International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 11, No.1 (Apr-2022)

https://dx.doi.org/10.12785/ijcds/1101108

A Modified Split-Radix Architecture-Based Key Scheduling Technique for Lightweight Block Ciphers

Sohel Rana[1] and Mohammod Abul Kashem[2]

[1,2]*Department of Computer Science and Engineering, Dhaka University of Engineering and Technology, Gazipur-1700, Bangladesh*

Received 21 Jun. 2021, Revised 4 Sep. 2021, Accepted 21 Oct. 20121, Published 15 Apr. 2022

Abstract: This research proposed MSBK, a novel key generation approach that introduces the split-radix butterfly architecture of fast Fourier transformation (FFT) in the field of cryptography. The MSBK cipher modified the butterfly architecture to fit accurately into cryptography. However, butterfly architecture has a higher avalanche effect which creates generated keys enough strong to protect information from different online and offline attacks. It also meets the standard of the Shannon properties: Confusion and diffusion. The proposed MSBK cipher consumes less power than earlier works. The MSBK technique's memory consumption and execution cycle were assessed using the FELICS (Fair Evaluation of Lightweight Cryptographic Systems) program, which operates on the Linux operating system. The suggested MSBK technique has also been simulated in MATLAB 2021a to evaluate the key sensitivity of multiple original and encrypted images using the histogram, correlation graph, and entropy. The negative correlation coefficient of images encrypted by the proposed cipher indicates that the differential and statistical attacks are quite impossible for an intruder. Furthermore, the number of change in pixel rate (NPCR) and the unified average change of intensity (UACI) parameters show that the suggested MSBK approach is robust to different statistical attacks.

Keywords: Split-radix Butterfly architecture, Avalanche Effect, Lightweight cryptography, Block Ciphers, Key Scheduling, FELICS, MATLAB.

1. Introduction and Overview

The lightweight block ciphers [1] refer to a subset of conventional encryption that aims to offer security for devices with limited resource capability. When considering computer communication, everyone wants to ensure that only the intended recipient receives the original information. To ensure this security problem, one must have to encrypt the information because only the intended recipient should interpret its content. Nevertheless, at the heart of lightweight block ciphers, there is still a distinct trade between lightweight and safety: that is how a reasonable degree of security is frequently accomplished in those kinds of resource-constrained systems. Cryptographic primitives that are suitable for resource-constrained systems [2] have been the emphasis of study over recent days. Now-a-days, the academic world has concentrated on developing encryption algorithms that are appropriate for these resource-limited systems. Traditional cryptographic techniques, such as RSA, generally work well enough on expensive devices; hence, lightweight techniques may not appear to require for these. The resource limited devices include micro-controllers, the Internet of Things (IoT), radio frequency identification devices (RFID), and wireless sensor networks (WSN). Also, as a consequence, lightweight encryptions are primarily driven for such purposes. For resource-constrained systems, several encryption techniques, such as DES [3] , AES [4] , and PRESENT [5] are employed. The lightweight block ciphers are designed to promise appropriate security as well as use as few execution cycles as feasible on many of these resource-constrained systems. The encryption schemes are often followed by Feistel architecture, such as SIMON [6],[7] , SIT [8] , Speck [7] etc. and some follow SPN structures like PRESENT, AES, etc. or both structures are used, such as DES, SIMON to provide sufficient security strength in cipher text.

In order to ensure the security of cryptographic algorithms, key scheduling must be done in a secure manner. Generally, different complex number theories are used in key generation techniques in order to make round keys strong. These approaches slow down the performance of devices with limited resources since they rely on complex number theories. To be efficient, key scheduling should have two characteristics: randomization to produce unique keys and an avalanche effect to assure high key sensitivity. Therefore, an attacker cannot readily predict plain-text or keys through statistical attacks on cipher text, a single bit change in the key should alter at least 50 % of the cipher text. The Avalanche effect is the term used to describe this phenomenon.

E-mail address: sohelresearch@gmail.com

A. *Motivation*

For most those interested in cryptography, it is a fascinating and hard subject to conduct research on. In order for data to be transmitted securely, it must be encrypted. This includes difficult mathematics, complicated coding, and sophisticated number theory (among other things). Because resource-constrained devices are becoming more common, lightweight block ciphers will be needed in the coming years to protect these devices. More than 70% of devices with insufficient resources are susceptible to cyberattacks, according to HP [9] . As a consequence of this, a balance must be maintained between performance and security. There have been several proposals for [10] complex number ciphers, such as prime factorization, modular arithmetic as well as GCD testing etc. By employing complex number theories to maintain Shannon's confusion and diffusion features (Avalanche Effect) [1] , traditional ciphers like RSA, AES become computationally costly, impairing the functionality of devices with limited resources. Hence, it has also become increasingly difficult to apply these advanced heavy-weight cryptographic techniques to tiny computing devices in order to maintain security. With the use of the [11] modified butterfly structure of FFT, we came up with a simple and fewer power-consuming key scheduling approach. More than half of what fulfills Shannon's confusion and diffusion qualities can be achieved using the MSBK approach.

