suitable for both hardware and software applications. In [16], the authors proposed the FPGA implementation on a Spartan-3 device and managed to store the constant key in RAM rather than key scheduling for efficient implementation. They achieved low hardware implementation but compromised with the latency. In [12], the authors developed a split datapath architecture such as 16-bit and 4-bit datapath of m-Crypton suitable for constrained devices. From these implementations, one can see that there was a slight decrease in area but throughput was reduced tremendously when the design moved from 16-bit datapath to 4-bit datapath. It was due to the increase in the number of control signals whereas 16-bit datapath and 4-bit datapath were the serial implementations of the m-Crypton cipher.

The performance and security of various lightweight encryption schemes other than m-Crypton cipher such as KLEIN, TEA, KATAN HIGHT etc., are used in resource-constrained applications. Those applications have been introduced to assess their memory efficiency, power utilization and performance analysis along with estimated degree of diffusion and confusion for security analysis. For each lightweight block cipher, every bit of data is getting altered by performing substitution cells, mixed columns, add round key and shift row operations. Among the lightweight ciphers, PRESENT cipher is generally used for highly resource-constrained applications. Generally, it operates on a word size of 64 bits and key sizes of 80 bits and 128 bits [17]. Moreover, PRESENT cipher consumed large cycles in the software implementation, so a new algorithm is being proposed called RECTANGLE lightweight block cipher. This algorithm provided very competitive performances in software and it was best suited for many platforms, which had very low areas in hardware but its throughput was not good. It supports word sizes of 64 bits and key sizes of 80 bits and 128 bits [18,19].

There have been other lightweight block ciphers proposed to get high throughput. CLEFIA is one of ciphers with a word size of 128 bits and key lengths of 256, 192 and 128 bits. It is very good in terms of hardware performance and gives better security to the information [20]. CAMELLIA is also one of a symmetric block cipher. It has a word size of 128 bits and key sizes of 256, 128 and 192 bits. CAMELLIA is best suited for smart cards and network systems with ultra-fast speed [21]. TWINE cipher gives good performance on embedded software devices, one more advantage of the TWINE

algorithm is that its hardware occupies very little space with acceptable throughput. There are two types of TWINE cipher. These are TWINE-80 and TWINE-128, here 80 and 128 signify their key sizes but both have a word size of 64 bits. These lightweight ciphers have hardware architectures for different key sizes [22].

A common architecture can be designed for lightweight block ciphers so that an architecture can be utilized for all available key sizes. The authors selected two lightweight cryptographic algorithms: ASCON and ISAP, as the security scheme. They proposed a unified architecture that supported the operation of both ASCON-128 and ISAP-A-128A(Enc), which reduced hardware resources without significant loss of speed through the reuse of structures. In this paper, ASCON and ISAP can be selected to satisfy the speed priority or strengthen the protection of passive side-channel attacks [23]. Moreover, the author proposed a flexible structure that can perform various configurations of CLEFIA cipher to support variable key sizes 128, 192 and 256 bits. This architecture provided a versatile implementation that supported different security levels using a variable key size [24]. In addition, a highly flexible and reconfigurable FPGA hardware accelerator was proposed for efficient inference of various CNNs. Two levels of optimization were performed in the work, (1) resource level: the dataflow and control logic of certain types of layers were merged and reused to reduce the design complexity: (2) performance level: several processing methods were proposed to process different types of convolutions [25]. Furthermore, the authors proposed a serial-based architecture of XXTEA lightweight block cipher supporting variable length block size. The variable length functionality incorporated in a single architecture gave the designer flexibility to work on different input block sizes. The serial implementation focused on achieving area optimization [26]. There are many similarities between AES and SM4 iterative round computation, similar non-linear S-box substitution and the same block width. Because of these similarities, the compact and unified implementation of the two algorithms was possible. The authors designed a unified S-box for AES-128 and SM4 successfully and proposed a kind of reconfigurable S-box logic in the pipeline circuit, which improved the unified coprocessor's working frequency [27].

In this paper, a flexible hardware architecture is designed for a lightweight m-Crypton block cipher and assessed their hardware performance in terms of hardware resources, maximum operating frequency, etc. The target m-Crypton block cipher takes a 64-bit plain block as input and converts it into 64-bit ciphertext using either 64-bit or 96-bit or 128-bit key sizes. There are twelve rounds, each round performs non-linear substitution, bit permutation, column-row transpose and key addition operations. Hence, a totally different way of cryptography, i.e. lightweight is to build

from the bottom level to a new optimized hardware. Moreover, make a slight change in basic architecture so that it can operate for all available key sizes in lightweight block ciphers.

3. Algorithm overview: m-Crypton cipher

m-Crypton is one of the lightweight block ciphers, which takes 64-bit block and variable key size as the inputs and converts them into a ciphertext [11]. There are twelve rounds in the encryption process.

Each round has four types of operations. These are

- Non-linear substitution
- Bit-permutation
- Colum-row transpose
- Key-addition

Now, let us discuss each operation briefly how they work. Basically, while doing these operations, the m-Crypton algorithm mostly uses nibbles in all operations so the first plain message breaks down into 16 nibbles ($h_0, h_1, h_2, \dots, h_{15}$) in the matrix format as given below and performs further operations. Here, column-wise, four nibbles are grouped and represented as in Equation (1):

$$H = \begin{bmatrix} h_0 h_1 h_2 h_3 \\ h_4 h_5 h_6 h_7 \\ h_8 h_9 h_{10} h_{11} \\ h_{12} h_{13} h_{14} h_{15} \end{bmatrix} \begin{bmatrix} H_r[0] \\ H_r[1] \\ H_r[2] \\ H_r[3] \end{bmatrix} \\ = (H_c[0], H_c[1], H_c[2], H_c[3]) \quad (1)$$

where $H_c[0]$ consists of h_0, h_4, h_8, h_{12} .

First, a non-linear substitution operation is performed, which is similar to the substitution in the PRESENT cipher. Here, the m-Crypton cipher uses four substitution boxes instead of one in the PRESENT cipher. The nibble of a plain message is replaced according to the position of the nibble and substitution box. The substitution will be selected according to the position of the nibble in the plain message.

3.1. Non-linear substitution (γ)

The non-linear substitution is performed using the four 4-bit substitution boxes S_i as given in Table 1, in which nibble will be taken as input and substituted with the corresponding nibble in the substitution box, as given in Table 2. Here, S-boxes (S_2 and S_0), (S_3 and S_1) are inverse to each other. $\gamma_i(a)$ is usually done for the i th row or column, i.e. for a 4-nibble word (h_0, h_1, h_2, h_3) as performed in Equation (2).

$$\gamma_i(a) = (S_i(h_0), S_{i+1}(h_1), S_{i+2}(h_2), S_{i+3}(h_3)) \quad (2)$$

If $i > 3$ in S_i , then $i = i \bmod 4$.

