A Low Cost FPGA based Cryptosystem Design for High Throughput Area Ratio

Muhammad Sohail Ibrahim*, Irfan Ahmed†, M. Imran Aslam†, Muhammad Ghazaal*,
Muhammad Usman*, Kamran Raza* and Shujaat Khan*

*Faculty of Engineering Science and Technology (FEST),
Iqra University, Defence View,
Karachi-75500, Pakistan
†Department of Electronic Engineering,
NED University of Engineering and Technology,
University Road, Karachi 75270, Pakistan

***Abstract*—Over many years, Field Programmable Gated Arrays (FPGA) have been used as a target device for various prototyping and cryptographic algorithm applications. Due to the parallel architecture of FPGAs, the flexibility of cryptographic algorithms can be exploited to achieve high throughputs at the expense of very low chip area. In this research, we propose a low cost FPGA based cryptosystem named as Secure Cipher for high throughput to area ratio. The proposed Secure Cipher is implemented using full loop unroll technique in order to exploit the parallelism of the proposed algorithm. The proposed cryptosystem implementation achieved a throughput of 4600Mbps for encryption. The logic resource utilization of this implementation is 802 logic elements(LE) which yields a throughput to area ratio of 5.735Mbps/LE.**

Keywords*—Encryption; Cryptosystem; Secure Cipher; AES; FPGA; Full loop unroll

I. Introduction

Data security has been a topic of major interest since decades. With the development of communication systems, the techniques of data exchange have been revolutionized hence the need of data integrity and authenticity has also elevated. Various cryptosystems have been proposed in this regard. A cryptosystem is a software or a hardware that can convert data from its original comprehensible form into a scrambled form in such a way that the original information can be disclosed to some selected persons only [1], [2], [3]. Cryptosystems have evolved over the years from Ceaser's cipher, which was based on just shifting of letters, to the modern AES (Advanced Encryption Standard) proposed by Joan Daemen and Vincent Rijmen[4].

Cryptographic hardware solutions have been yet another field of interest for many researchers [5], [6]. Various hardware cryptosystems have been proposed in which the choice of hardware may be microcontrollers, microprocessors, and custom ASICs based cryptosystems. Each of the aforementioned hardware offer some merits and demerits, for instance, a microcontroller based design might have low processing capability but such a design usually takes low time to market. Similarly, an ASIC based solution can achieve very high data rates and power efficiency but require high time to market.

The hardware based designs can be compared on the basis of the following performance metrics. Power consumption, time to market, and Non Recurring Engineering (NRE) cost etc. Microcontroller based designs can be a choice for hardware implementation of cryptosystems as these designs are low cost and low power solutions and require very low time to market but their performance is also very low. For high performance requirements, a microprocessor based solution can be opted but such designs run on high power and their cost is also very high. Another class of microprocessor based solutions offer low cost and low power designs, but such microprocessors based solutions also offer very low performance. Hardware based solutions with high performance and low power can be designed on custom ASIC platform. ASIC designs are usually produced in mass volumes, so their per unit cost is also low but these solutions have high time to market as the generation of ASIC designs is a very complex process and in case of any error in the design the ASIC solution is redesigned which increases the NRE cost. For a high performance solution with low cost and low power consumption, FPGA based design is another candidate. These designs have very low time to market and have very low NRE cost of FPGAs due to the reconfigurability. The speed and efficiency of FPGAs combined with their flexibility makes them very attractive for cryptographic applications. The ability to reconfigure an FPGA to use a different cryptographic algorithm on the fly or to be able to update, modify or even replace an outdated algorithm make them very useful for designing cryptosystems. Likewise, low power and subsequently high throughputs that FPGAs are capable of make them very useful in high speed communications links or servers that often require security.

A. FPGA based Cryptosystem

There have been many FPGA based cryptosystem designs which focused on obtaining high throughputs. These designs often fully unroll the iterative round structure of the cryptosystem and rely heavily on pipelining within each round to increase throughput. High throughput FPGA designs typically achieve throughput above 20 Gbps and are intended to use in solutions that need to handle multiple security sessions simultaneously.

An FPGA based implementation of AES proposed by

T. Hoang used an iterative looping technique to implement AES for a block size of 128-bits[7]. In [8] another compact implementation of AES on FPGA is proposed. AES with block size of 128-bits was targeted to be implemented on FPGA. The key objective of that implementation was to keep the design as small as possible. The design achieved a throughput of 166Mbps at the expense of 222 slices and 3 block RAMs of 4Kbits each. In [9], the design decisions that lead to area/delay trade-offs in a single chip FPGA based cryptosystem is explored for AES. The design achieved a throughput of 23.57Gbps with 16938 slices of hardware area. G. Rouvroy proposed an efficient solution to combine AES encryption and decryption in one FPGA design keeping focus on low area constraint[10]. The proposed design achieved a throughput of 208Mbps using 163 slices and 3 blocks RAM only. In another research[11], a high performance encryptor/decryptor core of AES is presented. The design was implemented on a single-chip FPGA using fully pipelined technique. It uses 5677 slices and resulted in 4121Mbps throughput. Similarly, in [12], a fully pipelined AES encryption only design is presented. The design implemented on a single FPGA chip achieved a throughput of 21.54Gbps using 84 block RAMs and 5177 slices. In [13], another low power and low cost hardware core of AES algorithm is proposed. The core was designed with a novel 8-bit architecture that supports encryption with a 128-bit key. The design produces 121Mbps throughput at 153MHz clock frequency. In [14], four different architectures for AES-128 bits algorithm implementation are proposed. The four design techniques proposed in [14] are accurate floor-planning, unrolling, pipelining and tiling. These architectures were derived for different area-delay trade-offs. In [15], an efficient pipelined hardware implementation of AES-128 is proposed. The implementation will stay efficient even after increasing the required number of rounds to encounter attacks.The iterative looping with multi-stage sub-pipelining AES architecture is proposed in [16]. The design achieved 1.33Gbps throughput at 425MHz operating frequency. The logic resource utilization of the design is 303 slices. Another low cost AES implementation was proposed in [17]. This implementation proposed a high throughput design by the introduction of parallel operation in folded architecture. This implementation produced 37.1Gbps throughput at the maximum operating frequency of 505.5MHz.

