Iraqi Journal for Computer Science and Mathematics

Journal Homepage: http://journal.esj.edu.iq/index.php/IJCM
e-ISSN: 2788-7421 p-ISSN: 2958-0544

SUMER: A New Family of Lightweight and Traditional Block Ciphers with Multi-Modes

Omar A. Dawood[1]

University of Anbar, College of Computer Science and Information Technology, Anbar, Iraq

*Corresponding Author: Omar A. Dawood

DOI: https://doi.org/10.52866/ijcsm.2024.05.03.011
Received February 2024; Accepted April 2024; Available online July2024

ABSTRACT: With the recent increase in the risks and attacks facing our daily lives and digital environment around us, the trend towards securing data has become inevitable. Block ciphers play a crucial role in modern crypto-applications such as secure network storage and signatures and are used to safeguard sensitive information. The present paper develops a new variant of the symmetric model called SUMER family ciphers with three equivalent modes: lightweight, conventional (traditional), and extended ciphers. SUMER name belongs to one of the oldest civilizations in Mesopotamia and stands for **S**ecure **U**niversal **M**odel **of** **E**ncryption **R**obust Cipher. The SUMER cipher is based on a simple and robust symmetric structure and involves solid algebraic theories that completely depend on the Galois Field GF(2^8). SUMER cipher is designed to work with two involutional structures of the Substitution-Permutation Network (SPN) and Feistel structure. These two involutional structures mean that the same algorithm is used for the encryption and decryption process, and only the algorithm of the ciphering key is used in reverse order in both structures. The SUMER lightweight structure is an elegant mode that does not need building an S-Box that requires a large amount of memory and a number of electronic logical gates as S-Box construction has been canceled and replaced by the on-fly computation clue, which does not need a reserved memory for building S-Box. SUMER family ciphers also can work in a traditional mode or as an extended mode with high margin security. This family of ciphers is applicable with multi-modes of various utilizations. The proposed ciphers are designed to be byte-oriented, showing good evaluation and results under several measurement tests for speed, time implementation, and efficiency.

Keywords: Block Cipher, Symmetric Cipher, Lightweight Cipher, Substitution and Permutation Network (SPN), Feistel Structure (FS), Involutional Structure, AES Cipher.

1. INTRODUCTION

The block cipher is considered one of the freshest topics in symmetric ciphers that has attracted the attention of a lot of researchers, designers and cryptanalysts around the world, for helping to solve several security problems of civil and military applications. Lightweight block ciphers are designed to be an appropriate solution and alternative to traditional block cipher with symmetric keys. Lightweight ciphers are basically based on efficiency and are dedicated to limited power and constrained devices, such as the Internet of Things (IoT), embedded devices, network sensors, and RFID tags [1]. The main difference between conventional and lightweight block ciphers lies in their design roadmap strategy and intention applications. The recent design orientation is towards constructing a fast and an elegant cipher with high performance in software implementation and constrained hardware of lightweight features [2]. Furthermore, the conventional (traditional) and lightweight block ciphers work to yielding secure ciphering designs with diverse modes to fulfil various needs and constraints. Lightweight block ciphers give precedence to speed and efficiency while traditional block ciphers focus on security and strength factors Thus, regarding the speed development for cryptanalysis and high-processing computers, it is time to dedicate a stronger cipher to defeat newly effective attacks [3].

The lightweight design requires neither a more complicated algebraic design nor a sophisticated building structure in its construction [4]. Often lightweight ciphers should work with a little E-gates design and do not need a very high-level security scheme. The basic work of the lightweight ciphers is to secure information in a constrained environment with low-cost requirements. The primary goal is to create a secure cipher with minimal hardware resources and efficient software implementation. Several conventional ciphers are unsuitable to adopt in embedded devices due to their higher power consumption and cost hardware implementation [5]. Although there are some concerns about the design of lightweight algorithms due to their moderate level of security,it is very necessary to understand that the light design does not produce weakness, nor is it a weak or non-robust cipher. Conversely, there are only fewer characteristics over traditional symmetric ciphers for the lightweight cipher, represented by an elegant internal structure

*Corresponding author: omar-abdulrahman@uoanbar.edu.iq

and efficiency [6]. Several cryptographers and designers all over the world work hard to innovate new lightweight algorithms and have produced many projects with distinct trends: lightweight stream ciphers, lightweight block ciphers, lightweight Message Authentication Code (MACs), lightweight Pseudo Random Number Generator (PRNG), and lightweight hash functions. There are minimal disparities and no explicit contradictions between the lightweight and the traditional cipher in terms of design perspective, as both types share the same concepts and strategies, except that the light model with fewer properties and less cost may produce minimal security margins [7]. Recently, numerous modern lightweight ciphers have surfaced, working with diverse structures and various mathematical and algebraic foundations, such as HIGHT [8], SEA [9], PRESENT [10], CLEFIA [11], Piccolo [12], KLEIN [13], DESL [14], mCrypton [15], LED-64 [16], KATAN, and KATANTAN [17]. The author mentioned all these lightweight algorithms to ensure that the readers are fully familiar with the design aspects of different types of algorithms and to stay up to date with the developed models to get a substantial scientific background about the newly upgraded algorithms compared with the proposed model.

This paper is outlined as follows: In Section 2, the motivation of the paper is presented. In Section 3, we discuss the literature survey with most related family block ciphers. In Sections 4 and 5, the proposed structure is generally described. The design strategy and the internal state operations are clarified in Sections 6 and 7. In Section 8, the experimental results of the implementation are produced. In Section 9, we conclude with the main finding and insight of the study.

2. MOTIVATION

The major motivation behind the proposed design of SUMER family ciphers is to build an iterated product of secure ciphers that satisfy the current demands and future security requirements and to submit secure and efficient ciphers with multi-mode for IoT applications and other restricted resources. The intended strategy was to develop a secure and fast block cipher with as minimal a number of electronic gates as possible. The currently introduced ciphers are desirable for many future pervasive applications with real-time and security needs. Our contribution is based on the idea of designing a new family of block ciphers with enhanced and acceptable security layer and flexibility factor with speed and robust construction at the same time. The SUMER cipher consists of repeated fixed stages along a certain structure with a round transformation parameterized by the round key. The round keys are derived from the ciphering key by means of the key schedule. SUMER cipher has adopted a new orientation in the lightweight family cipher. The proposed model tackles with changeable block sizes and variable cipher key lengths similar to the conventional block cipher but with a slightly lighter internal mathematical foundation. The proposed model can encrypt the plaintext with two different structures Feistel structure (FS) and Substitution Permutation Network (SPN) structure with three modes.

3. LITERATURE SURVEY

Loong is a proposed multi-cipher model by Bo-Tao Liu et al., 2019, a family of involutional lightweight block ciphers that are efficient in resource-constrained environments such as IoT and embedded systems applications. Loong is invented based on the new SPN structure with involutional of round transformation components to create a highly balanced symmetrical design. The symmetrical design is created by providing the same process for encryption and decryption and reducing the code and E-circuits for software and resources, with only a difference in the usage of the reverse order of the round constant. An involutional structure has been adopted, which can reduce resource consumption. The Loong round function uses two SubCells operations to create a more distinct S-Box with a lightweight design. The Loong structure could require around 1766 Gate Equivalents (GE) per bit using a 128-bit key and is lower than required by PRESENT and Piccolo. In addition, Loong is secure against differential and linear attacks [18].

High execution speed was the goal of Gregor Leander et al., 2022, which resulted in proposing SPEEDY cipher that acts as a family of lightweight and efficient block ciphers. SPEEDY is engineered as a block cipher with a low latency that is dedicated toward Metal–Oxide–Semiconductor (CMOS) hardware that operates in a constrained environment. The design includes a 6-bit S-Box with a 192-bit block and a key size that can achieve the security level of 128 bit with six rounds; however, the security level increases to match the 192 bit when operated with seven rounds. The latency evaluation showed that SPEEDY with five or six rounds and 192-bit size has a lower latency compared to other algorithms such as PRINCE, MANTIS, etc. The implementation speed of SPEEDY did not compromise its security, especially with using seven rounds that provide full security against cryptanalysis [19].

