Hindawi
International Journal of Intelligent Systems
Volume 2023, Article ID 6647416, 13 pages
https://doi.org/10.1155/2023/6647416

Research Article

ALICA: A Multi-S-Box Lightweight Cryptographic Algorithm Based on Generalized Feistel Structure

Jun Ye,[1,2] **Yabing Chen**,[1,2] **Fanglin An**,[1,2] **and Weili Jiang**[1,2]

[1]*School of Cyberspace Security, Hainan University, Haikou, China*
[2]*Key Laboratory of Internet Information Retrieval of Hainan Province, Haikou, China*

Correspondence should be addressed to Jun Ye; yejun@hainanu.edu.cn

Received 2 July 2023; Revised 5 October 2023; Accepted 16 November 2023; Published 5 December 2023

Academic Editor: Mohammad R. Khosravi

With the development of science and technology, IoT devices have already become ubiquitous in the public eye. Through the perception layer, the collected data are displayed or transmitted to the server backend for analysis. Due to the increasing integration of IoT devices into people's daily lives, privacy issues, such as data leaks, have received more attention. Most sensor nodes, such as temperature and pressure sensors in marine environments, have low computing power, storage capacity, and significant underlying heterogeneity, making it challenging to implement a standardized data security protection solution. Data security in the nodes is seriously challenged as a result. It is of great significance to design a lightweight block cipher for the Internet of Things (IoT) environment, which can ensure the security of node information. A new lightweight block cipher algorithm called ALICA is proposed in this paper, which is well-suited for the low computing power and heterogeneous device environment at the lower layers. A generalized Feistel structure and a linear structure with XOR and shift operations are used to achieve ease of software implementation. Two different S-boxes are used to create the nonlinear structure of the password, thereby enhancing the robust security of the cipher. In addition, the cipher adopts a design approach that prioritizes software efficiency while also considering hardware implementation efficiency. This makes it more suitable for low computing power, storage capacity, and resource-limited IoT sensor nodes at the lower layers.

1. Introduction

The Internet of Things (IoT) is the transmission and communication of data between terminal nodes in the Internet era. Through information gathering and communication, sensor devices form an interconnected network for transmitting information, enabling applications such as remote device tracking, dynamic logistics and goods management, and predictive analysis. IoT continuously expands and drives societal development, bringing extensive and profound impacts to various aspects of people's lives and offering significant convenience. However, while benefiting from the advancements in information technology, challenges arise in securing IoT devices due to their heterogeneous nature, limited computing capabilities of many IoT devices, constraints imposed by deployment environments, and resource limitations. Additionally, different IoT devices may utilize various cryptographic hardware acceleration modules, making it challenging to adopt a unified architecture for security protection. Moreover, some IoT devices lack built-in cryptographic modules or have outdated ones, leaving them vulnerable to security breaches related to password protection. Malicious individuals exploit these vulnerabilities in IoT terminal devices to engage in activities such as device destruction or stealing information. This leads to the silent theft, sharing, and dissemination of significant amounts of sensitive data, often for financial gain. To prevent data breaches and safeguard stored information, data encryption and secure communication in IoT devices play a crucial role. While efficient cryptographic hardware security modules are essential, the diverse nature of IoT devices limits deployment flexibility. Therefore, designing a lightweight cryptographic algorithm that prioritizes software implementation and is easily deployable becomes

a viable solution to address security concerns related to information and data leakage in resource-constrained IoT devices. The application of modern cryptography plays a vital role in ensuring data security. While encryption algorithms like AES [1] can provide data security, their implementation often requires a significant number of hardware gate circuits, resulting in a relatively large hardware footprint. This poses a challenge for deploying AES on low-power and low-computational devices, as it demands significant power consumption. In software implementation, the Advanced Encryption Standard (AES) can consume substantial amounts of memory and may not satisfy the output bandwidth demands of real-world applications. Additionally, it fails to meet the low-power and low-resource demands of IoT devices that have limited resources. In essence, traditional encryption implementations often struggle to deliver satisfactory performance in resource-limited environments, resulting in high latency and energy consumption. Therefore, they may not offer optimal data protection capabilities.

In recent years, several lightweight block cipher schemes have been proposed for various environments. As early as the 1980s and 1990s, the industry developed a series of lightweight encryption algorithms, including A5/1 [2]. This algorithm was utilized in the field of mobile communication as a cipher algorithm for communication encryption, commonly referred to as the GSM encryption algorithm. The design goal of the A5/1 algorithm was to efficiently perform encryption and decryption operations on devices with limited resources. However, despite being considered secure in the past, A5/1 has now been proven to be susceptible to brute-force attacks. This is due to linear attack vulnerabilities in its pseudorandom number generator design, which uses a 64-bit key. It is destined to be phased out with the advancement of computational power. Consequently, more and more people are turning to use A5/3 (also known as KASUMI) [3]. In 2005, Lim et al. improved the Crypton cipher and introduced the mCrypton [4] cipher, which offers three selectable security thresholds and is specifically designed for RFID tags. In 2007, Shibutani et al. proposed the Piccolo algorithm [5]. Piccolo is a block cipher algorithm that provides a security threshold with a 128-bit key size and utilizes the SPN structure. Piccolo's design goal is similar to achieve efficient encryption and decryption on embedded devices while demonstrating good performance in both software and hardware implementations. In 2011, Guo et al. introduced the LED cipher [6]. LED cipher offers two different encryption methods for keys of varying lengths, specifically the 64-bit and 128-bit versions, each with its own security threshold. It adopts the SPN structure, considering the resource constraints and power requirements of embedded devices. LED uses a simple S-box structure and efficient bit operations. Also in 2011, Suzaki et al. proposed the TWINE cipher [7], which is also based on the SPN structure. A notable feature of TWINE is its ability to select key and block lengths based on application requirements. For example, TWINE's key size and sequence length can be used in two different ways. Furthermore, TWINE provides variants with variable key and block lengths to adapt to various application scenarios. In 2011, Wu et al. introduced LBlock [8], which utilizes the Feistel structure. While this design logic reduces the cost of hardware and software implementations, it requires more iterative rounds as a trade-off to achieve sufficient cryptographic security. In 2012, Borghoff et al. proposed PRINCE [9], which utilizes a matrix structure and combines iterations involving permutation and linear transformation operations. The introduction of the matrix structure increases the complexity of cipher design logic and requires more memory. In 2015, Zhang et al. proposed a lightweight block cipher called RECTANGLE [10], which is suitable for hardware implementations and utilizes bit-slicing techniques. Similarly, the implementation of matrices in its design requires more memory. In 2017, Banik et al. introduced GIFT [11], a block cipher that utilizes the SPN structure with S-boxes generated using the Grøstl hash function. The use of the SPN structure makes logic implementation more complex compared to Feistel-structured ciphers, and it requires more memory. In 2019, Wu et al. proposed the uBlock algorithm [12], which is adaptable to various software and hardware platforms, taking into account the computing resources of modern microprocessors. However, despite mentioning that uBlock can be accelerated using SSE and AVX2 instructions, it is important to note that these instructions are not commonly supported on target devices. Therefore, the practical efficiency of this cipher in real-world applications still needs to be verified. In 2021, Kiran Kumar et al. introduced BRISI [13], which is based on ARX (add-rotate-XOR) operations. It adopts the Feistel structure and combines the characteristics of the BRIGHT and SIMON structures. Also in 2021, Kim et al. proposed PIPO [14], a bit-slice-oriented cipher. In 2021, Ying et al. introduced Shadow, which combines Feistel and ARX structures. In 2022, Gupta et al. introduced FUTURE [15], which aims to encrypt data within a single clock cycle in order to mitigate the high implementation cost of MDS matrices in lightweight block ciphers. This cipher is designed with a focus on hardware and may not be as suitable for software implementations. Cipher algorithms such as PIPO and Shadow, which are newly proposed, employ novel design logic and require further testing in terms of security and efficiency. As old cipher algorithms continue to face attacks and vulnerabilities in their design logic being exploited, it is necessary to introduce new cipher algorithms to counter the increasing computational capabilities and specialized attacks by adversaries. In 2023, following a review and third-party security analysis conducted by NIST's lightweight cryptography team, the Ascon series algorithms were chosen as the standard lightweight block cipher algorithms [16]. This marked the establishment of new cipher algorithm standards and opened up new research possibilities for researchers. When designing and using lightweight ciphers, it is crucial to consider specific application scenarios. Current cipher algorithms mostly use S-boxes as the nonlinear components of the cipher, particularly in lightweight block ciphers. In these ciphers, 4×4 S-boxes are often the optimal choice as they strike a balance between hardware implementation area, performance, security, and resource consumption. In the implementation of