B. *Contribution*

To address the security and resource usage concerns of lightweight devices, we offer a modified butterfly architecture-based key generation approach called MSBK. As a consequence, we have integrated the modified butterfly structure into the key scheduling process to produce round keys with high strength. MSBK employs basic logical operations such as XOR and XNOR to produce its output, which makes it computationally light. It also assures non-linearity because MSBK generates output using random numbers. The standard avalanche effect is strongly recommended to design a strong cipher. The average avalanche effect for the round keys is more than 50 percent. In AES and DES, S-BOX [12] performs this non-linear transformation. The Avalanche effect was tested by encrypting images in MATLAB 2021a. For example, the Linux-based FELICS (Fair Evaluation of Lightweight Cryptographic Systems) benchmark was used to analyze the MSBK's memory use and execution cycle. As a consequence, the suggested MSBK cipher produces appropriate security while consuming lower energy.

2. LITERATURE REVIEW

Nowadays, most of the cryptographic algorithms that are being suggested use advanced mathematical techniques like RSA [13] and ElGamal [14] , and also SPN [10] networks such as AES and Fiestel [2] architectures such as the DES technique. As a result, keys and cipher text will be more secure due to the increased confusion and diffusion in cipher text that results from the usage of these primitives. As these primitives are computationally costly, when implemented on devices with limited resources, their performance suffers. Cryptography's major challenge is to increase the "avalanche effect" while still using only lightweight mathematical operations. This section describes some of the most relevant research from the previous, as well as their shortcomings.

A. *Modified Butterfly architecture of FFT*

There are a number of ways to apply the DFT that are low-complexity. One of the most common is to utilize the FFT [10], which involves processing the signal for devices with minimal resources. Butterfly architecture is the heart of FFT computing. One of the most popular FFT methods is the Cooley-Tukey algorithm [11]. This algorithm is used to compute the complex series of FFT. This algorithm works in a divide-conquer manner to split the whole DFT problem into several possible smallest DFTs. The simple FFT consists of radix-2 butterfly architecture blocks. We used the same structure of butterfly block as in FFT. We just replaced complex computation and mathematical operations by two logical operations: XOR and XNOR that play a vital role to achieve a consistent avalanche effect in generated keys.

B. *Feistel Architecture*

Feistel architecture [8], [12] is a symmetric structure to achieve higher avalanche effect in cipher text to keep safe cipher text from different attacks. It is a repetitive structure. There are several rounds and each round has same operation with different keys in Feistel Network. Input data is divided into two halves. The right half does unchanged and also is transmuted by a round function which takes a round-key as input. The left half generates output by combining with the transformed output of right half using a bitwise operation XOR. After that left half and right half are exchanged to get input for the next round. The number of rounds is determined by the capability of the cipher installed on the device.

C. *Related works*

A neural network cipher was suggested by the authors of the studies [15],[16]. They developed a neural network-based encryption system which can be used in cryptography. Their neural network was trained using the Back-propagation architecture, which was used as part of supervised learning. There is no error propagation in the outputs of the neural network throughout the encryption and decryption process, the authors reported. Neural network based cryptosystem treats weights as keys. A neural network's added training phase, however, makes their method computationally costly. The authors of [17] evaluated the modern lightweight ciphers TEA, HIGHT, KATAN, and KLEIN, which are used in resource-constrained devices. These ciphers were implemented on the AVR Atmel ATtiny45 to test the avalanche effect and measure performance based on memory efficiency and energy consumption, as well as confusion and diffusion. Only performance metrics such as RAM, ROM, and execution cycles were taken into account. They did not, however, provide any results for determining the security strength of the ciphers reported. To achieve less data usage and reduced power consumption, the authors of [2] in 2018 proposed a new lightweight device cipher that comprises of 2 fundamental ideas in genetic algorithms: mutation and crossover. As part of the key scheduling process, non-linear bit shuffling was substituted for the matrix by the authors. Their proposed cipher was also put to the test on FELICS to compare execution time and memory consumption. MATLAB tools were used to analyze the security of cipher based on image encryption. In [19], authors presented a lightweight block cipher for resource-constrained devices based on feed forward neural network. In evaluation, they considered only visual evidence for security measurement by histogram and correlation graph of the encrypted images. For linear and non-linear transformations, the paper [8] proposed a cipher that coupled Feistel architecture with a Substitution Permutation Network (SPN). In their suggestion, the authors included key expansion and encryption in the cipher. In the key expansion stage, five distinct keys were generated using a 64-bit cipher key. F-functions are fitted as input for each of the four blocks (see Figure 1). For transforming the output of F-function, a 4x4 metric is employed. The only nonlinearity here comes from the use of the matrix. The F-function is made up of P and Q tables; the linear transformation is utilized in key expansion.

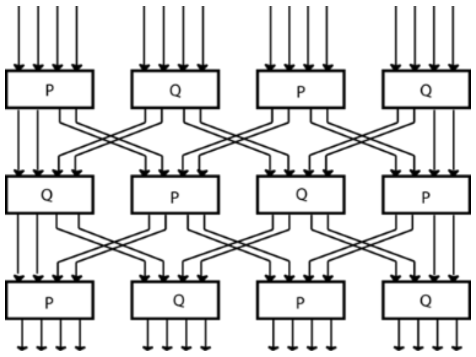


Figure 1. F-Function of SIT algorithm

For stronger key sensitivity, this study substituted the F-function with the MSBK block, which offers both non-linearity and avalanche effects.

