LOOP UNROLLING IMPLEMENTATION OF AN AES ALGORITHM USING XILINX SYSTEM GENERATOR

Alshaima Q. Al-Khafaji [1] **, M. F. Al-Gailani** [2]
[1,2] College of Information Engineering, Al-Nahrain University, Baghdad, Iraq
alshaimaalkhafaji7@gmail.com, m.falih@coie-nahrain.edu.iq
Received:26/10/2019, Accepted:22/12/2019

***Abstract*- Cryptographic algorithm is a tool that is used to secure the transmitted data on the network. The current standard algorithm the Advanced Encryption Standard (AES) is used to maintain the security and reliability of the encrypted data whether these data are stored in computer or in transmit. AES can be implemented either in hardware or software, however hardware implementation is more sensible for high speed applications. In this paper, AES- 128 algorithm is implemented in hardware in order to achieve a high-speed data processing. It is implemented on an FPGA platform using HLL language and Xilinx ISE software. The design is effectively optimized and Synthesizable with high accuracy using the conventional blocks of Xilinx System Generator. The results of implementation have enhanced the performance in terms of resource utilization, speed and power consumption as compared with other related works. The circuit operates at a maximum frequency of 800.000 MHz which offers a high throughput of 102.4 Gbps on virtex6 xc6vlx130t- 3ff1156, in addition it occupies only 2,405 slices.**

keywords: **Advanced Encryption Standard (AES), Field Programmable Gate Array (FPGA), High Level Language (HLL), Integrated Synthesis Environment (ISE).**

I. INTRODUCTION

The volume of information being transmitted through computer networks has become increasingly large. However, the big challenge is how to keep the privacy of these information, how to be sure that they are revealed by authorized party. Cryptography is the answer, which is initially proposed for military applications, it is currently used in any applications that require confidentiality like financial affairs, Automatic Teller Machine (ATM), E- commerce, Mobile network, credit card, health issues, etc. [1]. One of the most famous and secure algorithms of cryptography is the AES [2] [3]. The primary objective of implementing AES in hardware [4] is to ensure high speed realization for real time applications. In addition to the high throughput, the design should consider as well the area and power consumption. Thus, hardware implementation is very fast, and reliable compared to software implementations. Moreover, in long term it has lower costs by considering the requirement of software implementation to update occasionally between times [5]. The rest of the paper is organized as follows. Section - II presents the related work of the hardware implementations of the AES. Section - III shows a brief overview of the AES algorithm. Sections - IV explains the hardware design and implementation of the algorithm in addition to the key generation using Xilinx System Generator, while Section - V discusses the implementation results of the proposed design. The conclusion of the work is outlined in section - VI.

II. RELATED WORK

This section presents some of the previous related works: **G. P. Saggese 2003 [6]** presented a hardware implementation of the Advance Encryption Standard algorithm on FPGA XC2v6000- 8 device using VHDL approach. A loop unrolling architecture was implemented to increase the throughput of the design. The circuit reaches a maximum frequency of 158 MHz and achieves a throughput of 20.3 Gbps. **J. M. Granado- Criado 2010 [7]** proposed a pipelined and parallel

implementation of the Advanced Encryption Standard on XC2v6000- 6 of Xilinx using three hardware languages (VHDL, Handel- C and JBits) with a throughput of 24.92 Gbps. **Dong Chen 2010 [8]** implemented the AES algorithm on a Xilinx Virtex- 4 xc4vlx100 device using the composite field algorithm to realize SubByte operation. Sub- pipelined architecture was designed to improve the frequency and achieve higher throughput. It operates on a maximum frequency of 645.703 MHz with a throughput of 82.65 Gbps. **Pravin B. G. 2010 [9]** used Very High Speed Integrated Circuit Hardware Description Language (VHDL) and Virtex XCV600 FPGA to implement the AES algorithm. In order to increase the throughput of the design, a pipelining architecture was proposed for implementing the encryption and decryption algorithms. The throughput for both encryption and decryption are 352 MHz. **A. Arshad 2014 [10]** proposed a hardware implementation of Advance Encryption Standard algorithm on Virtex- 6 xc6vsx315t- 3ff1156 FPGA using High Level Language (HLL) approach. Pipelined architecture was implemented to increase the overall frequency and minimize the critical resources of the design. It operates on a maximum frequency of 288.19 MHz and offers a high throughput of 36.864 Gbps. **S. M. Umar Talha 2016 [11]** presented an efficient reconfigurable hardware implementation of an AES using High Level Language (HLL) on Virtex- 6 xc6vsx315t with the help of Xilinx System Generator. The implementation uses a pipelined architecture to reach a maximum frequency of 254.453MHz. **Kirat Pal Singh 2016 [12]** proposed an efficient hardware architecture design and implementation of the AES algorithm using VHDL on XC6vlx240t of Xilinx Virtex Family. The design was based on pipeline architecture to increase the frequency. It operates on a maximum frequency of 515.38 MHz.

