Optimization of Block Cipher with SIMON

Shylaja C
PG Scholar, VLSI & Embedded System
Dept. of Electronics & Communication
SJCE,Mysuru.

Shreekanth T
Assistant Professor
Dept. of Electronics & Communication
SJCE,Mysuru.

ABSTRACT
In cryptography, a block cipher is a deterministic algorithm operating on fixed-length groups of bits, called blocks, with an unvarying transformation that is specified by a symmetric key. Block ciphers are important elementary components in the design of many cryptographic protocols, and are widely used to implement encryption of bulk data. A block cipher is a method of encrypting text (to produce cipher text) in which a cryptographic key and algorithm are applied to a block of data (for example, 64 contiguous bits) at once as a group rather than applying it to one bit at a time. The primary purpose of encryption is to protect the confidentiality of digital data stored on computer systems or transmitted via the Internet or other computer networks.

A recent promising low-cost alternative of advanced encryption standard (AES) on reconfigurable platforms called the SIMON. It has been implemented as the construction of the round function and the key generation of SIMON, that enables bit-serial hardware architectures which can significantly reduce the cost. Encryption and decryption can be done using the same hardware and also propose the hardware architecture of the smallest block cipher ever published on field-programmable gate arrays (FPGAs) at 128-bit level of security.

Keywords
AES, SIMON, Block Cipher, Feistel , FPGA.

1. INTRODUCTION
Encryption is a part of our day-to-day life, mainly used to anticipate monitoring on our communications like phone calls, Internet, security system associations, making e-business and e-banking achievable and also where ever hiding the information from visible eyes are required. According to the history of advancement, old encryption functions have been broken & reconstructed to new form of encryption functions. Here one type of encryption functions are block ciphers. A block ciphers contains two functions: encryption function and decryption function. The data blocks of bytes with fixed size are processed by Block ciphers. Using a block cipher same key is used for both receiver end and sender ends of the channel. With the utilization of same key to encrypt the plaintext blocks, results in the same cipher text block.At present utilized block cipher is Rinjdael, and that was chosen as the winner of National Institute of Standards and Technology (NIST) supported test for new encryption standard in 2000. After conquering of the Advanced Encryption Standard (AES) challenge, presently Rinjdael cipher is frequently known as **AES**. With Consideration of a block ciphers are capable of encrypting the data only for fixed-sized blocks; they use the mode of operation with block cipher to encrypt bulk sets of data.

Block ciphers can be examined as vital block of the symmetric-key encryption system, relatively being utilized just to encode the messages. The execution of encryption is analytical, particularly on the server side of intelligence, the block ciphers are intended to be quick in hardware and software application. For block ciphers encryption, input is fixed-size plaintext data, which is converted into a same cipher text data. Thus this fixed length data is known as block size. The frequently block sizes utilized in the present block ciphers are 64 bits and 128 bits.

1.1 Overview of Cipher Model
An encryption of symmetric type has following 5 factors as depicted in Fig.1 [2].

a) Plaintext: It is a unique message or information which is given as input to the algorithm.

b) Encryption algorithm: This performs transformation of plaintext into encrypted data.
c) Secret key: This is additionally another data to the algorithm. The key be an independent value. Depending on the secret key it will results the distinctive output used at the time. The algorithm performs transformation rely on upon the key.

d) Cipher text: It is a specified output from the algorithm. The results rely upon the plaintext and key. For a particular data, different two cipher texts generated by two different keys. Therefore cipher text is an irregular stream of information & is incomprehensible.

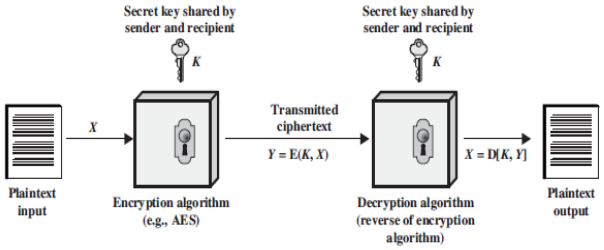


Fig.1: An encryption model of symmetric type.

