Design and Implementation of Pipelined AES Encryption System using FPGA

Mohamed Nabil, Ashraf A. M. Khalaf, Sara M. Hassan

Abstract: Nowadays, the data encryption became very important because of the usage of the data transmission in all the filed. The Advanced Encryption Standard (AES) that known as Rijndael algorithm is one of the most common encryption algorithms. The AES consists of 9 rounds in addition to the initial and final rounds that makes the AES consumes much time for encrypting the data. Of course the time consumption is considered one of the problems that face the information security. The more time the encryption system consumes to encrypt the data, the more chances increase for the hackers to break into the system. In this work, we find a new technique that can be used to increase the performance speed of the advanced encryption standard. The proposed algorithm methodology depends on the pipelined processing method for the processing time reduction. The paper includes a discussion of the design, the analysis and the implementation using Field Programmable Gate Array (FPGA) of the pipelined method to reduce the consumed numbers of clocks and speed up the processes. The AES is used to protect information and encrypt sensitive data and used in satellites, missiles, military application and other critical application. The paper describes the AES encryption system algorithm and the implementation of both the normal processing and the pipelined processing, and finally a comparison between the two algorithms.

Keywords: Encryption; Rijndael; Key; AES; Algorithm and Pipelined.

I. INTRODUCTION

The strength of the Data Encryption Standard (DES) has been weak to meet the new situation of the security systems and the attacks. A few years ago, the National Institute of Standards and Technology (NIST) determined the Rijndael algorithm as Advanced Encryption Standard (AES); this encryption method was developed by both Joan Daemen and Vincent Rijmen, to replace the DES. Now, Rijndael became the most common method that used encryption method to support bulk data encryption [1-3].

Manuscript published on January 30, 2020.
* Correspondence Author

Mohamed Nabil*, Department of Electronics and Communications, Faculty of Engineering, Minia University, Minia, Egypt.
E-mail: mohammednabel5050@gmail.com
Ashraf A. M. Khalaf, Department of Electronics and Communications, Faculty of Engineering, Minia University, Minia, Egypt.
E-mail: ashkhalaf@yahoo.com
Sara M. Hassan, Department of Electronics and Communications, Faculty of Engineering, Modern Academy, Cairo, Egypt.
E-mail: sara.hassan@eng.modern-academy.edu.eg

Researchers worked to update the encryption systems as they wanted to reach the best method to the security systems hard to attack and increase its performance. In the next paragraphs we will discuss the most common previous related studies.

The first encryption algorithm, Data Encryption Standard (DES) was used by NIST for the protection of important security information. However, the security of DES was reduced because of the short length key. Different attacks are capable of hacking DES in less than 255 complexities; in order to replace the DES it was important to find a stronger encryption algorithm [4, 5]. It was interesting to find an alternative encryption method instead of DES to avoid the attacks. Replacement processes of DES include three other algorithms that were found. First, the double DES then the triple DES with two keys and finally the triple DES with three keys. The disadvantages of the triple DES are: it has three times as many rounds as DES, so it is slow, and it uses a 64 bit block size; for efficiency and security, there is a need for a larger block size. These disadvantages of triple DES are not suitable for use nowadays [6]. The Rijndael algorithm was suitable for encryption so it became the standard. The AES algorithm was designed to increase the security level compared to the DES [4, 5].

There are many different types of encryption algorithms that are different in their encryption method, performance and strength. One of these systems is RSA (Rivest-Shamir-Adleman) which is an encryption technique that is used for encryption of data that are sent over the internet. RSA is used also in PGP (Post Graduate Program) and GPG (GNU Privacy Guard) programs. However, RSA is different from the triple DES; as it is asymmetric algorithms because it uses a pair of keys, one for the encryption process and another for the decryption. Finally, RSA makes it harder for the attackers to hack [7].

Encryption system is another algorithm used instead of DES which divides the data into segments of 64 bits and encrypts them separately. It is known by its speed and effectiveness. It used in software like e-commerce for payments security and passwords, where it used to protect passwords. It is also known as one of the most flexible encryption algorithms [8, 9].

Twofish is another algorithm that is used for the data encryption; Bruce Schneier is the designer of both Blowfish and. The used keys in are up to 256 bits, it is a symmetric algorithm due to the usage of one key.