Using multiple quasigroups to design a family of block ciphers was proposed by Kumar, Umesh, and V. Ch Venkaiah, 2022. The algebraic structure of quasigroups includes set and binary operations to satisfy certain properties and is utilized in the encryption and decryption operations to increase the confusion and diffusion. The proposed approach uses a maximum of 16! optimal quasigroups of order 16, which creates an S-Box with the highest degree of non-linearity and lower linearity differential. These characteristics provide the cipher with protection against various cryptanalytic attacks such linear and differential attacks. The proposed cipher was evaluated and passed the National Institute and Standard Technology NIST-STS tests [20].

Roberto Avanzi et al. proposed two versions (QARMA and QARMAv2) of a family tweakable lightweight block cipher in 2017 and 2023. The QARMAv2 is the newest version, which is a redesign and improvement over QARMA by enhancing security and allowing longer tweaks. The tweakable cipher is open to modification by changing the tweak value that is used along with the key to compute the permutation. Long tweaks can introduce a higher security bond by reducing tweak reuse. QARMAv2 builds on SPN structure by allowing two different block sizes, 64 bits (with key size 128) and 128 bits (with key sizes 128, 192, and 256). It also includes employing a diffusion layer to increase the resistance against differential and linear attacks. Furthermore, the low memory requirement and small code size made QARMAv2 a lightweight block cipher appropriate for a constrained-resource environment [21].

4. THE STRUCTURE OF SYMMETRIC BLOCK CIPHER

There are two main famous structures in the block cipher design: Feistel and SPN. These two structures can be applied as a lightweight or a traditional cipher, each of which has pros and cons, and the designer can design his algorithm according to the structure that he sees as convenient and fit for his needs. Therefore, both structures will traverse quickly.

4.1 FEISTEL NETWORK STRUCTURE

The Feistel network structure is the oldest structure invented by IBM cryptographer Horst Feistel, and its first use was in the Lucifer cipher, then altered to the Data Encryption Standard (DES) cipher. The input of plaintext in this structure is separated into two parts, left and right (L0 and R0), which are then connected. Feistel structure employs an arrangement of rounds that divide the entered block into two parts; one part should change the opposite part and then those two parts are swapped alternatively [22]. The round transformation in the Feistel network uses the same algorithm and the same code for encryption and decryption processes only the reversing order for the ciphering key. Feistel ciphers, in most cases, activate the complexity of attacks, which increases dramatically with the number of rounds. Some designers avoid using the Feistel structure due to the weak one-half of the Feistel round. As one-half of the Feistel round remains constant without exchange, this issue may be considered a weak point from the cryptographic point of view. Nevertheless, some excellent features make it an ideal structure and can be summed up by speed and round symmetry that needs no inverse structure. It is a simple structure with a straightforward implementation and provides excellent protection against structural attacks. The most well-known algorithms of Feistel structures are DES, IDEA, Blowfish, Twofish, RC6, MARS, and Camellia ciphers. In addition to the Feistel structure, several derived cipher structures involve balanced Feistel, unbalanced Feistel, and generalized Feistel structures [23].

4.2 SUBSTITUTION PERMUTATION NETWORK (SPN)

The SPN is another structure that is adopted in the design of many ciphers. It accepts the entry data as initial input and then splits it into numerous small bulks of a specific size. The round transformation of SPN is composed of three different invertible layers: a Non-Linear layer guaranteeing the confusion property, a Linear layer responsible for diffusion over several rounds, and the Key-addition layer comprises the XORed addition between the state array and the ciphering key [24]. This structure works with two algorithms, one for the encryption process and the other for the decryption process, in addition to the key generation algorithm. The parallelism notation in the SPN structure acts as the basic drawback for hardware implementations in opposition to what occurs in the software implementation. The most well-known algorithms that use SPN are the Rijndael, Serpent, and Square ciphers, etc. [25].

5. DESCRIPTION OF SUMER CIPHER

SUMER cipher is considered a general-purpose algorithm that is designed with the scalable structure of multi-mode for multi-utilizations. SUMER family ciphers produce several options for the users to select the suitable cipher modes according to the required security level and their application sensitivity. The proposed cipher design allows several compacted layers to be implemented recursively with as little as possible of the design area on the circuit board. The basic structure depends on the relative importance of features that involve encryption speed, key setup, working memory, hardware cost, low power consumption, less execution time, fewer code lines, low design cost, high throughput used, and other implementation parameters. The result certainly will be a highly flexible algorithm with a sufficient protection layer against sophisticated differential power analysis attacks and fault attacks. The principal innovation property in this model is the consistent structure that uses symmetric operations and abbreviated stages in the Encryption/Decryption process. The proposed cipher design combines a set of outstanding characteristics for different models in one algorithm. The proposed family ciphers encrypt different lengths of data with different sizes of ciphering keys according to the variable number of rounds, as shown in Table 1.

Table 1. SUMER Family Ciphers Characteristics

The Algorithm Mode	Structure	Plaintext Length	State Array	Key Length	Size-Key Array	Rounds
Involutional Lightweight SUMER-SPN	SPN	128-bit	4*4	128-bit	4*4	12
		128-bit	4*4	192-bit	4*6	14
		128-bit	4*4	256-bit	4*8	16
Extended SUMER-Feistel Structure	Feistel	256-bit	4*8	256-bit	4*8	10
		256-bit	4*8	384-bit	4*12	12
		256-bit	4*8	512-bit	4*16	14
Traditional SUMER – SPN	SPN	128-bit	4*4	128-bit	4*4	10
		128-bit	4*4	192-bit	4*6	12
		128-bit	4*4	256-bit	4*8	14

The proposed ciphers work with the involutional SPN structure of the lightweight mode and the other one with the uninvolutional SPN structure of the traditional mode. The round transformation of the SUMER lightweight mode consists of four main invertible stages that are implemented directly without taking care of their inverse operations. For instance, the first stage is built without a need to design an S-Box table. Thus, we get out of the familiar context, the non-linear layer depends on the invertible of a self-inverse affine equation with the self-inverse of a palindromic constant vector, which means that the same equation and the same constant vector are used in encryption and decryption processes, so there is no need for the inverse calculation. The computation value will be on-fly manner throughout the entry values to this stage without a need to own the S-Box tables that need at least 512 bytes of reserved memory for their indexed values. A knight shifting transformation is the second stage, which uses the same shifting steps in forward/backward operations. An inverse shifting process is also not needed as it has a self-inverse of knighting movements. The P-Box acts as the third stage in the proposed cipher selected as a new Maximum Distance Separable (MDS) of self-inverse property. The P-Box is selected with an involutional property to be convenient to the proposed structure with the same linear equation in a matrix form in forward/backward operations. The addition operation performed by XORing the ciphering key with the state array represents the last stage in the round transformation. Fig 1 shows the lightweight SUMER mode of the SPN structure. The traditional SUMER mode of SPN structure is XORed with the ciphering key stated in Fig 2 and the Extended SUMER mode with the Feistel structure of the 256-bit encrypted data as clarified in Fig 3.