Table 1. S-boxes S_i of m-Crypton cipher.

S_i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_0	4	15	3	8	13	10	12	11	5	7	14	2	6	1	9	0
S_1	1	12	7	10	6	13	5	15	11	2	0	8	4	9	14	3
S_2	7	14	12	2	0	9	13	3	15	5	8	6	4	11	1	10
S_3	11	0	10	7	13	6	4	12	14	3	9	1	5	15	8	2

Table 2. Substitution operation in matrix H.

h_0	h_1	h_2	h_3	\rightarrow	$S_0(h_0)$	$S_1(h_1)$	$S_2(h_2)$	$S_3(h_3)$
h_4	h_5	h_6	h_7		$S_1(h_4)$	$S_2(h_5)$	$S_3(h_6)$	$S_0(h_7)$
h_8	h_9	h_{10}	h_{11}		$S_2(h_8)$	$S_3(h_9)$	$S_0(h_{10})$	$S_1(h_{11})$
h_{12}	h_{13}	h_{14}	h_{15}		$S_3(h_{12})$	$S_0(h_{13})$	$S_1(h_{14})$	$S_2(h_{15})$

The transformation will look like this as given in Equation (3):

$$\gamma(H) = (\gamma_0(H_c[0])\gamma_1(H_c[1])\gamma_2(H_c[2])\gamma_3(H_c[3])) \tag{3}$$

3.2. Bit permutation (π)

Bit permutation is performed column-wise in the m-Crypton cipher. In this operation, four masking nibbles are given as $Q_0 = 1110_2, Q_1 = 1101_2, Q_2 = 1011_2$ and $Q_3 = 0111_2$. The column-wise nibbles are selected and the permutation operation is performed, as given in Equations (4) and (5). It is based on the position of nibbles and followed by xored to get one nibble.

$$\pi(H) = (\pi_0(H_c[0])\pi_1(H_c[1])\pi_2(H_c[2])\pi_3(H_c[3])) \tag{4}$$

$$p = \pi i(h) \Leftrightarrow p_j = \bigoplus_{(k=0)}^{(k=3)} (Q_{i+j+k_{\text{mod}4}} \cdot h_k) \tag{5}$$

where $Z \ll^k$: left rotation of a 16-bit word Z by k -bit positions. \bullet, \oplus : bit-wise logical operations for AND and XOR, respectively.

3.3. Column-to-row transposition (τ)

After bit permutation, the 4×4 nibble matrix is transposed by interchanging columns with the rows. The 4 bits at (i,j) th are interchanged with the 4 bits positioned at (j,i) th, as given in Equation (6):

$$P = \tau(H) \Leftrightarrow p_{j,i} = h_{i,j} \tag{6}$$

3.4. Key addition (σ)

$P = \sigma_K(H)$ is defined by $P_r[i] = H_r[i] \oplus K[i]$ ($0 \leq i \leq 3$), where $K = (K[0], K[1], K[2], K[3])$ is a round key. The round key will be generated from the key scheduling algorithm. The round key has a 64-bit size and it will be xored with the output of column-to-row transposition. Then, it completes one round out of a total of twelve rounds.

3.5. Key scheduling

The round key for each round of the encryption process is generated from the key scheduling algorithm. The given master key is updated for each round. The updating process involves substitution and permutation operations. For substitution purposes, the S-box (S_0) is used and masks are different in the key scheduling algorithm compared to the masks used in the round operation.

The key scheduling is performed by dividing the master key into 16-bit words like $K = \{K[i]_{i=0,1,2,3}^{t-1}\} = (K[0], K[1], \dots, K[t-1])$. The number of 16-bit words, i.e. the value of t depends on the master key size. It is 4, 6 and 8 for 64 bits, 96 bits, 128 bits, respectively. The round key generation uses a 16-bit round counter, which updates for each round. The round constant $C[i]$ consists of four identical nibbles, i.e. $C[i] = 0xC_iC_iC_iC_i$. The value of C_i is of 4-bit size, which is generated by x^i in $GF(2^4)$ defined by the irreducible polynomial $f(x) = x^4 + x + 1$, i.e. $C_0 = 1, C_1 = 2, C_2 = 3, C_3 = 8, C_4 = 3, C_5 = 6$, etc.

The key scheduling process for different key sizes is mentioned in this section. The key is stored in the register, which is modified by performing some operations stated below to update the master key. Then, the value of the register is replaced by a new value in each round. The key updating also involves left shifting of words by 3 or 8 bits based on the key sizes. The substitution S-box (S_0) is used for the first 16-bit keyword for generating the round key, i.e. $h = (h_0, h_1, h_2, h_3), S(h) = (S_0(h_0), S_0(h_1), S_0(h_2), S_0(h_3))$. After performing the substitution operation, xored operation is performed with a 16-bit round constant $C[i]$ value, which is used to produce D vectors (intermediate values of the round key). Then, different masks are used to extract the i th nibble to produce D_i . The masks are $Q_0 = 0xf000, Q_1 = 0 \times 0f00, Q_2 = 0 \times 00f0, Q_3 = 0 \times 000f$.

3.5.1. Key scheduling for 64 bits

The initial master key is stored in the V register followed by performing some operations. K_r represents the round key for each round. All the operations from

Equations (7–9) are done by taking the nibbles as input.

$$D \leftarrow S(V[0]) \oplus C[r], D_i \leftarrow D \bullet Q_i (0 \leq i \leq 3) \quad (7)$$

$$K_r \leftarrow (V[1] \oplus D_0, V[2] \oplus D_1, V[3] \oplus D_2, V[0] \oplus D_3) \quad (8)$$

$$V \leftarrow (V[1], V[2], V[3], V[0] \ll 3) \quad (9)$$

Likewise, the round key will be updated for the remaining rounds.