Twofish is considered one of the fastest encryption systems that appeared and can be used in hardware and software platforms. Additionally, Twofish can be used by any person. So, it is found in encryption programs like PhotoEncrypt and TrueCrypt [10].

The security system and the encryption techniques are still themes that are scalable on a daily basis. Hence security researchers must do a lot of lab to find new techniques to keep them save [11].

They are looking forward to determining the hackers by updating the security systems and the encryption techniques. Based on this point, we are trying to update one of the strongest encryption systems that are used in the information security. In this paper, we found a new technique that can be used to increase the performance speed of the advanced encryption standard. The AES algorithm increased the resistance against all of the known attacks and speed. It is also used for different usages [12, 13].

As we mentioned, until now, many researches used the AES encryption system; for example, the high speed efficient advanced encryption standard implementation for Soufiane Oukili and Seddik Bri (2017), the key-dependent Advanced Encryption Standard by Abdelrahman Altigani , Shafaatunnur Hasan , Bazara Barry and Siti Mariyam Shamsuddin (2018) [14, 15].

Today, a lot of applications use the AES algorithm for data encryption; however, many researchers worked to increase the security, performance and speed up the AES [16].

This paper is being considered as one of the researches that aim to increase the processing speed of the AES.

The AES encryption system has 9 rounds in addition to the initial round and the final round. The regular implementation uses 26 clocks to output the encrypted data. The clock consumption is considered one of the problems that face the information security experts. The more time the encryption system takes to encrypt the data, the more chances increase for the hackers to break into the system [17, 18]. This paper discusses the design, analysis and implementation of pipelined method to reduce the consumed numbers of clocks and speed up the processes [19].

This paper is organized as follows: Section II describes the AES encryption system and its steps. Section III shows the methodology that we use to reach the paper's goal for speeding up the encryption process by using the pipelined process. Section IV discusses the hardware implementation of the normal process and the pipelined process; then, the simulation and results are presented in section V, the conclusion is discussed in section V.

II. AES ENCRYPTION

The AES has a fixed text block size of 128 bits and the key sizes are 128, 192, or 256 bits [20]. Our design and implementation concentrate on the key of size 128 [13, 21].

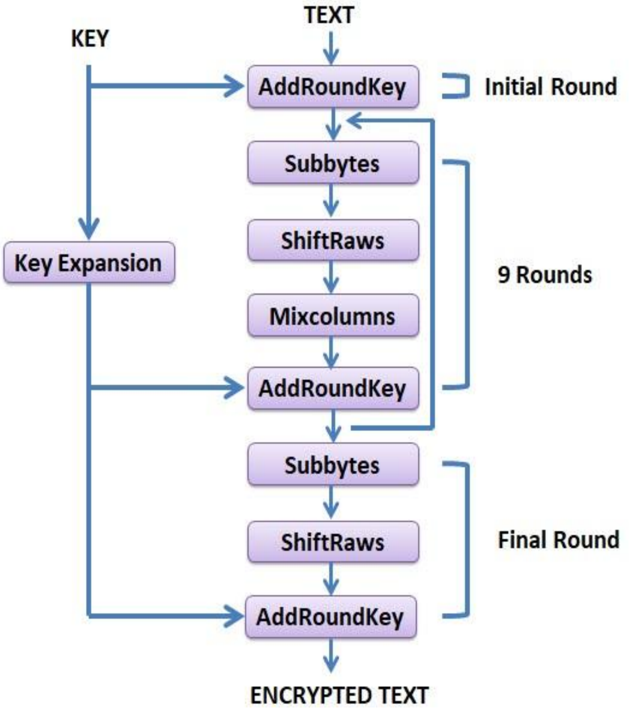


Fig. 1. The AES Encryption systems steps of each round

The input text and key of AES represented as a 4 × 4 array of bytes. Most of the AES processes are done in a particular finite field. Hence, if there are 16 bytes A_0, A_1, A_2…A_{15} these bytes are represented as two dimensional arrays [4, 16].

$$\begin{bmatrix} A_0 A_4 A_8 A_{12} \\ A_1 A_5 A_9 A_{13} \\ A_2 A_6 A_{10} A_{14} \\ A_3 A_7 A_{11} A_{15} \end{bmatrix}$$

The key matrix is used for converting the input which is called plaintext to the final encrypted text which called is ciphertext; the numbers of rounds are determined as per the rounds below [11, 22].