lightweight block cipher algorithms that utilize S-box substitutions, such as Piccolo, LED, LBlock, and uBlock, their performance in terms of logic is similar. This is because their performance is determined by the cipher design logic and clock frequency when implemented in hardware. In software implementations, cipher algorithms with S-box design logic do not offer a significant advantage compared to those that use arithmetic operations in ARX. Additionally, they require more memory storage for substitution tables. On the contrary, S-boxes offer greater security advantages and can achieve the required level of cryptographic security with fewer cipher iterations. ALICA, from its inception, is designed to be suitable for both software and lightweight hardware implementations. The goal is to minimize memory usage and employ an appropriate number of iterations while maintaining the required security level. ALICA utilizes fundamental instructions such as XOR and shifts to maximize the performance of the cipher on various platforms. This allows it to meet the demands of real-world environments with heterogeneous computing units and resource constraints. In this paper, we propose a novel lightweight cipher algorithm called ALICA, designed to be applicable to low-computational-power heterogeneous devices. ALICA employs a generalized Feistel structure, and linear processes are implemented through XOR and bitwise operations, making it easy to implement in software and deploy uniformly across diverse devices. While ensuring security, ALICA outperforms other cipher algorithms that use S-boxes in terms of performance. ALICA does not rely on round constants, which reduces its advantages over memory-intensive ciphers. ALICA does not depend on specific instruction accelerations, allowing it to perform well on various computing platforms.

2. ALICA

ALICA is a block cipher that employs a generalized Feistel network structure. It operates on input blocks of 96 bits and uses a 128-bit key. In ALICA, the initial key K is subjected to a key expansion algorithm to generate new subkeys k_i and k_{i+1}. These subkeys are then separately applied to the left-round function F_{left} and the right-round function F_{right}. The only difference between the two lies in the utilization of distinct 4×4 S-boxes.

Next, the plaintext input is divided into 32-bit segments and fed into the round functions for 24 iterations. After the final round, the resulting ciphertext is produced. The specific process of splitting the plaintext into smaller segments will be illustrated in the overall framework design of ALICA.

2.1. The Design of ALICA. PRESENT, uBlock, and FUTURE are lightweight cryptographic algorithms that utilize the substitution-permutation network (SPN) structure, as shown in Figure 1. In contrast, there are other structures employed by Simeck and LBlock, for example, the Feistel structure (as shown in Figure 2) and its derivative, the generalized Feistel structure. The ALICA cipher algorithm belongs to the generalized Feistel structure, as shown in Figure 3.

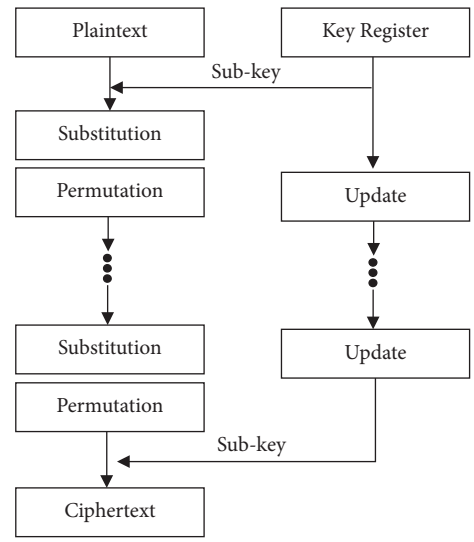

Figure 1: SPN structure.

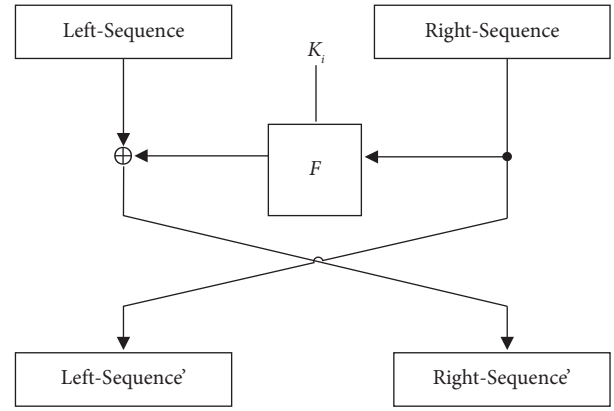

Figure 2: Feistel structure.