3.5.2. Key scheduling for 96 bits

All the operations from Equations (10–12) are used for 96-bit key scheduling.

$$D \leftarrow S(V[0]) \oplus C[r], D_i \leftarrow D \bullet Q_i (0 \leq i \leq 3) \quad (10)$$

$$K_r \leftarrow (V[1] \oplus D_0, V[2] \oplus D_1, V[3] \oplus D_2, V[4] \oplus D_3) \quad (11)$$

$$V \leftarrow (V[5], V[0] \ll 3, V[1], V[2], V[3] \ll 8, V[4]) \quad (12)$$

3.5.3. Key scheduling for 128 bits

All the operations from Equations (13–15) are used for 96-bit key scheduling.

$$D \leftarrow S(V[0]) \oplus C[r], D_i \leftarrow D \bullet Q_i (0 \leq i \leq 3) \quad (13)$$

$$K_r \leftarrow (V[1] \oplus D_0, V[2] \oplus D_1, V[3] \oplus D_2, V[4] \oplus D_3) \quad (14)$$

$$V \leftarrow (V[5], V[6], V[7], V[0] \ll 3, V[1], V[2], V[3], V[4] \ll 8) \quad (15)$$

Hence, each round operation of m-Crypton cipher applies the γ , π , τ and σ steps in order and is defined for round key K_i by Equation (16):

$$\rho_{K_i} = \sigma_{K_i} \circ \tau \circ \pi \circ \gamma \quad (16)$$

Finally, the encryption transformation E_K of m-Crypton cipher under the secret key K consists of an initial key addition σ_0 and 12 times repetitions of ρ . Then, a final output transformation, i.e. E_K can be defined as Equation (17). Here

$$E_K = \varphi \circ \rho_{K_{12}} \circ \rho_{K_{11}} \circ \dots \circ \rho_{K_2} \circ \rho_{K_1} \circ \rho_{K_0} \quad (17)$$

where $\varphi = \tau \circ \pi \circ \tau$.

4. Proposed flexible implementation

The implementation of the proposed architecture of m-Crypton cipher mainly occurs in the key updating part and round key generation part of the key scheduling block. However, the remaining blocks such as the

non-linear substitution block, bit permutation block and transpose block remain the same irrespective of the key sizes. The flexible structure for the key updating and round key generation can be used for all key sizes, which can be selected by the user among all available key lengths.

Now, let us discuss key updating part. First of all, the given input master key is padded by zeroes to make it a length of 128 bits, if the key length is less than 128 bits. After that, the first word is left shifted by 3 bits and the fourth word is left shifted by 8 bits in key lengths of 96 and 128 bits, respectively. This can be made very easily using multiplexers in the flexible architecture. The input master key is given to the multiplexers according to the available key lengths of 64, 96, and 128 bits. The output of the multiplexers is selected using the selection line provided for different selections of keys as shown in Figure 1.

Moreover, in the key updating block, the selection line inputs SEL(0) and SEL(1) are used to choose the particular bits of 128-bit key. The 64-bit and 96-bit keys are arranged by padding zeroes at the end so that all available keys of m-Crypton cipher can work for 128-bit keys only. It is incorporated by giving the input GND, as shown in Figure 2. The inputs for 64, 96 and 128 bits are connected at S_1 , S_2 , S_3 leaving the S_0 empty. For this, the values of the selection line are 2^1b01 , 2^2b10 and 2^3b11 for 64-bit, 96-bit, and 128-bit key sizes, respectively.

After the completion of the key updating part, the round key generation part is changed in the proposed architecture, which is used for the xored with the input plaintext. The proposed flexible architecture can be made suitable for all the key sizes by incorporating the multiplexers. The multiplexers select the correct 16-bit word in round key generation such as $V[127:112]$ for 64-bit and $V[63:48]$ for 96- and 128-bit key sizes. The output block performs the substitution operation of the first word using S-box(0) and further executes the xored operation with the round counter value generated for the round. It produces a 16-bit M value. The produced M value operates with the masks Q_0 , Q_1 , Q_2 and Q_3 and produces M_0 , M_1 , M_2 and M_3 , respectively. Furthermore, these values perform xored operation with the key stored in the register producing a round key for each round. The value of the round counter is generated using the irreducible polynomial $X^4 + X + 1$. The hardware architecture of the round key generation part is shown in Figure 3.

The operations of M_0 , M_1 , M_2 , M_3 are used for implementation of masks Q_0 , Q_1 , Q_2 and Q_3 , respectively. These are used as masks for producing the V vectors and making the circuits as simple and fast as possible. Figure 4 shows the operation of M_0 , M_1 , M_2 and M_3 to generate the masks Q_0 , Q_1 , Q_2 and Q_3 . The implementation of a round counter remains unchanged for all key sizes. It is the most simplified architecture, which performs only a 1-bit xor operation producing

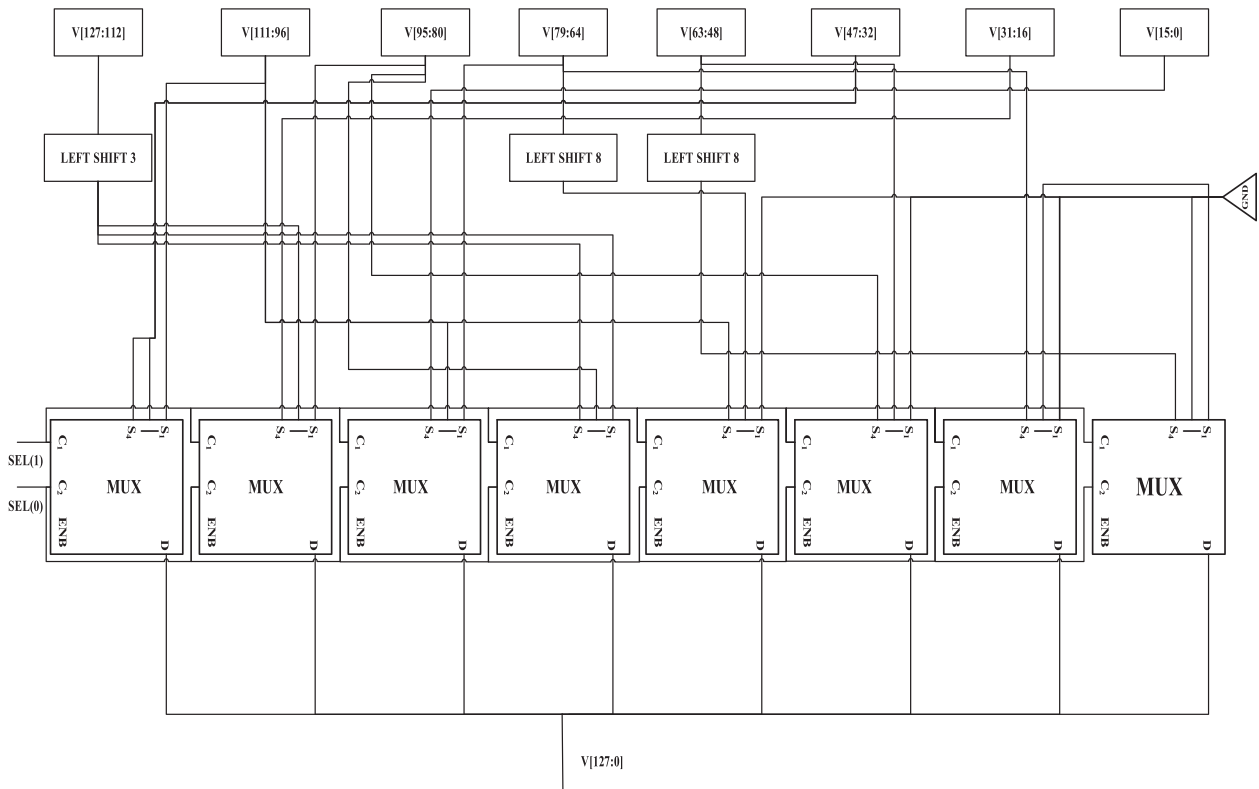


Figure 1. The proposed flexible architecture for key updating block.