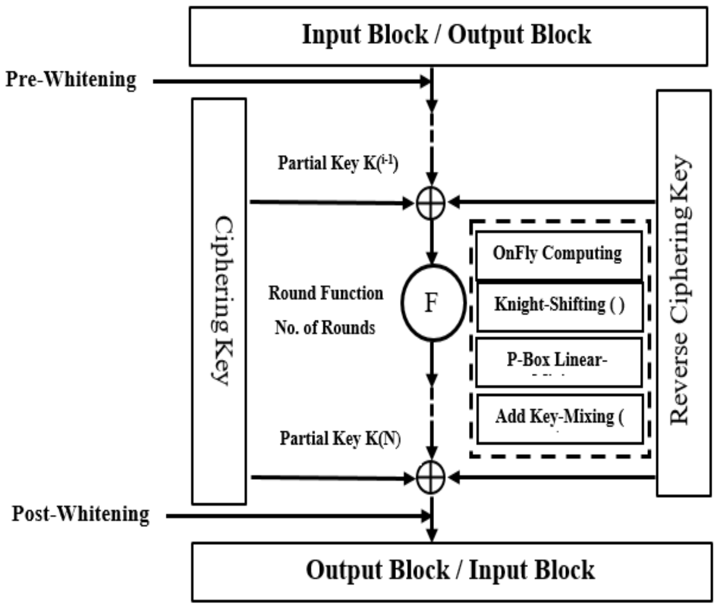

FIGURE 1. Lightweight of SUMER SPN Structure XORed with the Ciphering Key

The round transformation of the SUMER cipher of traditional mode also uses the same four main invertible stages implemented in lightweight mode. These stages are implemented directly with an involutional property except the S-Box stage as the stage does not use on-fly computation clue and needs the backward of an inverse operation such as the Advance Encryption Standard (AES) cipher as stated in Fig 2. The non-linear stage comprises the forward S-Box table of 256 byte and its inverse S-Box table of 256 byte. For the extended SUMER Feistel mode that accepts data of 256 bit and is separated into two parts (left, right) each with 128 bit, the entry data to the main Function (F) will be 128 bit for each round and the resultant is XORed with the second part of 128 bit and the ciphering key of 256 bit. In each round, the complex Function (F) is moved from right to left and vice versa to cover both the parts of the Feistel structure. The non-linear stage also comprises the forward S-Box table of 256 byte and its inverse S-Box table of 256 byte instead of the on-fly computation stage that exists in lightweight modes.

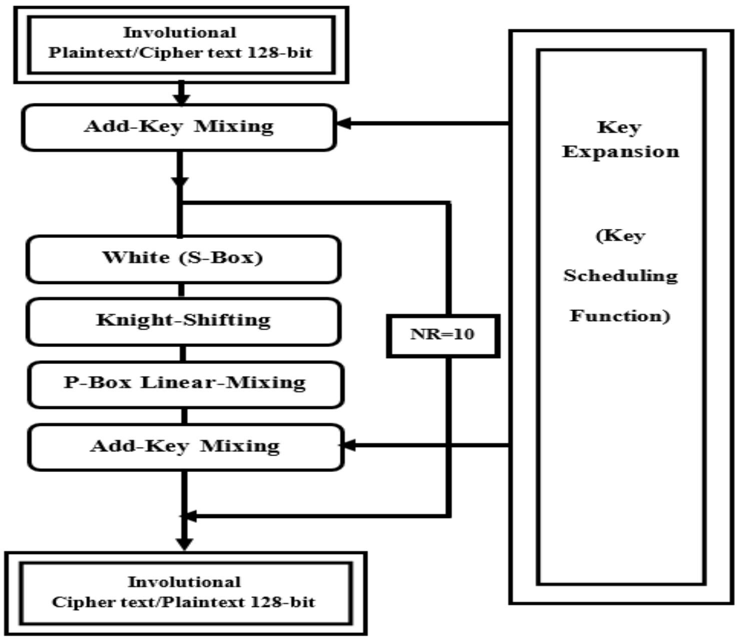
Involutional Plaintext/Cipher text 128-bit
Add-Key Mixing
White (S-Box)
Knight-Shifting
P-Box Linear-Mixing
Add-Key Mixing
NR=10
Key Expansion
(Key Scheduling Function)
Involutional Cipher text/Plaintext 128-bit

FIGURE 2. Traditional of SUMER SPN Structure

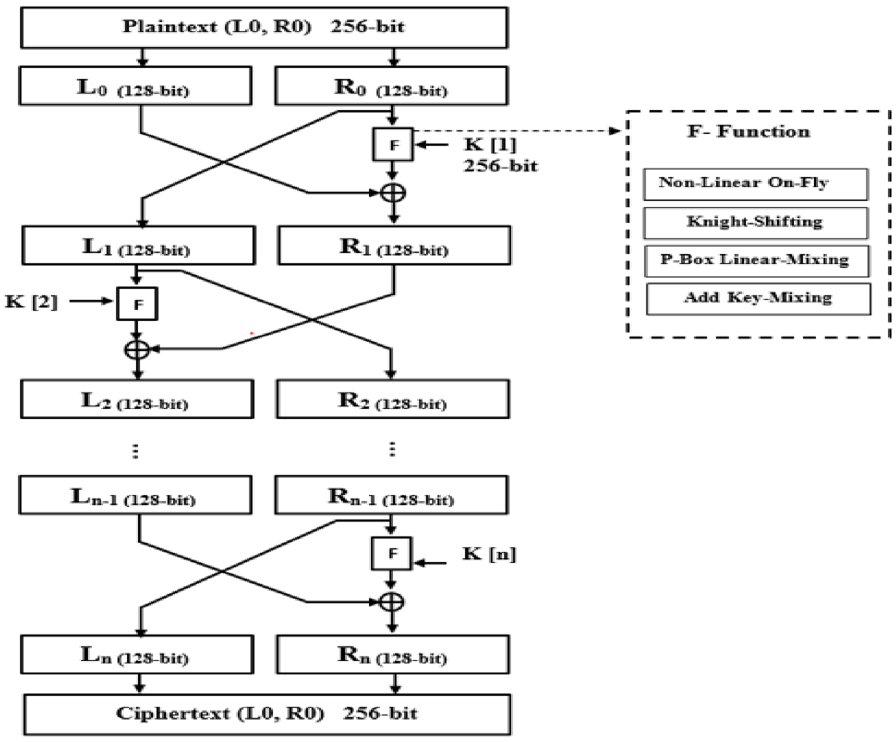

FIGURE 3. Extended of SUMER Feistel Structure

6. DESIGN GOALS AND STRATEGY

The principal goal and motivation behind the proposed cipher are to design and construct an elegant and lightweight block cipher subject to the standardization of lightweight criteria, along with satisfying the tractable implementation features in the limited software and the hardware areas without sacrificing either speed or security as much as possible. The basic idea behind the present design is to provide as low an implemented area in the hardware mapping board as possible with a small amount of electronic logical circuits. Designing an accepted, secure cipher roughly similar to the conventional cipher but with a lightweight flavor and characteristics is the other objective. Designing a lightweight cipher becomes a real demand in large-scale applications because of the future use and the urgent need of current requirements. The world has become more in need of lightweight and fast applications that require suitable block cipher algorithms in ubiquitous computing devices, which are assumed to be extensive in our daily life. It represents a departure from general-purpose applications to restricted device applications. As a result, the standard ciphers may be considered insufficient for lightweight applications and IoT aspects. IoT services and applications comprise a wide range of new different technologies that are increasingly used. So the proposed cipher is needed in many restricted environments, e.g., RFID tags, sensor networks, smart cards, tablets, and smartphones in addition to other new applications related to transposition, the health sector, mobile companies, industry tracking, etc. It is known that increasing the block size, key length, and number of rounds leads to an increased security margin. Therefore, the proposed cipher is designed to balance between the lightweight cipher and traditional cipher modes to be an effective model with multiple choices. The user has the option to choose which cipher mode can be used in his utilization, either lightweight or traditional cipher, according to the data sensitivity and the requirements.