The two requirements used for the conventional encryption are:
a) Require of strong algorithm such that if the opponent able to access one or more cipher texts would unable to decrypt the plaintext or discover the key for several cipher text is accompanied by the plaintext which generated by every cipher text.

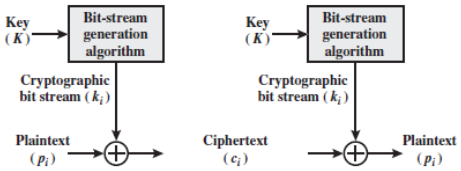
b) Receiver and sender should share the key which should be secure fashion. Here the only the key is kept in secure this is possible with symmetric encryption algorithm.

1.2 Block Cipher principles

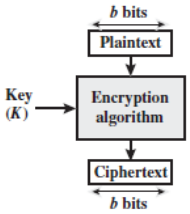
Currently many symmetric encryption algorithms used are relying upon the arrangement known as Feistel block cipher. It is critical to consider the configuration standards of Feistel cipher in which begins with the evaluation of stream ciphers and block ciphers.

1.2.1 Block Ciphers and Stream ciphers

A stream cipher can be defined as which encrypts the digital information stream either byte or bit at once. The cryptographic key stream is random in which the cipher is unbreakable means here key stream is provided in advance to both the users via independent and secure medium. In this method as Fig 2(a) shows bit stream generator, it is an algorithm of key control and generates the key stream in cryptographic method. Here both users shares only produced key, and individually results the key stream.A block cipher defined as the block of plaintext assumed as whole and equally generates the same length of cipher text. Typically block sizes utilized are 64 bits or 128 bits. A stream cipher uses the symmetric encryption as Fig 2(b) shows the same method followed by block cipher and also block ciphers are used in wide area than the stream cipher. The symmetric cryptographic applications utilize the block ciphers in required system.



(a) Bit-stream generator using Stream cipher.



(b) Block cipher

Fig.2: Stream Cipher and Block Cipher

1.2.2 Feistel cipher structure

Fig 3 shows the Feistel structure. For the encryption algorithm the inputs are plaintext of block length 2w bits and a key K. here the plaintext is equally divided into two halves, L_0 and R_0 .these two halves of the data processed through n rounds and combined the data to produce the cipher text block. Each round i has as inputs L_{i-1} and R_{i-1}, which is obtained from previous round, as well as a sub key K_i and from each other. In Fig 3 used 16 rounds, regardless of the possibility that any number of rounds can be actualized.All the round operation has similar configuration. A substitution method is performing on the left 50% of the information. The round function F is operated on the right 50% of the information and afterward taking exclusive-OR with result of round function F and left 50% of the information. The general structure of round function in every round is parameterized by the round sub key K_i.The subsequent substitution method follows a permutation which exchanges the two halves of the message.