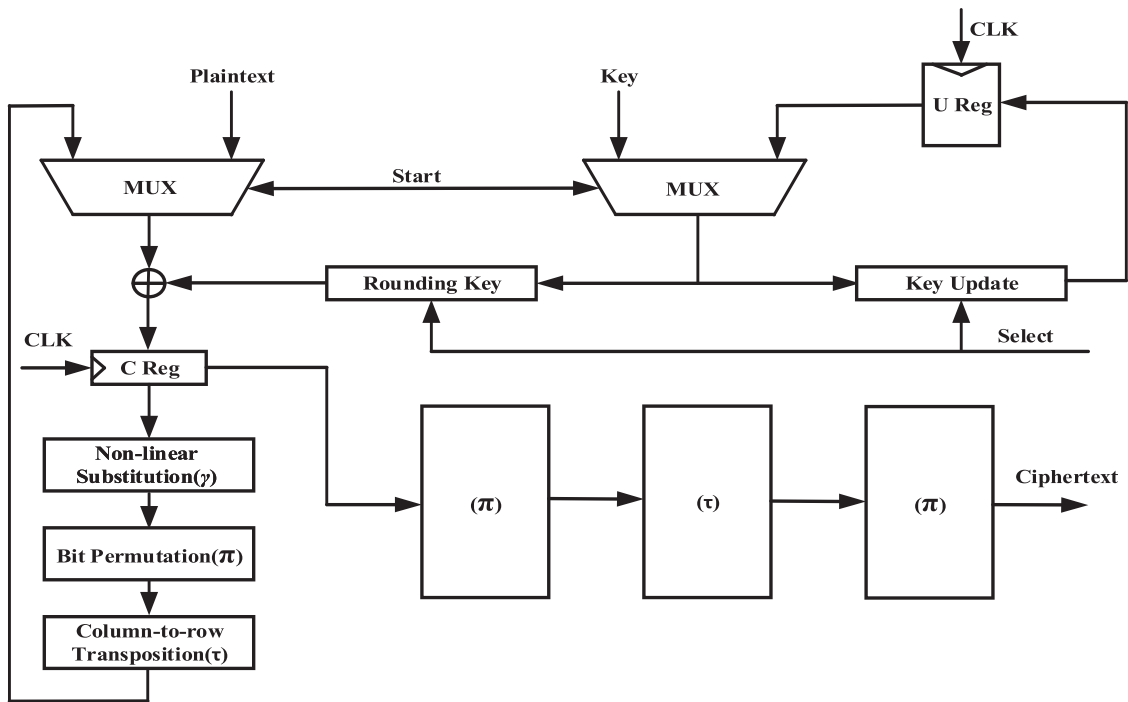


Figure 2. Encryption processes in the proposed flexible implementation of m-Crypton cipher.

4-bit round counter output and can be made 16 bits by repeating four times. The architecture of the generation of the round counter value is followed by the connections, as shown in Figure 5.

The proposed flexible architecture claimed the use of all available key variants for one common design. Furthermore, flexible architecture provided the option to

use a variant of key sizes according to the required multiple security levels. In home appliances IoT applications, security levels differ in accordance with the availability of hardware resources and strength of security. Some appliances like smartphones, computers, smart locks, etc., require a high level of security. Here, the security provided by the 128-bit key is sufficient

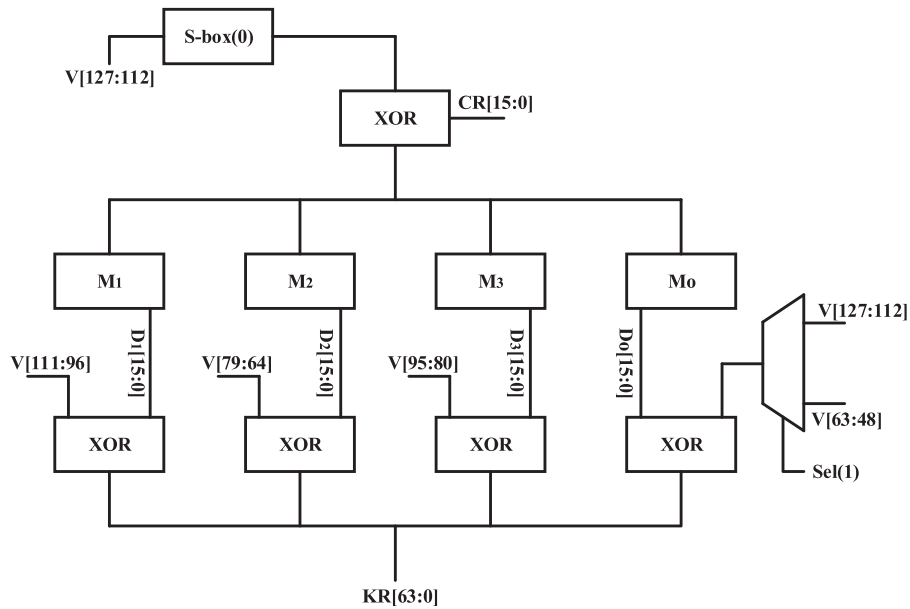


Figure 3. Modified round key generation in the proposed implementation.