III. AES ALGORITHM

In November, 2001 the Advanced Encryption Standard (AES) was announced officially by the National Institute of Standards and Technology (NIST) [3]. The algorithm is a block cipher based on SP- Network, it has a block size of 128- bit and a key length of 128, 192, or 256 bits. It is referred to as AES- 128, AES- 192, or AES- 256, depending on the key length. AES- 128 algorithm consists of ten rounds and each round consists of a sequence of four different transformations, called steps, which are SubByte, ShiftRow, MixColumn and AddRoundKey. These steps are identical at all rounds except the last which is applied without the MixColumn transformation. In addition, the value of the round key is differing from round to round and from the user supplied key.

- **SubByte Transformation**

  SubByte is a non- linear substitution of bytes, where each byte of the state is substituted with another using table lookup (S- box). The S- box is constructed by composing of two transformations: first, by taking the multiplicative inverse of the elements in the finite field GF(28) , where the element {00} is mapped to itself. Then by applying a certain affine transformation over GF(2).

- **ShiftRow Transformation**

  The elements in the last three rows of the state are cyclically shifted to the left over different numbers of offset, while first row remained unchanged. Second row is shifted 1- byte to the left, third row is shifted 2- byte to the left, and last row is shifted 3- byte to the left.

- **MixColumn Transformation**

  In MixColumn each column is processed separately. Each new byte of a column is a function of all four bytes in that

column. The MixColumn transformation can be defined by the following matrix using the irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$.

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix} \quad (1)$$

- **AddRoundKey**

  In this step, the 16 bytes of the State are XORed with the 16- byte of the round key.

IV. AES Hardware Design and Implementation

AES algorithm is designed in FPGA [13] using Xilinx System Generator [14]. The design of the AES- 128 is shown in Fig. 1. It is a Loop Unrolling Architecture where each round is implemented separately. In this architecture, the hardware required to implement each round is duplicated to the number of rounds, i.e. , 10. Thus, the throughput is increased with the cost of the area used. Accordingly, the design is suitable for applications that require high speed implementation.

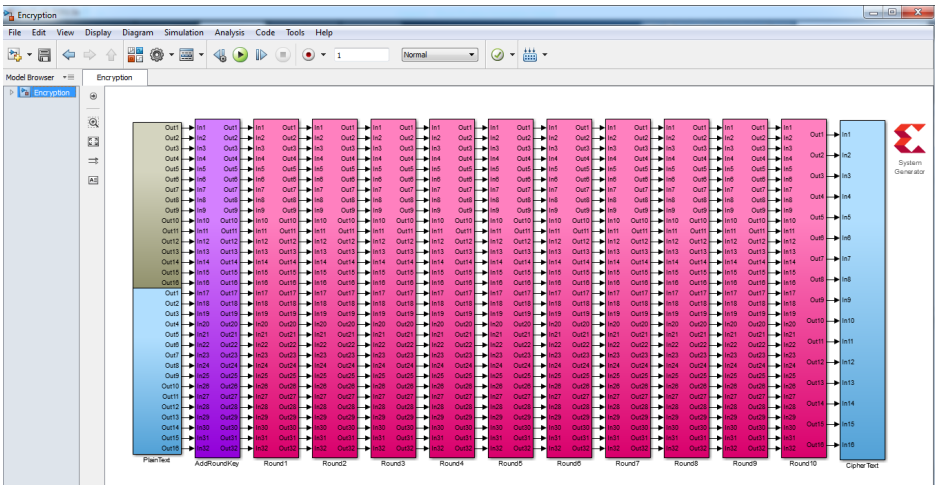

Figure 1: Loop unrolling architecture of an AES in system generator

- **SubByte**

  SubByte transformation is implemented using a lookup table. Read- Only- Memory (ROM) is used to store the 256 values of the s-box in the form of decimal numbers as shown in Fig. 2 This ROM is duplicated 16 times to implement the 16 bytes of the block in parallel at the same time.