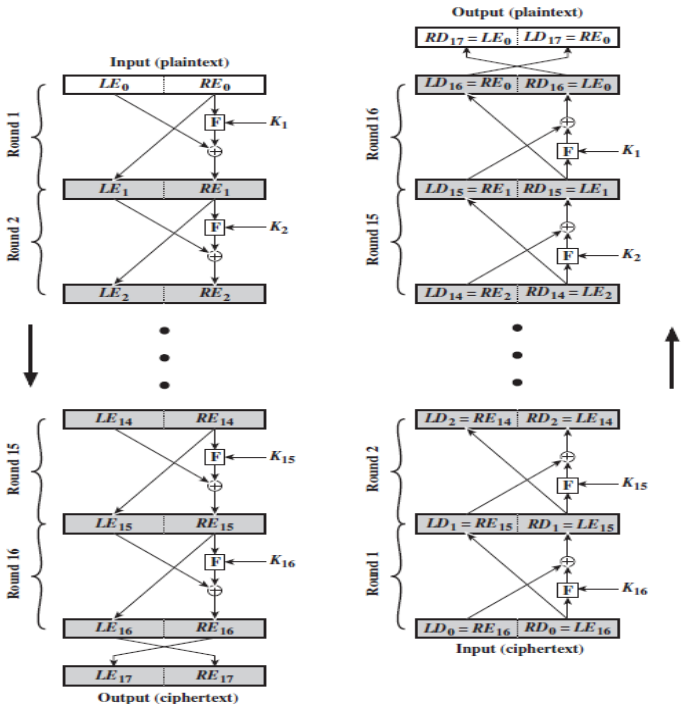


Fig.3: Feistel structure.

Parameters and design features for a Feistel network realization as follows:

a) Block size: considering the bigger block sizes give more prominent security yet lessened encoding and decoding rate for a specified algorithm. The superior security can be accomplished by more prominent diffusion. Generally for a 64-bit block size has been examine tradeoff and almost common in block cipher configuration. Though, the AES use a 128-bits block size.

b) Key size: Considering the bigger key sizes give more prominent security but reduced encoding and decoding rate. The superior security is accomplished by more prominent confusion. Now- a- days less use of 64 bits key sizes rather commonly used is 128 bit key size.

c) Number of rounds: The principle of Feistel cipher is that a solitary round which provides incomplete security however utilizing numerous rounds can expand the security. A common size is 16 rounds.

d) Sub key generation algorithm: Larger complication in algorithm is supposed to lead to more noteworthy trouble of cryptanalysis.

e) Round function F: Once more, more prominent intricacy regularly builds imperviousness to cryptanalysis.

The Feistel cipher follows two considerations in the design:

a) Fast software encryption/decryption: An encryption is implanted in applications or convenience functions that prohibit in the hardware implementation. Therefore considering mainly on speed of execution of the algorithm.

b) Ease of analysis: The algorithm is briefly and evidently defined, and then it is ease to evaluate, that algorithm for cryptanalytic vulnerabilities and consequently can add to a more elevated amount of quality.For example- DES.

2 METHODOLOGIES

The block ciphers are purposefully used on controlled devices, where a few devices have been intended for particularly to perform well on altered Application- Specific Integrated Circuits (ASICs), and along these lines it can be acknowledged by circuits with insignificant power necessities.Here proposed two groups of adequately streamlined block ciphers, SIMON and SPECK [4], which is adoptable to give incredible execution in both hardware and programming application. SIMON is specialized for efficient hardware implementations and SPECK is specialized for efficient software realizations. Also both families comprise of algorithms with different ranges of key and block size that can employ on several of applications.The design of SIMON and SPECK are families of block cipher: each supports block sizes of 32,48,64,96 and 128 bits, with three key sizes along with each block size. Each family supports ten algorithms in all. Table 1 shows the different block and key sizes, in bits, for SIMON and SPECK.

Table 1: Simon Block And Key Size.

Block size	Key size
32	64
48	72,96
64	96,128
96	96,144
128	128,192,256

The SIMON block cipher utilized a block of n-bits is assigned SIMON 2n, and mn-bits key is indicated as Simon 2n/mn, where n is 16,24,32,48,or 64 and m is 2,3 or 4. For example, SIMON 64/128 indicates the SIMON with 64bit blocks of plaintext & key of 128-bit [5].

Typically SIMON utilizes the well-known Feistel rule of activity. The algorithm organized to miniature in hardware which is simple at different levels to serialize the data.

2.1 Round Functions

Simon 2n encryption utilized the subsequent operations of n-bit words:

a) Bitwise AND: This operation performed on random two bits of n-bit words.

b) Bitwise XOR: This operation performed on the result of bitwise AND operation and one the bit from lower block and resultant again XOR-ed with one of random bit from upper block and finally XOR-ed with a key.

c) Left and Right circular shift, S^j and S^{-j} by j bits, respectively.

