Computers, Materials & Continua
DOI:10.32604/cmc.2021.015519
Article

Tech Science Press

LBC-IoT: Lightweight Block Cipher for IoT Constraint Devices

Rabie A. Ramadan[1,2,*], Bassam W. Aboshosha[3], Kusum Yadav[1], Ibrahim M. Alseadoon[1], Munawar J. Kashout[1] and Mohamed Elhoseny[4,5]

[1]College of Computer Science and Engineering, University of Ha'il, Ha'il, 81481, Saudi Arabia
[2]Department of Computer Engineering, Faculty of Engineering, Cairo University, Cairo, 12613, Egypt
[3]Department of Computer Engineering, Higher Institute of Engineering, Elshorouk Academy, Cairo, Egypt
[4]Faculty of Computers and Information, Mansoura University, Egypt
[5]College of Computer Information Technology, American University in the Emirates, United Arab Emirates
[*]Corresponding Author: Rabie A. Ramadan. Email: rabie@rabieramadan.org
Received: 25 November 2020; Accepted: 15 January 2021

Abstract: With the new era of the Internet of Things (IoT) technology, many devices with limited resources are utilized. Those devices are susceptible to a significant number of new malware and other risks emerging rapidly. One of the most appropriate methods for securing those IoT applications is cryptographic algorithms, as cryptography masks information by eliminating the risk of collecting any meaningful information patterns. This ensures that all data communications are private, accurate, authenticated, authorized, or non-repudiated. Since conventional cryptographic algorithms have been developed specifically for devices with limited resources; however, it turns out that such algorithms are not ideal for IoT restricted devices with their current configuration. Therefore, lightweight block ciphers are gaining popularity to meet the requirements of low-power and constrained devices. A new ultra-lightweight secret-key block-enciphering algorithm named "LBC-IoT" is proposed in this paper. The proposed block length is 32-bit supporting key lengths of 80-bit, and it is mainly based on the Feistel structure. Energy-efficient cryptographic features in "LBC-IoT" include the use of simple functions (shift, XOR) and small rigid substitution boxes (4-bit-S-boxes). Besides, it is immune to different types of attacks such as linear, differential, and side-channel as well as flexible in terms of implementation. Moreover, LBC-IoT achieves reasonable performance in both hardware and software compared to other recent algorithms. LBC-IoT's hardware implementation results are very promising (smallest ever area "548" GE) and competitive with today's leading lightweight ciphers. LBC-IoT is also ideally suited for ultra-restricted devices such as RFID tags.

Keywords: Security; internet of things; cryptographic algorithms; block cipher; lightweight algorithms

1 Introduction

Secure and reliable cyberspace is one of the most critical issues facing humanity. The fragility and insecurity in cyberspace have exposed businesses and individuals to unexpected and dangerous attacks. Achieving safe cyberspace requires a careful balance between technologies and social needs to resolve significant scientific obstacles and achieve safety and trust in cyberspace. New advances in cyberspace technologies, social change, and new spaces would also require reevaluating privacy, security, and cyberspace trust relationships. Cybersecurity is also one of the hot topics of today's research. It refers to the array of instruments, devices, procedures, security principles, protection safeguards, guidelines, risk management techniques, programs, planning, best practice, surveillance, processes, systems, and cyberattack controls. At the same time, the corporate and customer properties, including devices, staff, services, networks, technology, applications, utilities, telecommunications systems, and now connected through the Internet, and their information is shared and/or stored in the virtual world. Moreover, cybersecurity is a critical issue when political, military, private, financial, and medical institutions collect and store their data on computers and other devices. Consequently, sensitive information, like financial and personal data, or some other form of information for which improper entry or distribution could have adverse consequences on a large portion of that information [1,2].

In modern digital communication technologies, cryptography has become the primary method for maintaining the necessary digital security. It guarantees the core protection components such as authorization, authentication, confidentiality, non-repudiation, and integrity to all cyberspace data exchanges. In other words, the data produced by countless tiny networked devices such as the Internet of Things (IoT) would need a new class of cryptographic protections against cyber-attacks [3].

Many modern cryptography algorithms were applied to resource-restricted devices; however, the results were not acceptable. The transition from desktop to small and tiny computers raises a variety of security problems and privacy concerns. It is still a challenge to implement desktop cryptographic algorithms on resource-limited devices where reliability and efficiency are still important for their applications. Lightweight encryption is a subfield of cryptography that is developed especially for resource-limited devices [4].

This paper proposes a new ultra-lightweight cryptographic algorithm for IoT applications, named LBC-IoT. The algorithm differs from our previous work [5] in its structure and design, where the round function, permutation shift, and key generation function are different. Besides, the evaluation methodologies are also changed where it is designed for very limited devices in terms of their footprint. The algorithm satisfies constrained resource devices' requirements in terms of its footprint and simplicity; it is a rigid algorithm against most of the recent attacks. The algorithm performance is compared to many of the current similar security algorithms and proves its efficiency.

The paper begins with a summary of the literatures in Section 2, while the LBC-IoT architecture and its functionalities are discussed in Section 3. The implementation description is discussed in Section 4, while the output analysis is provided in Section 5. The paper conclusion is depicted in Section 6.

2 Literature Review

Many Lightweight Block Cipher systems were designed in the last two decades to achieve performance advantages over NIST's Advanced Encryption Standard (AES) [6], particularly AES-128. The algorithms presented in the literature can be classified into two classes:

- Adaptive algorithms: They are based on a modified version of the well-investigated and trusted ciphers.
- Progressive algorithms: this means that new ciphers are designed to have low hardware implementation costs.

Adaptive algorithms are one type of algorithms that could be suitable for limited-resource devices. Poschmann et al. [7–10] laid the foundations of this class, and his co-authors have presented several valuable contributions in this area. Lightweight Data Encryption Standard (DESL) could be the best example of such algorithms. It is an improved version of the DES algorithm with smart implementation. It uses a single S-Box, where DES uses eight S-boxes. Such smart S-Box design makes DESL resistant to many of the cryptanalytic attacks, including linear and differential attacks. This also allows saving a part of ROM for tables' storage, where almost more than 85% of memory space occupied by traditional DES has been saved [9]. Another example is revisited GOST, where the GOST standard does not specify a set of S-boxes [10]. Poschmann et al. [8] have exploited the idea that the standard does neither specify if the S-boxes used shall be different or not. Thus, with a small area footprint in mind, they proposed an efficient new version of the GOST algorithm using a single, level headedly chosen S-box. The proposed S-box is used eight times and still perfectly matches the standard specifications. This modification makes this version highly suitable to be deployed for low-cost passive RFID-tags. The required development area is reduced to 651GE, as given in [10].