- **ShiftRow**

  ShiftRow transformation is implemented by reconnecting the wires according to the shift operation explained in section III without involving any logics.

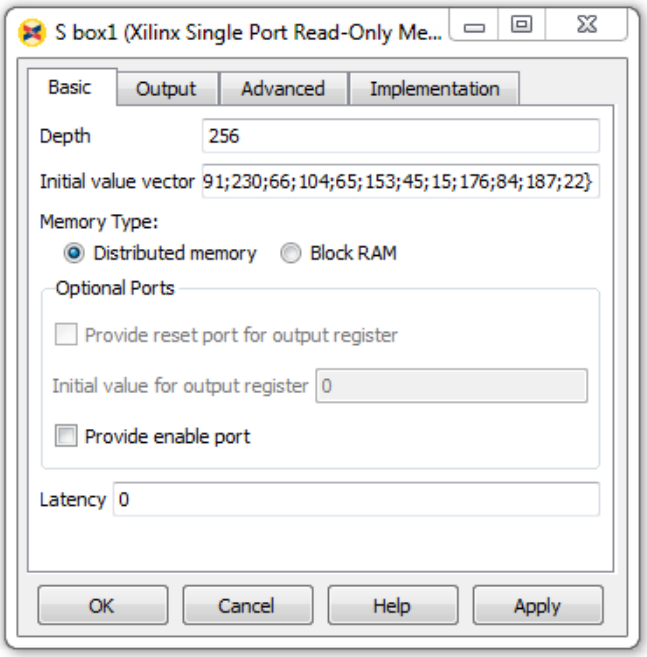

Figure 2: ROM block setting

- **MixColumn**

  MixColumn transformation processes one word (one column, 4- byte) at a time according to (1). It multiplies the state (intermediate data) with a fixed array, noting that this multiplication is polynomial. Each new byte is a result of XORing the bytes of the word after multiplying them with the mentioned array. Fig. 3 shows the architecture of the MixColumn for the first 32-bit word. One of the bytes of the word is multiplied by 2. Using Xilinx multiplier costs three latency, therefore multiplier 2 is designed using the circuit shown in Fig. 4, which cost nothing in hardware. The circuit works as follow, if the value of the entered number is greater than or equal 128, i.e. the most significant bit of the byte is one, the output byte is computed by shifting the input byte 1-bit to the left, and XORing the result with the constant 27 (the value of the irreducible polynomial in decimal). Otherwise, the XOR operation is skipped. From this circuit Multiplier 3 is designed by XORing the designed multiplier 2 with the input data as shown in Fig. 5.
- **AddRoundKey**

  At AddRoundKey transformation the 16- output byte of the MixColumn operation is XORed with the 16- byte of the round sub Keys.
- **Deign of Round Key Generation**

  AES- 128 requires generating in total sub- keys of 44 words (32- bit each), 4- word sub- key for each round in addition to the 4- word of the initial key addition, which is actually the user provided key. The words are generating by simply XORing the word preceding the word to be generated in one location with that word located four places backward from the new one, for example, Word 5 is generated by XORing Word 4 with Word 1. This algorithm is

applied to three out of four words of each round. For the first word from each round which is located in a place its value modulo 4 is 0, a different algorithm is used. It is achieved by passing the word immediately preceding the word to be generated through the G function. This function as shown in Fig. 6 consists of three steps, first step maps the values of the 4- byte by passing them through the substitution box. This layer is implemented as in the algorithm using S- box which its value is calculated in advance and stored in a ROM.

Next step is the rotate word which rotates the 4- byte one byte to the left. This layer is executed by rearranging the output according to the algorithm without involving any logic. The last step is to XOR the results of the previous step with the round constant. This constant is used to break the symmetry as its value is changed at each round. It starts with a value of one and then this value is multiplied by 2 every round, noting that this multiplication is polynomial multiplication. The output of the G function is then XORed with the column which is three columns behind. The design is shown in Fig. 7.