For $k \in GF(2)^n$, the key-dependent SIMON 2n round function is the two-stage Feistel map R_k: $GF(2)^n$ x $GF(2)^n$ $GF(2)^n$ x $GF(2)^n$ defined by

$$R_k(x, y) = (y \oplus f(x) \oplus k, x), \quad (1)$$

Where $f(x) = (Sx \,\&\, S^8x) \oplus S^2x$ and k is the round key.

The SIMON key schedule takes a key and it generate from a sequence of T key words $k_0,\ldots\ldots,k_{T-1}$, where T is the number of rounds. The encryption map is the structure R_{kT-1} ….R_{k1}, R_{k0}, read from right to left [8].

The operation of round function R_{ki} on the two words of sub cipher (x_{i+1}, x_i) at the i^{th} step of this process.

2.2 Key Schedules

The SIMON follows the exactly same operation of symmetric with respect to the circular shift operation on n-bits words. The key schedules of SIMON uses a sequence of 1-bit round constants designed for the purpose of slide properties and circular shift symmetries. Table 2 shows the different SIMON parameters.

Table 2: SIMON parameters.

Block size 2n	Key size mn	Word size n	Key words m	Const seq	Rounds T
32	64	16	4	Z_0	32
48	72,96	24	3,4	Z_0,Z_1	36,36
64	96,128	32	3,4	Z_2,Z_3	42,44
96	96,144	48	2,3	Z_2,Z_3	52,54
128	128,192, 526	64	2,3,4	Z_2,Z_3,Z_4	68,69,72

The cryptographic separation between the different versions of SIMON having the same block size by defining five such sequences: $Z_0,\ldots..,Z_4$. Designers provides these sequences Z_0,Z_1 with period 31 and Z_2,Z_3,Z_4 with 62 [6]. Z0 is defined as below:

$Z_0=(Z_0)_0,(Z_0)_1,(Z_0)_2\ldots\ldots=$
11101000100101011000011100110…

Where (Z_j)I is the i^{th} bit of Z_j.

The method of Z_0 to Z_4 sequences generation is given in[1]. Let $c = 2^n - 4 = (2^n-1)$ 3 = 0xff……fc. For SIMON 2n with m key words (k_{m-1}…..k_1,k_0) and constant sequence Z_j , round keys are generated by

$$k_{i+m} = \begin{cases} c \oplus (z_j)_i \oplus k_i \oplus (I \oplus S^{-1})S^{-3}\, k_{i+1} & \text{if } m = 2, \\ c \oplus (z_j)_i \oplus k_i \oplus (I \oplus S^{-1})S^{-3}\, k_{i+2} & \text{if } m = 3, \\ c \oplus (z_j)_i \oplus k_i \oplus (I \oplus S^{-1})(S^{-3}\, k_{i+3} \oplus k_{i+1}) & \text{if } m = 4, \end{cases} \quad (3)$$

for $0 < i < T - m$.

2.3 SIMON Block Cipher

The architecture of SIMON block cipher consists of parallelism of encryptions which involves round functions and key generation blocks. For given 128 bit plain text and a 128 bit cipher key, SIMON block generates 128 bit cipher text in 68 rounds. The above block diagram describes the dimension of parallelism for the block ciphers. Depending on the parallelism choice at each dimension the hardware implementation can range from n-parallel encryptions per clock cycle to one-bit of one round encryption per clock cycle. Hence to minimize the cost, we used a bit-serialized architecture in which inputs of all operators are one bit. To implement SIMON in a bit-serialized method, we have to first bit-serialized the round function and key generation [6]. Fig.4 shows the generic block diagram for parallelism of encryptions and rounds.