On the other hand, Progressive algorithms represent another class that suggests new techniques suitable for limited-resource devices. Large number of algorithms proposed within this class in the last few years such as PRESENT [11], HIGHT [12], CLEFIA [13], mCrypton [14], KATAN [15], KLEIN [16], LED [17], PICCOLO [18], SIMON/SPECK [19], PRINCE [20], TEA [21], XTEA [21], XXTEA, and SLIM [5]. In the following paragraphs, those algorithms are briefly described.

PRESENT [11] is one of the most remarkable representatives of lightweight block ciphers. It introduces the concept of serial implementation. However, it suffers from different cryptanalysis methods. HIGHT was proposed in [12]; it is built on a generalized Feistel network. Lately, it has been adopted as an ISO standard in South Korea. However, a related-key rectangle and a related-key differential attack have been mounted on 26 and 28 rounds of HIGHT. Sony developed CLEFIA [13]; it has been adopted as an ISO standard as a lightweight cryptography algorithm. The differential cryptanalysis was the strongest method for attacking the reduced-round of CLEFIA. mCrypton was designed based on Crypton [22]. However, 7-round cryptanalysis of mCrypton was introduced [22]. KLEIN [16] is a software-oriented block cipher and a hardware efficient structure, as it combines 4-bit S-boxes with the AES Mix Column transformation. The various key lengths of KLEIN offer flexibility and a moderate security level. The most successful security analyses to KLEIN are represented in [23], which was resistant to differential attack. Different versions are now available where the secret key could be in full 12, reduced 13, and 14 rounds for KLEIN-64, 80, and 96, respectively. LED [17] is a family of AES-like lightweight ciphers. It supports arbitrary key lengths between 64 and 128 bits, where the two most-relevant versions are LED-64 and LED-128. Related-Key cryptanalysis is applied to 12 and 16 rounds

(out of 32) for LED-64, 16, and 24 rounds (out of 48) for LED-128. PICCOLO [18] is a lightweight block cipher that adopts a generalized Feistel network structure (GFS) with two versions, Piccolo-80 and PICCOLO-128 [24]. It suffers from meet-in-the-middle attacks on 14-round PICCOLO-80 and 18-round PICCOLO-128. Differential attacks up to 13 rounds and 15 rounds of PICCOLO-80 and 128, respectively can also be mounted. Furthermore, PICCOLO full round biclique cryptanalysis is given in [18]. SIMON and SPAC [19] are lightweight block cipher families optimized for ease, compatibility, and well-functioning in hardware and software based on ARX structures. They are vulnerable to both discrete and linear cryptanalysis [25]. PRINCE [20] is a lightweight block cipher that focuses on minimal latency; it fulfills hardware implementation restrictions and real-time security needs. It is founded on the so-called FX structure. It is also vulnerable to differential [26], Accelerated Exhaustive Search [26], meet-in-the-middle, biclique, a related-key, and linear attacks [27].

There are also algorithms from the 1990s, such as TEA, XTEA, and XXTEA [21], which consist of simple round structures that make them suitable for constrained environments. TEA and XTEA suffer from different types of attacks like differential and related-key attack [28]. TEA was redesigned to be XXTEA to avoid such attacks. However, it suffers from a chosen-plaintext attack [29]. SLIM [5] is the most recent ultra-lightweight block cipher designed for the Internet of Health Things and has one of the smallest footprints till writing this paper.

Tab. 1. Summarizes the most well-known lightweight block cipher algorithms concerning their structures, block size, key size, number of rounds, area in terms of GE, CMOS technology in μm, and the possible attacks.

Table 1: Comparison of lightweight block ciphers

Cipher	Cryptographic properties					Implementation properties	
	Block size	Key size	Struct.	Rounds	Attacks	Tech. used	Area (#GE)
AES	128	128	SPN	10	• Impossible differential, 7-rounds AES-128	0.13 μm	3100
		192		12		–	–
		256		14	• Related-key boomerang, full AES-192 and full AES-256 • Biclique (full AES)	–	–
CLEFIA	128	128	GFN	18	• Integral (12, 13, 14 rounds)	0.09 μm	4950
		192		22	• Improbable differentials (13, 14, 15 rounds)	–	–
		256		26		–	–
DESL	64	56	Feistel	16	• None	0.18 μm	1848
DESLX	64	184	Feistel	16	• None	0.18 μm	2168
GOST revisited	64	256	Feistel	32	• 2D-MitM	0.18 μm	651/1017
HIGHT	64	128	GFS	32	• Saturation (22 rounds) • Impossible diff. (26 rounds) • Related Key rectangle (full cipher) • Biclique (full cipher)	0.25 μm	3048

(**continued.**)

Table 1: Continued

Cipher	Cryptographic properties					Implementation properties	
	Block size	Key size	Struct.	Rounds	Attacks	Tech. used	Area (#GE)
KLEIN	64	64	SPN	12	• Differential (KLEIN-64, 8 rounds)	0.18 μm	1360/2032
		80		16			1530/2202
		96		20	• Truncated Differential (KLEIN-64, full cipher)		1700/2372
KATAN	32	80	Stream-cipher-like	254	• Differential (KATAN32, 115 rounds)	0.13 μm	802
	48					–	–
	64				• Multi-dimensionnal MitM (175-rounds KATAN32, 130-rounds KATAN48 and 112-rounds KATAN64)	0.13 μm	1054
KTANTAN	32	80	stream-cipher-like	254		0.13 μm	462
	48				• 3-subsets MitM (full cipher)	–	–
	64				0.13 μm	688	
		256		32			
LED	64	64	SPN	32	• *Ad Hoc* (12 rounds of LED-64, 32 rounds of LED-128)	0.18 μm	966
		128		48			1265
mCrypton	64	64	SPN	12	• MitM 7-rounds mCrytpon-64/96/128	0.13 μm	2420
		96					2681
		128			• MitM 8- and 9-rounds mCrytpon-128		2949
Piccolo	64	80	GFN	25	• Biclique (full Piccolo-80; 28-round Piccolo-128) • Related-key impossible diff[, 14-rounds Piccolo-80, 21-rounds Piccolo-128	–	683/1136
		128		31		–	758/1196
PRESENT	64	80	SPN	31	• Statistical saturation, up to 24-rounds • Multi-dimensionnal linear, 26-rounds • Truncated differential, 26-rounds	0.18 μm	1075/1570
		128					1391/1884
PRINCE	64	128	SPN	12	• Reflection attack, 6 rounds • Sieve-in-the-Middle, 8 rounds • Multiple differentials, 10 rounds	0.09 μm/0.13 μm	3286/3491
SIMECK	32	64	Feistel	32	• Differential (22/28/35 rounds SIMECK-32/48/64)	0.13 μm	549/765
	48	96		36			778/1117
	64	128		44			1005/1484

(continued.)