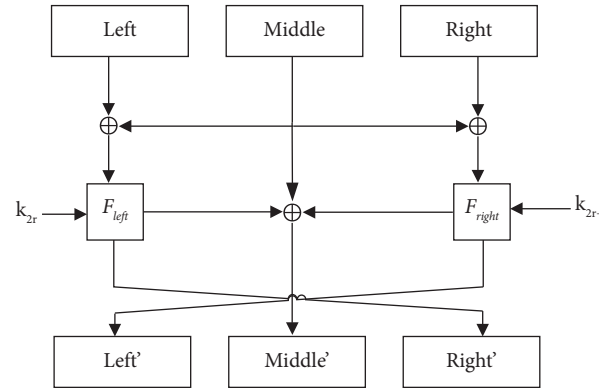

Figure 3: Generalized Feistel structure proposed in this paper.

In ALICA, the 96-bit plaintext sequence is divided into three groups: left, middle, and right, each consisting of 32 bits. The left and right groups are then separately fed into the left-round function and the right-round function. The results from both functions are XORed together to produce the intermediate sequence for the next round. The logical structure is illustrated in the diagram provided.

The F-function is composed of a 4×4 S-box and a bitwise shift operation. The design of the S-box is determined by the specified design scheme, and the output is XORed at the byte level before undergoing a shift operation. In the following sections, a detailed analysis and study of each component will be presented.

2.2. The Design of the F-Function. The F-function is the most crucial component in the Feistel structure of the cipher, and the complexity of its design directly affects the security aspects such as resistance to linearity and differentials. In ALICA, which adopts the generalized Feistel structure, the left-round function F_{left} and the right-round function F_{right} only differ in the internal S-boxes they use, while their fundamental structures remain the same. The F-function takes two types of sequences as inputs. The first type is the XOR result of a 32-bit plaintext sequence, and the second type is a 32-bit subkey sequence generated by the key expansion scheme. The F-function consists of S-box substitutions, byte-wise XOR operations, and shift operations. Using A–H to represent 4-bit random sequences, the F-function can be depicted as shown in the following figure.

2.3. The Design of S-Box. The S-box is the only nonlinear component in the ALICA cipher structure and plays a crucial role in ensuring the security and robustness of the cipher algorithm. ALICA employs three 4×4 S-boxes in its design, and their logical designs are identical. Although using larger S-boxes, such as the 8×8 S-box in AES (1000 GEs), provides better resistance against various cryptographic attacks, their larger hardware implementation area or higher memory consumption in software implementations makes them unsuitable for deployment. Similarly, 6×6 (300 GEs) and 6×4 (300 GEs) S-boxes are also not preferred choices for lightweight cipher designs. On the other hand, the hardware implementation of a 4×4 S-box only requires 28 GEs, significantly reducing the hardware implementation area and memory usage of the nonlinear component in cipher designs. Hence, the 4×4 S-box is the optimal choice for designing lightweight ciphers. However, reducing the size of the S-box design inevitably leads to a decrease in its security. Furthermore, Mishra et al. [17] analysed that the S-box in the PRESENT cipher is not optimally designed and cannot effectively resist differential attacks. Therefore, the design and selection of the 4×4 S-box should strive to improve its security performance.

The security of an S-box can be evaluated by analysing its cryptographic properties, including nonlinearity, differential uniformity, avalanche effect, algebraic degree, and distribution of terms. Different cryptographic properties determine the ability of an S-box to withstand various types of attacks. Generally, an S-box with better cryptographic properties provides stronger security.

Nonlinear mapping is a common design approach for many S-boxes, and researchers have developed powerful S-boxes using mapping transformations. Among them, the linear fractional transformation (LFT) is one of the mapping methods commonly used by researchers. In mathematics, a linear fractional transformation (LFT) can be roughly represented by the formula (1), which is generated by modifying four parameters: a, b, c, and d, in order to create a dynamic S-box.

$$z = \frac{\mathrm{ax} + b}{\mathrm{cx} + z}. \tag{1}$$

While the design concept of dynamic S-boxes can make it harder for attackers to identify design flaws in a cipher algorithm, implementing LFT requires additional memory storage for data. Furthermore, the performance of the algorithm's division operation is about ten times slower compared to bitwise XOR and shift operations on computers. In division operations, it is necessary to determine whether each subtraction of operands results in an overflow before deciding on the input for the next subtraction. Unlike multiplication, division operations cannot be parallelized, which further diminishes their advantages in lightweight cipher design. The use of multiple S-boxes in cipher algorithm design can enhance cryptographic strength without relying on dynamic S-boxes, as it eliminates these disadvantages.

The specific method for designing the S-box is as follows.

By addressing the issue of data overflow in the generation of S-boxes, Zahid et al. [18] proposed the use of the triple fraction transformation (CFT), which constrains the transformation within the finite field GF (2^4). The CFT, as shown in equation (2), is an extension of the LFT, where the original linear transformation is converted into a nonlinear transformation. The introduction of nonlinear characteristics makes the generation of S-boxes more complex. However, on the contrary, the probability of generating S-boxes with good cryptographic properties within a limited time frame increases.

$$C(z) = \frac{1}{\alpha(z)^3 + \beta} \left(\mathrm{MOD}\left(2^n + 1\right)\right), \alpha, \beta, \mathrm{z} \in \mathrm{Z}, \tag{2}$$

where $\alpha(z)^3 + \beta \neq 0$.

Let $\alpha = 3$ and $\beta = 4$. The S-box generated by these two values is used in the left function F_{left} of the ALICA cipher, referred to as the left S-box in this article, as listed in Table 1.

Next, this paper will analyze the left S-box generated using the proposed method for constructing S-boxes. It will evaluate its encryption strength using widely used standards to measure the performance of S-boxes.

2.3.1. Injectivity. A function f is injective if and only if for every $\forall y \in Y$ and a unique $x \in X$ such that $f(x) = y$. In other words, when $x_1 \neq x_2$, $f(x_1) \neq f(x_2)$.

2.3.2. Nonlinearity. A good S-box should possess high nonlinearity rather than linearity. The nonlinearity of an n-bit Boolean function f is calculated using the following formula:

$$\mathrm{NL}(f) = \frac{2^{n-1} - 1}{2} \max_{a \in 0,1^n, a \neq 0} \left| \frac{1}{2^n} \sum_{x \in 0,1^n} (-1)^{f(x) \oplus \langle x,a \rangle} \right|. \tag{3}$$

Here, $<x, \alpha>$ represents the bitwise dot product. The value proposed in this paper is 7.

Table 1: Left S-box of the ALICA cipher algorithm.