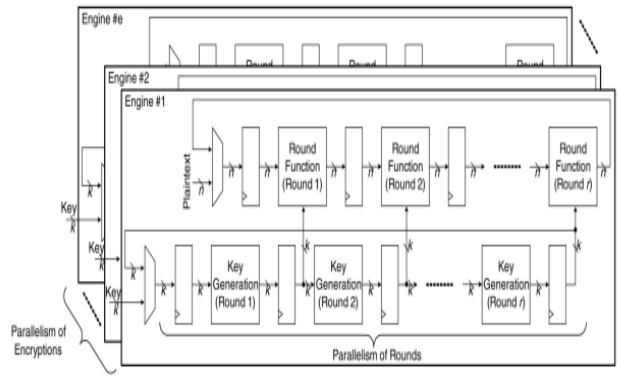


Fig 4: Generic block diagram for the parallelism of encryptions and rounds.

The basic function of block ciphers is to provide secure on stored data or information from the third party using symmetric encryption methods. They perform operation on fixed-size blocks of plain text and resulting in a block of cipher text for each. The most commonly used block sizes in currently block ciphers are 64 bits and 128 bits [6].

2.4 SIMON: Feistel-Based Low-Cost Block Cipher

This work focuses on the 128/128 configuration of SIMON with security level equivalent to AES-128. This configuration uses the inputs of 128-bit of plaintext and 128-bit key to generate 128-bit cipher text in 68 rounds [3].Fig.5 shows the round operation of SIMON. 128-bit data has been divided into two equal halves refers as upper block and lower block. The round function performs logic operations on the most significant 64-bits (the upper half block) and the result is XOR-ed with least significant 64-bits (the lower half block) and the 64-bit round key k_i. At the end of each round, the contents of the upper block is transferred to the lower block as the new generated values are written back into the upper block.

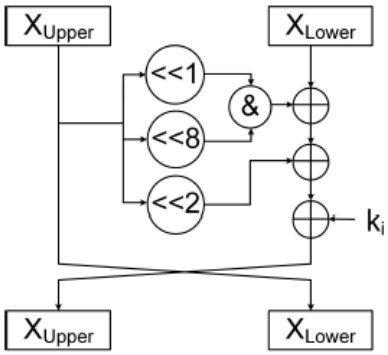


Fig.5: Feistel round configuration.

The SIMON round operation consists of three 64-bit shift operators (shift left one, shift left two, and shift left eight) three 64-bit XOR operators and one 64-bit AND operator. Fig.6 shows the key generation of SIMON. The key generation function performs logic operations on most significant of 64-bits and the result is XOR-ed with the least significant 64-bits and the 64-bit round constant Z_i. The constant is a design-time constant value that is uniquely tuned for each configuration. The full key generation consists of two 64-bit shift operators (shift right three and shift right four) and three 64-bit XOR operators.

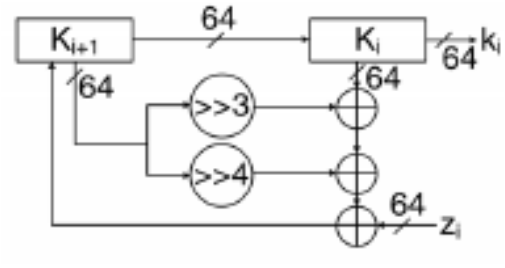


Fig.6: Key generation of SIMON for 128/128 configuration.

2.5 Hardware Architecture

The bit serialized implementations is suitable for Field Programmable Gate Array (FPGA) resources. These hardware architectures can support Look Up Tables (LUTs) can be efficiently designed as bit-serial memory elements. FPGAs have different configurations of LUT with different sizes. For examples, Spartan-3 has LUT [13] each act as 8x1 First-In-First-Out (FIFO), similarly on Spartan -6 has increased size of LUT inputs, where each act as 32x1 FIFO.Typically the FIFO-based implementations can reduce the control cost because no need of address generation circuitry, they are internally addressed and only required of control on one bit input and output. Moreover, in FIFO it can read from and write into at the same clock cycle. Therefore can overlap the compute and transfer function so that the round operation can complete within 64 clock cycles.The Fig.7 shows the hardware architecture based on FIFO for round operation. Here the one the objectives is analyzed. Each one bit per clock cycle will enable the processing of four bits of the LUT and can read at the same clock cycle. With the help of these bits are used in LUT to perform the logic operations of SIMON round operation.