3. PROPOSED MSBK CIPHER

As described in this section, the suggested method uses 64-bit block-sized symmetric key block ciphers. The proposed algorithm consists of Key scheduling as well as encryption process. The key expansion generates round keys which are used to encrypt plaintext with the help of encryption process. In the suggested technique, five different round keys for 5 rounds of encryption are generated using an MSBK-based key generation process. Thus the encryption process must be strong enough so that the intruders cannot be able to break the cipher. The key is the most important part of the encryption and decryption process. This key is essential to the encryption of the cipher text. Security is compromised if the key used to produce cipher text is compromised. The round keys have a size of 64 bits each.

A. MSBK Structure

The Modified Split-radix Butterfly for Key (MSBK) structure is very simple which is based on the concept of the modified split-radix butterfly structure of FFT. This function have three layers namely input layer $X = [x_0, x_1, x_2, x_3]$, middle layer and output layer $Y = [y_0, y_1, y_2, y_3]$. This function takes four 4-bit numbers as input and output has the same size as input. Two basic bitwise logical operations: XOR and XNOR are used to produce the output. The modified split-radix butterfly comprises two hidden layer operations from full butterfly architecture for first two input sequences like x_0 and x_2 (see figure 2). In the split-radix architecture, first two input sequences like x_0 and x_2 is directly connected to output layers as shown in Figure 2 . Also two random numbers R_1 and R_2 is applied to generate the output.

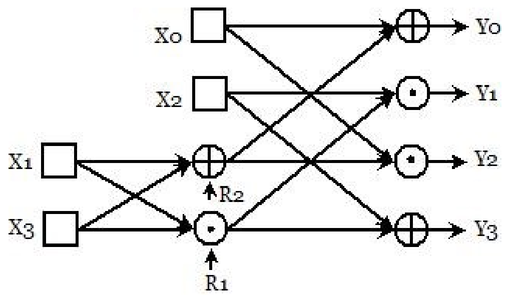


Figure 2. Internal structure of MSBK block

The pseudo random number R_1 and R_2 are generated by the following equation 1, 2, and 3 .

$$X = \sum_{i=0}^{4} x_i \tag{1}$$

$$R_1 = X \quad mod \quad m \tag{2}$$

$$R_2 = X \quad mod \quad m \tag{3}$$

where $x_i = [x_0, x_1, x_2, x_3]$ four inputs and m is a prime number in between 2 and 16. The pseudo random number guarantees that the MSBK block maintains nonlinearity in generated keys. The following equations produce the final output of the MSBK block.

$Y_0 = X_0 \oplus X_1 \oplus X_3 \oplus R_2$
$Y_1 = X_2 \odot X_1 \odot X_3 \odot R_1$
$Y_2 = X_0 \odot X_1 \oplus X_3 \oplus R_2$
$Y_3 = X_2 \oplus X_1 \odot X_3 \odot R_1$
Thus, output layer Y is calculated to feed into the next block as input.

B. Key expansion with MSBK

The key scheduling design of a cipher should be as complex as feasible to implement in resource-constrained devices In order to protect data against various statistical attacks. The generated round keys must have high sensitivity. A single-bit key difference would not be enough for an attacker to decrypt encoded text with a key varying only by a single bit from the secret key. The MSBK structure was employed in order to generate round keys including a higher unpredictability. Figure 3 shows the internal functional block diagram of the proposed MSBK-based key expansion process.

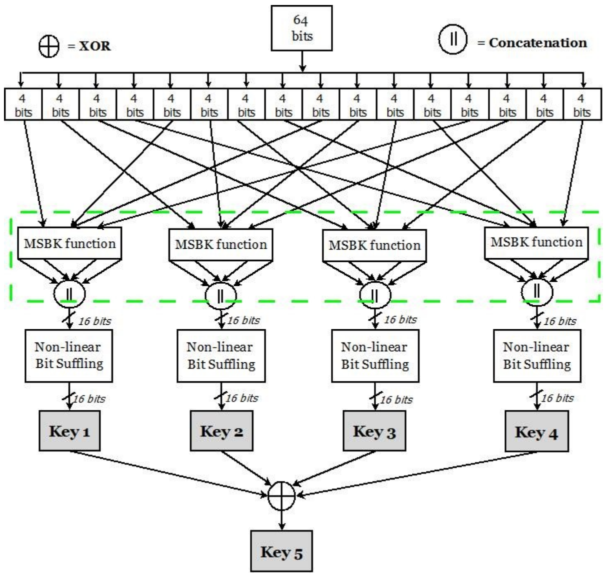


Figure 3. MSBK-based key expansion

To produce five distinct keys for 5 rounds of encryption, the proposed technique requires a 64-bit cipher key as input (see figure 3). There are 16 networks created from 64 bits of cipher key split by 4 bits. In each network, 16 networks of 4 bit data are interconnected. Employing four MSBK blocks, the suggested technique operates on four 4-bit (0-15) numerical numbers. According to equation 3.4, the first permutation of 16 segments of the input cipher key (K) yields four integers for each MSBK block.

$$MSBK_i = \|_{j=1}^{4} K_{((j-1)+i)} \tag{4}$$

The first four round keys are numbered from i = 1 to 4 (see figure 3). According to equation 3.4, each MSBK block receives input from four segments. The nonlinear bit shuffling used in earlier research was duplicated in order to achieve sufficient diffusion (linear transformation) in the created key sets. This is followed by the generation of four 16-bit keys (K1, K2, K3, and K4), which are derived from the output of four non-linear bit shuffling networks. K5 will be generated by applying the XOR function to each of the first four keys.