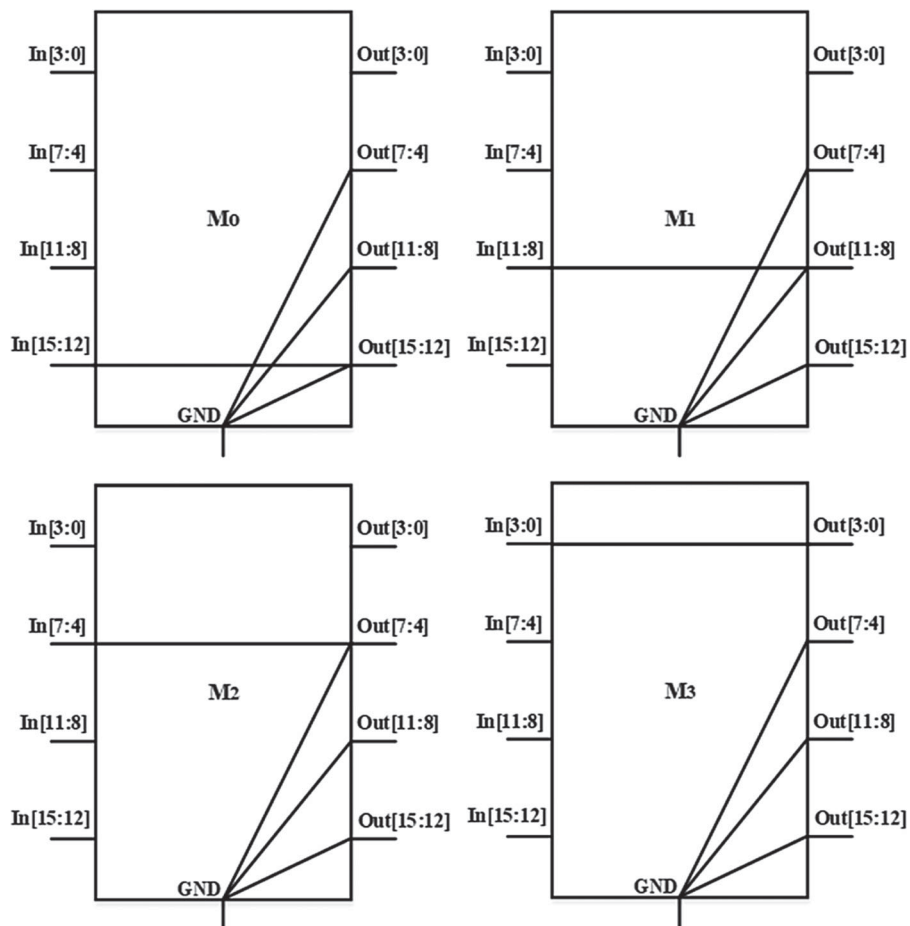


Figure 4. Operations of M_0 , M_1 , M_2 , M_3 in the round key generation.

and appropriate. On the other hand, some systems like healthcare systems, logistics and tracking applications, etc. need a moderate level of security. Security provided by 96-bit key is sufficient and appropriate for those applications. Moreover, some appliances like smart TVs, controlling air conditioners etc., need

a low level of security. Therefore, 64-bit key security is enough and does not have a large number of hardware resources. Hence, devices connected through the internet in IoT applications can be connected with multiple security levels with limited hardware resources.

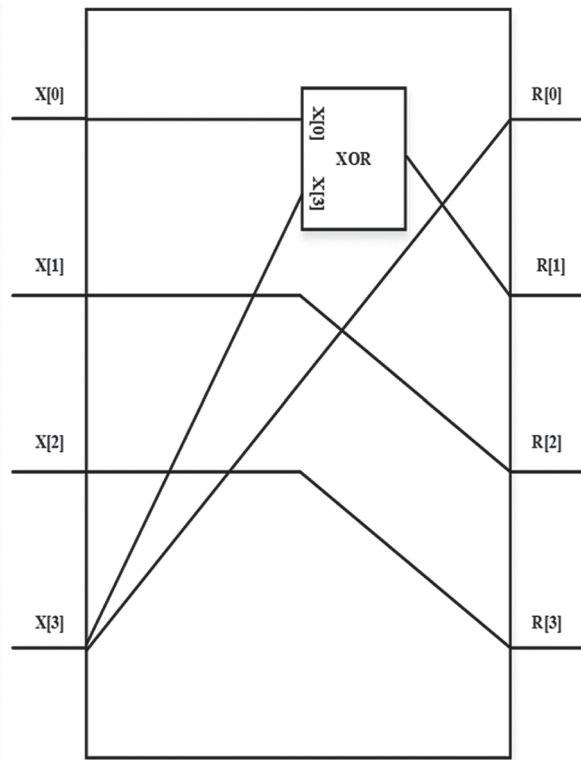


Figure 5. Generation of round counter values.

5. Results and comparison

The proposed flexible architecture of m-Crypton cipher is implemented on different FPGA platforms, such as Spartan-3, Virtex-4 and Virtex-5 devices. The architecture is implemented in such a way that it provides a hardware solution to introduce secure implementation for resource-constrained applications. The proposed hardware architecture is synthesized in *Xilinx* using the *ISE* design suite and simulation is performed using the *ISIM* simulator. Each FPGA family has a different number of LUTs, flip-flops and slices, as well as different maximum frequencies and throughput values. The architecture realized on FPGA is evaluated using various parameters such as area and speed. The speed metric evaluates performance based on frequency and latency. Parameters, such as Slices, LUTs and flip-flops, are used to describe the required hardware area. Moreover, latency is defined as the time duration between inserting the input of plaintext into an algorithm and getting an output of ciphertext. Hence, latency is equal to the number of clock cycles required to reach the first output.

Table 3 shows hardware performance metrics obtained for the proposed work on different FPGA devices. The maximum operating frequency values are high, which suggests high-speed implementation. The results achieved for Virtex-5 and Virtex-4 for maximum operating frequencies 453.23 and 417.31 MHz, respectively, are better than the those obtained on