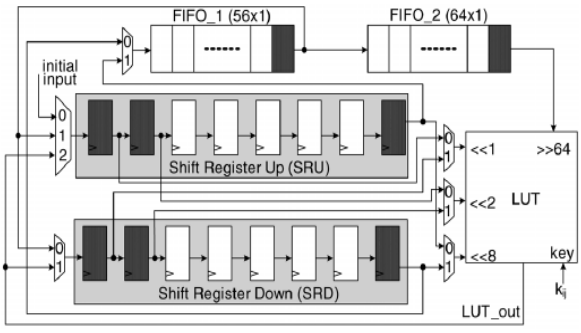


Fig.7: Block diagram of round operation for the bit-serialized architecture.

The upper block of 64-bits is stored in the 8 bit shift registers so as to enable the parallel process, which is followed by the 56x1 sized FIFO. In order to eliminate the problem of bubbles of scheduling that can occur at the beginning of the each round, so we can implement the two ping-pong register sets, where each consists of 8 register. At even rounds, the output of the LUT is written into the lower shift registers down (SRD), and at odd rounds, output of the LUT is written into the upper shift registers up (SRU). Similarly in order to enable the circular access pattern, at even rounds the output of the FIFO is written back into the SRU and at the odd rounds it is written back into the SRD. Here FIFO_2 input is connected to FIFO_1 output to perform transfer operation.

The application of the key generation uses the operating principle of the round function, but this uses right-shifted values at the inputs of LUT. Therefore the registers are placed after the FIFO. Fig.8 shows the how the bit-serialization of the SIMON round operation performs.

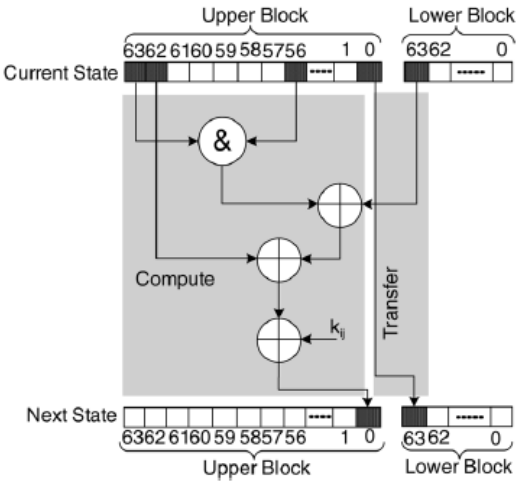


Fig.8: Bit serialized SIMON round function. K_{ij} stands for the key bit for the i^{th} round and j^{th} bit of the state.

In the first clock cycle the bits processing are highlighted. The round operation consists of two components, compute and transfer. Compute requires reading three single-bits from the upper block, one bit from the lower block, and performing logic operations on these bits using also a single-key bit. Transfer writes the output bit of compute into the upper block of the next state, and transmits one bit from the upper block of the current state to the lower block of the next state.

3 RESULTS & DISCUSSIONS

Testing is essential for ensuring the total quality of the product. Testing process helps to identify completeness, correctness, security and quality of the developed embedded product. This chapter gives the information regarding results and discussions of bit-serialized simulation.

3.1 Xilink Simulation Results

Using Feistel structure the 128-bit data is equally divided into two halves of data and performs the logic operations on random bits of upper block of 64-bit data. The resulted data is XOR-ed with lower block of plaintext, and using key generation of SIMON for the 128/128 configuration. Fig.9 shows the key generation of 128-bits for same length of the plaintext. The SIMON round operation is also involved in this key generation.

Fig.9: SIMON key generation.