1) 10 rounds if 128-bit keys.
2)12 rounds if 192-bit keys.
3)14 rounds if 256-bit keys.

Each round consists of various steps. In each step, the key is changed due to some processes on the key of the previous round as shown in Fig. 3.

In the initial round, the text is Xored with the input key to produce the input of the next round while the second round starts with the subbytes steps [5, 23, 24].

A.The Subbytes Steps

In this step, every element of the text matrix is replaced with another due to the S-Box [25]. The first 4 bites of the every element of the text matrix represent the number of the row of the S-Box that is shown in Fig. 3 while the second 4 bites represent the number of the column. Accordingly, the value of the element in the S-Box is due to the row and the column that will replace the value of the element of the text matrix as shown in Fig. 2 [2, 26].

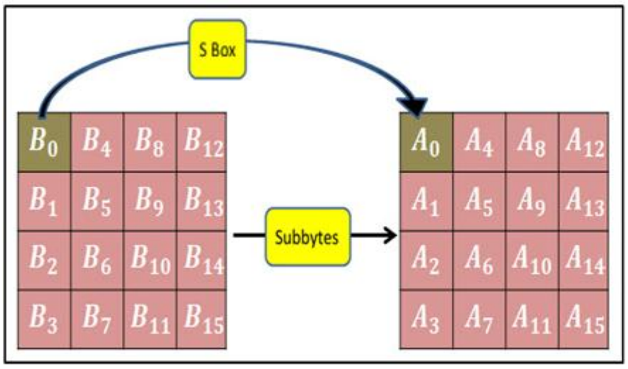
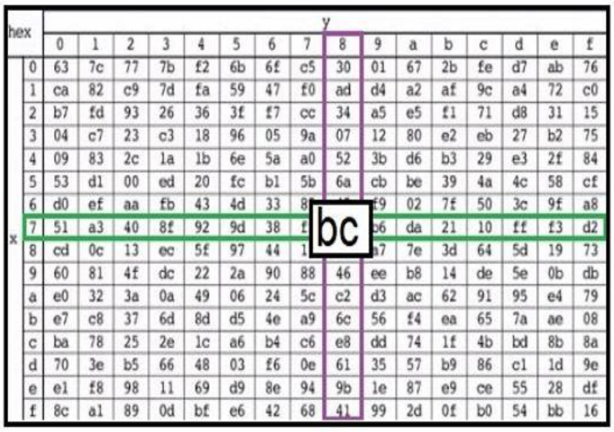

S Box
B_0 B_4 B_8 B_{12}
B_1 B_5 B_9 B_{13}
B_2 B_6 B_{10} B_{14}
B_3 B_7 B_{11} B_{15}
Subbytes
A_0 A_4 A_8 A_{12}
A_1 A_5 A_9 A_{13}
A_2 A_6 A_{10} A_{14}
A_3 A_7 A_{11} A_{15}

Fig. 2. The Subbytes step



hex	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	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	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	8	[illegible]	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	1	[illegible]	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Fig. 3. The S-Box of the AES

B. The ShiftRaws Step

The element of every row is shifted due to the number of the rows, the first row is not shifted, the second row is where one element is shifted to the left, the third row is where the two elements are shifted to the left and the final row is where the three elements are shifted to the left as shown in Fig. 4 [4, 5, 16, 23].