Table 1: Continued

Cipher	Cryptographic properties					Implementation properties	
	Block size	Key size	Struct.	Rounds	Attacks	Tech. used	Area (#GE)
SIMON	32	64	Feistel	32	• Differential (up to 21/22/28/35/46 rounds SIMON-32/48/64/96/128) • Linear (20 rounds SIMON-32) • Impossible diff. (19/20/26 rounds SIMON-32/48/64) • Multi-Dim. Linear (23/25/31/38/53 rounds SIMON-32/48/64/96/128)	–	–
	48	72/96		36			– /763
	64	96/128		42/44			838/1000
	96	96/144		52/54			984/ –
	128	128/192/256		68/69/72			1317/–/–
SPECK	32	64	ARX	22	• Differential (up to 14/15/19/17/19 rounds SPECK-32/48/64/96/128) • Rectangle (11/12/14/16/18 rounds SPECK-32/48/64/96/128)	–	–
	48	72/96		22/23			–/884
	64	96/128		26/27			984/1127
	96	96/144		28/29			1134/–
	128	128/192/256		32/33/34			1396/–/–
XTEA	64	128	Feistel	64	• Related-key rectangle 36 rounds-MitM 23 rounds	0.13 μm	3490

3 LBC-IoT Encryption and Decryption Specification

In this section, we specify the overall structure of LBC-IoT and its design principles. The selection of each component of LBC-IoT is motivated to achieve a well-balanced trade-off between security, performance, and resource requirements for specific resource-constrained IoT devices. It is also designed to achieve strictness against the various forms of attacks.

3.1 Structure of the LBC-IoT Algorithm

LBC-IoT is a symmetric enciphering scheme in which both the enciphering and deciphering procedures use the same key. The only distinction between the two processes is the reverse order of the subkeys. In LBC-IoT, two essential design issues are taken into consideration, security and simplicity. It achieves immunity against the exhaustive search attack using NIST recommendations report for key length (key length $\geq$ 80). Besides, it accomplishes both confusion and diffusion concepts. Moreover, a compact 4-bit S-box is designed based on high nonlinearity properties to fulfill the confusion requirements. Nevertheless, a combination of the two permutations boxes (P1, P2) and two halves swap because of the Feistel structure's nature are used for the data diffusion process. Consequently, simplicity is accomplished by selecting the S-box size and using simple internal processes in the round function.

An LBC-IoT block cipher is characterized by using 32-bit plaintext of blocks manipulated by an 80-bit key. This algorithm's design's fundamental feature is to have the smallest footprint area suitable for the different IoT applications. The framework was designed to be simply implemented in both software and hardware. Also, LBC-IoT consists of 32 rounds using 32 subkeys, each of 16-bit produced from the 80-bit key. Fig. 1a demonstrates the simple round configuration of the LBC-IoT encryption algorithm.

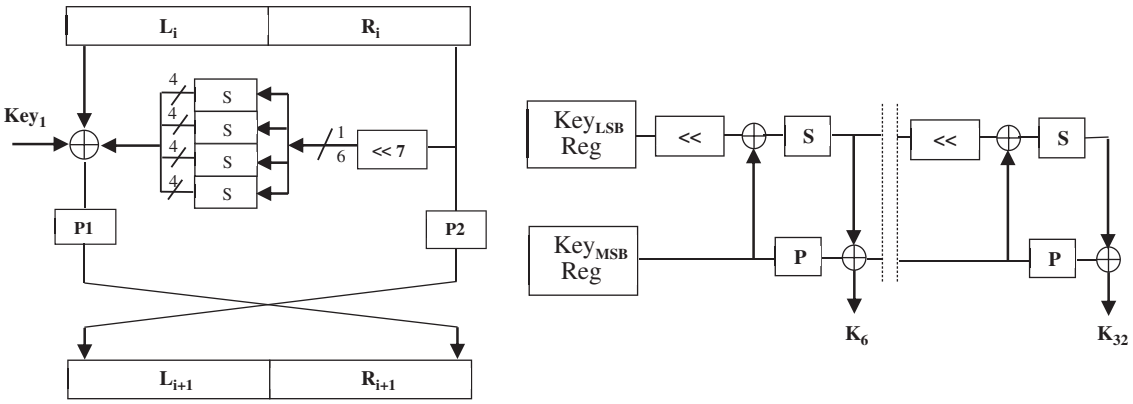

Figure 1: (a) Round structure of LBC-IoT encryption, (b) key generator structure

3.2 Single Round Processing

A detailed framework of LBC-IoT can be illustrated by looking at the internal configuration of one round. Here, the 32-bit input is divided into equal sixteen-bit halves, known as L_i or R_i. The processing of each round is summarized in Eqs. (1) and (2), where the input right half is circularly left-shifted by 7-bit. The output of the shifting process is forwarded to a substitution box that is repeated four times. The input left half, and the round sub-key are mixed through the addition of modulo 2. Finally, the output is forwarded to a permutation box P1 to become the subsequent round's right half input. Likewise, the input right half is sent to a permutation box P_2 to become the left half input of the following round; see Eqs. (1) and (2).

$$L_i = P_2(R_{i-1}) \tag{1}$$

$$R_i = P_1[L_{i-1} \oplus K_i \oplus S(R_{i-1} \ll 7)] \tag{2}$$

3.2.1 Substitution Layer

LBC-IoT is designed with NIST recommendations in mind. As can be noticed, the LBC-IoT cipher involves S-Boxes as the only nonlinear building blocks of the LBC-IoT cipher. Besides, there was a recommendation for the replacement of 8×8 S-boxes with a smaller 4×4 to suit the constrained environments. On the contrary, the authors of [30] demonstrated that DES had been broken when inefficient S-boxes are used. For similar DES frameworks, the métier of the security algorithms depends mainly on the power of the used S-boxes. LBC-IoT is a block cipher with a single 4×4 S-box repeated four times per round shown in Fig. 1a. The criteria of the S-box selection considered the immunity against various types of cryptanalysis. Tab. 2 gives the LBC-IoT S-box.