SL	***Cipher Keys(64 bits)***	***Cipher text (64 bits)***	***No. of bit changed***	***Avalanche Effect (%)***
1	*0x1000000000000001*	0xd9bca81c1d41fe01	*36*	*56.25*
	0x1000000000000004	0x13c10b4a82272460		
2	*0x5555555555555554*	0x52f4f57878257569	*40*	*63.49*
	0x5555555555555555	0x9c2f0f7996108fbe		
3	*0xabababababababab*	0x7a099c9711c641b4	*33*	*51.56*
	0xababababababababaa	0x211f22faab8722ba		

TABLE I. Avalanche effect of MBFK based key generation approach

C. Avalanche effect of MSBK-based key expansion

The proposed MSBK based key scheduling process generates round keys with higher avalanche effect. We generated quite thousands keys to evaluate the avalanche effect of MSBK based key generation approach. In best case analysis, it provides avalanche effect up to 63.49%. However, the average avalanche effect of proposed key scheduling is above 50 percent which meets the standard of Shannon confusion properties. Table I illustrates three ciphertext pairings for three different pairs of cipher keys, each of which varies by one bit.In this case, the plaintext used is 0xabcd123487650135.

D. Bit shuffling block(Non-Linear)

Nonlinear bit-shuffling was utilized, as has been described in the literature [2]. Each non-linear bit shuffling block receives a concatenated 16-bit input from the corresponding MSBK block. Each 16-bit input is mixed with a random integer that is calculated from the 16-bit input. A linear feedback shift register generates a pseudo random integer using the 16-bit input as a seed. These are then XORed. In the bit shuffling block, the result is passed on. A typical in-place permutation is performed by the bit shuffling block. Bit shuffling and the perfect shuffling block progressively receive the result of the XOR operation to produce adequate diffusion in the produced keys. The position of non-linear bit shuffling is as seen in figure 3 .

E. Encryption and decryption

The authors of the study [2] suggested a cryptographic method that we utilized. Genetic algorithms have two fundamental operators: mutation and crossover, which are used in the Feistel architecture with G-function [2]. Figure 4 illustrates the graphical view of a single round of encryption among five rounds, and equation 5 describes the mathematical operations of a single round.

$$R_{i,j} = \begin{cases} X_j \odot K_i & ; \quad j = 1 \quad and \quad 4 \\ X_{j+1} \oplus G_{li} & ; \quad j = 2 \\ X_{j-1} \oplus G_{ri} & ; \quad j = 3 \end{cases} \tag{5}$$

Where the range of i equals to number of round (in this case i is 1 to 5).
As illustrated in Figure 4, the 64-bit data is divided into four 16-bit parts (X_1, X_2, X_3, and X_4). Swapping, XOR, and XNOR operations are undertaken among the divided blocks as per the Feistel architecture to improve the avalanche effect in ciphertext. Two separate X-NOR operations are

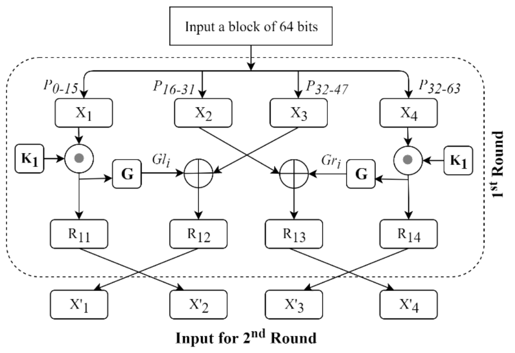


Figure 4. A single round of encryption process

conducted between the round key and the leftmost (X_1) and rightmost (X_4) blocks. The result ($R_{i,j}$) of the X-NOR operation is then supplied as an input to the G-function, which produces the output Gl_i or Gr_i, where $i$represents the round number and j indicates one of the four segments of the data blocks. The third block (X_3) and the left G-function's output Gl_i are XORed again, as are the second block (X_2) and the right G-function's output Gr_i. Then, excluding the last round, a swapping operation is performed among the four blocks in such a way that the four input segments of the second round X'_1, X'_2, X'_3, X'_4 will be R_{12}, R_{11}, R_{14}, and R_{13}, as illustrated in Figure 4. The process of encrypting plaintext into ciphertext in a single round is represented by equation 5. Finally, each four blocks are concatenated to form the ciphertext block (of 64 bits). The decryption procedure is the inverse of the encryption procedure. Round keys will be used in reverse order this time, with the 5th key (K5) being utilized in the first round.

F. G-function

As part of our investigation, we mimicked the G-function from prior studies. A genetic algorithm's mutation and crossover operators are used to develop the G-function [2] . First, it divides 16 bits evenly into two eight-bit parts. A substitution box is used to convert the middle 4 bits of two 8-bit data. Once both outputs of the S-Box have been transformed by a two-point crossover operator, a coin flip mutation takes place. Last but not least, we have the 16-bit output created for the encryption process as G_{li} or G_{ri} as shown in Figure 4.