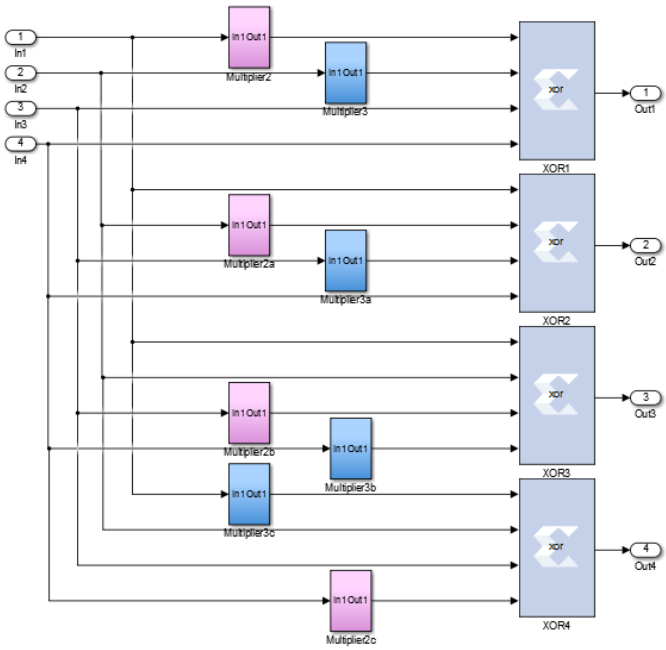

Figure 3: MixColumn architecture

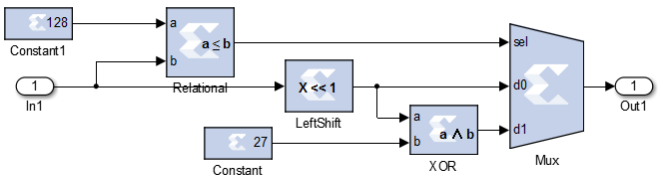

Figure 4: Multiplier by 2 circuit

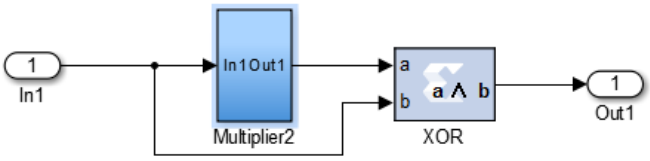

Figure 5: Multiplier by 3 circuit

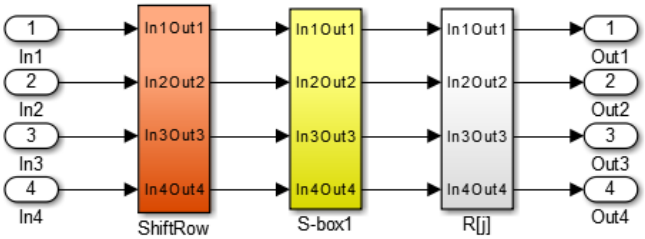

Figure 6: Design of G-function

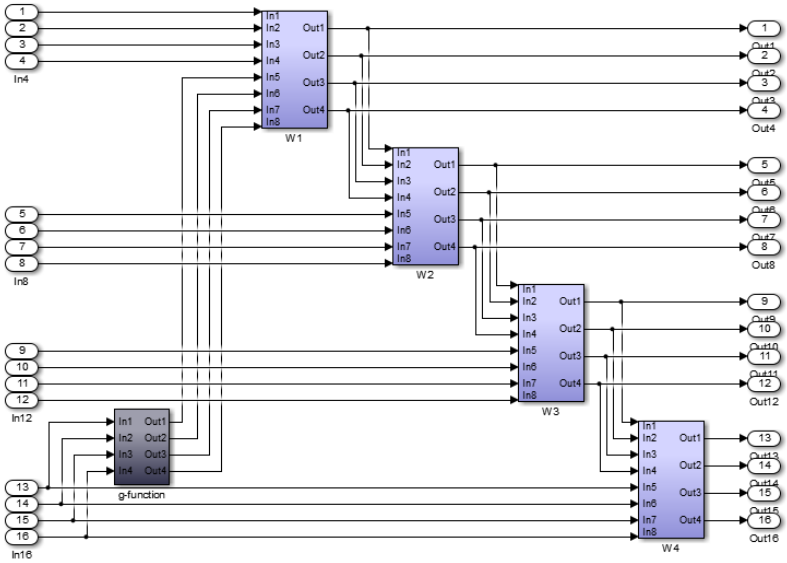

Figure 7: Design of round key generation

V. Hardware Implementation Result & Discussion

The cipher has been implemented on the target device Xilinx xc6vlx130t- 3ff1156 FPGA, which is chosen very carefully to adhere the requirements of the proposal. The algorithm is design using Xilinx System Generator that generates the necessary netlist file which is synthesized and simulated using Xilinx ISE Foundation 14.7 [15] and ModelSim [16]. The block of data is processed in only one cycle this is due to the selected architecture and the efficient design of the layers.