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

2.3.3. Strict Avalanche Criterion (SAC). Webster et al. [19] proposed the strict avalanche criterion (SAC) as a standard for a high-performance S-box. The average SAC value of the proposed left S-box in this paper is approximately 0.4974, which satisfies the strict avalanche criterion. SAC values for each value of the S-box are calculated and presented in Table 2.

2.3.4. Linear Probability. When evaluating the security of an S-box, an important metric is its linear probability. The linear probability of an S-box refers to the likelihood of a linear correlation between the input and output bits of the S-box. Specifically, assuming the input bits of the S-box are $x_1, x_2, \ldots, x_n$, and the output bits are $y_1, y_2, \ldots, y_m$, the linear probability of the S-box can be defined using the following equation:

$$P\left(a_1x_1 + a_2x_2 + \ldots + a_nx_n = b_1y_1 + b_2y_2 + \ldots + b_my_m\right). \tag{4}$$

Here, $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_m$ are arbitrary nonzero bit values, excluding all zeros and all ones. A smaller linear probability of an S-box implies that it is more difficult for attackers to exploit linear relationships for cryptographic attacks.

The linear probability of an S-box is calculated using the following equation:

$$\mathrm{LP} = \max\left|\frac{\#\{x \in Z \mid x \cdot A_x = S(x) \cdot B_x\}}{2^n} - \frac{1}{2}\right|, A_x, B_x \neq 0. \tag{5}$$

Here, A_x and B_x represent input masks and output masks, specifically, $Z \in \{0, 1, 2, ..., 15\}$. The maximum linear probability of the left S-box proposed in this paper is 0.25, indicating that the S-box possesses resistance against linear cryptanalysis.

2.3.5. Differential Uniformity. Differential uniformity is a measure of the first-order nonlinearity of an S-box and is also known as the minimum nonzero element value in the table of differential distribution. It represents the minimum magnitude between the differences (input differentials) of any two given inputs and the differences (output differentials) of their corresponding outputs. Differential uniformity is a measure of the uniformity of a permutation function across various input differentials. The method for calculating differential uniformity is as follows:

(1) Calculate the set of output differentials for each input differential in the S-box

(2) For each set of output differentials, calculate the number of distinct elements

Table 2: The SAC of left S-box.

0.5	0.4375	0.4375	0.4375
0.4375	0.4375	0.4375	0.4375
0.5	0.5	0.5	0.4375
0.4375	0.5	0.4375	0.4375

(3) Take the minimum value of the number of distinct elements among all sets of output differentials as the differential uniformity value of the S-box

A higher differential uniformity indicates a stronger first-order nonlinearity of the S-box, thus enhancing its security in cryptography. Based on the above analysis, the calculation formula for differential uniformity (DU) is given by equation (6), and the calculation formula for differential probability (DP) is given by equation (7):

$$\mathrm{DU} = \min\left[\left|\#\{\triangle y \mid S(x) \oplus S(x + \triangle x) = \triangle y\}\right|\right], \quad \triangle y, \triangle x \in F_{2^n},\ \triangle x \neq 0, \tag{6}$$

$$\Delta_p(\Delta x, \Delta y) = \frac{|a|S(a \oplus \Delta x) \oplus S(a) = \Delta y|}{2^n}. \tag{7}$$

Here, Δx represents the input differential, α represents a bit in the input differential, and $\mathbb{F}_{2^n}$ represents the n-bit binary field. Table 3 presents the differential distribution table of the proposed left S-box in this paper. Table 4 displays the linear approximation table of the proposed left S-box in this paper. Lastly, Table 5 illustrates the DU values of the proposed left S-box. According to the differential distribution table, the minimum DU value is 6. The differential probability (DP) of the left S-box is 0.25.

2.4. The Linear Transformation of F-Function. According to Shannon's theorem, the security and robustness of a cipher can be achieved through the principles of confusion and diffusion. The ALICA cipher employs a generalized Feistel structure, which inherently provides diffusion capabilities. However, this alone is not sufficient. Due to the lightweight constraints imposed by the designed S-boxes (nonlinear components), the design of the diffusion component also needs to balance diffusion capabilities with hardware implementation area and memory footprint. In ALICA, the linear components of the round function F are designed with a focus on high efficiency in software implementation. This is achieved by utilizing only bitwise cyclic shift and bitwise XOR operations on bit sequences. The specific design logic can be referred to in Figure 4. In the upcoming sections, this paper will represent the linear components using logical symbols.

Let M, N, P, and Q be 8-bit random sequences generated by the S-boxes within the round function F (not specifically referring to the left- or right-round function). k_i represents the subkey of the i-th round, composed of 32 bits. The generation of subkeys will be detailed in Section 3. The linear transformation L in the round function can be represented by Algorithm 1.

Table 3: Differential distribution table of the left S-box.

Output	Input															
	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	0	0	2	4	2	2	2	2	0	0	0	4	0	0	2
2	0	4	0	4	2	0	2	0	0	2	0	2	0	0	0	0
3	0	0	2	0	2	2	2	0	0	0	0	2	2	2	0	2
4	0	0	4	0	0	4	0	0	2	0	2	0	0	2	0	2
5	0	2	2	0	0	0	2	2	0	0	4	0	2	0	2	0
6	0	2	0	4	0	0	2	0	0	2	0	0	4	0	2	0
7	0	0	0	2	0	0	2	4	0	4	2	0	0	0	0	2
8	0	2	0	0	4	0	0	2	2	0	0	0	2	2	0	2
9	0	2	2	0	0	2	0	2	2	2	0	4	0	0	0	0
A	0	0	2	0	2	2	0	2	0	2	0	0	4	2	0	0
B	0	0	2	2	2	0	2	0	2	2	0	0	0	2	0	2
C	0	2	0	0	0	2	0	0	0	2	2	2	0	2	2	2
D	0	0	0	2	0	2	0	0	0	0	4	2	0	2	4	0
E	0	2	2	0	0	0	0	0	4	0	0	0	2	0	4	2
F	0	0	0	0	0	0	2	2	2	0	2	4	0	2	2	0

Table 4: Linear approximation table of the proposed left S-box.

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

Table 5: DU values of the left S-box.

—	7	7	6
6	7	8	7
7	8	6	6
6	6	5	8

Differential branch number refers to the number of branches required to determine the critical differentials through differential analysis. The branch number represents the number of bits observed during the differential analysis. The higher the branch number of the linear transformation in a cryptographic system, the stronger its resistance to differential analysis. The linear branch number is an indicator that expresses the linearity properties of a linear transformation in a cryptographic algorithm. It is defined as the expected number of output bits that differ when there is a change in the input. A higher linear branch number indicates a larger portion of the algorithm's linear transformation. This means that changing input bits will have a greater impact on more output bits. This implies a higher complexity of the linear transformation, making it more difficult for attackers to exploit it in cryptographic attacks.