7. ROUND INTERNAL OPERATIONS

The initially entered plaintext is treated as the matrix of numbers that are XORed with the secret ciphering key to produce intermediate ciphertext. At the beginning and the end of an algorithm, the key addition process is implemented according to the whitening concepts. The state array can be pictured as a square array of 4*4 dimensions, which is manipulated in four iterated basic stages. The same main stages of the round transformation are used in all categories modes as follows:

7.1 NON-LINEAR STAGE

This main stage in the round transformation affects the whole algorithm design modes. The user has two options for selecting the S-Box type. If the user selects the lightweight cipher mode, then the algorithm will involve On-Fly-non-linear mathematical computation that is not needed for building S-Box, and consequently less electronic gates will be needed. Otherwise, if the user selects the traditional cipher mode, then the algorithm will contain two tables of S-Boxes that require at least 512 byte for their construction in forward and backward steps like the AES cipher. The round transformation in the lightweight and traditional cipher modes has no huge difference in their internal design, but only in the first stage of S-Box design. The others stages of the round transformation remain unchanged in all cipher modes.

A. On-Fly Computation

This is the first stage of the data transformation process in SUMER cipher lightweight mode. On-fly computation considers a new prominent change in which a new output byte replaces each input byte throughout on-fly computations. The basic notation behind this stage is to discard and eliminate the S-Box table, which requires a reserved memory, and replace it with an invertible self-inverse of non-linear affine mapping equation. Then XORed with a self-inverse constant vector of value (A5) (66), (C3), or (99) in hexa-notation can be used in encryption and decryption processes without needing an inverse vector as stated in Equation (1). The (A5) vector (10100101) is selected as a palindromic value to be compatible with the self-inverse of affine mapping in both directions of encryption and decryption operations. The core idea for the nature work of this stage starts by taking the affine multiplication for each entry value in the state matrix XORed with (A5) vector sequentially. Taking the multiplicative inverse, the outcome values get the modular arithmetic under any selected irreducible polynomial of order eight from the existing 30 irreducible equations. The resultant value in the next step again maps the same affine multiplication XORed with the same (A5) vector as stated in Equation (2). The main notation behind the repeating multiplication is to provide full confusion and to apply high non-linearity on each entry value of any non-zero byte where x € Galois Field GF(2^8), and it is substituted by the following transformation:

$$Y = AX^{-1} \oplus b \qquad (1)$$

$$Y' = AY \oplus b \qquad (2)$$

Where A represents 8*8 of a constant affine matrix transformation and b represents the palindromic constant vector (A5) of 8*1 that increases the complexity of the reversing operation and makes the analytical training as hard as possible. The inverse operation is implemented by repeating the same steps of the forward operation with the same sequence that can be summarized as follows:

1. Takes the Affine Transform Multiplication XORed with Constant Vector (A5) according to Equation (3).
2. Take the Multiplicative Inverse under any irreducible polynomial for example $x^8 + x^5 + x^4 + x^3 + x^2 + x + 1$, as shown in Table 2.
3. Repeat (1).

Ex. According to 100011011 irreducible polynomial
On-Fly Forward Operation
Input value (B7)
Affine (B7) = E2 $\oplus$ V(A5) = 47
M-Inverse (47) = FC
Affine (FC) = A9 $\oplus$ V(A5) = 0C
The On-Fly Backward Operation
Input value (0C)
Affine (0C) = 59 $\oplus$ V(A5) = FC
M-Inverse (FC) = 47
Affine (47) = 12 $\oplus$ V(A5) = B7

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1&1&1&0&0&1&1&0 \\ 0&1&1&1&0&0&1&1 \\ 1&0&1&1&1&0&0&1 \\ 1&1&0&1&1&1&0&0 \\ 0&1&1&0&1&1&1&0 \\ 0&0&1&1&0&1&1&1 \\ 1&0&0&1&1&0&1&1 \\ 1&1&0&0&1&1&0&1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \qquad (3)$$

$$\begin{bmatrix} 1&1&1&0&0&1&1&0 \\ 0&1&1&1&0&0&1&1 \\ 1&0&1&1&1&0&0&1 \\ 1&1&0&1&1&1&0&0 \\ 0&1&1&0&1&1&1&0 \\ 0&0&1&1&0&1&1&1 \\ 1&0&0&1&1&0&1&1 \\ 1&1&0&0&1&1&0&1 \end{bmatrix} * \begin{bmatrix} 1&1&1&0&0&1&1&0 \\ 0&1&1&1&0&0&1&1 \\ 1&0&1&1&1&0&0&1 \\ 1&1&0&1&1&1&0&0 \\ 0&1&1&0&1&1&1&0 \\ 0&0&1&1&0&1&1&1 \\ 1&0&0&1&1&0&1&1 \\ 1&1&0&0&1&1&0&1 \end{bmatrix} = \begin{bmatrix} 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \end{bmatrix}$$

B. White S-Box Design

The SUMER cipher of the traditional mode uses the white S-Box design instead of on-fly computation that is constructed according to the design steps of AES S-Box. The design procedure starts by taking the multiplicative inverse according to the certain irreducible polynomial for each value in the table as stated in Table 2 and then mapping the same affine transform in Equation (3) is then XORed with the same palindromic constant vector (A5) in hexa representation. The result of these mathematical operations will be a new forward S-Box, as shown in Table 3.

Table 2. The Multiplicative Inverse according to Irreducible Polynomial of $x^8 + x^5 + x^3 + x + 1$

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	01	E6	95	E6	DF	BB	A4	FA	85	C8	55	AC	CE	52	69
1	7D	27	D7	F8	64	59	BF	A3	56	50	67	9A	29	33	A1	98
2	AB	91	86	E8	FE	E1	7C	11	32	1C	B9	30	CA	76	C4	3D
3	2B	B8	28	1D	A6	B1	4D	3F	81	61	8C	5A	C5	2F	4C	37
4	C0	F4	DD	44	43	DC	74	FC	7F	8F	E5	C6	3E	36	9D	DA
5	19	57	0E	68	C9	0B	18	15	65	15	3B	8D	62	97	8B	6F
6	80	39	5C	96	14	58	9B	1A	53	0F	CD	D9	B3	9E	8A	5F
7	D5	F2	A5	06	46	FD	2D	CB	F7	E2	82	ED	26	10	8E	48
8	60	38	7A	EC	FB	09	22	E9	B4	C2	6E	5E	3A	5B	7E	49
9	AA	21	D2	B7	E7	02	63	5D	1F	A0	1B	66	DB	4E	6D	B2
A	99	1E	BE	17	07	72	34	B0	F1	EF	90	20	0C	CF	BD	D1
B	A7	35	9F	6C	88	C3	D3	93	31	2A	DE	05	D0	AE	A2	16
C	40	F5	89	B5	2E	3C	4B	E4	0A	54	2C	77	D8	6A	0D	AD
D	BC	AF	92	B6	F3	70	F9	12	CC	6B	4F	9C	45	42	BA	04
E	FF	25	79	F6	C7	4A	03	94	23	87	EB	EA	83	7B	F0	A9
F	EE	A8	17	D4	41	C1	E3	78	13	D6	08	84	47	75	24	E0

Table 3. The Forward of White S-Box

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	A5	68	65	25	E3	1E	E5	CD	A0	B9	5E	F0	A3	F2	91	CC
1	27	7D	8D	3B	58	A9	29	AC	A6	0A	0E	6A	BF	96	37	F1
2	C2	52	EF	E7	97	44	EA	B4	5B	20	85	C0	C5	1F	07	54
3	24	48	72	ED	56	EB	42	CF	8E	A2	1A	FF	CA	13	8F	A1
4	30	62	78	E1	80	B5	84	0C	BC	4C	73	9C	02	6C	0B	19
5	DA	6B	67	01	93	9D	17	C7	95	83	F8	D7	F4	FE	7B	60
6	43	63	53	33	4E	64	A7	8C	5C	AA	A4	4F	70	5D	B6	05
7	16	CE	00	09	7A	C1	88	08	34	12	D8	1D	B0	79	81	B8
8	6F	AE	46	D0	6D	06	87	2A	11	AB	AD	C8	35	32	71	75
9	0F	D1	77	47	E8	3E	39	9E	76	FA	41	C3	D4	14	FB	BD
A	3C	BB	E4	18	C4	28	F7	26	98	86	9F	1C	FC	3F	B2	21
B	9B	3A	90	36	2D	66	BA	C9	0D	E9	2E	5F	EC	38	61	D5
C	D6	AF	E0	DC	DE	99	EE	BE	50	3D	45	D2	82	9A	31	6E
D	7F	F5	04	8A	03	B3	F6	E2	69	57	D9	C6	2C	4D	D3	92
E	5A	E6	10	F9	51	23	F3	A8	4A	22	B1	7C	15	8B	55	59
F	4B	94	7E	DB	1B	FD	DF	DD	2F	40	CB	74	B7	49	2B	89

For the forward S-Box steps, the backward white S-Box is constructed in reverse order. The main clue starts by mapping the same affine transform in Equation (3) XORed by the same palindromic constant vector of (A5). The last step in the construction of a backward white S-Box involves taking the multiplicative inverse to the whole values according to the same irreducible polynomial. The backward white S-Box will be the outcome of these sequential operations, which can be seen in Table 4.