Fig.10 shows that using bit-serialized hardware architecture for a plaintext of 128-bit and a key of 128-bit used in which SIMON block generates 128-bit Cipher text in 68 rounds. With the help of key expansion used in plaintext to convert into cipher text. When the data_ready is 11 the architecture generates the cipher text and when data_ready is 10 data starts loading into the architecture and performs operations.

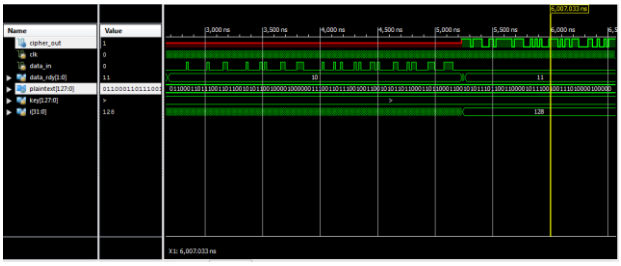


Fig.10: Cipher output using SIMON key generation in bit-serialized fashion.

The SIMON algorithm with round operation for example m = 4 can be explained in 5 control steps (CS). And sub-operations have been allocated in each control step follows most two ALUs, two shift-registers and one register are required. Fig.11 shows the optimal state for scheduling, allocation and binding.

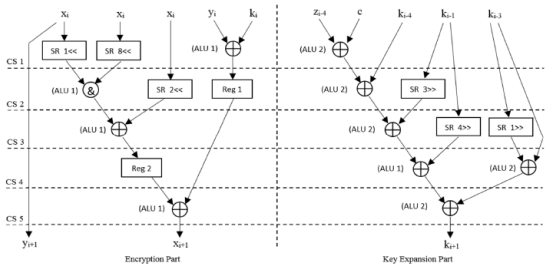


Fig.11: Scheduling, Allocation and Binding with 5 CS.

The allocation further divided into sub-tasks, such as functional unit allocation, memory unit allocation and interconnects allocation. The number of functional unit in each step for performing operations, memory units for storing data values and interconnects for data transformation.The scheduling, allocation and binding can repeat for m = 2,3 which is easier compare with m=4. In Fig.12, j << means circular shift to left by j bits and j >> means a circular shift to right by j bits. Therefore X_i is transformed into Y_{i+1} within in 5 CS. The key expansion done for i = 4 to 31 and the round key will be obtained for each i, and then the encryption is done with the obtained round key.In this project the performance of bit-serialized architecture has been studied by means of XILINX simulation. The comparison is done with the results obtained from the previous work on low-cost block cipher and stream cipher implementations. The main requirement is to achieve the minimum cost, because of tool automation tends to minimize LUT count and increase the number of utilized slices. Hence, this results the area cost of an IP-block is the number of utilized slices.The comparison of previous work with different techniques is as shown in Fig12 that target to a commonly used FPGA; the bit-serialized architecture is the smallest implementation that ever had been in the literature. The area result of block cipher HIGHT, it uses 91 slices, where as our project costs 36 slices. The proposed architecture is also smaller than all other introduced low-cost stream ciphers [14],[15]. The security

level of 128-bits is same; the cost of smallest stream cipher is GRAIN [15], uses 48 slices. The proposed architecture of SIMON is 86% smaller than AES, 60% smaller than existing block cipher and even 25% smaller than the stream cipher.

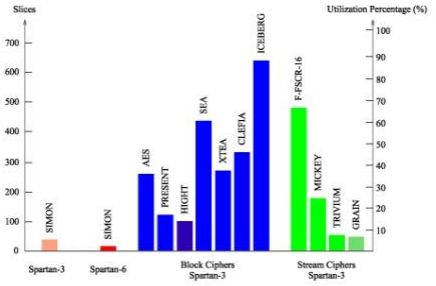


Fig.12: Implementation results of SIMON and comparison with earlier work on low-cost block cipher and stream cipher implementations.