Table 2: The proposed S-box in LBC-IoT

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	0	8	6	D	5	F	7	C	4	E	2	3	9	1	B	A

Based on the previous table, the 4-bit S-box is selected with optimal bit-slice representation in the core function of the LBC-IoT. Bit-slice representation is proved to be strong against linear and differential properties as well as it has the lowest area footprint of 4-bit S-boxes [31]. The S-Box generation process was inspired by [32], where the focus of the selection was based on linear and the differential properties of the S-box. The optimality of the selected S-Box could be described as follows:

Let S be a 4×4 S-box. If S fulfills the following conditions, we call S an optimal S-box:

1) S is a bijection.
2) $\mathrm{Lin(S)} = 1/4$.
3) $\mathrm{Diff(S)} = 1/4$.

Therefore, three major properties are considered in the design of the S-Box which are bijection, linearity, and differentially.

a. Bijection Property

For an n-bit input function f to be a bijection, it has to fulfill a necessary and adequate condition that any linear combination of a Boolean function has Hamming weight 2^{n-1}. In other words, any potential input vector is mapped to a single output vector. This formation is accomplished by LBC-IoT S-box, as seen in Tab. 2.

b. Linearity property

To test LBC-IoT resistance to linear cryptanalysis, the Linear Approximation Table (LAT) needs to be defined first. For the given S-box generated from a mapping of $f: \mathrm{F_2^n} \to \mathrm{F_2^m}$, the linear approximation table entry $\mathrm{LAT(a \cdot b)}$ is described by Eq. (3) as:

$$LAT\,(a \cdot b) = \#\left\{x \in F_2^n \mid a \cdot x = b \cdot f\,(x)\right\} - 2^{n-1} \tag{3}$$

where $a \in F_2^n$, and $b \in \{F_2^m\}/0$, and $a \cdot x$ denotes the inner product of the vectors a and x evaluated over $\mathrm{F_2}$. Tab. 3 displays the linear approximation of the LBC-IoT S-box. The linear approximation entry with the greatest absolute values is denoted by $\mathrm{L_{max}}$. The lower the $\mathrm{L_{max}}$ value, the higher is the resistance of S-box to linear cryptanalysis.

c. Differential property

Once more, to verify the resistance of LBC-IoT to differential cryptanalysis, the Differential Distribution Table (DDT) needs to be defined first. For a specified S-box built from a mapping of $f: \mathrm{F_2^n} \to \mathrm{F_2^m}$, the Differential Distribution table entry $\mathrm{N_{\Delta x \Delta y}}$ is described by Eq. (4) as:

$$N_{\Delta x \Delta y} = \#\left\{x \epsilon F_2^n \mid f\,(x \oplus \Delta x) \oplus f\,(x) = \Delta y\right\} \tag{4}$$

where $\Delta x \in F_2^n$, and $\Delta y \in \{F_2^m\}$.

The entry $\mathrm{N_{oo}} = 2^{\mathrm{n}}$ is not taken into account since it has no cryptographic meaning. For $\Delta \mathrm{x} \neq 0$, the greatest entry in the differential distribution table is denoted by $\mathrm{D_{max}}$. Tab. 4 shows the differential distribution of the LBC-IoT S-box. As the $\mathrm{D_{max}}$ gets lower, the resistance of S-box to differential cryptanalysis becomes higher. Tab. 5 summarizes the differential distribution values observed, and the linear approximation tables for the S-box are summarized. As can be seen in the table, the DPPmax is $4/16 = 2^{-2}$, and the maximum bias likelihood equals $\pm 4/16$. It becomes clear that the proposed S-box is strong, as recommended in [32].

Table 3: Linear approximation table of the LBC-IoT S-box

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	4	0	0	4	0	0	−4	0	0	4	0	0
2	0	0	4	0	0	0	0	0	4	4	0	0	0	0	−4	0
3	0	4	0	0	2	0	0	2	0	−2	2	4	2	−2	−2	−2
4	0	0	2	−4	0	2	4	0	2	−2	0	2	−2	0	2	2
5	0	0	4	0	0	0	0	0	−4	0	0	0	4	0	0	4
6	0	4	0	0	2	0	0	2	0	2	2	−4	−2	−2	2	2
7	0	0	−2	0	4	−2	0	−4	2	2	0	2	2	0	2	2
8	0	4	0	0	−2	4	0	−2	0	2	−2	0	2	2	2	−2
9	0	0	−2	4	0	2	4	0	−2	2	0	2	−2	0	−2	2
A	0	0	2	4	−2	−2	0	2	2	0	−2	2	0	−2	4	0
B	0	−4	0	0	2	4	0	2	0	2	2	0	2	−2	2	−2
C	0	0	2	0	0	−2	0	0	−2	2	4	2	−2	4	2	−4
D	0	0	−2	0	−2	2	−4	2	2	0	2	2	0	2	0	4
E	0	0	2	4	2	2	0	−2	2	−4	2	−2	0	2	0	0
F	0	0	−2	0	−2	−2	4	2	2	0	2	−2	4	2	0	0

Table 4: Differential distribution table of the LBC-IoT S-box

I/O XOR Diff.	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	4	0	0	0	0	0	0	4	0	4	4	0	0	0	0
2	0	0	4	2	0	2	4	0	0	0	0	2	0	2	0	0
3	0	0	0	2	0	0	0	2	2	2	2	0	2	2	2	0
4	0	4	0	0	0	2	0	2	0	4	0	0	0	2	0	2
5	0	0	0	0	0	2	0	2	4	0	4	0	0	2	0	2
6	0	0	4	2	4	0	0	2	0	0	0	2	0	0	0	2
7	0	0	0	2	0	2	0	0	2	2	2	0	2	0	2	2
8	0	0	0	0	4	0	4	0	0	0	0	0	4	0	4	0
9	0	0	0	0	2	2	2	2	0	0	0	0	2	2	2	2
A	0	0	4	2	0	2	0	0	0	0	0	2	0	2	4	0
B	0	0	0	2	2	2	2	0	2	2	2	0	0	0	0	2
C	0	4	0	0	0	2	0	2	0	4	0	0	0	2	0	2
D	0	4	0	0	2	0	2	0	0	0	0	4	2	0	2	0
E	0	0	4	2	0	0	0	2	0	0	0	2	4	0	0	2
F	0	0	0	2	2	0	2	2	2	2	2	0	0	2	0	0

Moreover, the beauty of the selected S-box is the simple hardware implementation, where it consists of 3 AND gates, 1 OR gate, and 4 XOR gates. Besides, its algebraic degree is 3, with differential probability 2^{-2} and linear probability 2^{-1}, which provide a high rigidity profile against the most effective cryptanalyses attacks linear and differential cryptanalysis. In addition, bit-slice implementation supports immunity against side-channel attacks.