Table 4. The Backward of White S-Box

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	72	53	4C	D4	D2	6F	85	2E	77	73	19	4E	47	B8	1A	90
1	E2	88	79	3D	9D	EC	70	56	A3	4F	3A	F4	AB	7B	05	2D
2	29	AF	E9	E5	30	03	A7	10	A5	16	87	FE	DC	B4	BA	F8
3	40	CE	8D	63	78	8C	B3	1E	BD	96	B1	13	A0	C9	95	AD
4	F9	9A	36	60	25	CA	82	93	31	FD	E8	F0	49	DD	64	6B
5	C8	E4	21	62	2F	EE	34	D9	14	EF	E0	28	68	6D	0A	BB
6	5F	BE	41	61	65	02	B5	52	01	D8	1B	51	4D	84	CF	80
7	6C	8E	32	4A	FB	8F	98	92	42	7D	74	5E	EB	11	F2	D0
8	44	7E	CC	59	46	2A	A9	86	76	FF	D3	ED	67	12	38	3E
9	B2	0E	DF	54	F1	58	1D	24	A8	C5	CD	B0	4B	55	97	AA
A	08	3F	39	0C	6A	00	18	66	E7	15	69	89	17	8A	81	C1
B	7C	EA	AE	D5	27	45	6E	FC	7F	09	B6	A1	48	9F	C7	1C
C	2B	75	20	9B	A4	2C	DB	57	8B	B7	3C	FA	0F	07	71	37
D	83	91	CB	DE	9C	BF	C0	5B	7A	DA	50	F3	C3	F7	C4	F6
E	C2	43	D7	04	A2	06	E1	23	94	B9	26	35	BC	33	C6	22
F	0B	1F	0D	E6	5C	D1	D6	A6	5A	E3	99	9E	AC	F5	5D	3B

7.2 KNIGHT SHIFTING AND SHUFFLING

Knight shifting or knight moving is a new transposition cipher that changes the positions of values from one place to another place, not randomly and without changing their values. This stage distributes the placements of bytes according to the knight moving and is similar to the one in chess in which the knight piece goes two squares towards, backward, or side-to-side and then turns to the left or right at a 90-degree angle, exactly similar to the shape of letter "L". For the decryption process, the same operation is implemented and is similar to the one for the encryption process, but with a reverse direction, as shown in Fig 4.

P1	P2	P3	P4
P5	P6	P7	P8
P9	P10	P11	P12
P13	P14	P15	P16

P10	P8	P5	P11
P3	P13	P16	P2
P15	P1	P4	P14
P6	P12	P9	P7

FIGURE 4. The Distribution Map of Permuted Byte Positions

In this case, it will need entirely eight movements for the eight values (positions) to replace with their corresponding eight values according to the knight-shifting method as stated in Fig 5. The state array consists of 16 bytes, and the movements' actions will be half of the position number n/2. Hence, the diffusion property will propagate and cover the whole values of the state array.

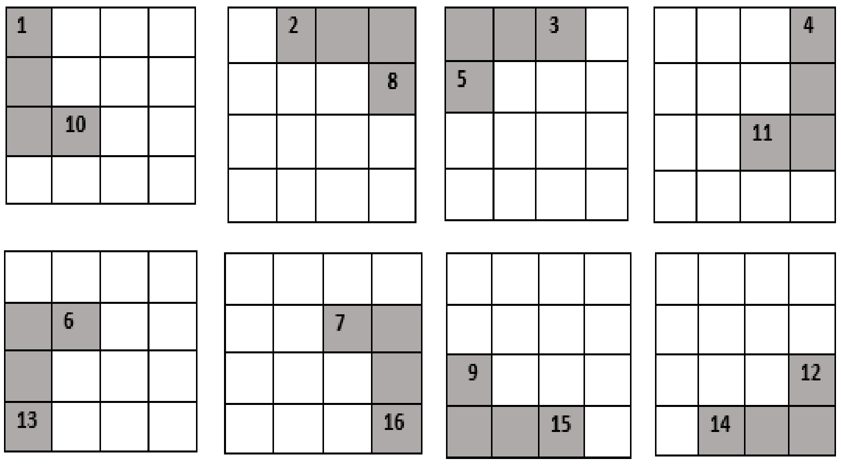

FIGURE 5. The Knight-Shifting Movements

7.3 P-BOX OF SELF-INVERSE MIXING

Self-Inverse Mixing is the third stage in the round transformation that is responsible for the confusion and diffusion properties together. This is a linear equation of polynomial mathematics with degree four over GF(2^8). The new linear equation is selected among several equations that have gained with the self-inverse of the invertible feature as clarified in Equation (4) below:

$$c(x) = 03x^3 + 05x^2 + 03x + 04 \qquad (4)$$

and co-prime to x^4+1. The mathematical preliminaries for the linear equations are fundamental to the content of this work, specifically for this section devoted to studying and generating the MDS code and providing a basis for the weight spaces of certain polynomial notations of the general linear group that are relative to determine the minimum distance of certain weight spaces. If C is a k-dimensional subspace of V(n, q), then C is called [n, k]-code, So it consists of qk codewords, for some integers k and the C which have the dimension k. It has qk codewords if the minimal distance of C is d. Then, it is called [n, k, d] code and the minimum distance d acts as a collection of n-tuples (codewords) over an alphabet of size q such that any two codewords have at most n-d common coordinates and some pair of codewords that may have n-d common coordinates. Therefore, it represents the minimum hamming distance of the code d. This can be addressed as a circulate matrix multiplication with the self-inverse and invertible matrix at the same time where a is the byte collection before transformation and b represents the vector after transformation. Therefore, there is no need to mention the decryption process as it has the same operation in the processes in both directions. Let b(x) = c(x) ⊗ a(x). The usage of this operation is performed on all columns one by one according to the following formula: for $0 \le c < Nb$ where Nb = 4 no of bytes in the word

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 04 & 03 & 05 & 03 \\ 03 & 04 & 03 & 05 \\ 05 & 03 & 04 & 03 \\ 03 & 05 & 03 & 04 \end{bmatrix} = \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \otimes \begin{bmatrix} 04 & 03 & 05 & 03 \\ 03 & 04 & 03 & 05 \\ 05 & 03 & 04 & 03 \\ 03 & 05 & 03 & 04 \end{bmatrix} = \begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix}$$

The implication of this multiplication can be imagined as the entry of 4 bytes, likewise, a column that is multiplied by a fixed MDS matrix in encryption and decryption. As it is known for this type of matrix, the multiplication process by itself produces the identity matrix, this evidence with a reducible equation of $(x^4 + 1)$ over GF(2) shown in Equation (5) below:

$$a(x) = b(x) \mod (x^4 + 1) \qquad (5)$$

$$\begin{bmatrix} 04 & 03 & 05 & 03 \\ 03 & 04 & 03 & 05 \\ 05 & 03 & 04 & 03 \\ 03 & 05 & 03 & 04 \end{bmatrix} \times \begin{bmatrix} 04 & 03 & 05 & 03 \\ 03 & 04 & 03 & 05 \\ 05 & 03 & 04 & 03 \\ 03 & 05 & 03 & 04 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

7.4 ADDING KEY MIXING STAGE

The Key-Mixing stage is the last in each round that is denoted by the Key-Addition process. It represents the adding of byte by byte and bit by bit of the plaintext and the corresponding ciphering key of the same size of ciphering key over G(2) or larger size of ciphering key. There is no need for the inverse process as it is a self-inverse operation and is implemented under the addition of XOR.