Although maximum distance separable (MDS) structures have inherent advantages in the design of cryptographic algorithms, their implementation cost in terms of software and hardware is relatively high. Table 6 shows the XOR numbers required for linear layers using different methods, which limits their widespread adoption in lightweight cryptography. In designing the linear transformation of ALICA, a compromise approach is taken, utilizing a cost-effective strategy for linear transformation. The linear transformation L constructed in this paper possesses the following characteristics:

(1) The differential branch number of L is 8, and the linear branch number is 4

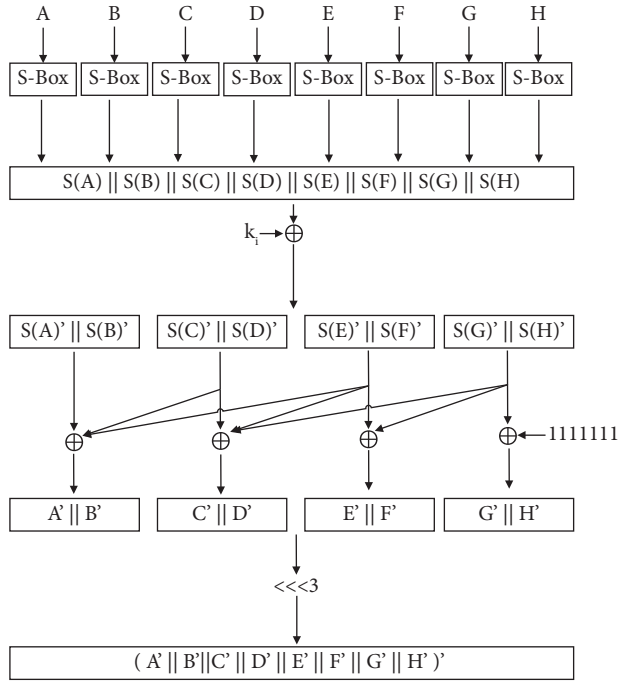

Figure 4: The structure of the F-function.

Input: Random 8-bit sequences M, N, P, and Q, subkey k_i.
$M\|N\|P\|Q = (M\|N\|P\|Q) \oplus k_i$
$M = M \oplus N \oplus P$
$N = N \oplus P \oplus Q$
$P = P \oplus Q$
$Q = Q \oplus 11111111$
$M\|N\|P\|Q = (M\|N\|P\|Q) \ll 3$

Algorithm 1: Linear components of F-function.

(2) The operations involved in L include XOR and bit shifting, making it easy to implement in both software and hardware with a relatively low implementation cost

2.5. The Subkey Generation Scheme of ALICA. The subkey generation scheme in a cryptographic algorithm refers to the process of generating the subkeys necessary for the algorithm. The ALICA cryptographic algorithm allows for the derivation of the entire set of subkeys for the data encryption process from a single computation of the key. Therefore, even though the purpose of this paper is to design a lightweight cipher, the subkey generation process is designed to ensure key diffusion during the encryption process and is more complex than the encryption process itself.

In the design of the ALICA cryptographic algorithm, subkeys are generated separately for the left-round function F_{left} and the right-round function F_{right} ALICA generates a total of 48 subkeys. The values of left-k are derived from k_{2r}, and the values of right-k are derived from k_{2r+1}, where r represents the iteration round number from 0 to $n-1$. The subkey generation scheme reuses the F-function component. The subkey input parameter of the F-function is set to 0, and the S-box used in the F-function is replaced with a key S-box. The design concept of the key S-box has been explained in the previous section and can be referenced in Figure 5. The specific key S-box is shown in Table 7.

Table 6: Required XOR numbers.

Block cipher	Linear layer	Cost
AES	MDS matrix	108 XORs
Piccolo	MDS matrix	52 XORs
Future	MDS matrix	52 XORs
ALICA	Bit permutation	7 XORs

The following section will describe in detail the process of generating subkeys from the key K. In the ALICA cryptographic algorithm, the key K is 128 bits, and each subkey k has a length of 32 bits. The specific subkey generation scheme is outlined in Algorithm 2.

Let Bitset $[i]$ be a value represented as $x_7x_6x_5x_4x_3x_2x_1x_0$, $x_i \in \{0, 1\}, 0 \le x \le 7$. The XOR function for Bitset $[i]$ can be expressed using the following formula:

$$\mathrm{XOR}\left(x_7x_6x_5x_4x_3x_2x_1x_0\right) = x_7 \oplus x_6 \oplus x_5 \oplus x_4 \oplus x_3 \oplus x_2 \oplus x_1 \oplus x_0. \tag{8}$$

3. Security Analysis of ALICA

3.1. Differential Cryptanalysis. By calculating the number of active S-boxes in the single-round function of the ALICA algorithm, we can use the program to determine the minimum number of active S-boxes in a complete encryption round, which consists of 24 rounds. The calculation results are shown in Table 8.

According to the second section of this paper, it is known that the maximum differential probability for both the left S-box and the right S-box in ALICA is 2^{-2}. By calculation, it is found that the maximum differential probability for 24 rounds of the ALICA cipher is $(2^{-2})^{108} = 2^{-216}$, which is much smaller than the time complexity required for brute-force key search, which is 2^{128}. Therefore, the complete round of the ALICA cipher is sufficient to resist differential analysis.

3.2. Linear Analysis. The ALICA cipher algorithm uses left and right S-boxes with a maximum linear bias of 2^{-2}. Therefore, applying the stack lemma, it can be concluded that the maximum linear bias probability for 24 rounds of the ALICA algorithm is $(2^{-2})^{108} = 2^{-216}$, which is smaller than the security boundary of 2^{-216}. Hence, the ALICA cipher algorithm with the full number of rounds is capable of resisting linear analysis.

3.3. Random Test of Subkey. The results of the NIST randomness tests, which are based on the calculation of the subkey sequence generated from the initial key, are presented in Table 9. The randomly tested number sequence passes all 10 out of 10 tests in each category, indicating excellent performance in various aspects without any significant deviations.

Frequency and Block Frequency Tests. These tests assess the balance between 0 and 1 bits in the random number sequence. A pass rate of 10/10 indicates that the sequence exhibits good balance and shows no apparent bias.