Besides the AES, various other algorithms are also used to design FPGA based cryptosystems. S. Singh recently proposed a hardware implementation of RSA algorithm[18]. The authors have implemented RSA encryption using left to right radix-2 montmgomery multiplier on Xilinix Spartan-3 device. The design had a logic area utilization of 503 slices. The RSA algorithm FPGA implementation achieved 79.546MHz maximum clock frequency. In [19], an encryption scheme for real-time video streaming and its FPGA implementation has been proposed.

The demand of lightweight cryptographic algorithms has greatly increased due to the development and use of low resource devices for communication. In [20] a lightweight cipher named HIGHT, that provides adequate security at limited resource utilization is proposed along with its FPGA implementation. The authors presented pipelined and scalar (LUT) implementations of HIGHT with a claim of 18 times improved throughput at 60% less power consumption in pipelined design as compared to their LUT based design. In [21], the Minalpher algorithm and its implementation on various FPGA devices with simple and pipelined architecture is proposed. The performance of Minalpher algorithm was evaluated on resource constrained hardware.

The encryption process in standard algorithms is usually carried out by creating confusion and diffusion in the data. This objective is achieved by various operations such as shifting, transposition, various logical operation, and multiplication operations. Modern advancements in the field of data security suggest the use of algorithms that can be embedded in resource constrained devices such as smart phones, PDAs, etc [22]. Such devices have low on-board resources of memory and chip area, therefore, it is suggested to use algorithms with as low as possible complexity with adequate security. For this purpose, many researchers have proposed lightweight ciphers. The hardware implementation of such a lightweight block cipher named LEA is proposed in [23]. The algorithm was generally intended for software efficiency, therefore, the S-BOX structure was designed to have simple addition, rotation and XOR operations. The authors proposed a custom ASIC design which achieved a throughput of 533.3, 457.1, and 400 Kbps for key sizes of 128, 196, and 256 bits respectively at the operating frequency of 100KHz only. Furthermore, the design achieved 800Mbps throughput at 100MHz operating frequency for the key size of 256 bits. A full loop unroll architecture based FPGA implementation of a lightweight cryptographic algorithm named Secure Force is presented in [24]. The design achieved a throughput of 3.43Gbps at 53.5MHz operating frequency. In [25], an algorithm named Triple Hill Cipher, that can secure any binary data such as video, images, or audio data is proposed. The FPGA implementation of the algorithm achieved the maximum operating frequency of 528MHz at the expense of 4636 slices only.

B. Motivation and Organization of Paper

The ability of an FPGA to process data in parallel has attracted many researchers to use FPGA as a target device for the implementation and prototyping of a cryptosystem. Apart from keeping the algorithm efficient and lightweight, many programming techniques can be adopted to achieve high throughputs while keeping the chip area to the minimum. Such techniques include pipelining, full loop unrolling, sub-pipelining, partial loop unrolling etc [26].

In this paper, we propose a novel cryptosystem named Secure Cipher and its FPGA implementation. The rest of the paper is organized as follows; in section II, the proposed algorithm and its implementation is discussed. The experimental setup, evaluation criteria, and results are discussed in section III followed by the conclusion in section IV.

II. Proposed Cryptosystem and FPGA Implementation

The primary goals of any hardware cryptographic implementation are high throughput, low latency, low chip area, high operating frequency, and low power dissipation [27]. Since all these goals can never be achieved in a single hardware implementation, therefore, trade-offs are generally considered . These trade-offs are generally between delay or latency and chip area or resource utilization.

A. Secure Cipher

Many lightweight encryption algorithms have been proposed that are computationally inexpensive [28] The proposed Secure Cipher is low complexity encryption algorithm based on Feistal structure. It is a block cipher that consists of 5 encryption rounds only. Each encryption round consists of five logical and mathematical operations that operate on 8-bit data. This creates adequate confusion and diffusion in the data to confront various types of attacks. The proposed cryptosystem consists of the following blocks.

1) Key Generation Block: Key generation block generates five keys for each encryption and decryption round. The key generation block takes a 128-bit key as an input and generates round keys(K_r) of size 32 bits for each encryption/decryption round. The key generation block performs logical operations such as XOR and XNOR, fixed matrix multiplication, and left shift. Each of the logical notations have been displayed in figure I.

TABLE I: Notations and their Functions

Operation	Multiplication	XOR	XNOR
Notation	$\otimes$	$\oplus$	$\odot$

The input key (K) is an array of 128-bits, which is divided into 4 halves of 32-bits each. Each block of 32-bits is arranged in the form of a 4×8 matrix. Shift row operation is applied to each of the 4 matrices. Each of the shifted matrices are then arranged in an 4×8 matrix column-wise, on which XNOR logical operations are performed. The results of XNOR operation are stored in 4 matrices of the size 4×8 in column-wise fashion. These matrices then undergo a shift row operation and then multiplied with 4 individual fixed matrices of the size 8×4. The four fixed matrices labelled FM_1, FM_2, FM_3, and FM_4 are defined in equations (1),(2),(3), and (4) respectively. The detailed diagram of key generation block is shown in figure 1.

$$FM_1 = \begin{bmatrix} 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ 64 & 32 & 16 & 8 & 4 & 2 & 1 & 128 \\ 32 & 16 & 8 & 4 & 2 & 1 & 128 & 64 \\ 16 & 8 & 4 & 2 & 1 & 128 & 64 & 32 \end{bmatrix} \quad (1)$$

$$FM_2 = \begin{bmatrix} 8 & 4 & 2 & 1 & 128 & 64 & 32 & 16 \\ 4 & 2 & 1 & 128 & 64 & 32 & 16 & 8 \\ 2 & 1 & 128 & 64 & 32 & 16 & 8 & 4 \\ 1 & 128 & 64 & 32 & 16 & 8 & 4 & 2 \end{bmatrix} \quad (2)$$

$$FM_3 = \begin{bmatrix} 128 & 32 & 64 & 8 & 16 & 1 & 4 & 2 \\ 64 & 128 & 8 & 1 & 32 & 4 & 2 & 16 \\ 1 & 16 & 4 & 32 & 128 & 8 & 64 & 2 \\ 32 & 2 & 128 & 4 & 16 & 64 & 1 & 8 \end{bmatrix} \quad (3)$$

$$FM_4 = \begin{bmatrix} 2 & 16 & 64 & 128 & 1 & 32 & 4 & 8 \\ 64 & 1 & 4 & 16 & 32 & 128 & 16 & 2 \\ 1 & 128 & 32 & 16 & 4 & 2 & 64 & 8 \\ 4 & 1 & 128 & 32 & 64 & 8 & 16 & 2 \end{bmatrix} \quad (4)$$