4. RESULT AND DISCUSSION

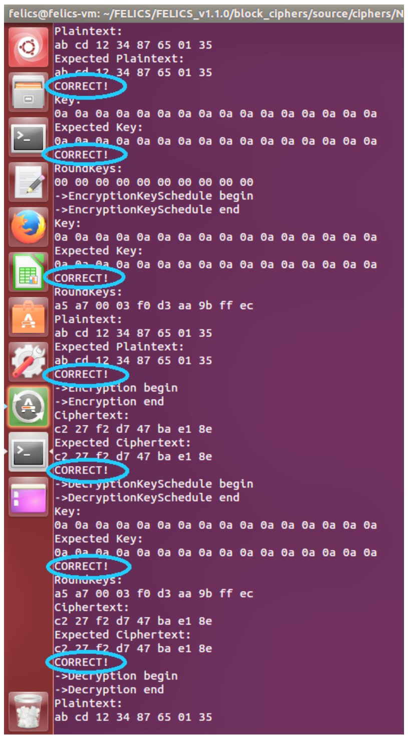
We implemented the suggested MSBK cipher in C. As an IDE, we utilized CodeBlocks. Machine-specific coding was not used in the development of the code. As well, we utilized a Linux-based benchmark program called FELICS to analyze memory consumption and execution cycles. FELICS is a free and open-source tool that anybody may use. Encrypt and decrypt pictures are considered to test the security of our proposed MSBK encryption in MATLAB 2021a.

A. Evaluation Aspects

Key sensitivity, histogram, correlation, UACI, and NPCR are used to assess the algorithm's security. For key generation, encryption, and decryption, we also considered at the algorithm's memory use and processing time cycles for encryption, decryption, and key generation.

B. FELICS implementation

FELICS is another tool utilized by the MSBK cipher to evaluate memory use as well as execution cycles. FELICS [18] , a command-line interface similar to GCC, can be used to test and build any newly designed lightweight block cipher (GNU Compiler Collection). They included proper guidelines to enable a smooth implementation procedure in their website (www.cryptolux.org/). Using FELICS, we have built and tested MSBK cipher. Three unique scenarios may be utilized to put the code to the test. FELICS is a utility that runs on the Linux environment. The testing status of proposed ciphers for each round is depicted in Figure 5 , indicating whether or not the intended ciphertext can be produced after encryption using round keys. The round marks in Figure 5 indicate that the encryption was performed correctly in each round and that the plaintext was also deciphered successfully.



```
felics@felics-vm: ~/FELICS/FELICS_v1.1.0/block_ciphers/source/ciphers/N
Plaintext:
ab cd 12 34 87 65 01 35
Expected Plaintext:
ab cd 12 34 87 65 01 35
CORRECT!
Key:
0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a
Expected Key:
0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a
CORRECT!
RoundKeys:
00 00 00 00 00 00 00 00 00 00
->EncryptionKeySchedule begin
->EncryptionKeySchedule end
Key:
0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a
Expected Key:
0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a
CORRECT!
RoundKeys:
a5 a7 00 03 f0 d3 aa 9b ff ec
Plaintext:
ab cd 12 34 87 65 01 35
Expected Plaintext:
ab cd 12 34 87 65 01 35
CORRECT!
->Encryption begin
->Encryption end
Ciphertext:
c2 27 f2 d7 47 ba e1 8e
Expected Ciphertext:
c2 27 f2 d7 47 ba e1 8e
CORRECT!
->DecryptionKeySchedule begin
->DecryptionKeySchedule end
Key:
0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a
Expected Key:
0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a 0a
CORRECT!
RoundKeys:
a5 a7 00 03 f0 d3 aa 9b ff ec
Ciphertext:
c2 27 f2 d7 47 ba e1 8e
Expected Ciphertext:
c2 27 f2 d7 47 ba e1 8e
CORRECT!
->Decryption begin
->Decryption end
Plaintext:
ab cd 12 34 87 65 01 35
```

Figure 5. Testing the MSBK cipher by using FELICS

The MSBK cipher is simulated using FELICS, an open source benchmark tool for lightweight cryptography. It is employed with the aim of evaluating performance across various platforms (such as AVR, MSP, ARM, and PC). In the AVR architectural scenario, we collect performance

CIPHER	Key Length	Size of Block	RAM	Size of Code	Cycles (Key Generation)	Cycles (Encryption)	Cycles (Decryption)	Cycles (Total Execution)
AES	128	128	720	23090	3274	5423	5388	14085
HIGHT	128	64	288	13476	1412	3376	3401	8189
LEA	128	128	432	3700	4290	3723	3784	11797
PRESENT	80	64	274	1738	2570	7447	7422	17439
Simon	96	64	188	1370	2991	1980	1925	6896
Speck	96	64	124	2552	1509	1179	1411	4099
SIT	64	64	22	826	2130	876	851	3857
G-cipher	64	64	34	1228	1630	792	789	3211
MSBK cipher	64	64	34	1228	1416	792	789	2997

TABLE II. Comparison of Different Lightweight Algorithms on AVR Architecture

metrics such as execution cycles (encryption, decryption, and key generation), RAM footprint, and binary code size for the MSBK cipher as well as the other reported ciphers. It is simple to compare the new encryption to earlier ciphers. Table II compares the performance of several ciphers for execution cycles in the AVR architecture for single block of plaintext. It can be observed that the proposed method has the approximately lowest cycles for key generation, encryption, and decryption among the methods considered. As a consequence, MSBK requires less time to encrypt and decrypt the data.

Using bar charts, Figure 6 compares many existing ciphers with the recommended approach. Each cipher has a bar chart that shows the number of clock cycles required to generate keys, cipher text, plaintext and cipher text, and the overall number of clock cycles required. The graph in figure 6 clearly indicates that the MSBK technique needs fewer clock cycles than most of the other ciphers that have been presented.