Cumulative Sums Test. This test checks for variations in the cumulative sums of the random number sequence. A pass rate of 10/10 suggests that the sequence performs well in this regard and does not show statistically significant deviations.

Runs and Longest Run Tests. These tests detect the presence of repeated sequences in the random number sequence. A score of 0.534146 indicates that there are relatively few repeated patterns in the sequence, and a pass rate of 10/10 suggests that there are no apparent repeated sequences.

Rank Test. This test evaluates whether the random number sequence is uniformly distributed within a specific range of values in a matrix. A score of 0.534146 indicates that the sequence performs well in this aspect.

FFT (Fast Fourier Transform) Test. This test assesses the distribution of frequency-domain features in a random number sequence. A score of 0.739918 indicates that the sequence also exhibits good randomness in the frequency domain.

Randomness security analysis is essential for assessing and determining whether a sequence is secure enough to protect sensitive information from attacks. Based on the comprehensive results, it can be concluded that the tested random number sequence has passed all NIST randomness tests, demonstrating a high level of randomness and security.

3.4. The Avalanche Effect of ALICA. The avalanche effect in cryptographic algorithms refers to the phenomenon where even a slight alteration in the input leads to a substantial and unpredictable alteration in the output. In cryptography, the avalanche effect is a highly crucial feature because it ensures the sensitivity and randomness of the algorithm to input changes, thereby enhancing the security of the encryption.

Using the key "00||01||02||03||04||05||06||07||08||;09||0A||0B||0C||0D||0E||0F," the text "00||00||00||00||00||00||00||00||00||00||00||00||00||00||00||01||00||00||00||01||00||00||00||01||00||00||00||02||00||00||00||02||...|x|00||00||00||7F||00||00||00||7F||00||00||00||7F," which consists of 128×96 bits of plaintext data, was encrypted. The ciphertext obtained is depicted in Figure 6. When modifying a single bit in the encryption key and employing the key "01||01||02||03||04||05||06||07||08||09||0A||0B||0C||0D||0E||0F" to encrypt the identical plaintext information, the resulting ciphertext is depicted in Figure 7. The computation of the dissimilar bit count between the two 128×96-bit ciphertext characters before and after yields a total of 6204 bits, with a calculated probability of alteration amounting to 50.48%. This observation suggests that the encryption algorithm demonstrates exceptional avalanche properties.

4. Performance of ALICA

HIGHT [20] is a lightweight encryption algorithm that South Korea adopted as a standard for lightweight cryptography. It provides a security threshold of 128 bits, similar to ALICA. HIGHT was designed specifically for 8-bit processors. However, with the majority of IoT devices now equipped with more advanced architectures and higher-bit processors, HIGHT's advantages are no longer optimal on the next-generation x86 platforms. On older ARM architecture processors, such as the Qualcomm Snapdragon 410 platform, HIGHT still retains its advantages. However, when comparing ALICA to Piccolo and uBlock on ARM, ALICA demonstrates a greater advantage. The initial design of uBlock relies on instruction acceleration on the processor platform. The comparison results shown in the experimental graph provide a more realistic representation of the underlying heterogeneous IoT device environment, without utilizing instruction acceleration. The performance of the four mentioned encryption algorithms on the ARM (Snapdragon 410) platform, all providing a 128-bit key security threshold, is illustrated in Figure 8. Similarly, the performance on the x86 (Intel i5 9500) platform is shown in

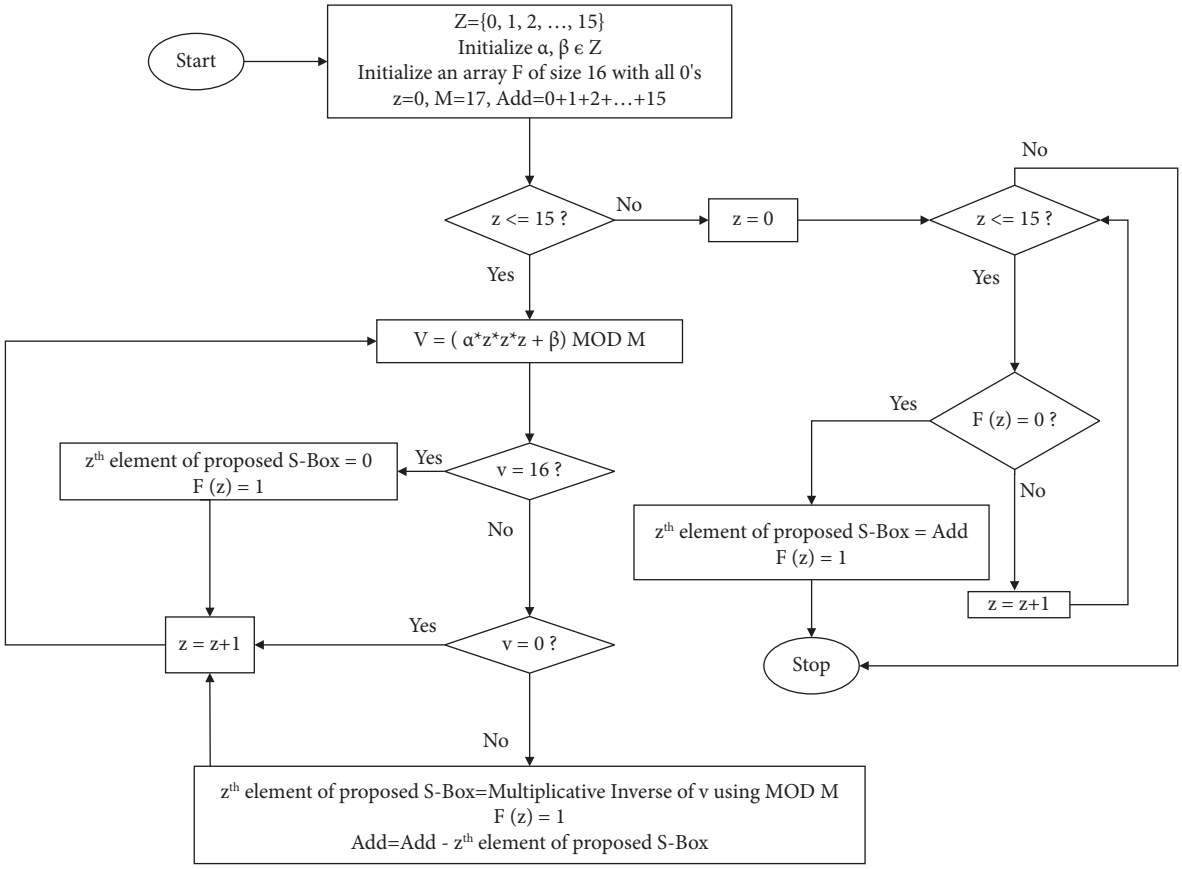

Figure 5: The structure of S-box generation.