The Table 3 shows the area/performance tradeoff of our design. The systematic exploration of parallelism yields a trade off in performance of a factor of 109 times for a variation in area of 21 times. These factors provides much larger than general automatic design space exploration using FPGA tools.

Table 3: SIMON – Area versus Throughput.

Design (x,y,z)	Area (Slice)	Max.Frequency(MHz)	Throughput(Mbps)
(1,1,1)	36	136	3.6
(1,1,128)	399	135	216.8
(1,2,128)	766	135	392.4

The Feistel ciphers compared to S-P ciphers, suit better to low-cost FPGA architectures. S-P network uses S-box and they not efficiently serialized beyond the operand size and S-box. Thus, bit-serialized architectures are not feasible for them. Hence Feistel ciphers are easily bit-serialized. FIFOs can efficiently realize using LUTs, therefore FPGAs well-adjusted for bit-serialized implementations.

4. CONCLUSIONS

This project represented a low-cost, bit-serialized architecture for the block cipher SIMON, and also shows the new area records for block ciphers.The Rijndael, in accordance with the requirement for the AES, is a block cipher with a variety of bits of key size, the variable bits of block size and variable rounds. For 128/128/10 i.e., using 128-bit cipher key, 128-bit block size and 10 rounds. The Rijndeal cipher uses the substitution permutation network structure. Therefore considering Rijndeal and with other evaluation of results this proposed work is efficient because of symmetric key cipher standard, and increase in the level of security due to the application of 128-bit key.

This proposed architecture is even smaller than the other block ciphers and stream ciphers. The bit-serialized architecture of SIMON provides only 36 slices of area. Finally it can be concluded that SIMON shows 50% of area deduction as compared to AES for its ASIC implementation, and also shown that with the reduction of 86% of SIMON is a stronger alternative to AES for low-cost FPGA applications.

5. REFERENCES

[1] AdyinAysu,Ege Gulcan and Patrick Schaumont,"SIMON says:Break Area Records of Block Ciphers on FPGAs", IEEE Embedded Systems Letters, Vol.6, No.2, June 2014.

[2] William Stallings fifth edition of "Cryptography and Network Security Principles and Practice".

[3] T. Eisenbarth, S. Kumar, C. Paar, A. Poschmann, and L. Uhsadel.A Survey of Lightweight-Cryptography Implementations. In IEEE Design & Test, Volume 24, Issue 6, pages 522–33, 2007. 21.

[4] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, Louis Wingers, "The simon and speck of lightweight block ciphers," National Security Agency 9800 Savage Road, Fort Meade, MD 20755, USA, June 2013.

[5] C. D. Cannière, O. Dunkelman, and M. Kneževi´c. KATAN and KTANTAN - A Family of Small and Efficient Hardware-Oriented Block Ciphers. In CHES 2009, Lecture Notes in Computer Science, No. 5747, pages 272–88. Springer-Verlag, 2009. 5, 19, 20.

[6] T. Good and M. Benaissa, "AES on FPGA from the fastest to the smallest," in Cryptographic Hardware and Embedded Systems CHES 2005, ser. Lecture Notes in Computer Science, J. Rao and B. Sunar, Eds. Berlin, Germany: Springer-Verlag, 2005, vol. 3659, pp. 427–440.

[7] D. Hwang, M. Chaney, S. Karanam, N. Ton, and K. Gaj, "Comparison of FPGA-targeted hardware implementations of eSTREAM stream cipher candidates," in Proc. State of the Art of Stream CiphersWorkshop, SASC 2008 , Lausanne, Switzerland, Feb 2008, pp. 151–162.

[8] P. Bulens, K. Kalach, F.-X. Standaert, and J.-J. Quisquater, "FPGA implementations of estream phase-2 focus candidates with hardware profile," in Proc. State of the Art of Stream Ciphers Workshop, SASC 2007, Bochum, Germany, Jan. 2007.