The results show that the AES- 128 loop unrolling architecture operates on a maximum frequency of 800 MHz and offers a high throughput of 102.4 Gbps. The design occupies 2,405 slices out of 20,000 (12%). The total power consumption of the circuit is 3.477 W. The simulation result is shown in Fig. 8 using ModelSim simulator. The values of the input key and plaintext as well the generated ciphertext are:

Input Key: 2b7e151628aed2a6abf7158809cf4f3c

Plain Text: 3243f6a8885a308d313198a2e0370734

Cipher Text: 3925841d02dc09fbdc118597196a0b32

The performance of the algorithm is clearly enhanced as shown in Table I, which presents a comparison between the obtained results with other related works.

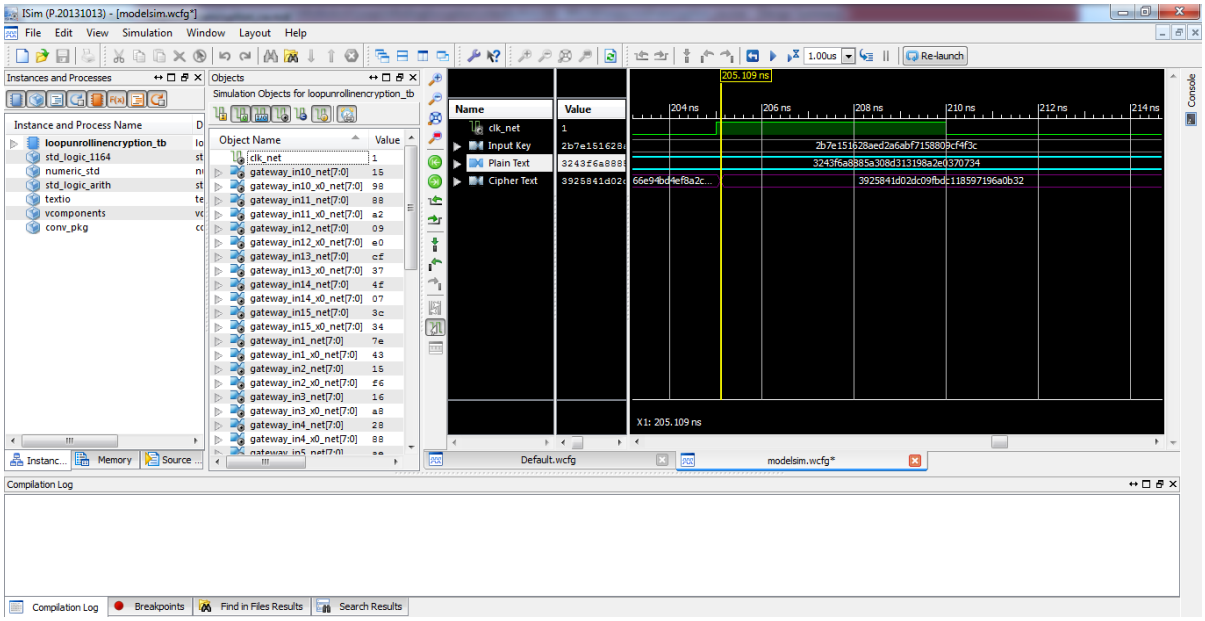

Figure 8: Simulation result

TABLE I
COMPARISION RESULT

Implementation	Devices	Frequency (MHz)	Throughput (Gbps)
Reference [3]	XC2v6000-8	158	20.3
Reference [4]	XC2v6000-6	194.7	24.9
Reference [5]	Virtex-4 xc4vlx100	645.7	82.6
Reference [6]	Virtex XCV600	140.390	-
Reference [7]	Virtex-6 xc6vsx315t	288.19	36.8
Reference [8]	Virtex-6 xc6vsx315t	254.45	-
Reference [9]	Virtex-6 XC6vlx240t	515.38	-
Proposed work	Virtex-6 xc6vlx130t-3	800	102.4

VI. CONCLUSIONS

In this paper, efficient implementation of an AES in term of throughput, area, and power consumption is presented using Xilinx system generator. The chosen methodology was to optimize the implementation speed, thus loop unrolling

architecture is proposed taking into consideration the amount of utilized resources and wasted power. The obtained results are promising compared to other related works, hence the design is actualized competently as planned.