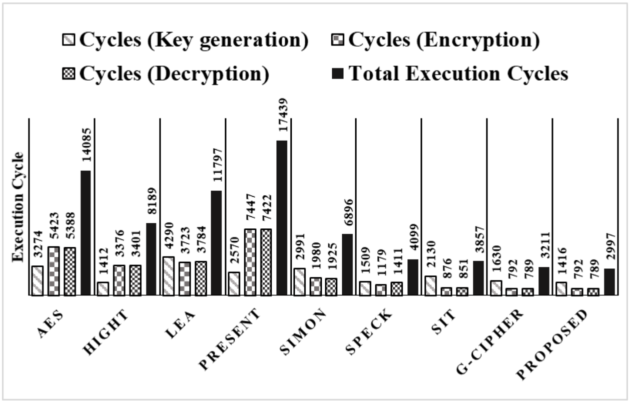


Figure 6. Execution Cycle Comparison for Hardware Implementation

C. Security Analysis using MATLAB

In MATLAB®, the MSBK cipher is also evaluated, which encrypts an image and then decrypts it with the correct key for graphical reflection of key sensitivity. The images are then decrypted using an incorrect key that differs by only one bit from the original key. This is also a test for the keys' avalanche effect. Despite only one bit change in the original keys, the ciphertext is unrecognizable. Figure 7 illustrates that the encrypted pictures for the MSBK encryption can only be deciphered with the right key. Hence, the statistical attacks are ineffective to break the MSBK cipher [8].

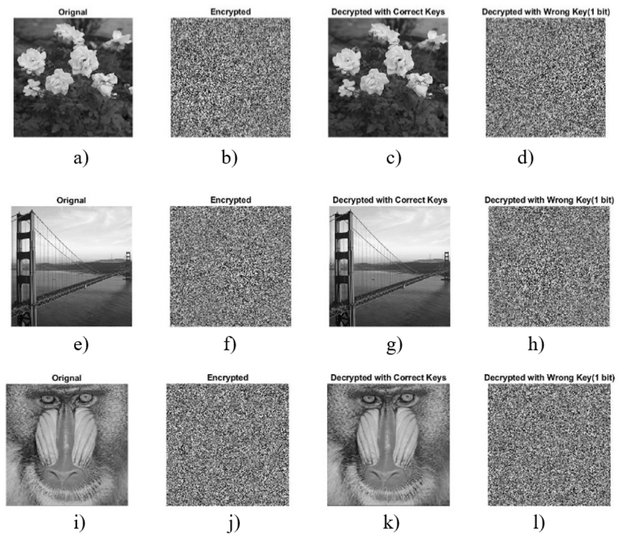


Figure 7. Statistical Analysis of Key Sensitivity

Figure 8 depicts the correlation between the original image and the encrypted image. As we can see from the graph, there is a linear link between the two images with a strong positive correlation value. Although the cipher image correlation graph reveals a great degree of unpredictability, as seen by the negative values. Negative correlation values for encrypted pictures, thus, point to the MSBK cipher as a robust encryption scheme.

The histograms of the original images are shown in Figure 9: a) Flower, e) Bridge, i) Baboon, and encrypted images: b) Flower, f) Bridge, and j) Baboon The vertical line represents the number of pixels, while the horizontal line represents the image's intensity. Encrypted images depict a uniform distribution in a histogram, indicating the MSBK cipher's security strength. As a consequence, statistical attacks such as chosen ciphertext and chosen plaintext are not susceptible to this cipher.

Using correlation coefficients for (a) flower, (e) bridge, and (i) baboon images, the security strength analysis of the MSBK cipher is shown in Table III. Original images have a greater positive correlation coefficient. The correlation coefficient of encrypted images, on the other hand, reveals a significant degree of unpredictability, i.e., negative values. A strong cipher is one that has a negative correlation value for encrypted images. Also number of pixel change rate (NPCR) is almost 100 percent that indicates the strong security of the proposed cipher.

The score of UACI for all tested images meet the security standard as shown in Table III for the proposed MSBK cipher. So, the security strength of the proposed MSBK cipher is as strong to prevent online and offline attacks.

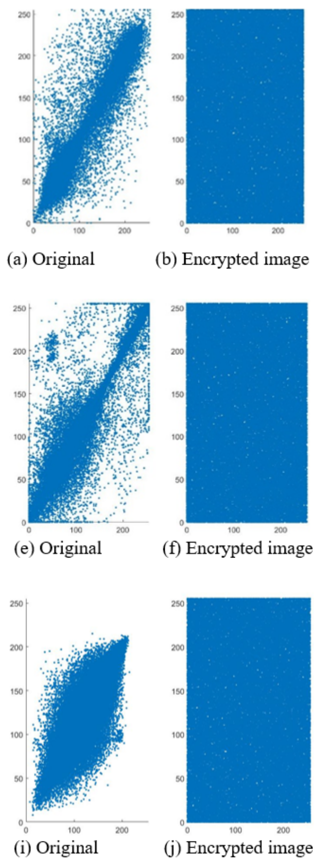


Figure 8. Correlation graphs for original and encrypted images

Images		Correlation coefficients	UACI	NPCR (%)
Flower (a)	Original	0.9569	23.1466	99.5987
	Encrypted	-0.0005		
Bridge (e)	Original	0.9626	14.7873	99.5941
	Encrypted	-0.0044		
Baboon (i)	Original	0.8198	13.3488	99.5575
	Encrypted	-0.0015		

TABLE III. Correlation Coefficients of Original and Encrypted Image

		Bridge	Baboon	Flower
Entropy H(S)	Encrypted image	7.9976	7.9972	7.9974
	Original image	7.5856	7.2316	7.2658

TABLE IV. Results of Information Entropy

The typical result of entropy is 8 that relates for the real randomness for a grayscale image. Here, proposed approach MSBK scores almost 8 of entropy for three different images as shown in Table IV.