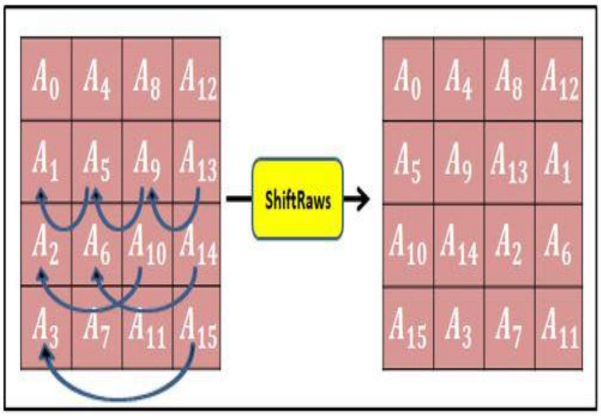

A_0 A_4 A_8 A_{12}
A_1 A_5 A_9 A_{13}
A_2 A_6 A_{10} A_{14}
A_3 A_7 A_{11} A_{15}
ShiftRaws
A_0 A_4 A_8 A_{12}
A_5 A_9 A_{13} A_1
A_{10} A_{14} A_2 A_6
A_{15} A_3 A_7 A_{11}

Fig. 4. The ShiftRaws step

C. The Mixcolumns Step

The Mixcolumns step processes column by column, treating each column as a four term polynomials. The result of this process makes a new four bytes generated as shown in Fig. 5 [9, 13].

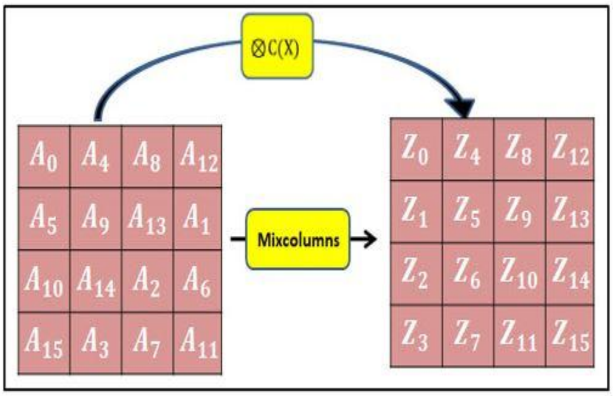

⊗C(X)
A_0 A_4 A_8 A_{12}
A_5 A_9 A_{13} A_1
A_{10} A_{14} A_2 A_6
A_{15} A_3 A_7 A_{11}
Mixcolumns
Z_0 Z_4 Z_8 Z_{12}
Z_1 Z_5 Z_9 Z_{13}
Z_2 Z_6 Z_{10} Z_{14}
Z_3 Z_7 Z_{11} Z_{15}

Fig. 5. The Mixcolumns step

D. The AddRoundKey Step

In this step, the text is Xored with the key to produce the input text of the next round as shown in Fig. 6.

The key of every round is changed due to some processes to produce a new key for each round. These changes make the AES safer and more difficult to be attacked [27].

The previous steps are repeated for every round till the round before the final one. The final round preforms the previous steps except for the mixcolumns step meanwhile in the normal processing of the AES that occurs every round [28].

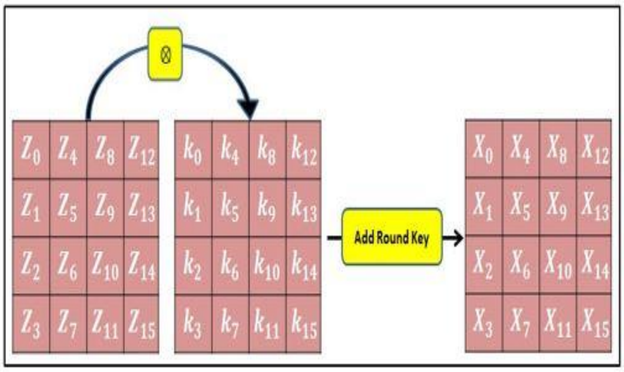

⊗
Z_0 Z_4 Z_8 Z_{12}
Z_1 Z_5 Z_9 Z_{13}
Z_2 Z_6 Z_{10} Z_{14}
Z_3 Z_7 Z_{11} Z_{15}
k_0 k_4 k_8 k_{12}
k_1 k_5 k_9 k_{13}
k_2 k_6 k_{10} k_{14}
k_3 k_7 k_{11} k_{15}
Add Round Key
X_0 X_4 X_8 X_{12}
X_1 X_5 X_9 X_{13}
X_2 X_6 X_{10} X_{14}
X_3 X_7 X_{11} X_{15}

Fig. 6. The AddRound step

As we mentioned, the AES encryption processes consist of 9 rounds in addition to the initial and the final rounds [29].