REFERENCES

[1] William Stallings, " Cryptography and Network Security Principles and Practice" , Seventh Edition, Pearson, Pearson Education Limited, 2017.
[2] Joan Daemen and V. Rijmen, " The Design of Rijndael: AES-The Advanced Encryption Standard" , Berlin Heidelberg: Springer, 2002.
[3] National Institute of Standard and Technology (NIST), " FIPS 197 Announcing the ADVANCED ENCRYPTION STANDARD (AES)" , Computer Security Resource Center (CSRC), November 26, 2001. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf
[4] Steve Kilts, " Advanced FPGA Design Architecture, Implementation, and Optimization" , John Wiley & Sons, Inc. , Hoboken, New Jersey 2007.
[5] Khose, P. N., and Prof. Vrushali G. Raut, " Implementation of AES Algorithm on FPGA for Low Area Consumption" , International Conference on Pervasive Computing (ICPC), 2015.
[6] G. P. Saggese, A. Mazzeo, N. Mazzocca, and A. G. M. Strollo, " An FPGA- based Performance Analysis of the Unrolling, Tiling, and Pipelining of the AES algorithm" , in Proc. FPL, pp. 292- 302, 2003.
[7] J. M. Granado- Criado, M. A. Vega- Rodriguez, J. M. Sanchez-Perez, and J. A. Gomez- Pulido, " A new methodology to implement the AES algorithm using partial and dynamic reconfiguration" , Integr. VLSI J., vol. 43, pp. 72- 80, 2010.
[8] Dong Chen, Gouchu Shou, Yihong Hu, Zhigang Guo," Efficient Architecture and Implementations of AES" , IEEE ICACTE2010, pp. V6- 295- V6- 298.
[9] Pravin B. Ghewari, Mrs. Jaymala K. P., and AMIT B. C., " Efficient Hardware Design and Implementation of AES Cryptosystem" , International Journal of Engineering Science and Technology, Vol. 2(3) , 2010.
[10] A. Arshad, K. Aslam, D. Kundi, and A. Aziz, " FPGA Implementation of Advance Encryption Standard Using Xilinx System Generator", Asian Journal of Applied Sciences, vol. 2, Issue 02, April 2014.
[11] S. M. Umar Talha1, Mir Asif, Hammad Hussain, Ali Asghar, and Hadi Ameen, " Efficient Advance Encryption Standard (AES) Implementation on FPGA Using Xilinx System Generator" , 2016 IEEE.
[12] Kirat P. S. , and Shiwani D. , " An Efficient Hardware design and Implementation of Advanced Encryption Standard (AES) Algorithm" , International Journal of Recent Advances in Engineering & Technology (IJRAET) V- 4 I- 2 For National Conference on Recent Innovations in Science, Technology & Management (NCRISTM) ISSN (Online) : 2347- 2812, 2016.
[13] Juan Jose Rodriguez Andina, Eduardo de la Torre Arnanz, and Maria Dolores Valdes Pena, " FPGAs Fundamentals, Advanced Features and Applications in Industrial Electronics" , CRC Press, Taylor & Francis Group, LLC, 2017.
[14] Esteban Tlelo- Cuautle, Jose de Jesus Rangel- Magdaldaleno, and Luis Gerardo de la Fraga, " Engineering Applications of FPGAs Chaotic Systems, Artificial Neural Networks, Random Number Generators, and Secure Communication Systems" , Springer International Publishing Switzerland, 2016.
[15] Xilinx, " ISE In- Depth Tutorial" , UG695 (v14.1) April 24, 2012.
https://www.xilinx.com/support/documentation/sw-manuals/xilinx14-1/ise-tutorial-ug695.pdf.
[16] Mentor Graphics Corporation, " ModelSim User's Manual" , Software Version 10. lc, 2012.
https://www.cc.gatech. edu/ hadi/teaching/cs3220/ doc/modelsim/ModelSim-Users-Manual-v10.lc.pdf.