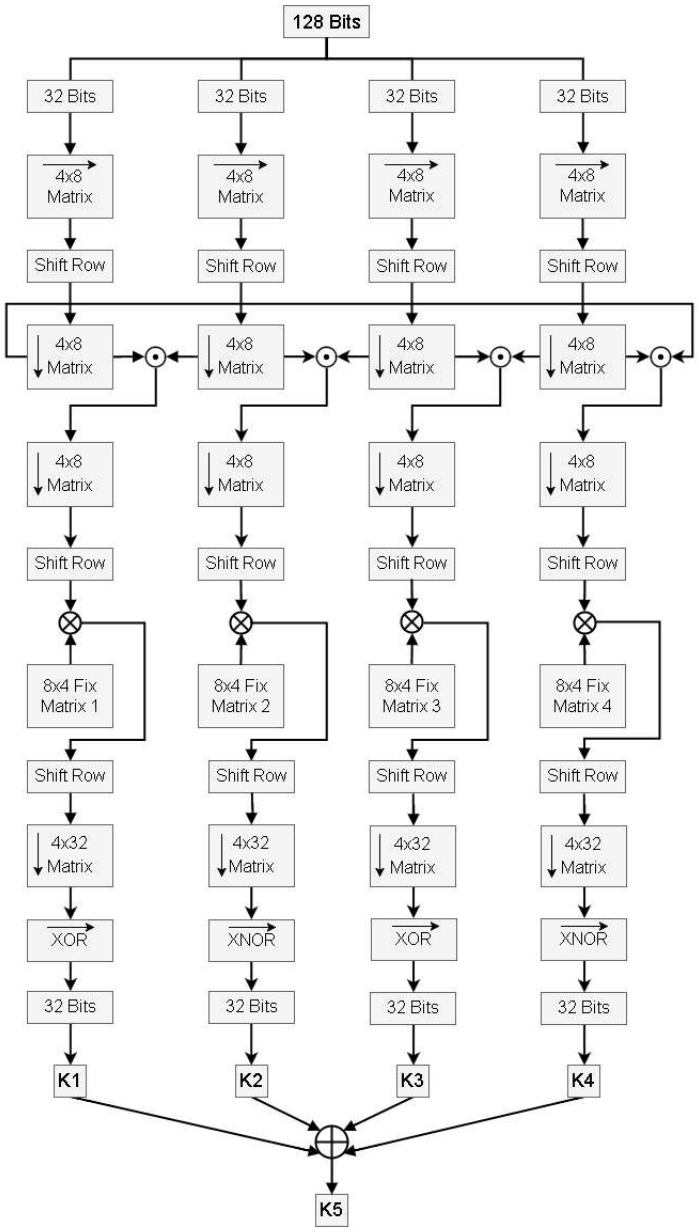


Fig. 1: Key expansion

2) Encryption Block: The encryption process consists of very simple operations of XOR, XNOR, left shift (LS), swapping operations, and Substitution Boxes (SBOX). The 128-bits wide plain text (X) is divided into two parts of 64-bits each, and these 64-bit halves are further divided into 32-bits. Swapping of 32-bits is performed in each round in order to alter the position of data hence increasing the complexity of the cipher. The round keys (K_r) are XNOR with the data in each round as shown in figure 2.

The block labelled "F" in figure 2 is the principle block in encryption process as it contains the substitution boxes. Figure 3 presents the process of the F function. The 32 bits input of F block are divided into 4 halves of 8-bits each. The first 8

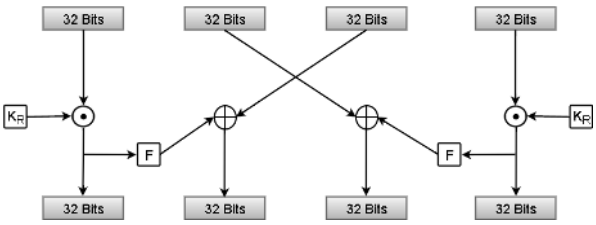


Fig. 2: A single encryption round

bits are moved to SBOX1 without performing any left shift operation. The second, third, and fourth 8 bit halves are left shifted by 1, 2, and 3 bits respectively. These left shifted halves are then moved to moved to the substitution boxes (SBOX1, SBOX2, SBOX3, and SBOX4) and then the results of each SBOX is concatenated to form 32 bits again. The structure of each of the SBOX is shown in figure 4.

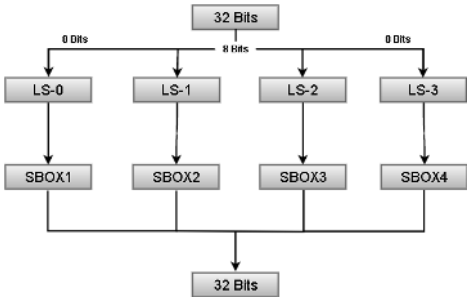


Fig. 3: F Function

An example of the substitution of data using SBOX is shown in figure 5. Each of the substitution box is generated in such a way that the output of any two or more than two substitution boxes cannot be the same despite the chance of having exactly the same selection byte. The SBOX operation takes place when the result of 32 bits data and round key K_R is divided into 4 halves of 8 bits each. Each of these 8 bit halves go through left shift operation as shown in figure 3. The 8 bit data is then moved to substitution boxes as the selection byte for the respective SBOX. The SBOX transformation takes place as; the 2 bits from MSB and 2 bits from LSB of the selection byte concatenate to give the row number of the SBOX, and the remaining 4 bits make the column number. In the example shown in figure 5, the output of the SBOX is $8C$, which is the $SB1_{(0,15)}$ entity of the SBOX1.

B. FPGA implementation

The overall hardware architecture of the Secure Cipher is based on loop unrolling technique. It is reported in [29] that loop unrolling is the main technique to achieve higher degrees of parallelism in reconfigurable hardware such as FPGA. It is also reported that loop unrolling increases the area but can also improve the throughput. The Secure Cipher is implemented on Altera Cyclone II EP2C35F672C6N FPGA using Verilog HDL.