Table 5: Summary of the non-linear properties of LBC-IoT block cipher S-boxes.

LBC-IoT S-Box	D_{max} (Differential distribution table)	L_{max} (Linear approximation table)
	4	±4

3.2.2 Permutation Layer

In general, the permutation is a rearrangement process. In order to enhance the diffusion of the standard Feistel structure, along with the two-half swapping owing to the design of the Feistel structure between rounds, LBC-IoT uses two permutation boxes P1 and P2 between rounds. Here, the permutation is the last phase of the LBC-IoT function. The permutation box accepts 16-bit and permutes them using a certain rule producing a 16-bit output. The permutation process is given in Tab. 6 below.

Table 6: Permutation table

Permutation																
x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
P1 (x)	13	10	7	12	9	14	3	2	5	16	15	4	1	6	11	8
P2 (x)	5	8	16	12	3	11	2	13	4	1	14	6	9	15	7	10

In this paper, the FIPS conventions have been considered where FIPS number of bits from left to right starts at 1. When choosing a mixing layer, our emphasis on hardware reliability includes a linear layer that can be applied with a minimal number of processing components, i.e., transistors. Besides, for simplicity, a standard bit-permutation with no fixed point is chosen to avoid linearity analysis.

3.3 Key Generation

For 32 rounds and a block of 32-bit, 32 sub-keys (16-bit) are generated from the 80-bit encryption key (see Fig. 1b). The generation scheme is as follows:

- The first five sub-keys, labeled K1, K2, K3, K4, and K5 are taken directly from the original key, with K1 is equal to the first (least significant) 16-bits, K2 corresponding to the next 16-bits, and so on. Then, the 80-bit key passes to a divider that results in two 40-bit quantities, labeled Key_{MSB} and Key_{LSB}; each half is thereafter treated separately.
- At each round, Key_{LSB} goes through a circular left shift by three bits, and then the produced output is XORed by the Key_{MSB}. The output of the XORing operation is forwarded to a substitution layer. The output of the S- boxes and the permutated Key_{MSB} are manipulated using XOR operation to produce the round sub-key.

3.4 LBC-IoT Decryption

The decryption process is the same as that of encryption LBC-IoT decryption is done using ciphertext as an input to the same LBC-IoT structure. On the other hand, the decryption sub-key is applied in the reverse order with another sub-key selection.

a. Hardware Implementation

One of the critical issues of lightweight encryption architecture is hardware implementation. The architecture challenge is minimizing the algorithm implementation area. The plaintext and the encryption key occupy a fixed size of the memory depending on the technology used, while the encryption algorithm (Round Function, Key schedule, and Control logic) is the main challenge in the implementation, as can be seen in Fig. 2a. As mention in section 2, Poshmann et al. in [11] have introduced the first modern lightweight encryption algorithm called PRESENT. Although the serial implementation triggers a rise in latency and decreases in throughput, in [11], they recommend that serial implementation is the most effective implementation for lightweight cryptographic algorithms, primarily it is implemented on constraint resources environment.

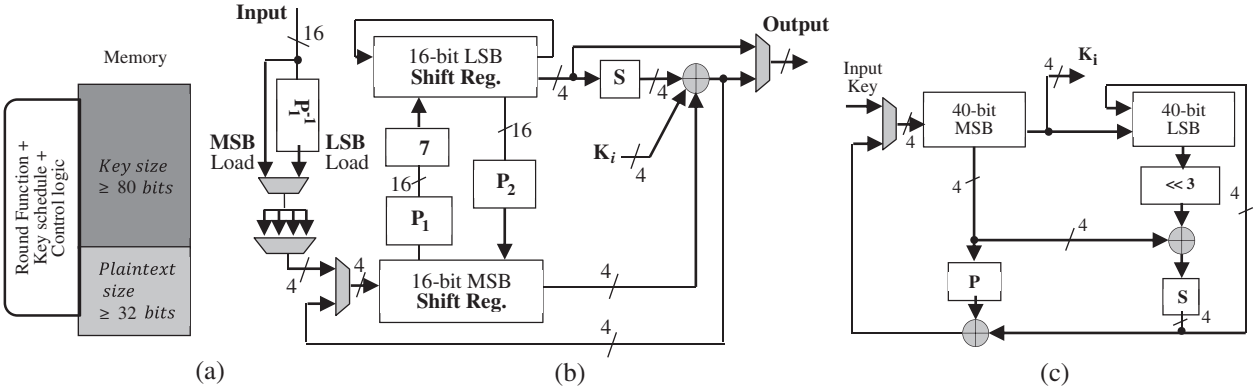

Figure 2: (a) Block cipher hardware perspective, (b) 4-bit data-path architecture, (c) key scheduling architecture

Circuit area and power are the essential hardware resources. The area is usually computed in gate.

Generally, a round-based implementation of ciphers can be done straightforwardly, while a serialized implementation creates some challenges for a hardware designer. Although these obstacles exist, intelligent cost-wise implementation of LBC-IoT with a data path width of 4-bit is given. Fig. 2b shows the 4-bit data-path architecture of the LBC-IoT round, and Fig. 2c shows the 4-bit data-path architecture of the LBC-IoT key generation scheduling. Thus, we spare the details of the former architecture and focus on the latter with a data path width of 4 bits. The most challenging is the permutation step since it permutes the whole state. Thus, it is not possible to operate on 4-bit chunks, but instead, we must operate on the entire state (16-bit). The details of our architecture are given in Fig. 2b.