Table 7: Left S-box of the ALICA cipher algorithm.

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

```
Input: key sequence grouping KEY, 0 ≤ i ≤ 15
Keys = new int [48]
for num in range (48):
   for i in range (2):
      index = (2 ∗ num + i) mod 16
      sbox_output = S_BOX[ (KEY [index] >>> 4)&0x0f ≪ 4|S_BOX[KEY [index] & 0x0f]
   KEY [index] = sbox_output
   xor_value = 0
   for i in range (16):
   if (XOR (KEY [i])):
      xor_value = (xor_value << 1)|0x01
      KEY [i] = (KEY [i] << 1)|0x01
   else:
      xor_value = xor_value << 1
      KEY [i] = KEY [i] << 1
   new_key = (xor_value << 16)|(KEY [(2 ∗ num) mod 16] << 8 & 0xff00)|(KEY [(2 ∗ num) mod 16] & 0xff)
   Keys [num] = new_key
Output: Keys
```

Algorithm 2: Subkey generation algorithm.

Table 8: Number of differential active S-boxes of ALICA.

R	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
S-Box	0	8	12	16	20	24	28	32	40	44	48	52	56	60	64	72	76	80	84	88	92	96	104	108

Table 9: Random test table for subkey sequence of ALICA.

C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	P value	Proportion	Statistical test
3	2	2	0	0	1	1	0	0	1	0.350485	10/10	Frequency
1	0	2	1	2	1	0	3	0	0	0.350485	10/10	Block frequency
4	1	2	0	0	0	1	1	0	1	0.122325	10/10	Cumulative sums
3	2	1	0	1	0	0	3	0	0	0.122325	10/10	Cumulative sums
3	2	1	1	1	1	1	0	0	0	0.534146	10/10	Runs
3	0	0	0	1	1	2	1	1	1	0.534146	10/10	Longest run
2	0	2	1	0	1	2	2	0	0	0.534146	10/10	Rank
1	1	1	3	0	1	1	1	0	1	0.739918	10/10	FFT

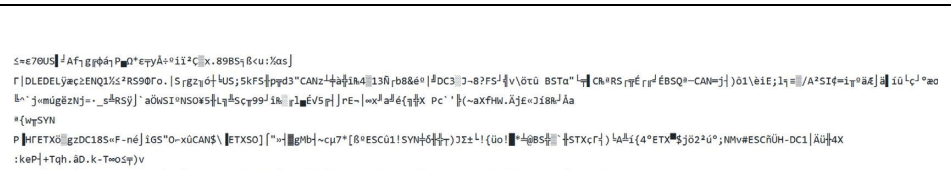

Figure 6: Ciphertext by original key.

Figure 7: Ciphertext by changed key.

Figure 9. ALICA, similar to uBlock and Piccolo, incorporates an S-box substitution during the algorithm design, whereas HIGHT utilizes an ARX design logic. It can be anticipated that on the x86 platform, ALICA's performance is on par with HIGHT or even superior. The memory usage for storing S-boxes and round constants in the cryptographic algorithm is depicted in Figure 10. In the future, the algorithm employs MDS matrices and round constants, which results in consuming twice the memory compared to other lightweight block cipher algorithms. ALICA's memory usage during the encryption process is comparable to that of other lightweight encryption algorithms.

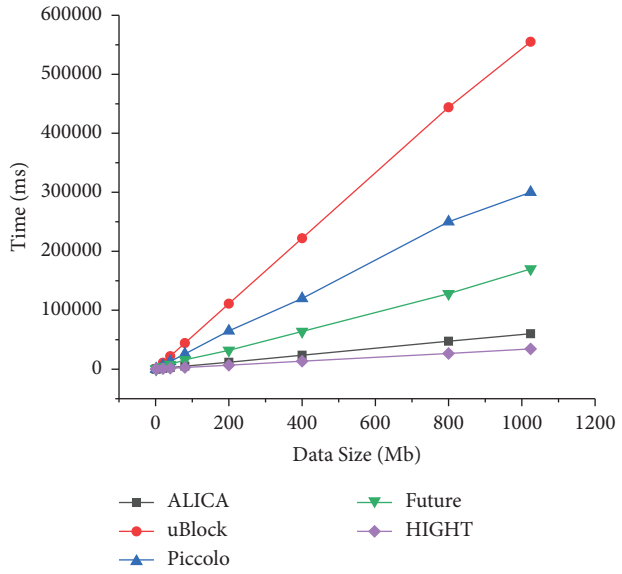

Figure 8: Cryptographic algorithm performance of ARM.

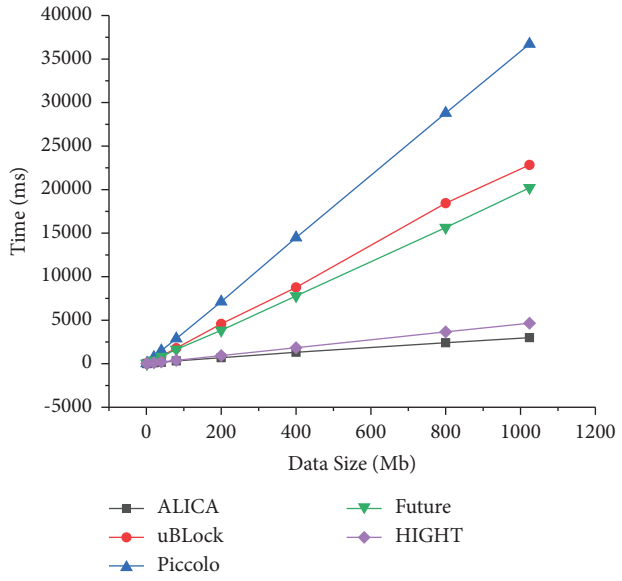

Figure 9: Cryptographic algorithm performance of x86.