As stated earlier that the design was lead out using full loop unroll technique. In this implementation, the iterations of

SB1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	63	7C	77	7B	F2	6B	6F	C5	30	O1	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	O4	C7	23	C3	18	96	O5	9A	O7	12	80	E2	EB	27	B2	75
4	O9	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	OO	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	O2	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
10	E0	32	3A	0A	49	O6	24	5C	C2	D3	AC	62	91	95	E4	79
11	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	O8
12	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
13	70	3E	B5	66	48	O3	F6	0E	61	35	57	B9	86	C1	1D	9E
14	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
15	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Substitution box # 1

SB2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	O4	93	8O	C9	24	BC	5F	88	E5	47	OF	81	5E	9O	6D	34
1	75	38	4C	OB	1C	8O	8E	DB	92	74	B4	C4	2D	E9	A6	F1
2	OC	D8	O7	B3	3A	6A	O7	4C	FO	C5	77	68	F4	B7	6D	5D
3	D9	36	CC	87	81	26	E5	27	B7	96	B2	6B	E1	92	D6	1O
4	9B	AA	3F	CO	1D	EO	89	92	6O	F3	OE	41	9B	D1	BE	EC
5	AC	A5	73	9O	93	69	8O	D5	A1	65	23	AF	81	E5	C5	32
6	72	FE	BD	5B	CD	BF	3D	53	DE	E7	72	21	8F	B3	B1	O5
7	2A	4E	6F	OF	5E	62	BF	AO	61	36	6D	49	62	89	AB	E8
8	74	67	AF	2A	78	A8	FC	A7	E5	24	OF	O5	59	7B	6F	EC
9	8F	EA	25	D5	7C	DA	78	34	54	BO	EB	A9	3O	ED	6O	35
10	CD	38	6O	68	6O	47	AD	6O	B2	41	8A	DE	CF	75	88	C7
11	36	EC	12	24	5C	CC	C2	A1	F6	AC	BB	EF	C2	28	7A	13
12	B9	CA	5F	35	8O	3O	71	34	54	25	75	E5	A1	66	44	3C
13	D4	57	D7	FF	OB	74	DF	F2	37	E8	8O	CA	A6	D4	C8	3F
14	9B	32	69	3A	E3	DA	B2	BO	DC	13	7O	5O	AO	98	9E	B6
15	4A	E6	2O	24	B8	42	C5	1F	8A	6B	2F	CB	8A	95	88	91

Substitution box # 2

SB3	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	82	89	7F	11	A5	4A	41	41	75	64	42	CC	14	83	6C	6D
1	A5	1B	A9	22	C8	5F	C4	9F	B2	FB	89	1D	D5	E1	8O	12
2	15	4C	E2	55	12	79	O7	2C	B8	OO	39	D7	7E	A1	13	49
3	88	48	B5	F1	72	8F	5O	28	O7	71	E4	C1	53	OA	27	79
4	EE	C8	79	88	77	24	44	DB	65	3O	73	7B	DE	14	23	85
5	DD	35	B6	51	86	D7	1E	B3	21	89	EB	BC	31	34	AB	C2
6	1F	58	OD	FA	B4	A4	97	46	OC	BF	96	9A	7B	D3	DB	A4
7	23	A5	36	76	69	52	E2	1D	DE	2F	53	EC	E6	F7	54	29
8	BA	7D	59	4E	4E	15	B3	O7	A9	AF	C7	A3	E4	DE	3A	1A
9	97	27	D9	O1	35	9B	19	1B	49	74	DB	7C	D6	3O	ED	FE
10	B1	29	D1	EO	86	FO	5D	61	A9	B3	C2	CO	FF	53	FE	1D
11	9B	A3	3A	EE	59	92	48	92	26	A8	B3	CE	F6	BO	45	59
12	49	7F	2C	AA	C1	75	AA	6D	C7	41	EF	DO	C4	DD	95	7D
13	1O	A3	43	C3	BC	C4	88	OO	AE	8D	72	D4	45	41	2E	39
14	9D	5B	EO	8O	AB	38	AD	D1	22	48	15	A6	65	C1	37	16
15	D4	5C	45	82	8O	3B	C6	5O	O5	56	DC	FO	68	BD	86	6F

Substitution box # 3

SB4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	96	9D	79	6E	11	77	9F	67	BB	EF	1O	37	17	98	1A	D8
1	A6	4B	OF	75	3C	B2	CE	D8	61	29	15	56	8E	A3	F8	45
2	84	94	OD	BO	A1	4A	41	61	E5	59	D7	DE	F1	5E	D1	O9
3	7O	B4	87	AF	C9	AC	DA	O2	OF	7D	1C	2C	F9	45	O2	FB
4	DE	2E	31	2C	69	F8	CB	A3	B1	7C	6D	AO	14	EC	5B	C6
5	95	52	C2	6A	48	6A	71	BE	82	79	56	CD	2C	B8	21	DC
6	55	24	97	88	3A	CA	OB	34	91	EO	6D	98	44	11	FE	9D
7	57	25	98	C1	D3	54	CF	FC	5D	58	33	88	4O	16	O1	D9
8	8O	97	O3	FC	83	EF	O7	C8	3A	5A	OC	BC	D5	6D	C4	C2
9	C1	EB	55	36	A5	BD	B6	A2	1F	4D	45	A8	54	26	EF	7C
10	2C	15	37	1F	OD	62	88	46	85	21	FA	FO	19	B7	8C	68
11	D8	D3	69	3B	OD	BA	FE	7E	1O	25	BO	D8	BE	5D	34	D7
12	DO	41	8F	7B	D4	O8	C5	E7	31	14	17	2C	5C	5E	A7	9D
13	EO	F3	79	23	44	O2	OB	5C	6E	A7	5B	D5	CD	81	63	4C
14	B9	3E	7A	39	17	CA	O8	E1	O6	2B	E7	99	F7	F3	95	1C
15	E7	O4	31	DF	2F	3A	57	8A	99	FC	9D	AB	31	F7	8E	6F