A. SUMER Round Key Generation

The key generation process acts as the core part that plays a vital role in securing the algorithm. The ciphering key is affected strongly by the strength of the proposed algorithm and should be kept confidential. As it is known, the key generation algorithm is a one-way algorithm that is implemented directly in a straightforward manner and should be reversed backward of the decryption process. The same proposed key generation function is used in all modes that generate the ciphering sub-keys by depending on the round transformation and the complex functions of (F). The primary operations in the key generation algorithm include bitwise XORed operations, 4 bytes shifting operations, complement operations, right rotate operations, and XORed two constant words of 32 bit. The ciphering key is key-dependent on the sub-byte operation that interleaved with the non-linear stage in the round transformation. The key-scheduling algorithm is generated according to the complex function and some simple operations. The results are XORed to yield a new ciphering sub-key. The incoming word to the state array flows as taking the sub-byte operation or on-fly computation XORed to the two constant vectors (Constant Vector1= E8b47391) and (Constant Vector2= A642C712) for byte shifting and mapping to the complement operation (Com) as stated in Fig 6.

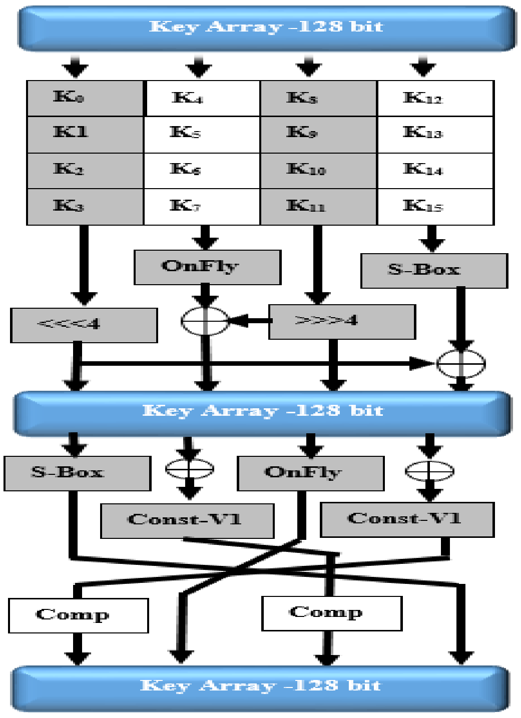

FIGURE 6. Key Scheduling Round

The proposed model is a lightweight design that works with the variable length of the ciphering key. Several new active attacks that have been recently proposed fall under the title of related key attacks that should be encountered to defeat such effective attacks. The algorithm uses the key-dependent round transformation supported by two levels for each round. The ciphering key eliminates any symmetric key sets between the current round and the next one. For the weak or semi-weak keys, the constant vectors are used to remove any clustering bytes or similarity of values. Therefore, various key lengths of the SUMER cipher produce a flexibility and a high level of security margin with different rounds. As the key is one-way, there is no problem in implementing the key addition in both models with variable length.

8. ANALYSIS AND IMPLEMENTATION RESULTS

The lightweight ciphers are built to process a little data with low bandwidth. With these limitations, it has become hard to optimize the algorithms with lightweight cryptography trends. The designers of lightweight algorithms address the design scheme either for a low level of privacy or to a certain level of algebraic immunity (resistance), which is optimized for specific needs to manipulate small amounts of information with limited bandwidth and low power consumption. The design of lightweight ciphers presents the developers with three critical options or three challenges of design objectives: security level, cost, and performance. In practice, it is simple to enhance any two of these three objectives, but it is a very hard task to enhance all these objectives together. It is easy to produce a reasonable plan for balancing between the security and performance factors, or security and cost requirements at the expense of cost or performance. However, in case of the choice of security factor, it will need a large dedicated circuit area that consequently increases the cost requirement and decreases the performance at the same time. From another perspective, submitting an ideal balance is possible with a tradeoff between the reliability of the algorithm and the cost matter, but with restricted potential for the algorithm. The key size and the number of rounds factors specify the strength of the algorithm as a concept of increasing the number of rounds that inevitably increases the security margin.

The SUMER family cipher deals with sufficient key size and number of rounds that prevent reduced or square attacks. The linear and differential attacks are not workable with the strong involutional structure of different modes. In a non-linear stage, the main purpose behind the on-fly mathematical computation is to defeat the interpolation attack that works to find the coefficients of the polynomial. The SUMER model has a balanced structure with involutional round stages in the implementation process in all modes as the encryption and decryption roughly take the same elapsed time in execution processes. The balanced structure closes the door in front of the side-channel and timing attacks that wholly depend upon the implementation issues. The SUMER modes use the key-dependent feature in key scheduling of sub-keys that gives an effective countermeasure against the linear and differential attacks. There is less concern for the ciphering key attacks or the related key attacks on SUMER cipher modes that are considered less effective against the adopted key generation strategy.

The related key attack becomes very hard because there is no weak key or semi-weak key in the key generation procedure by adding two constant words. The main idea of two constant words is to eliminate the similarities of cipher sequences and to frustrate the correlation attack. As the SUMER model works with a robust byte-mathematical design, structural and integral attacks seem impossible with the proposed byte-oriented structure design. The design complexity substantially affects the number of Gate Equivalent (GE) of the logical circuits, in particular, the construction of S-Boxes tables and the length of the ciphering key algorithm. In this regard, the design of a lightweight model can be implemented in software and hardware together or sometimes merely for the software applications due to design features that may pass the permissible range of GE. The SUMER model supports several options for the security levels according to the importance of encrypted data. In case of a high margin of security with less performance, the SUMER extended mode will be the better choice in which the algorithm encrypts the data of 256 bit with three different lengths of keys (256 bit, 384 bit, and 512 bit). According to security and performance, The SUMER model also supports a traditional mode of SPN structure like the AES cipher. For lightweight applications, the SUMER lightweight mode is implemented for a restricted environment with low cost and an acceptable security level. The NIST randomness statistical tests test the proposed cipher and produce reasonable results without any negative biases to the randomness threshold. The future labor will witness new optimizations in designing ciphers that include the improvement of a mixed structure such as block and stream ciphers to optimize a hybrid cipher. Another respected work is to construct a symmetric cipher that gets together the two diverse structures (FN) and (SPN) in a single dynamic algorithm. The GE and other impact parameters are simulated and compared with other traditional and lightweight ciphers as illustrated in Table 2. The security aspects of the SUMER cipher need more deep analysis and precise investigation from the researchers and specialized academics by attacking the proposed model to test it precisely. I would be satisfied and thankful to accept any negative or positive consequences to evaluate the strength of the proposed cipher modes, and I shall accept any scientific criticism that would correct the algorithm and work to the best. The ModelSim hardware simulator tool, which tests benchfiles with “.vhd,” implemented the proposed ciphers, and the simulated results are illustrated in the following Table 5. The implementation was under an HP laptop, core-i7, 5500U CPU, and 2.40 GHz, RAM 8 GB, and Windows 64 bit. All proposed algorithm modes are programmed with Visual Studio C# language, using Notepad++ free source editor, which supports Hardware Description Language (HDL). Notepad++ edit and write HDL codes and are then integrated with the simulator.