The Gate Equivalent (GE) depends on a particular cell library. To check on the efficiency of the LBC-IoT design, ARM standard cell library for the IBM 8RF (0.13 micron) ASIC process is utilized [26]. The areas of some basic gates in this library are defined as NOT (0.75), NAND (1.00), AND (1.25), OR (1.25), XOR (2.00), XNOR (2.00), 2–1 MUX (2.25), D flip-flop (4.25), 1-bit full Adder (5.75), and scan flip–flop (6.25). For LBC-IoT, to store the 80-bit key, it requires about $80 \times 4.25 = 340$ GE, and to store the 32-bit data state, there is a need for about $32 \times 4.25 = 136$ GE.

The round structure consists of the following three sub-functions (XOR, S-box, and permutation):

- **Shifting Process:** The right-hand side (LSBs) of the entry data is passed through a left circular shifting process by 7 steps, which does not require any gates.
- **Substitution-layer:** The non-linear S-box layer consists of a single 4-bit S-box (4-AND, 4-XOR), which requires about (13 GE).
- **Add Round Key:** The key addition procedure is a 4-bit XOR operation used to achieve this mixing process, which requires about ($2 \times 4 \times 2 = 16$ GE).
- **Diffusion-layer:** At the end of each round function, a permutation process is executed, which can be implemented by simple wiring with no area cost.

Besides, the standard round of LBC-IoT requires three 2-to-1 MUXes and a single 4-to-1 MUX; a single 2-to-1 MUX cost 2.25 GE and 4-to-1 MUX costs 6.25 GE. Consequentially, the multiplexing (selecting) process requires ($3 \times 2.25 + 1 \times 6.25 = 13$ GE). Tab. 7 gives the entire encryption process area in gate equivalents.

Table 7: Area estimation of the hardware implementation of LBC-IoT in GE

Component	Gate Count
Registers	
Left register (16-bit)	$16 \times 4.25 = 68$ GE
Right register (16-bit)	$16 \times 4.25 = 68$ GE
Round function	
Xor	$2 \times 4 \times 2 = 16$ GE
Muxes	$3 \times 2.25 + 1 \times 6.25 = 13$ GE
Substitution layer	13 GE
Total	
178 GE	

The key-scheduling architecture consists of four sub-functions (XOR, S-box, permutation, and shifting).

- **Shifting Process:** In LBC-IoT, the right-hand side (LSBs) of the key segments is passed through the left circular shifting process by 3 steps, which not requires any gates.
- **Permutation Layer:** In LBC-IoT, the left-hand side (MSBs) of the key segments is passed through the permutation process (P1), which not requires any gates as well.
- **Substitution-layer:** In LBC-IoT, Single S-box is used for key scheduling to reduce the overhead in the implementation of the datapath. The non-linear S-box layer consists of a single 4-bit S-box (3-AND, 1-OR, and 4-XOR), which requires about (13 GE).
- **XORing Process:** In LBC-IoT, the key generator requires two 4-bit XOR operations to manipulate the different inputs, which requires about ($2 \times 7.5 = 15$ GE).

Finally, a single 2-to-1 MUX is required to select between the input key and the result that appears at the bottom of the previous round's data-path; a single 2-to-1 MUX costs 2.25 GE. Consequentially, GEs calculations of this architecture for its hardware implementation are shown in Tab. 8. Then, the overall hardware implementation area of the LBC-IoT cipher is 548.25 GE. It is the smallest ever implementation area for cipher until writing this article to the best of our

knowledge. The closest implementation was the one proposed in simeck32-64, where the required area was 549 GE; however, it is controlled by the 64-bit key.

Table 8: GE calculation for key scheduling

Component	Gate count
Shift registers	
Left register (40-bit)	$40 \times 4.25 = 170$ GE
Right register (40-bit)	$40 \times 4.25 = 170$ GE
Key function	
Xor	$2 \times 7.5 = 15$ GE
Mux	$1 \times 2.25 = 2.25$ GE
Substitution layer	13GE
Total	
370.25 GE	

b. Software Implementation

This section focuses on the efficient software implementation of 4×4-bits-boxes where it is an integral part of several primitive cryptographies. We are investigating bit-sliced implementations, shown in Fig. 3, which claimed to have a very effective implementation for various block ciphers. The bit-slicing concept is that processors work with registers larger than one bit where bitwise operations can be parallelized. This can be up to a factor of 128 on modern CPUs. However, the upcoming expansion of Intel's AVX would allow for 256-bit operands.

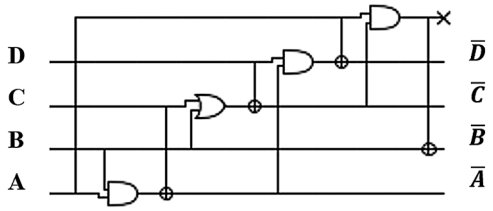

Figure 3: 4-bit S-box with optimal bit-slice representation

For processors with a restricted set of instructions (i.e., AND, OR, MOVE, XOR, NOT operations), we suggest low-cost encryption schemes (i.e., tiny code size and memory) [11]. We just need to take the number of ANDs and ORs into account. Tab. 9 gives an efficient instructions sequence "set of routines" for finding the shortest program to implement the proposed S-boxes (A, B, C, D are the four input bits).

c. LBC-INoT Analysis

In this section, LBC-IoT cryptographic strength is described as follows:

- **Block length:** The block length should be long to avoid any statistical analysis. As shown in the LBC-IoT design, the block length is a 32-bit block, which is enough to resist any statistical analysis.
- **Key Length:** To prevent exhaustive key searches, the key should be long enough. LBC-IoT has an 80-bit key length that makes it secure, even in the future.

- **Confusion:** Here, the ciphertext should be complicated and should rely on the plaintext and key. The goal is to confuse the determination of how ciphertext statistics depend on plaintext statistics. LBC-IoT uses a strong 4-bit S-box satisfying the confusion requirements.