Substitution box # 4

Fig. 4: Substitution Boxes

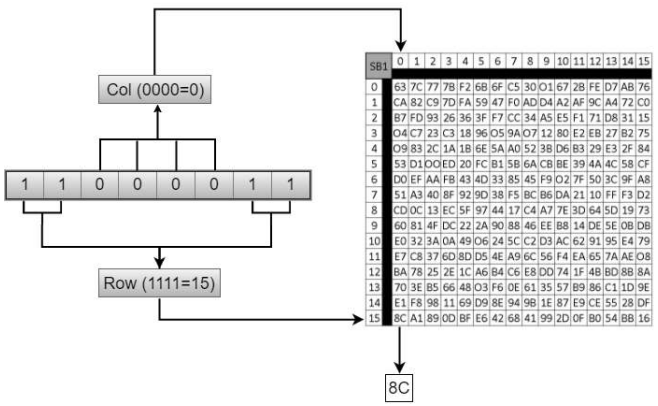


Fig. 5: Sbox Transformation

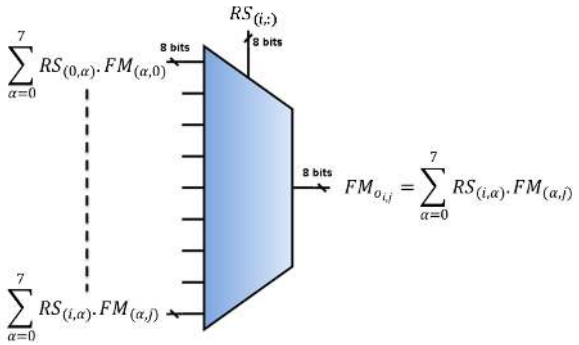


Fig. 6: Fixed Matrix Multiplication

every loop in the algorithm are unrolled in such a way that the output of each iteration becomes the input of the successive loop iteration. The hardware design of every sub-module of the Secure Cipher will be described in the respective subsections.

1) Key Genration Module: Key generation block generates the keys for individual rounds. This block takes 128 bits as an input key and performs various logical operations. This is to create enough confusion and diffusion in the input key in order to eliminate the chance of generation of weak keys. The key expansion process of the proposed Secure Cipher relies mainly on logical operations such as OR, AND, XOR, XNOR, Left shift, transposition, and permutation operations. Permutation and transposition operations are mapped by substitution.

Another operation included in key expansion block is the fixed matrix multiplication operation. There are four fixed matrices of the size 8×4, and these matrices hold fixed 8 bit integer values. As illustrated in figure 1, the output of the XNOR operation is arranged in an 4×8 matrix row-wise on which a left shift operation is applied. Each of these shifted matrices of 32 bits are multiplied with the fixed matrix which results into a 4×4 matrix of 128 bits. The obtained 4×4 matrix then goes through a left shift operation. We observed that the fixed matrix multiplication produces results from a finite set of numbers, as there involves multiplication of a binary 1 or 0 with an 8 bits wide number. Therefore, instead of using hard multiplier blocks, we transformed the fixed matrix multiplication problem into fixed look up tables. Each entity of the result of fixed matrix multiplication is defined by the equation shown as the output of the look up table presented in figure 6.

In figure 6, RS is defined as the row shifted matrix of size 4×8, FM is defined as the fixed matrix of size 8×4. The result of the equation is defined as the $(i, j)th$ entity of the matrix labelled as fixed matrix multiplication output FM_o.

The hardware of the fixed matrix multiplication is illustrated in figure 6. The select line of the look up table is the 8 bits wide row of the left shifted matrix RS, which depicts that the 8 bits wide output FM_o of the look up table can be selected form 256 possible input combinations. Each of the input is the product of ith element of the row of RS matrix with the jth element of the column of fixed matrix FM.

2) Encryption Module: The encryption module of Secure Cipher consists of simple logical operations (AND, OR, XOR, and XNOR) and substitution boxes (SBOX). The encryption module takes 128 bits plain text as an input and divides it into 4 halves of 32 bits each. The encryption process continues as illustrated in figure 2. Since encryption is an iterative process, therefore full loop unroll technique is employed which unrolls all the five encryption rounds. The block diagram of loop unrolled encryption block is shown in figure 7.

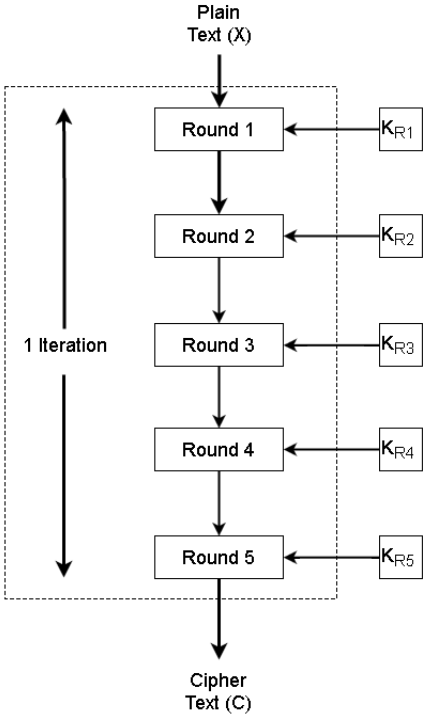


Fig. 7: Full Loop Unrolled Encryption

Each encryption round takes the output of the previous round and a round key K_R as an input. As mentioned earlier that the F function block displayed in figure 3 is the block of principle importance in encryption. Each of the SBOX in F function is an array of the size 16×16, which performs substitution. The hardware of SBOX, as shown in figure 8 is also a look up table which selects its output from 256 standard

values. The selection of output is displayed in figure 5. The encryption process is the same in all of the five rounds. At the end of the $5th$ round, the 32 bits wide outputs are concatenated to form the cipher text or encrypted message.