Table 5. Comparison Between SUMER (Modes) and Other Lightweight Cipher Modes

Algorithm	Structure	Key Size	Block Size	Area-Gate Equivalent	Throughput (Mbit/s)
Lightweight SUMER-128-SPN*	SPN	128	128	987	10.4
Extended SUMER-256 - Feistel*	Feistel	256	256	3430	92
Conventional SUMER-128- SPN*	SPN	128	128	2780	64.7
AES-128 [20]	SPN	128	128	3100	80
DESL-56 [14]	Feistel	56	64	1848	44.4
LED-64 [16]	SPN	64	64	966	5.1
LED-80 [16]	SPN	80	64	1040	3.4
LED-96 [16]	SPN	96	64	1116	3.4
LED-128 [16]	SPN	128	64	1265	3.4
KlEIN-64 [13]	SPN	64	64	1220	N/A
KLEIN-80 [13]	SPN	80	64	1478	N/A
KLEIN-96 [13]	SPN	96	64	1528	N/A
PRESENT-80 [10]	SPN	80	64	1570	200.0
PRESENT-128 [10]	SPN	128	64	1391	11.4
KATAN-64 [17]	Stream	64	64	1054	25.1
mCrypton-96 [1z]	SPN	96	64	2681	492.3
SEA-96 [9]	Feistel	96	96	3758	103.0
HIGH-128 [8]	GFN	128	64	3048	188.0
Piccolo-80 [12]	GFN	64	80	1136	237.04
Piccolo-128[12]	GFN	64	128	1196	193.9
CLEFIA-128[11]	GFN	128	128	4950	355.6

The Table 5 above shows the results of gate equivalents and the throughput for the SUMER cipher with multi-modes compared with various famous algorithms that have been evaluated and tested by ModelSim. The time implementation of elapsed time for the encryption and decryption process is measured in milliseconds. It is valuable to assess the trade-offs between the security factor, throughput, and gate count when choosing an encryption cipher for a particular application. The aforementioned algorithms were compared under fixed conditions to give a fair evaluation. The outcomes give wonderful insights into the effectiveness of each algorithm with its structure for a particular task. Furthermore, the results provide excellent perception of the performance of these cryptographic ciphers and identify the best for resource utilization. The throughput test compensates by measuring the number of bits or bytes that can be handled per millisecond and the effect on the performance of the algorithm. Regarding throughput, SUMER lightweight mode has a processing speed of 10.4 cycles per byte, while SUMER traditional mode has throughput rates of 64.7. The extended mode works with 92 cycles per byte. These indicators for the throughput showed that the proposed lightweight mode presented a much faster processing speed compared to traditional and extended modes. When resource-restricted environments are considered, the higher throughput of the extended cipher mode may have a great advantage when fast encryption and decryption are desirable. Regarding the gate equivalent that represents the amount of electronic gate for hardware implementation. The GE for each algorithm offers the feasibility of algorithm work on resource count. The SUMER cipher with three modes provides efficient implementation, but the best one with the lower gate equivalent of 987 GE and high throughput is the SUMER lightweight cipher.

Table 6. S-Box Design Characteristics

Algebraic Properties	Lightweight Mode	Extended Mode	Traditional Mode
Correlation Immunity	Zero	Zero	Zero
Algebraic Degree	Seven	Seven	Seven
Algebraic Complexity	252/255	252/255	11/255
Bijection	Yes	Yes	Yes
Strict Avalanche Criteria (SAC)	½	½	½
Non-Linearity	112	114	114
Differential Uniformity	4	4	4
Power Mapping	Onfly	S-box	S-box
Key Dependent S-Box	Yes	Yes	Yes

The S-Box design for the proposed family ciphers gave excellent results in non-linearity and confusion properties under several algebraic criteria, as stated in Table 7 above. The strength of the S-Box concerning the resistance against linear and differential attacks is measured in the correlation community factor. A good correlation is shown between the input and output of the proposed three modes, which makes analyzing or deducing the key of the S-Box hard. The proposed ciphers exhibit a low degree of 7, which relatively participate in the defense against algebraic attacks. The algebraic complexity factor states a high complexity in the algebraic equations through which the S-Box is designed, which approximate the irreducible polynomial and make the attacks more sophisticated. The proposed S-Box validated the bijection factor, as each input value has a unique corresponding output value and ensures that the S-Box is invertible. The Strict Avalanche Criteria (SAC) factor proves the effectiveness of the algorithm when changing a certain input bit for the proposed S-Box values effectively changes half of its output bits consecutively. Therefore, the SAC prevents the attacker from exploiting the ciphertext along the rounds. The SAC plays a vital role in measuring the S-Box deviation from a linear transformation. The strength of any algorithm depends on the non-linearity degree for the S-Box that produces a powerful resistance against the linear cryptanalysis attacks. Another important factor is the differential uniformity, which measured the mapping of output differentials for all possible input differentials. The proposed S-Box showed a lower value for the differential uniformity of 4, which means it is more coherent against the differential cryptanalysis.

The power mapping points to the properties of the S-Box that enable the attacker to try to compute the encryption and decryption operation by analyzing the power consumption. Consequently, information about the secret key is gained. The proposed S-Box of SUMER works with two mechanisms: either on-fly computing that closes the door in front of side-channel attacks or a lookup table that affects power consumption. The mathematical operations are computed by exploiting the electromagnetic emission for the processor. The key-dependent S-Box feature works to eliminate any similar pattern for the sub-keys generation and prevents weak keys from appearing. The key-dependent S-Box enhances the security of the algorithms and effects dramatically so that any change in encryption will impact the S-Box work.

Table 7. **Time Implementation of Three SUMER Modes**

The Algorithm Mode	Structure	Key Length	Size-Key Array	Rounds	Encryption Time	Decryption Times
Involutional Lightweight SUMER-SPN	SPN	128-bit	4*4	12	000.25	000.25
		192-bit	4*6	14	000.32	000.32
		256-bit	4*8	16	000.45	000.45
Extended SUMER-Feistel Structure	Feistel	256-bit	4*8	10	00.85	00.94
		384-bit	4*12	12	00.1	00.13
		512-bit	4*16	14	00.22	00.31
Traditional SUMER – SPN	SPN	128-bit	4*4	10	000.41	000.46
		192-bit	4*6	12	000.53	000.6
		256-bit	4*8	14	000.74	000.81

The encryption and decryption time for any algorithm depends on the size of the encryption block, key length, number of rounds, and the mathematical basis. According to the Table 7 above, the SUMER lightweight mode showed clearly that the encryption and decryption operations take the same time because of the balanced involutional structure. Regarding the SUMER traditional cipher, the algorithm encrypts the plaintext to the ciphertext a little faster than the decryption process due to the extra computation for the inverse operations. The extended SUMER cipher is the slowest cipher among the SUMER modes. The decryption process for the extended cipher is also slower than the encryption process. The slowness increases significantly with the increase in the key length and the number of rounds. The speed of the algorithm is very important for some applications like real-time and IoT applications, especially some scenarios that require fast data access and speed processing. In general, even the slowness in the decryption process is still reasonable compared with other algorithms. There is no better algorithm in our proposal ciphers as each one is designed for specific purposes in which the lightweight mode is considered a good choice in restricted devices. The traditional cipher will be a good one if the applicable scenario needs less than 128 bits with three changeable keys. In case of big data applications and military-grading utilizations, the extended mode will be a better option.

9. CONCLUSIONS

A practical of secure proposed SUMER family ciphers with three distinctive equivalent modes has been suggested. SUMER ciphers are multi-mode algorithms that work with three modes: lightweight, traditional, and extended. The lightweight mode can be implemented with an involutional structure of SPN for securing embedded devices. It can also be implemented with low-resource areas and highly constrained hardware in a limited environment, especially for IoT applications. The traditional mode can be implemented so far, similar to the AES cipher implementation. The SUMER extended mode is a high-level secure algorithm that would be used for securing a wide variety of applications at a top-secret level. Ciphers of the SUMER family are suitable algorithms for general-purpose applications and for a large-scale multi-utilization fields. Enough amendment and software/hardware optimization have been introduced for the lightweight structure to be compatible with two different structures. Each of these structures employs the on-fly computation feature of an elegant lightweight construction, which enables very efficient execution in several platforms. The proposed algorithms present an optimal tradeoff among different metrics of security margin and the performance property according to the utilization nature and the aforementioned measurement tests. The SUMER family ciphers are adaptive ciphers with high key agility and an involutional structure, which work with the same stages in the round transformation of the SPN and Feistel structures but with different block and key sizes.