Table 9: The proposed S box Logic operations (a, b, c, d are the four input bits)

Instruction	Expression
Z = A **AND** B	Z = A **&** B
X = C **XOR** Z	X = C $\bigoplus$ Z
Y = B **OR** C	Y = B | C
H = D **XOR** Y	H = D $\bigoplus$ Y
N = D **AND** X	N = D **&** X
O = N **XOR** A	O = N $\bigoplus$ A
S = H **AND** A	S = H **&** *A*
L = S **XOR** B	L = S $\bigoplus$ B
MOV A′, X	=> A′ = X
MOV B′, L	=> B′ = L
MOV C′, H	=> C′ = H
MOV D′, O	=> D′ = O

Table 10: Hardware implementations comparison of lightweight block ciphers on 0.13 μm technology

Algorithm	Block size (bits)	Key length (bit)	Tech. (μm)	Network structure	Area in GE
AES-128 [33]	128	128	0.13	SPN	3100
CLEFIA [34]	128	128	0.13	GFN	2488
NOEKEON [35]	128	128	0.13	SPN	2880
LED [36]	64	128	0.13	SPN	3194
PRINCE [20]	64	128	0.13	SPN	2953
SIMON [19]	64	128	0.13	SPN	1026
SPECK [19]	64	128	0.13	SPN	1005
XTEA [21]	64	128	0.13	Feistel	2521
Piccolo-128 [18]	64	128	0.13	GFN	758
SEA [35]	96	96	0.13	Feistel	2,562
mCrypton-96	64	96	0.13	GFN	2681
mCrypton-128	64	96	0.13	GFN	2949
SIMON [19]	48	96	0.13	SPN	796
SPECK [19]	48	96	0.13	SPN	778
PRESENT-80 [11]	64	80	0.13	SPN	2195
RECTANGLE [37]	64	80	0.13	SPN	1111
Piccolo-80 [38]	64	80	0.13	GFN	683
LBC-IoT	**32**	**80**	**0.13**	**Feistel**	**548**
mCrypton-64 [14]	128	64	0.13	SPN	2420
KLEIN [23]	64	64	0.13	SPN	1432
SIMON [19]	32	64	0.13	SPN	562
SPECK [19]	32	64	0.13	SPN	549

- **Diffusion:** Each plaintext bit should impact each ciphertext bit, and each key bit should affect each ciphertext, extending the plaintext bit's statistical structure over several bits of ciphertext. via LBC-IoT, a permutation process is carried out, and the exchange of the two halves of the plain text in each round. This configuration takes two (n/2)-bits values extracted from the plaintext as input and two (n/2)-bits sub-keys derived from the key and outputs (n/2)-bits. Each output bit in the first round relies on each input bit's plaintext and each subkey bit. In the algorithm, this simple structure is replicated m times.

The hardware implementation result of LBC-IoT is shown in Tab. 10 with a comparison with the other algorithms. The lightweight block ciphers were implemented on 0.13 μm technology. As shown in the table, LBC-IoT could have the minimum GE among the current lightweight algorithms. Besides, it proposes an 80-bit key size and a block size of 32 bits achieving the same performance as other algorithms. Moreover, it follows the Feistel structure, which could be the simplest among the other structures.

4 Conclusion

A proposed lightweight cryptography algorithm entitled "LBC-IoT" has been introduced in this paper. The goal of the proposed algorithm was to provide a practical and secure cipher for low-resource applications. The benefit of the proposed LBC-IoT is its simplicity in terms of the used functions and the utilized compact S-boxes. Based on our analysis of LBC-IoT, we are confident to conclude that LBC-IoT hardware implementation has a minimum GE among the current algorithms reported in the literature. Besides, the algorithm has a minimum software footprint, which makes it suitable for limited-resource devices. Moreover, LBC-IoT follows the recent standards and up-to-date recommendations. Nevertheless, it has high immunity against the different attacks, including linear, nonlinear, side-channel attacks, etc. The future work involves examining LBC-IoT in IoT applications such as healthcare and the military, where data is critical to be transferred through wireless channels. Hardware implementation is planned on FPGA and other recent programmable devices such as Arduino and Raspberry Pi.

Funding Statement : This project has been funded by Scientific Research Deanship at University of Ha'il—Saudi Arabia through Project Number RG-20019.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] E. Bertino, V. Casola, A. Castiglione and W. Susilo, "Security and privacy protection vs. sustainable development," *Computer Security*, vol. 76, pp. 250–251, 2018.

[2] C. M. Colombini, A. Colella, M. Mattiucci and A. Castiglione, "Cyber threats monitoring: Experimental analysis of malware behavior in Cyberspace," in *Int. Conf. on Availability, Reliability, and Security*, Regensburg, Germany, pp. 236–252, 2013.

[3] A. E. S. B. W. Aboshosha, M. M. Dessouky and R. Ramadan, "LCA-Lightweight cryptographic algorithm for IoT constraint resources," *Menoufia Journal of Electronics Engineering Resources*, vol. 28, pp. 374–380, 2019.

[4] D. N. Serpanos and A. G. Voyiatzis, "Security challenges in embedded systems," *ACM Transactions on Embedded Computing Systems*, vol. 12, no. 1, pp. 1–10, 2013.

[5] B. Aboushosha, R. A. Ramadan, A. D. Dwivedi, A. El-Sayed and M. M. Dessouky, "SLIM: A Lightweight block cipher for Internet of Health Things," *IEEE Access*, vol. 8, pp. 203747–203757, 2020.

[6] K. A. McKay, L. Bassham, M. S. Turan and N. Mouha, *Report on Lightweight Cryptography*. Gaithersburg, MD: National Institute of Standards and Technology (NIST), 2017.

[7] T. Eisenbarth, S. Kumar, C. Paar, A. Poschmann and L. Uhsadel, "A survey of lightweight-cryptography implementations," *IEEE Design and Test of Computers*, vol. 24, no. 6, pp. 522–533, 2007.

[8] G. Leander, C. Paar, A. Poschmann and K. Schramm, "New lightweight DES variants," in *Int. Conf. on Fast Software Encryption*, Berlin, Heidelberg. Springer Berlin Heidelberg, 2007.

[9] A. Poschmann, G. Leander, K. Schramm and C. Paar, "New light-weight crypto algorithms for RFID," in *IEEE Int. Symp. on Circuits and Systems*, New Orleans, LA, USA, pp. 1843–1846, 2007.

[10] A. Poschmann, S. Ling and H. Wang, "256 Bit standardized crypto for 650 GE–GOST revisited," in *Int. Workshop on Cryptographic Hardware and Embedded Systems*, Santa Barbara, USA, pp. 219–233, 2010.

[11] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann *et al.,* "PRESENT: An ultra-lightweight block cipher," in *Int. Conf. on Cryptographic Hardware and Embedded Systems*, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 450–466, 2007.