D. Energy Consumption

The execution cycles of an algorithm on a certain device must be known before calculating the overall power utilized by an algorithm [19]. It is possible to calculate the power consumption of an algorithm on a certain device by

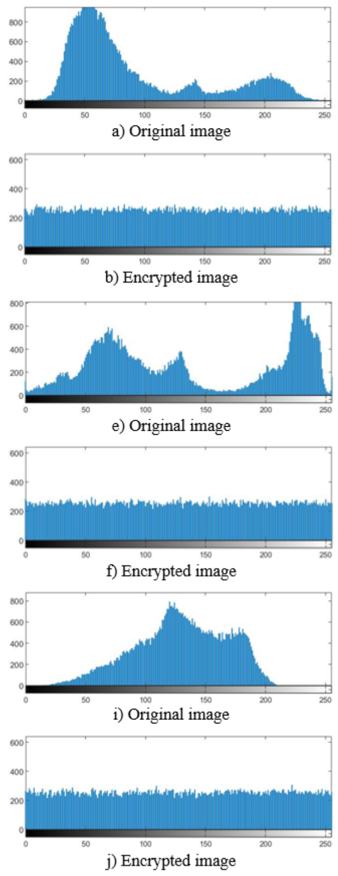


Figure 9. The histogram of original images and encrypted images

applying the following equation 6.

$$E = V_{cc} * I * N * \Gamma \tag{6}$$

When Γ refers to the time period in seconds, V_{cc} equals the operating voltage of the specific device, and I is the current in Amperes. It needs both the clock period Γ and the number N, which is the necessary number of execution cycles to calculate the power consumption. The time period of a specific device may be calculated if$f$ is the operating frequency in Hertz as $\Gamma = \frac{1}{f}$ second per cycle.

As specified by the absolute maximum rating (AMR) of the datasheet, the Atmega88/168 typically operates at 6V. There's a maximum current of 200mA, and it runs at 20 MHz. A comparison of the energy usage of several existing cryptosystems and the suggested MSBK scheme is shown in Figure 10. It is also noticeable from the bat graph that the suggested technique utilizes less energy than others. Finally, the MSBK cipher is faster in terms of encryption and decryption, reflecting that it uses a reduced amount of power than other ciphers.

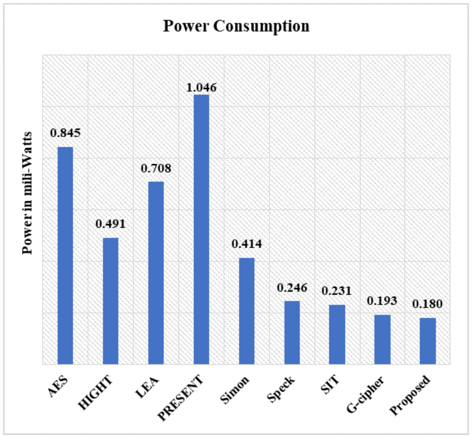


Figure 10. Energy consumption comparison of ciphers

5. Conclusions and Future Work

Devices with limited resources face a number of challenges in terms of security strength as well as performance. To achieve this goal, this study proposes a lightweight security technique on the butterfly architecture of the FFT. Compared to current cryptographic algorithms, the MSBK cipher employs fewer total execution cycles and needs a reduced amount of power over other reported ciphers. It is clear from the histogram, correlation graphs, and scores of NPCR, UACI, and image entropy that the MSBK cipher is resistant to most statistical attacks. This is because the real key for the MSBK encryption can only be used to recover encrypted images. For devices with limited resources, the MSBK algorithm would provide good security. Later, the proposed method's security level will be strengthened with the help of further mathematical analysis like the Chi-Square test to evaluate the randomness of generated keys.

Ethical disclosures

Conflict of interest: According to the study of authors, there is no conflict of interest in this research.

References

[1] W. Stallings, "Cryptography and Network Security Principles and Practices, Fourth Edition, Publisher: Prentice Hall," p. 592, November 16, 2005, $ISBN : 10 : 0 - 13 - 187316 - 4$.

[2] S. Rana, S. Hossain, H. I. Shoun, and D. M. A. Kashem, "An effective lightweight cryptographic algorithm to secure resource-constrained devices," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 11, 2018 10.14569/$IJACSA$.2018.091137 . [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2018.091137

[3] M. Noura, H. N. Noura, A. Chehab, M. M. Mansour, and R. Couturier, "S-DES: An efficient and secure DES variant," *2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)*, no. Article ID:8371019, pp. 1–6, 2018 *DOI* : 10.1109/$MENACOMM$.2018.8371019.