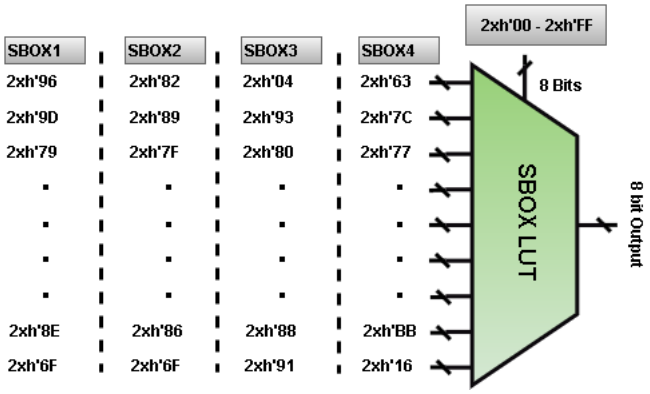


Fig. 8: Substitution box look up table

III. Experimental Setup

The security evaluation of cryptosystems is done on the well known parameters such as key senstivity test based on strict avalanche criterion(SAC), entropy, histogram, and correlation[30], [31], [32], [33], [34]. The hardware designs of cryptographic algorithms are generally compared on the basis of their logic resource utilization or area, propagation delay or latency, throughput, power consumption, and maximum operating frequency [35], [36], [37].

The target device for the proposed cryptosystem implementation is a low cost Altera Cyclone II EP2C35F672C6N FPGA. The details of the aforementioned evaluation parameters will be described in later subsections.

A. Evaluation Parameters

The performance of the Secure Cipher is evaluated on the following performance metrics. The results related to security were performed on MATLAB software. And the hardware performance evaluation parameters such as area, propagation delay, and throughput were performed on Altera Cyclone II FPGA using Quartus II 12.1 sp1 edition software.

1) Key Sensitivity: Key sensitivity of cryptosystems is tested on the basis of Strict Avalanche Criterion (SAC). The SAC states that "If a function is to satisfy the strict avalanche criterion, then each of its output bits should change with a probability of one half whenever a single input bit is complemented"[38]. For key sensitivity test, the the cipher text should change with a probability of 50 %.

2) Image Entropy and Correlation: Entropy is the measure of information content of the data. The entropy of the encrypted data should be high so that the data cannot be recognized after encryption. And correlation is defined as the measure of similarity between the adjacent pixels of an image. For an efficient cryptosystem, the results of correlation of an encrypted image should be as low as possible so as to ensure that the data is scrambled adequately.

3) Histogram: For the security related testing, we performed the tests on image data since the results in the visual form can be understood easily. The histogram of an image before encryption shows the intensity variation of the image pixels. For an encrypted image, the pixel intensity should be uniform. This shows the randomness created in the image after encryption.

4) Area: The area in FPGAs is measured in terms of the logic units or circuits being used by the design. For Altera Cyclone II FPGA family, the resource utilization or area is measured in terms of the number of logic elements (LE), whereas for Xilinx Spartan FPGAs, the term logic circuits (LC) is used. A logic element (LE) contains a 4 input Look-Up Table (LUT), a D flip-flop, and a register for carry chain connection.

In [26], it is reported that the cryptosystems designed with full loop unroll technique may have larger area on hardware as compared to partial loop unrolled architectures, but such designs can achieve high throughputs.

5) Propagation Delay: Propagation delay is defined as the maximum amount of time that exists between the edges of signal when it propagates from input to the output of a given circuit, so, it is the amount of time for the slowest signal to propagate from input to output in a circuit. The propagation delay can be greater if the circuit has complex operations and large area. In general, the propagation delay can be high for full loop unroll designs, but it can be low if the algorithm's flexibility is properly utilized. For instance, in the proposed algorithm, the fixed matrix multiplication is the most complex mathematical operation and it can cause higher delays even if hardware multiplier blocks are used to perform multiplication. But instead of using the multiplier blocks, we propose to implement this multiplication on a simple look-up tables problem which is very low in terms of complexity as compared to the conventional multiplication operation. Such look-up tables implementations cause much less propagation delays as compared to hard multiplier blocks.

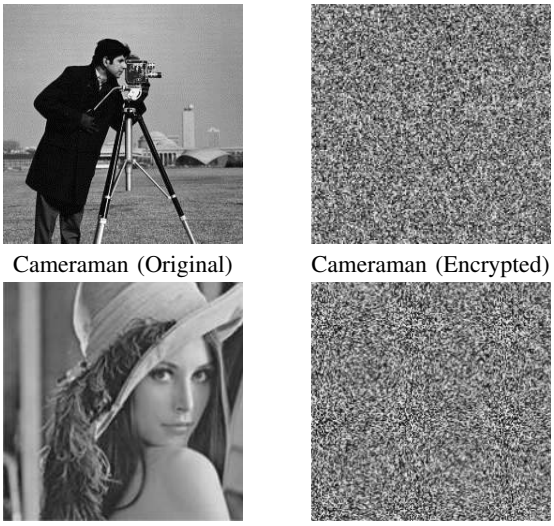
6) Throughput: Throughput is referred as the primary measure of speed for a hardware based cryptosystem. For hardware implementation of algorithms, throughput is the measure of the amount of data (in bits) processed per unit time. Modern hardware cryptosystems posses high speed data links, therefore, their throughputs should be high enough to be in orders of $Mbps$ to $Gbps$ so as to utilize the high data link speeds.

B. Results

The evaluation parameters related to security have been described in section III-A. The proposed Secure Cipher performs adequately in terms of security. The visual testing results of image encryption using the proposed Secure Cipher have been displayed in figure 9. The security related tests have been performed on images of the size 256×256 named Cameraman and Lena. It can be seen in the figure 9 that the encrypted images are impossible to identify visually.

The key sensitivity is tested on strict avalanche criterion (SAC). Based on the SAC, we calculated the mean percentage avalanche value for 1000 variations in the input key and plain

text and achieved the mean percentage avalanche value of 54.55%.



Cameraman (Original) Cameraman (Encrypted)

Lena (Original) Lena (Encrypted)

Fig. 9: Image encryption visual results

As mentioned in section III-A, the histogram of the encrypted image should be uniform so that each pixel contains nearly the same information content. The histogram results of the original and encrypted images have been shown in figure 10. Whereas the correlation results of original and encrypted images have been shown in figure 11.