[12] D. Hong, J. Sung, S. Hong, J. Lim, S. Lee *et al.,* "HIGHT: A new block cipher suitable for low-resource device," in *Int. Workshop on Cryptographic Hardware and Embedded Systems*, Yokohama, Japan, pp. 46–59, 2006.

[13] M. B. Somasagar, "Clefia-a encryption algorithm using novel s-box architecture," *International Journal of Engineering Resources*, vol. V9, no. 7, pp. 825–827, 2020.

[14] C. H. Lim and T. Korkishko, "mCrypton–a lightweight block cipher for security of low-cost rfid tags and sensors," in *Int. Workshop on Information Security Applications*, Jeju Island, Korea, pp. 243–258, 2006.

[15] C. De Cannière, O. Dunkelman and M. Knežević, "KATAN and KTANTAN—a family of small and efficient hardware-oriented block ciphers," in *Int. Workshop on Cryptographic Hardware and Embedded Systems*, Lausanne, Switzerland, pp. 272–288, 2009.

[16] Z. Gong, S. Nikova and Y. W. Law, "KLEIN: A new family of lightweight block ciphers," in *Int. Workshop on Radio Frequency Identification: Security and Privacy Issues*, Amherst, USA, pp. 1–18, 2012.

[17] P. L. Lee, C. L. Yeh, J. Y. S. Cheng, C. Y. Yang and G. Y. Lan, "An SSVEP-based BCI using high duty-cycle visual flicker," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 12, pp. 3350–3359, 2011.

[18] K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita *et al.,* "Piccolo: An ultra-lightweight blockcipher," in *Int. Workshop on Cryptographic Hardware and Embedded Systems*, Nara, Japan, pp. 342–357, 2011.

[19] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks *et al.,* "The SIMON and SPECK lightweight block ciphers," in *Proc. of the 52nd Annual Design Automation Conf.*, New York, NY, USA, pp. 1–6, 2015.

[20] J. Borghoff, A. Canteaut, T. Guneysu, E. B. Kavun, M. Knezevi *et al.,* "A low-latency block cipher for pervasive computing applications," in *Int. Conf. on the Theory and Application of Cryptology and Information Security*, Beijing, China, pp. 208–225, 2012.

[21] M. H. Al Meer, "Programmable SoC for an XTEA encryption algorithm using a co-design environment replication performance approach," *Journal of Computer Communication*, vol. 5, no. 11, pp. 40–59, 2017.

[22] C. H. Lim, "A revised version of CRYPTON: CRYPTON V1.0," in *Int. Workshop on Fast Software Encryption*, Rome, Italy, pp. 31–45, 1999.

[23] V. Lallemand and M. Naya-Plasencia, "Cryptanalysis of KLEIN," in *Int. Workshop on Fast Software Encryption*, London, UK, pp. 451–470, 2015.

[24] B. W. Aboshosha, M. M. Dessouky, R. A. Ramadan and A. El-Sayed, "Evaluation of lightweight block ciphers based on general feistel structure (GFS)," *WAS Science Nature*, vol. 2, no. 1, pp. 1–7, 2020.

[25] A. Biryukov, A. Roy and V. Velichkov, "Differential analysis of block ciphers SIMON and SPECK," in *Int. Workshop on Fast Software Encryption*, London, UK, pp. 546–570, 2015.

[26] A. Canteaut, T. Fuhr, H. Gilbert, M. Naya-Plasencia and J. R. Reinhard, "Multiple differential cryptanalysis of round-reduced PRINCE," in *Int. Workshop on Fast Software Encryption*, London, UK, pp. 591–610, 2015.

[27] J. Jean, I. Nikolić, T. Peyrin, L. Wang and S. Wu, "Security analysis of PRINCE," in *Int. Workshop on Fast Software Encryption*, Singapore, pp. 92–111, 2014.

[28] S. Hong, D. Hong, Y. Ko, D. Chang, W. Lee *et al.,* "Differential cryptanalysis of TEA and XTEA," in *Int. Conf. on Information Security and Cryptology*, Seoul, Korea, pp. 402–417, 2004.

[29] E. Yarrkov, "Cryptanalysis of XXTEA," *IACR Cryptology*, vol. 254, pp. 3–72, 2010.

[30] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," *Journal of Cryptology*, vol. 4, no. 1, pp. 3–72, 1991.

[31] B. Ullrich, M. Canniere, C. D. Indesteege, S. Kucuk, O. Mouha *et al.,* "Finding optimal bitsliced implementations of 4×4-bit S-boxes," in *Symmetric Key Encryption Workshop*, Lyngby, Denmark, 2011.

[32] G. Leander and A. Poschmann, "On the classification of 4 Bit S-Boxes," in *Int. Workshop on the Arithmetic of Finite Fields*, Madrid, Spain, pp. 159–176, 2007.

[33] P. Hamalainen, T. Alho, M. Hannikainen and T. D. Hamalainen, "Design and implementation of low-area and low-power AES encryption hardware core," in *9th EUROMICRO Conf. on Digital System Design*, Dubrovnik, Croatia, pp. 577–583, 2006.

[34] T. Akishita and H. Hiwatari, "Very compact hardware implementations of the blockcipher CLEFIA," in *Int. Workshop on Selected Areas in Cryptography*, Toronto, ON, Canada, pp. 278–292, 2012.

[35] T. Plos, C. Dobraunig, M. Hofinger, A. Oprisnik, C. Wiesmeier *et al.,* "Compact hardware implementations of the block ciphers mcrypton, NOEKEON, and SEA," in *Int. Conf. on Cryptology in India*, Kolkata, India, pp. 358–377, 2012.

[36] J. Guo, T. Peyrin, A. Poschmann and M. Robshaw, "The LED block cipher," in *Int. Workshop on Cryptographic Hardware and Embedded Systems*, Nara, Japan, pp. 326–341, 2011.

[37] W. Zhang, Z. Bao, D. Lin, V. Rijmen, B. Yang *et al.,* "RECTANGLE: A bit-slice lightweight block cipher suitable for multiple platforms," *Science China Information Sciences*, vol. 58, no. 12, pp. 1–15, 2015.

[38] L. Ertaul and S. K. Rajegowda, "Performance analysis of CLEFIA, PICCOLO, TWINE Lightweight block ciphers in IoT environment," in *Int. Conf. of Security and Management*, Las Vegas, USA, 2017.