[4] Y. Y. Z. W. J. W. C. H. Biao Xing1, DanDan Wang1, "Accelerating DES and AES algorithms for a heterogeneous many-core processor." *International Journal of Parallel Programming (2021)*, vol. 2021, no. 463 - 486, p. 49, 2021, *DOI* : 10.1007/s10766 − 021 − 00692 − 4.(2021) . [Online]. Available: https://doi.org/10.1007/s10766-021-00692-4.(2021)

[5] R. Chatterjee and R. Chakraborty, "A Modified Lightweight PRESENT Cipher For IoT Security," *2020 International Conference on Computer Science, Engineering and Applications (ICCSEA)*, no. Article ID:9132950, pp. 1–6, 2020 *DOI* : 10.1109/$ICCSEA$49143.2020.9132950.

[6] A. L. Tomer Ashur, "An account of the iso/iec standardization of the simon and speck block cipher families." *Security of Ubiquitous Computing Systems.Edited by © 2020 Springer Nature Switzerland AG. Part of Springer Nature,Vol License CC BY 4.0*, vol. 2021, no. 63 - 78, p. 15, 2021, *DOI* : 10.1007/978−3−030−10591−4−42021 .

[7] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK lightweight block ciphers," *Proceedings of the 52nd Annual Design Automation Conference*, pp. 1–6, 2015.

[8] M. Usman, I. Ahmed, M. I. Aslam, S. Khan, and U. A. Shah, "SIT: A lightweight encryption algorithm for secure internet of things," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 1, 2017, 10.14569/$IJACSA$.2017.080151 . [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2017.080151

[9] S. L. T. T. P. Wang, Professor S. Chaudhry and H. Li, ""the internet of things: a security point of view"," *Internet Research*, vol. 26, no. 2, pp. 337–359, 2016.

[10] B. Preneel, "Understanding cryptography: A textbook for students and practitioners." 2009, *Springer, DOI* : 10.1007/978 − 3 − 642 − 04101 − 3.

[11] M. G. Fahad Qureshi and O. Gustafsson, "Unified architecture for 2, 3, 4, 5, and 7-point DFTs based on winograd fourier transform algorithm," *Electronics Letters, Institution of Engineering and Technology (IET)*, vol. 49, no. 5, pp. 348–349, February, 2013, *DOI* : 10.1049/el.2012.0577.

[12] A. A. D. M. A. K. Sohel Rana, Wadud, "A survey paper of lightweight block ciphers based on their different design architectures and performance metrics," *International Journal of Computer Engineering and Information Technology*, vol. 11, no. 6, 2019.

[13] T. K. A. Ekhlas Abbas Albahrani, "An image encryption scheme based on lorenz hyperchaotic system and RSA algorithm," *Security and Communication Networks*, vol. 2021, no. Article ID: 5586959, p. 18, 2021, *DOI* : 10.1155/2021/5586959 . [Online]. Available: http://dx.doi.org/10.1155/2021/5586959

[14] M. E. Haque, S. Zobaed, M. U. Islam, and F. M. Areef, "Performance analysis of cryptographic algorithms for selecting better utilization on resource constraint devices," *2018 21st International Conference of Computer and Information Technology (ICCIT)*, no. Article ID:8631957, pp. 1–6, 2018 *DOI* : 10.1109/$ICCITECHN$.2018.8631957.

[15] V. K. M. J. Eva Volna, Martin Kotyrba, "Cryptography based on neural network," *Proceedings 26th European Conference on Modelling and Simulation*, no. 386 - 391, p. 5, 2012, *DOI* : 10.7148/2012 − 0386 − 0391.

[16] S. Rana, M. R. H. Mondal, and A. H. M. S. Parvez, "A new key generation technique based on neural networks for lightweight block ciphers," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 6, 2021, DOI : 10.14569/$IJACSA$.2021.0120623 . [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2021.0120623

[17] K. Papapagiannopoulos, "High throughput in slices: the case of present, prince and katan64 ciphers," *Radboud University Nijmegen, Department of Digital Security*, 2016.

[18] J. G. D. K. Y. L. C. L. P. D. Dinu, A. Biryukov, "FELICS – fair evaluation of lightweight cryptographic systems," *University of Luxembourg*, July 2015.

[19] L. H. Utsav Banerjee and S. Koppula, "Power-based side-channel attack for aes key extraction on the atmega328 microcontroller," *Massachusetts Institute of Technology*, December 2015.

Sohel Rana(1st Author): He pursued his M.Sc. in ICT degree in 2021 from Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh. Received B.Sc. in CSE from Dhaka University of Engineering and Technology (DUET), Gazipur, Bangladesh. Now working in Bangladesh University of Business and Technology(BUBT) as a Lecturer from 2018. Interested research areas: Cryptography, Network Security etc.

Dr. Mohammod Abul Kashem (2nd Author): He started his career as an Assistant Professor at the dept. of CSE, Dhaka University of Engineering & Technology (DUET) in the year 2003. He joined the Department after completion of his B.Sc. and M.Sc.Engg. Degrees from State University "Lvivska Polytechnica", Ukraine in 1996 and 1997 respectively. In 2001, he earned Ph.D. in Control Systems and Processes from National University "Lviv Politechnic" Ukraine. Subsequently, Dr. Kashem completed his Post Doctorate fellowship from University Lumiera Lyon2, France. (Erusmus Mundas Scholarship, European Commission), 2016 and He was appointed as professor at the CSE Department of DUET in the year of 2013.