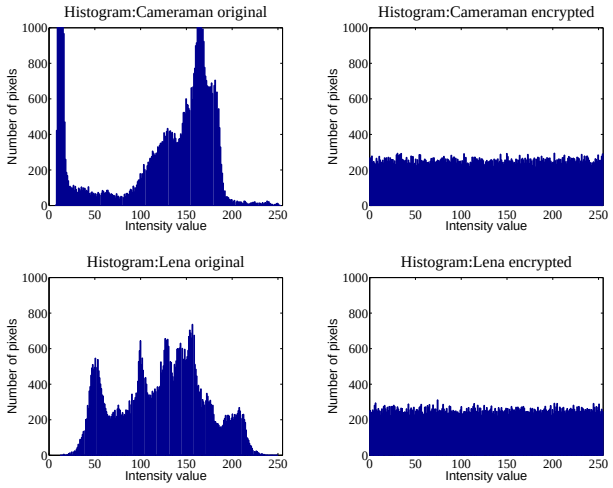


Fig. 10: Histogram of original vs. encrypted images

The proposed Secure Cipher was implemented on Altera DE2 board with Cyclone II EP2C35F672C6N FPGA. The design was synthesized using Quartus II 12.1 sp1 edition. The FPGA implementation results are listed in table II. In [26], it is reported that the hardware implementations with full loop unroll architectures may occupy high area. But the proposed Secure Cipher has low algorithmic complexity such that the resource utilization of the proposed Secure Cipher is lower than [15], [25], [23], and [39].

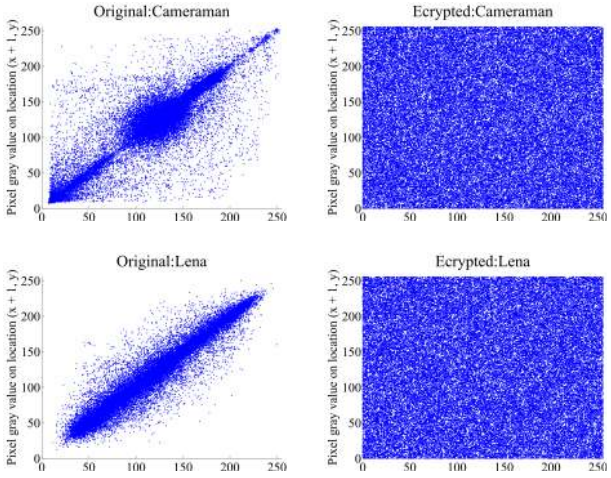


Fig. 11: Correlation of original vs. encrypted images

The throughput to area ratio should be high for a hardware cryptosystem as it shows the contribution of a single LE in the speed of the hardware design. It is evident from the results displayed in table II that the proposed Secure Cipher has higher throughput to area ratio than the designs presented in [15], [25], and [39] as mentioned in table II.

TABLE II: Comparison of Implementation Results

Design	Device	Propagation Delay (ns)	Throughput (Mbps)	Area (LEs)	Throughput Area Ratio (Mbps/LE)
HIGHT (2016) [20]	Cyclone II	14.97	4275.2	632	6.76
Triple Hill (2014) [25]	Virtex-4	1.894	67581.8	4636	14.57
LEA (2014) [23]	Cyclone III	200	650.19	813	0.8
DES (2015) [39]	Vertix II	2.182	278.26	303	0.918
Secure Cipher	Cyclone II	13.925	4600	802	5.735

IV. Conclusion

Reconfigurable hardware devices such as FPGAs play a vital role in assessing the performance of cryptographic block ciphers on hardware platform. The proposed cryptosystem named Secure Cipher was designed on FPGA using Full Loop Unroll architecture. The hardware performance results are promising in terms of area, and throughput as the complete design was implemented on Altera Cyclone II FPGA using 802 LE only. And the proposed system has a throughput of 4600Mbps with 5.735Mbps/LE throughput to area ratio. Whereas the proposed Secure Cipher ensures adequate security with a percentage SAC value of 54.55%. For future considerations, the pipelined design of the proposed cryptosystem can be implemented which would help in evaluating the flexibility of the proposed Secure Cipher.

REFERENCES

[1] S. Khan, M. Ebrahim, and K. A. Khan, "Performance evaluation of secure force symmetric key algorithm," 2015.

[2] M. Ebrahim, S. Khan, and U. Khalid, "Security risk analysis in peer 2 peer system; an approach towards surmounting security challenges," *arXiv preprint arXiv:1404.5123*, 2014.

[3] M. Ebrahim, S. Khan, and S. S. U. H. Mohani, "Peer-to-peer network simulators: an analytical review," *arXiv preprint arXiv:1405.0400*, 2014.

[4] J. Daemen and V. Rijmen, "Aes proposal: Rijndael," 1999.

[5] S. Khan, M. S. Ibrahim, K. A. Khan, and M. Ebrahim, "Security analysis of secure force algorithm for wireless sensor networks," *arXiv preprint arXiv:1509.00981*, 2015.

[6] M. Ebrahim, S. Khan, and U. B. Khalid, "Symmetric algorithm survey: A comparative analysis," *International Journal of Computer Applications (0975 – 8887)*, vol. 61, no. 20, 2014.

[7] T. Hoang *et al.*, "An efficient fpga implementation of the advanced encryption standard algorithm," in *Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF), 2012 IEEE RIVF International Conference on*. IEEE, 2012, pp. 1–4.

[8] P. Chodowiec and K. Gaj, "Very compact fpga implementation of the aes algorithm," in *Cryptographic Hardware and Embedded Systems-CHES 2003*. Springer, 2003, pp. 319–333.

[9] J. Zambreno, D. Nguyen, and A. Choudhary, "Exploring area/delay tradeoffs in an aes fpga implementation," in *Field Programmable Logic and Application*. Springer, 2004, pp. 575–585.

[10] G. Rouvroy, F.-X. Standaert, J.-J. Quisquater, and J.-D. Legat, "Compact and efficient encryption/decryption module for fpga implementation of the aes rijndael very well suited for small embedded applications," in *Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on*, vol. 2. IEEE, 2004, pp. 583–587.

[11] F. Rodrıguez-Henrıquez, N. Saqib, and A. Dıaz-Perez, "4.2 gbit/s single-chip fpga implementation of aes algorithm," *Electr. Lett*, vol. 39, no. 15, pp. 1115–1116, 2003.

[12] A. Hodjat and I. Verbauwhede, "A 21.54 gbits/s fully pipelined aes processor on fpga," in *Field-Programmable Custom Computing Machines, 2004. FCCM 2004. 12th Annual IEEE Symposium on*. IEEE, 2004, pp. 308–309.