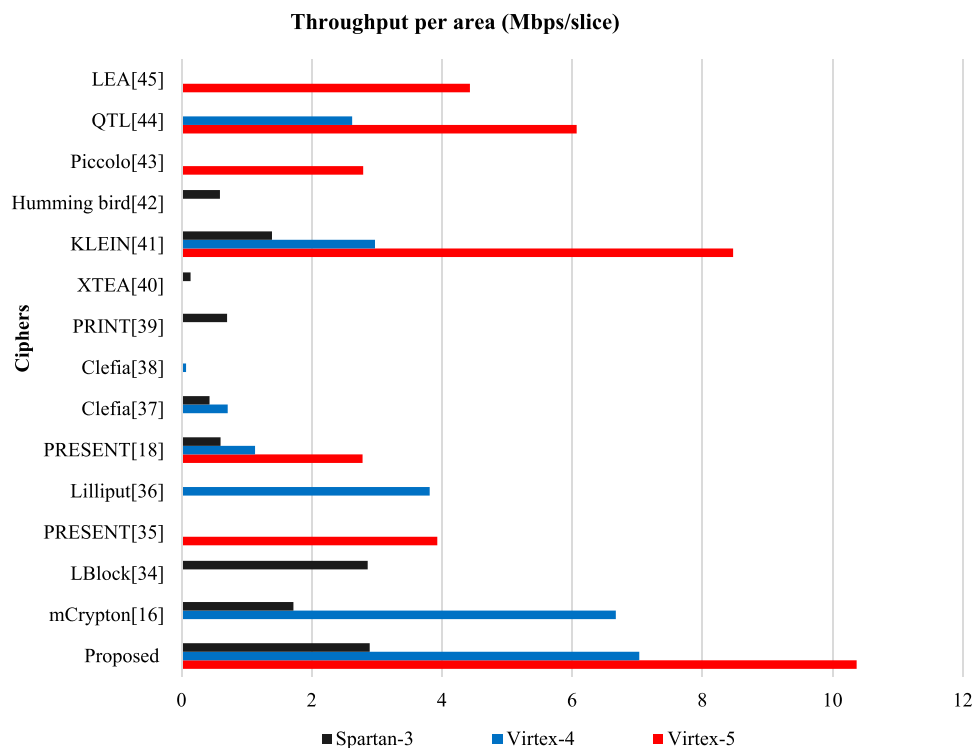
other ciphers for the same FPGA devices. The hardware implementation results of the proposed architecture are compared with the implementation of different lightweight and conventional block ciphers given in Table 3. As can be seen from Table 3 throughput is high for the proposed implementation compared to other state-of-the-art implementations of ciphers such as LBlock [26], XXTEA [26], SIT [28], LED [29], m-Crypton [16], LBlock [33], PRESENT [34], Lilliput [35], PRESENT [17], Clefia [36], Clefia [37], PRINTCIPHER [38], XTEA [39], KLEIN [40], Humming bird [41], Piccolo [42], QTL [43], LEA [44], etc., for the same FPGA devices. Moreover, the proposed implementation has high hardware efficiency in terms of throughput per area (Mbps/slice) compared to other state-of-the-art implementations of ciphers such as m-Crypton [16], LBlock [33], PRESENT [34], PRESENT [17], Clefia [36], Clefia [37], PRINTCIPHER [38], XTEA [39], KLEIN [40], Humming bird [41], Piccolo [42], QTL [43], LEA [44], etc., for the same FPGA devices. Hence, the proposed flexible implementation of m-Crypton lightweight block cipher showed higher speed in terms of throughput (Mbps). Figure 6 shows that the proposed flexible architecture of m-Crypton lightweight block cipher achieved higher efficiency in terms of throughput per area (Mbps/slice) than other lightweight block ciphers in respective FPGA devices.

6. Conclusion

This paper proposed a flexible hardware architecture of a lightweight m-Crypton block cipher for high-speed resource-constrained devices. The proposed flexible architecture has been implemented on different FPGA devices, such as Spartan-3, Virtex-5, Spartan-6, Spartan-7 and Virtex-7. The proposed work is suitable for reprogrammable architecture in FPGAs as this architecture is a highly attractive design option for hardware implementation of encryption algorithms. The proposed flexible architecture claimed the use of all available key variants for one common design. Furthermore, flexible architecture provided the option to use a variant of key sizes according to the required multiple security levels. In home appliances and IoT applications, security levels differ in accordance with the availability of hardware resources and strength of security. Some appliances like smartphones, computers, smart locks, etc., require a high level of security. Here, the security provided by the 128-bit key is sufficient and appropriate. On the other hand, some systems like healthcare systems, logistics, tracking applications, etc., need a moderate level of security. Security provided by 96-bit key is sufficient and appropriate for those applications. Moreover, some appliances like smart TVs, controlling air conditioners, etc., need a low level of security. Therefore, 64-bit key security is enough and does not have a large number of hardware resources.

Table 3. FPGAs implementation results of m-Crypton and other ciphers.

Ciphers	Device	Flip-flop	LUTs	Slices	Max frequency (MHz)	Throughput (Mbps)	Throughput per area (Mbps/slice)
ASCON + ISAP(ENC) [23]	Virtex-7	–	1614	467	367.00	3767.00	8.07
LBlock [26]	Spartan-3	151	223	–	181.76	363.52	–
XXTEA [26]	Spartan-3	161	327	–	97.99	97.99	–
SIT [28]	Spartan-3	83	462	240	62.07	794.53	3.31
LED [29]	Spartan-3	–	328	–	207.59	17.20	–
PRESENT [29]	Spartan-6	–	287	–	245.60	12.76	–
ECC + PRESENT [30]	Virtex-5	–	5016	1663	147.00	–	–
KECCAK-CORE [31]	Cyclone-V	–	–	4639	–	11264	2.43
AESHA [32]	Virtex-6	–	–	3428	328.15	14876.13	4.34
m-Crypton [16]	Spartan-3	–	–	375	302.00	646.00	1.72
LBlock [33]	Virtex-4	149	224	140	466.82	933.64	6.67
LBlock [33]	Spartan-3	149	224	138	197.58	395.17	2.86
PRESENT Iterative [34]	Virtex-5	200	285	87	250.89	341.64	3.93
PRESENT Serial [34]	Virtex-5	203	237	70	245.76	53.32	4.21
Lilliput [35]	Virtex-5	149	230	68	397.60	848.21	12.47
Lilliput [35]	Virtex-4	149	347	178	318.37	679.19	3.81
PRESENT [17]	Virtex-5	201	239	73	431.78	203.19	2.78
PRESENT [17]	Virtex-4	201	265	152	364.56	171.19	1.13
PRESENT [17]	Spartan-3	201	264	151	194.63	91.59	0.60
ClefiA [36]	Virtex-4	–	–	1222	122.59	871.75	0.71
ClefiA [37]	Virtex-4	–	–	14069	146.00	1038.22	0.07
ClefiA [36]	Spartan-3	–	–	1086	65.51	465.85	0.43
PRINTCIPHER [38]	Spartan-3	–	–	210	147.73	147.70	0.70
XTEA [39]	Spartan-3	–	–	254	62.62	35.78	0.14
KLEIN [40]	Virtex-5	141	327	110	407.84	932.19	8.47
KLEIN [40]	Virtex-4	141	459	255	331.13	756.87	2.97
KLEIN [40]	Spartan-3	141	461	263	159.39	364.32	1.39
Humming bird [41]	Spartan-3	120	473	273	40.10	160.40	0.59
Piccolo [42]	Virtex-5	200	604	249	282.24	694.74	2.79
QTL [43]	Virtex-5	72	287	119	180.68	722.72	6.07
QTL [43]	Virtex-4	72	417	214	140.14	560.56	2.62
LEA [44]	Virtex-5	832	715	271	311.00	1206.30	4.43
AES [45]	Virtex-5	286	274	137	77.29	61.83	0.45
Proposed	Virtex-5	196	491	233	453.23	2417.21	10.37
Proposed	Virtex-4	196	751	316	417.31	2225.65	7.03
Proposed	Spartan-3	196	718	354	191.98	1023.89	2.89