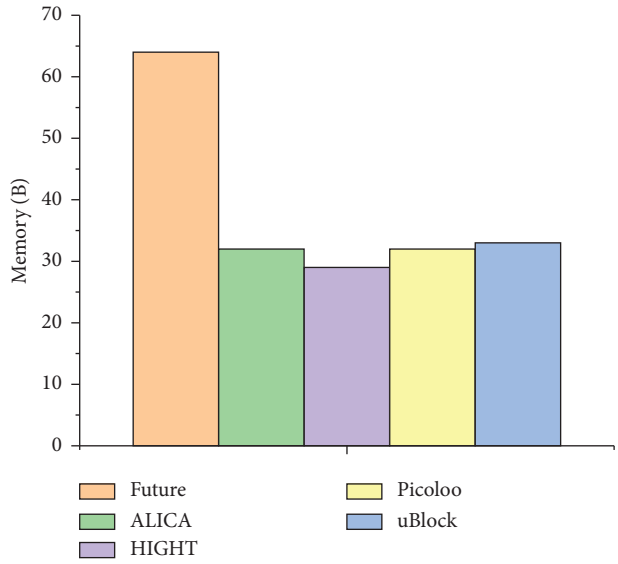

Figure 10: Memory size.

5. Conclusion

This paper presents a summary of recent literature on lightweight block ciphers. By analysing the components of lightweight block ciphers, the paper proposes a new lightweight cipher called ALICA. ALICA employs a generalized Feistel structure with a key size of 128 bits and a block size of 96 bits. The overall structure of ALICA enables efficient hardware implementation and minimal software memory usage through the reuse of the F-function. ALICA demonstrates good deployment and applicability on devices with limited hardware resources and strict software size constraints. The security of ALICA is evaluated by calculating the success probability of differential attacks based on the number of active S-boxes after iterations. The calculated security level of ALICA is significantly higher than the threshold for brute-force attacks on lightweight ciphers. In terms of performance, this paper compares ALICA with the HIGHT cipher algorithm and demonstrates the advantages of the proposed solution on high-end processors. The next step in this research is to further investigate the design and application of various lightweight cipher algorithms on constrained devices. Additionally, specific research should be conducted on the additional security aspects of the proposed cipher mentioned in this paper.

Data Availability

The original data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (No. 62162020) and the Science Project of Hainan University (No. KYQD(ZR)20021).

References

[1] Y. Ou and L. Li, "Research on a high-order AES mask anti-power attack," *IET Information Security*, vol. 14, no. 5, pp. 580–586, 2020.

[2] D. Jovan, "cryptanalysis of alleged A5 stream cipher," *Advances in Cryptology—EUROCRYPT'97*, vol. 1233, pp. 239–255, Springer, Cham, Switzerland, 2001.

[3] W. Johan, "Design principles of the KASUMI block cipher," in *Proceedings of the Helsinki University of Technology Seminar on Network Security*, Helsinki, Finland, December 2000.

[4] C. H. Lim and T. Korkishko, "mCrypton–a lightweight block cipher for security of low-cost RFID tags and sensors," *International workshop on information security applications*, vol. 3786, pp. 243–258, 2005.

[5] K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, and T. Shirai, "Piccolo: an ultra-lightweight blockcipher," in *Proceedings of the Cryptographic Hardware and Embedded Systems–CHES 2011: 13th International Workshop*, vol. 6917, pp. 342–357, Nara, Japan, September 2011.

[6] J. Guo, T. Peyrin, A. Poschmann, and M. Robshaw, "The LED block cipher," in *Proceedings of the Cryptographic Hardware and Embedded Systems–CHES 2011: 13th International Workshop*, vol. 6917, pp. 326–341, Nara, Japan, September 2011.

[7] S. Tomoyasu, K. Minematsu, S. Morioka, and E. Kobayashi, "Twine: a lightweight, versatile block cipher," *ECRYPT Workshop on Lightweight Cryptography*, vol. 2011, Springer, Berlin, Germany, 2011.

[8] W. Wu and L. Zhang, "LBlock: a lightweight block cipher," in *Proceedings of the Applied cryptography and network security: 9th international conference, ACNS 2011*, vol. 6715, pp. 208–225, Nerja, Spain, June 2011.

[9] B. Julia, A. Canteaut, G. Tim et al., "PRINCE–a low-latency block cipher for pervasive computing applications," in *Proceedings of the Advances in Cryptology–ASIACRYPT 2012: 18th International Conference on the Theory and Application of Cryptology and Information Security*, vol. 7658, pp. 208–225, Beijing, China, September 2012.

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

[11] S. Banik, S. K. Pandey, T. Peyrin, Y. Sasaki, S. M. Sim, and Y. Todo, "GIFT: a small present: towards reaching the limit of lightweight encryption," in *Proceedings of the Cryptographic Hardware and Embedded Systems–CHES 2017: 19th International Conference*, vol. 10529, pp. 321–345, Taipei, Taiwan, September 2017.

[12] W. Wu, L. Zhang, Y. Zheng, and L. Li, "The block cipher uBlock," *Journal of Cryptologic Research*, vol. 6, pp. 690–703, 2011.

[13] V. G. Kiran Kumar and C. Shantharama Rai, "Design and implementation of novel BRISI lightweight cipher for resource constrained devices," *Microprocessors and Microsystems*, vol. 84, pp. 104–267, 2021.

[14] H. Kim, Y. Jeon, G. Kim et al., "A lightweight block cipher with efficient higher-order masking software implementations," *International Conference on Information Security and Cryptology*, vol. 12593, pp. 99–122, Springer, Cham, Switzerland, 2021.

[15] K. C. Gupta, S. K. Pandey, and S. Samanta, "FUTURE: a lightweight block cipher using an optimal diffusion matrix," *International Conference on Cryptology in Africa*, vol. 13503, pp. 28–52, Springer, Cham, Switzerland, 2022.

[16] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "Cryptanalysis of Ascon," in *Proceedings of the Topics in Cryptology--CT-RSA 2015: The Cryptographer's Track at the RSA Conference 2015*, vol. 9048, pp. 371–387, San Francisco, CA, USA, April 2015.

[17] R. Mishra, M. Okade, and K. Mahapatra, "Optimized s-box architectures of present cipher for resource constrained applications," in *Proceedings of the 2020 IEEE International Symposium on Smart Electronic Systems (ISES)*, pp. 304–307, Chennai, India, December 2020.

[18] A. Zahid, M. Arshad, and M. Ahmad, "A novel construction of efficient substitution-boxes using cubic fractional transformation," *Entropy*, vol. 21, no. 3, p. 245, 2019.

[19] A. F. Webster and S. E. Tavares, "On the design of S-boxes," *Advances in Cryptology—Crypto'85*, vol. 218, pp. 523–534, Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[20] B. J. Mohd, T. Hayajneh, Z. A. Khalaf, and K. M. Ahmad Yousef, "Modeling and optimization of the lightweight HIGHT block cipher design with FPGA implementation," *Security and Communication Networks*, vol. 9, no. 13, pp. 2200–2216, 2016.