[13] P. Hämäläinen, T. Alho, M. Hännikäinen, and T. D. Hämäläinen, "Design and implementation of low-area and low-power aes encryption hardware core," in *Digital System Design: Architectures, Methods and Tools, 2006. DSD 2006. 9th EUROMICRO Conference on*. IEEE, 2006, pp. 577–583.

[14] G. P. Saggese, A. Mazzeo, N. Mazzocca, and A. G. Strollo, "An fpga-based performance analysis of the unrolling, tiling, and pipelining of the aes algorithm," in *Field Programmable Logic and Application*. Springer, 2003, pp. 292–302.

[15] N. Nedjah, L. de Macedo Mourelle, and C. Wang, "A parallel yet pipelined architecture for efficient implementation of the advanced encryption standard algorithm on reconfigurable hardware," *International Journal of Parallel Programming*, pp. 1–16, 2016.

[16] M. El Maraghy, S. Hesham, and M. A. Abd El Ghany, "Real-time efficient fpga implementation of aes algorithm," in *SOC Conference (SOCC), 2013 IEEE 26th International*. IEEE, 2013, pp. 203–208.

[17] K. Rahimunnisa, P. Karthigaikumar, S. Rasheed, J. Jayakumar, and S. SureshKumar, "Fpga implementation of aes algorithm for high throughput using folded parallel architecture," *Security and Communication Networks*, vol. 7, no. 11, pp. 2225–2236, 2014.

[18] S. Singh and P. S. Jassal, "Synthesis and analysis of 32-bit rsa algorithm using vhdl," 2016.

[19] F. Sbiaa, S. Kotel, M. Zeghid, R. Tourki, M. Machhout, and A. Baganne, "A format-compliant selective encryption scheme for real-time video streaming of the h. 264/avc," *International Journal of Advanced Computer Science & Applications*, vol. 1, no. 7, pp. 386–396, 2016.

[20] B. J. Mohd, T. Hayajneh, Z. A. Khalaf, A. Yousef, and K. Mustafa, "Modeling and optimization of the lightweight hight block cipher design with fpga implementation," *Security and Communication Networks*, 2016.

[21] M. Kosug, M. Yasuda, and A. Satoh, "Fpga implementation of authenticated encryption algorithm minalpher," in *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)*. IEEE, 2015, pp. 572–576.

[22] D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B.-S. Koo, C. Lee, D. Chang, J. Lee, K. Jeong *et al.*, "Hight: A new block cipher suitable for low-resource device," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2006, pp. 46–59.

[23] D. Lee, D.-C. Kim, D. Kwon, and H. Kim, "Efficient hardware implementation of the lightweight block encryption algorithm lea," *Sensors*, vol. 14, no. 1, pp. 975–994, 2014.

[24] S. Khan, M. S. Ibrahim, H. Amjad, K. A. Khan, and M. Ebrahim, "Fpga implementation of 64 bit secure force algorithm using full loop-unroll architecture," in *2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*. IEEE, 2015, pp. 1–6.

[25] A. A. Khalaf, M. S. A. El-karim, and H. F. Hamed, "A triple hill cipher algorithm proposed to increase the security of encrypted binary dataand its implementation using fpga," *Journal Editorial Board*, vol. 1, no. 3, p. 752, 2014.

[26] A. J. Elbirt, W. Yip, B. Chetwynd, and C. Paar, "An fpga-based performance evaluation of the aes block cipher candidate algorithm finalists," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 9, no. 4, pp. 545–557, 2001.

[27] S. Khan, M. S. Ibrahim, M. Ebrahim, and H. Amjad, "Fpga implementation of secure force (64-bit) low complexity encryption algorithm," *International Journal of Computer Network and Information Security*, vol. 7, no. 12, p. 60, 2015.

[28] M. Usman, I. Ahmed, 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 (IJACSA)*, vol. 8(1), no. 51, 2017.

[29] B. Buyukkurt, Z. Guo, and W. A. Najjar, "Impact of loop unrolling on area, throughput and clock frequency in roccc: C to vhdl compiler for fpgas," in *Reconfigurable Computing: Architectures and Applications*. Springer, 2006, pp. 401–412.

[30] A. Kumar and M. N. Tiwari, "effective implementation and avalanche effect of aes," *International Journal of Security, Privacy and Trust Management (IJSPTM)*, vol. 1, no. 3/4, pp. 31–35, 2012.

[31] S. Shivkumar and G. Umamaheswari, "Performance comparison of advanced encryption standard (aes) and aes key dependent s-box-simulation using matlab," in *Process Automation, Control and Computing (PACC), 2011 International Conference on*. IEEE, 2011, pp. 1–6.

[32] M. Zeghid, M. Machhout, L. Khriji, A. Baganne, and R. Tourki, "A modified aes based algorithm for image encryption," *International Journal of Computer Science and Engineering*, vol. 1, no. 1, pp. 70–75, 2007.

[33] D. S. A. Elminaam, H. M. Abdual-Kader, and M. M. Hadhoud, "Evaluating the performance of symmetric encryption algorithms." *IJ Network Security*, vol. 10, no. 3, pp. 216–222, 2010.

[34] J. W. Yoon and H. Kim, "An image encryption scheme with a pseudorandom permutation based on chaotic maps," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 12, pp. 3998–4006, 2010.

[35] K. Gaj, E. Homsirikamol, and M. Rogawski, "Fair and comprehensive methodology for comparing hardware performance of fourteen round two sha-3 candidates using fpgas," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2010, pp. 264–278.

[36] K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita, "Camellia: A 128-bit block cipher suitable for multiple platformsdesign andanalysis," in *International Workshop on Selected Areas in Cryptography*. Springer, 2000, pp. 39–56.

[37] S. Anis *et al.*, "Fpga implementation of parallel particle swarm optimization algorithm and compared with genetic algorithm," *International Journal of Advanced Computer Science & Applications*, vol. 1, no. 7, pp. 57–64.

[38] A. Webster and S. E. Tavares, "On the design of s-boxes," in *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 1985, pp. 523–534.

[39] M. Abdelwahab *et al.*, "High performance fpga implementation of data encryption standard," in *Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE), 2015 International Conference on*. IEEE, 2015, pp. 37–40.