**Figure 6.** Throughput per area (Mbps/slice) comparison of the proposed flexible implementation of m-Crypton Cipher.

Hence, devices connected through the Internet in IoT applications can be connected with multiple security levels with limited hardware resources. Moreover, the

proposed architecture achieved a 10.37 throughput-to-area ratio better than other block ciphers. Hence, this proposed design utilized high speed to transform

plaintext into ciphertext. Hence, the proposed design can be used in high-bandwidth applications and high-end RFID smart devices. However, this architecture used extra hardware to select a key size among available key sizes. The hardware performance of the proposed designs can be further enhanced by making appropriate designs with different optimization techniques. The future scope of this work includes possibilities of improvement in FPGA design to make an optimum trade-off among area, power consumption and latency.

Disclosure statement

No potential conflict of interest was reported by the author(s).

ORCID

Pulkit Singh  <http://orcid.org/0000-0001-8196-5768>

S. V. S. Prasad  <http://orcid.org/0000-0003-0940-9243>

Shipra Upadhyay  <http://orcid.org/0000-0002-7078-8716>

Rajan Singh  <http://orcid.org/0000-0001-8510-4113>

References

- [1] Flint D. RFID tags, security and the individual. *Comput Law Secur Rep.* 2006;22(2):165–168. doi:10.1016/j.clsr.2006.01.009
- [2] Hatzivasilis G, Fysarakis K, Papaefstathiou I, et al. A review of lightweight block ciphers. *J Cryptogr Eng.* 2018;8:141–184. doi:10.1007/s13389-017-0160-y
- [3] Singh P, Acharya B, Chaurasiya RK. A comparative survey on lightweight block ciphers for resource constrained applications. *Int J High Perform Syst Archit.* 2019;8(4):250–270. doi:10.1504/IJHPSA.2019.104953
- [4] Elbirt J, Yip W, Chetwynd B, et al. An FPGA-based performance evaluation of the AES block cipher candidate algorithm finalists. *IEEE Trans Very Large Scale Integr VLSI Syst.* 2001;9(4):545–557. doi:10.1109/92.931230
- [5] Bui DH, Puschini D, Bacles-Min S, et al. AES datapath optimization strategies for low-power low-energy multisecurity-level Internet-of-Things applications. *IEEE Trans Very Large Scale Integr VLSI Syst.* 2017; 25(12):3281–3290. doi:10.1109/TVLSI.2017.2716386
- [6] Leander G, Paar C, Poschmann A, et al. New lightweight des variants. *Lect Notes Comput Sci.* 2007;4593: 196–210. doi:10.1007/978-3-540-74619-5_13
- [7] Kong JH, Ang LM, Seng KP. A comprehensive survey of modern symmetric cryptographic solutions for resource constrained environments. *J Netw Comput Appl.* 2015;49:15–50. doi:10.1016/j.jnca.2014.09.006
- [8] Bogdanov A, Knudsen LR, Leander G, et al. Present: an ultra-lightweight block cipher. Berlin: Springer; 2007. p. 450–466. doi:10.1007/978-3-540-74735-2
- [9] Guo J, Peyrin T, Poschmann A, et al. The LED block cipher. In: Preneel B, Takagi T, editors. *International Workshop on Cryptographic Hardware and Embedded Systems.* Berlin: Springer; 2011. p. 326–341. doi:10.1007/978-3-642-23951-9_22
- [10] Beaulieu R, Shors D, Smith J, et al. Simon and speck: block ciphers for the internet of things. In: *Proceedings of the 52nd Annual Design Automation Conference – DAC '15;* 2015. p. 1–6. doi:10.1145/2744769.2747946
- [11] Lim CH, Korkishko T. mCrypton – a lightweight block cipher for security of low-cost RFID tags and sensors. In: Song JS, Kwon T, Yung M, editors. *International Workshop on Information Security Applications.* Berlin: Springer; 2006. p. 243–258. doi:10.1007/11604938_19
- [12] Banik S, Bogdanov A, Isobe Tet al. Midori: a block cipher for low energy. In: *International Conference on the Theory and Application of Cryptology and Information Security.* Springer; 2015. p. 411–436. doi:10.1007/978-3-662-48800-3_17
- [13] Rajesh S, Paul V, Menon VG, et al. A secure and efficient lightweight symmetric encryption scheme for transfer of text files between embedded IoT devices. *Symmetry (Basel).* 2019;11(2):293. doi:10.3390/sym11020293
- [14] Singh P, Acharya B, Chaurasiya RK. Lightweight cryptographic algorithms for resource-constrained IoT devices and sensor networks. In: Sharma SK, Debnath NC, Bhushan B, editors. *Security and privacy issues in IoT devices and sensor networks.* Academic Press; 2021. p. 153–185. doi:10.1016/B978-0-12-821255-4.00008-0
- [15] Daemen J, Rijmen V, Leuven KU. AES proposal: Rijndael complexity; 1999. p. 1–45. Available from: <http://ftp.csci.csusb.edu/ykarant/courses/w2005/csci531/papers/Rijndael.pdf>
- [16] Abbas YA, Hameed AS, Alwan SH, et al. Efficient hardware implementation for lightweight mCrypton algorithm using FPGA. *Indones J Electr Eng Comput Sci.* 2021;23(3):1674–1680. doi:10.11591/ijeecs.v23.i3.pp1674-1680
- [17] Lara-Nino CA, Diaz-Perez A, Morales-Sandoval M. Lightweight hardware architectures for the present cipher in FPGA. *IEEE Trans Circuits Syst I Regul Pap.* 2017;64(9):2544–2555. doi:10.1109/TCSI.2017.2686783
- [18] Zhang W, Bao Z, Lin D, et al. Rectangle: a bit-slice lightweight block cipher suitable for multiple platforms. *Sci China Inf Sci.* 2015;58(12):1–15. doi:10.1007/s11432-015-5459-7
- [19] Upadhyay S, Singh P, Pandey AK, et al. Comparative performance analysis of present lightweight cipher for security applications in extremely constrained environment. In: Jain S, Marriwala N, Singh P, et al., editors. *International Conference on Emergent Converging Technologies and Biomedical Systems.* Berlin: Springer; 2023. p. 511–521. doi:10.1007/978-981-99-8646-0_40
- [20] Shirai T, Shibutani K, Akishita T, et al. The 128-bit blockcipher CLEFIA (extended abstract). In: Biryukov A, editor. *Fast software encryption. FSE 2007. Lecture Notes in Computer Science, Vol. 4593.* Berlin: Springer. 2007. p. 181–195. doi:10.1007/978-3-540-74619-5_12
- [21] Aoki K, Ichikawa T, Kanda M, et al. *Camellia*: a 128-bit block cipher suitable for multiple platforms – design and analysis. In: Stinson DR, Tavares S, editors. *Selected areas in cryptography. SAC 2000. Lecture Notes in Computer Science, Vol. 2012.* Berlin: Springer. 2001. p. 39–56. doi:10.1007/3-540-44983-3_4
- [22] Suzaki T, Minematsu K, Morioka S, et al. Twine: a lightweight block cipher. *Lect Notes Comput Sci.* 2013;7707:339–354. doi:10.1007/978-3-642-35999-6_22
- [23] Guo C, Wang Y, Chen F, et al. Unified lightweight authenticated encryption for resource-constrained electronic control unit. In: *29th IEEE International Conference on Electronics, Circuits and Systems (ICECS).*