Funding

None

ACKNOWLEDGEMENT

The Author would like to thank his students in M.A for their support and helping in programing and testing the introduced algorithms and theirs vectors tests.

CONFLICTS OF INTEREST

The author declares no conflict of interest.

REFERENCES

[1] A. Carlson, S. R. Mikkilineni, M. W. Totaro, R. B. Wells, and R. E. Hiromoto, "Equivalence of Product Ciphers to Substitution Ciphers and their Security Implications", in *2022 International Symposium on Networks, Computers and Communications (ISNCC)*, Jul. 2022, pp. 1–6. https://doi.org/10.1109/ISNCC55209.2022.9851719.

[2] G. Sravya, M. O. V. P. Kumar, Y. Sudarsana Reddy, K. Jamal, and K. Mannem, "The Ideal Block Ciphers - Correlation of AES and PRESENT in Cryptography," in *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*, Dec. 2020, pp. 1107–1113. https://doi.org/10.1109/ICISS49785.2020.9315883.

[3] E. B. Kavun, "A Power Reduction Technique Based on Linear Transformations for Block Ciphers", in 2022 IFIP/IEEE 30th International Conference on Very Large Scale Integration (VLSI-SoC), Oct. 2022, pp. 1–6. https://doi.org/10.1109/VLSI-SoC54400.2022.9939628.

[4] O. A. Dawood, A. M. Sagheer, and S. S. Al-Rawi, "Design Large Symmetric Algorithm for Securing Big Data", in *2018 11th International Conference on Developments in eSystems Engineering (DeSE)*, Sep. 2018, pp. 123–128. https://doi.org/10.1109/DeSE.2018.00026.

[5] O. A. Dawood, "Fast lightweight block cipher design with involution substitution permutation network (SPN) structure", *Indones. J. Electr. Eng. Comput. Sci.*, vol. 20, no. 1, p. 361, Oct. 2020. http://doi.org/10.11591/ijeecs.v20.i1.pp361-369.

[6] Hkiri, Amal, Mouna Karmani, and Mohsen Machhout. "Implementation and Performance Analysis of Lightweight Block Ciphers for IoT applications using the Contiki Operating system." 2022 IEEE 9th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT). IEEE, 2022. http://doi.org/10.1109/SETIT54465.2022.9875503.

[7] K. A. McKay, L. Bassham, M. S. Turan, and N. Mouha, Report on lightweight cryptography, National Institute of Standards and Technology, Gaithersburg, MD, Mar. 2017. https://doi.org/10.6028/NIST.IR.8114.

[8] Hong, Deukjo, Jaechul Sung, Seokhie Hong, Jongin Lim, Sangjin Lee, Bon-Seok Koo, Changhoon Lee, et al. 2006. "HIGHT: A New Block Cipher Suitable for Low-Resource Device". In *Cryptographic Hardware and Embedded Systems - CHES 2006*, 46–59. Springer Berlin Heidelberg. https://doi.org/10.1007/11894063_4.

[9] Standaert, François-Xavier, Gilles Piret, Neil Gershenfeld, and Jean-Jacques Quisquater. 2006. "SEA: A Scalable Encryption Algorithm for Small Embedded Applications". In *Smart Card Research and Advanced Applications*, 222–36. Springer Berlin Heidelberg. https://doi.org/ 10.1007/11733447_16.

[10] Bogdanov, A., L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe. 2007. "PRESENT: An Ultra-Lightweight Block Cipher". In *Cryptographic Hardware and Embedded Systems - CHES 2007*, 450–66. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-74735-2_31.

[11] Shirai, T., Shibutani, K., Akishita, T., Moriai, S., Iwata, T.: "The 128-bit Blockcipher CLEFIA (Extended Abstract)". FSE 2007. LNCS, vol. 4593, pp. 181-195. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74619-5_12

[12] Shibutani, K., Isobe, T., Hiwatari, H., Mitsuda, A., Akishita, T., Shirai, T.: "Piccolo: An ultra-lightweight blockcipher". In: CHES. (2011). https://doi.org/10.1007/978-3-642-23951-9_23.

[13] Gong, Z., Nikova, S., Law, Y.-W.: "KLEIN, a new family of lightweight block ciphers". In: Proceedings of The 7th Workshop on RFID Security and Privacy 2011. LNCS. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25286-0_1

[14] Leander, G., Paar, C., Poschmann, A., Schramm, "K.: New Lightweight DES Variants".In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 196–210. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74619-5_13

[15] Lim, C.H., Korkishko, T.: "mCrypton – A Lightweight Block Cipher for Security of Low-Cost RFID Tags and Sensors". In: Song, J., Kwon, T., Yung, M. (eds.) WISA 2005. LNCS, vol. 3786, pp. 243–258. Springer, Heidelberg (2006). https://doi.org/10.1007/11604938_19.

[16] J. Guo, T. Peyrin, A. Poschmann, and M. Robshaw, "The LED Block Cipher" In B. Preneel and T. Takagi (Eds.), CHES 2011, LNCS 6917, pp. 326–341, Springer, 2011. https://doi.org/10.1007/978-3-642-23951-9_22.

[17] De Canni`ere, C., Dunkelman, O., Knezevic, M.: "KATAN and KTANTAN – A Family of Small and Efficient Hardware-Oriented Block Ciphers". In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 272–288. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04138-9_20.

[18] Liu, Bo-Tao, et al. "Loong: A family of involutional lightweight block cipher based on SPN structure." IEEE, pp.136023-136035, Access 7, (2019). https://doi.org/10.1109/ACCESS.2019.2940330

[19] Leander, Gregor, et al. "The speedy family of block ciphers-engineering an ultra-low-latency cipher from gate level for secure processor architectures.", pp. 510-545, Cryptology ePrint Archive (2021). https://doi.org/10.46586/tches.v2021.i4.510-545

[20] Kumar, Umesh, and V. Ch Venkaiah. "A Family of Block Ciphers Based on Multiple Quasigroups." Cryptology ePrint Archive (2022).

[21] Avanzi, Roberto, et al. "The qarmav2 family of tweakable block ciphers." IACR Transactions on Symmetric Cryptology 2023.3, pp. 25-73, (2023). https://doi.org/10.46586/tosc.v2023.i3.25-73

[22] V. Nachef, J. Patarin, E. Volte. "Feistel ciphers Security proofs and cryptanalysis". Springer International Publishing, pp.1-309, 2017. https://doi.org/10.1007/978-3-319-49530-9

[23] J. Holden, *The Mathematics of Secrets: Cryptography from Caesar Ciphers to Digital Encryption*. Princeton University Press, ISBN:9780691183312, pp. 392, 2018.

[24] B.-T. Liu, L. Li, R.-X. Wu, M.-M. Xie, and Q. P. Li, "Loong: A Family of Involutional Lightweight Block Cipher Based on SPN Structure", *IEEE Access*, vol. 7, pp. 136023–136035, 2019. https://doi.org/10.1109/ACCESS.2019.2940330.

[25] A. Carlson, S. R. Mikkilineni, M. W. Totaro, R. B. Wells and R. E. Hiromoto, "Equivalence of Product Ciphers to Substitution Ciphers and their Security Implications," *2022 International Symposium on Networks, Computers and Communications (ISNCC)*, Shenzhen, China, pp. 1-6, 2022. https://doi: 10.1109/ISNCC55209.2022.9851719.