- IEEE; 2022. p. 1–4. doi:10.1109/icecs202256217.2022.9971118
- [24] Rashidi B. Efficient and flexible hardware structures of the 128 bit CLEFIA block cipher. *IET Comput Digit Tec.* 2020;14(2):69–79. doi:10.1049/iet-cdt.2019.0157
- [25] Wu X, Ma Y, Wang M, et al. A flexible and efficient FPGA accelerator for various large-scale and lightweight CNNs. *IEEE Trans Circuits Syst I: Regul Pap.* 2021;69(3):1185–1198. doi:10.1109/TCSI.2021.3131581
- [26] Kamble A, Mishra Z, Acharya B. Hardware implementations of LBlock and XXTEA lightweight block ciphers for resource-constrained IoT application. *Int J High Perform Syst Archit.* 2023;11(3):169–178. doi:10.1504/IJHPSA.2023.130223
- [27] Xu Y, Deng F, Xu W, et al. Unified coprocessor for high-speed AES-128 and SM4 encryption. In: 2022 IEEE 6th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC). IEEE; 2022. p. 640–644. doi:10.1109/iaeac54830.2022.9929737
- [28] Mishra Z, Acharya B. High throughput and low area architectures of secure IoT algorithm for medical image encryption. *J Inf Secur Appl.* 2020;53:102533. doi:10.1016/j.jisa.2020.102533
- [29] Rashidi B. Flexible structures of lightweight block ciphers PRESENT, SIMON and LED. *IET Circuits Devices Syst.* 2020;14(3):369–380. doi:10.1049/iet-cds.2019.0363
- [30] Rashid M, Sonbul OS, Arif M, et al. A flexible architecture for cryptographic applications: ECC and PRESENT. *Comput Mater Contin.* 2023;76:1009–1025. doi:10.32604/cmc.2023.039901
- [31] Maache A, Kalache A. Design and implementation of a flexible multi-purpose cryptographic system on low cost FPGA. *Int J Electr Comput Eng Syst.* 2023;14(1):45–58. doi:10.32985/ijeces.14.1.6
- [32] Khalid A, Aziz A, Wang C, et al. Resource-shared crypto-coprocessor of AES Enc/Dec with SHA-3. *IEEE Trans Circuits Syst I: Regul Pap.* 2020;67(12):4869–4882. doi:10.1109/TCSI.2020.2997916
- [33] Singh P, Acharya B, Kumar R. Low-area and high-speed hardware architectures of LBlock cipher for Internet of Things image encryption. *J Electron Imaging.* 2022;31(3):1–29. doi:10.1117/1.JEI.31.3.033012
- [34] Hanley N, O'Neill M. Hardware comparison of the ISO/IEC 29192-2 block ciphers. In: 2012 IEEE Computer Society Annual Symposium on VLSI. IEEE; 2012. p. 57–62. doi:10.1109/ISVLSI.2012.25
- [35] Singh P, Acharya B, Chaurasiya RK. Efficient VLSI architectures of LILLIPUT block cipher for resource-constrained RFID devices. In: 2019 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT). IEEE; 2019. p. 1–6. doi:10.1109/CONECCT47791.2019.9012869
- [36] Singh P, Patro KAK, Chaurasiya RK, et al. Hardware-software co-design framework of lightweight CLEFIA cipher for IoT image encryption. *Sādhanā.* 2022;47(213):1–7. doi:10.1007/s12046-022-01994-0
- [37] Kryjak T, Gorgon M. Pipeline implementation of the 128-bit block cipher CLEFIA in FPGA. In: 2009 International Conference on Field Programmable Logic and Applications. IEEE; 2009. p. 373–378. doi:10.1109/fpl.2009.5272264
- [38] Okabe T. Efficient FPGA implementations of PRINTCIPHER. *Int J Emerg Technol Innov Res.* 2016;3(4):76–85. Available from <http://www.jetir.org/papers/JETIR1604017.pdf>
- [39] Kaps J-P. Chai-tea, cryptographic hardware implementations of xTEA. In: Chowdhury DR, Rijmen V, Das A, editors. International Conference on Cryptology in India. Springer; 2008. p. 363–375. doi:10.1007/978-3-540-89754-5_28
- [40] Singh P, Acharya B, Chaurasiya RK. High throughput architecture for KLEIN block cipher in FPGA. In: 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON). IEEE; 2019. p. 64–69. doi:10.1109/IEMECONX.2019.8877021
- [41] Fan X, Gong G, Lauffenburger K, et al. FPGA implementations of the hummingbird cryptographic algorithm. In: 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST). IEEE; 2010. p. 48–51. doi:10.1109/hst.2010.5513116
- [42] Ramu G, Mishra Z, Singh P, et al. Performance optimised architectures of Piccolo block cipher for low resource IoT applications. *Int J High Perform Syst Archit* 2020;9(1):49–57. doi:10.1504/IJHPSA.2020.107175
- [43] Shrivastava N, Singh P, Acharya B. Efficient hardware implementations of QTL cipher for RFID applications. *Int J High Perform Syst Archit* 2020;9(1):1–10. doi:10.1504/IJHPSA.2020.107173
- [44] Mishra Z, Nath PK, Acharya B. High throughput unified architecture of LEA algorithm for image encryption. *Microprocess Microsyst* 2020;78:103214. doi:10.1016/j.micpro.2020.103214
- [45] Canright D. A very compact S-box for AES. *Lect Notes Comput Sci* 2005;3659:441–455. doi:10.1007/11545262_32