In the normal process, the beginning of the encryption process does not start till the previous encryption process is performed as shown in Fig. 7.

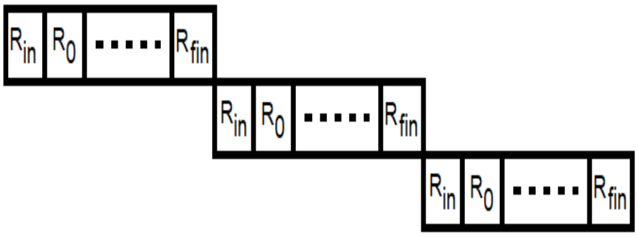

R_{in} R_0 ..... R_{fin}
R_{in} R_0 ..... R_{fin}
R_{in} R_0 ..... R_{fin}

Fig. 7. The Normal AES processing

Retrieval Number: E6475018520/2020©BEIESP
DOI:10.35940/ijrte.E6475.018520
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

This processing algorithm makes the encryption process very slow, consumes much of clocks and makes the system not suitable for a large number of communication applications. Because of this problem, the normal AES algorithm makes some restrictions on the data rate [27, 30].

In addition to the previous problem, the design should include a large memory for storing the input bits of the next encryption processes till the previous process would be finished. This makes the system use more resources and consume more hardware components; accordingly, this leads to a high budget [10].

So, these problems that can occur in the normal AES as there are the high processing time, usage of more resources and the high cost.

III. PROPOSED PIPELINED ALGORITHM

The pipelined algorithm is a good solution for the previous problems; as every clock has anew encryption process that will be started as shown in Fig. 8.

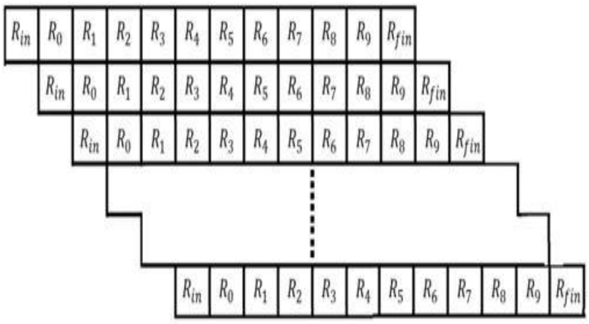


Fig. 8. The Pipelined AES processing

If we compared between the pipelined algorithm to the normal one, we find that the pipelined is faster than the normal; as every new clock has a new encryption process that will be started and another one will be finished. This algorithm makes the AES more suitable for many communication applications unlike the normal AES. The pipelined technique is based on designing a system that makes anew encryption process that starts every clock. Hence, it is unnecessary to wait the present process to compete in order to begin anew process [30]. This idea makes the system faster and more reliable for several applications. The algorithm is simulated using Matlab simulation program to make sure of achieving the goal of this paper. Fig. 11 shows the number of clocks that are used to encrypt various lengths of different messages, in case of the normal and the pipelined algorithm. The graph shows that in all the cases, the pipelined is finishing the encryption process before the normal one and that is expected [8, 10, 13, 31].

The proposed algorithm is achieved by using the following steps. First, every element of the two matrix consists of 8 bits and the input bits is inputted a bit by bit. Hence, each 8 serial bits will be converted in parallel and saved in the memory till element 16 are completed.

The beginning of the encryption process will be started after receiving the 16 elements of the key and the text. The input data will be saved during the encryption is being performed to avoid any losses of the data. This leads to the down conversion of the 50 MHz frequency to 6.25 MHz. Hence, every 160 nanosecond anew element is passed to the memory.

The data will be passed to the encryption system every 16 elements and don't wait the finishing of the encryption process. As every clock anew encryption process will be started

IV. HARDWARE IMPLEMENTATION

In this paper the implementation of the normal and the pipelined algorithms is performed to compare between the two algorithms from the view of many parameters. The design is performed using Xilinx Spartan 6 FPGA SP605 Evaluation Kit (XC6SLX45T-3C in FGG484 package), by writing a VHDL code on Xilinx ISE14.7, and simulated by using the ModelSim 6.5 simulator [4, 32].

Fig 9 shows the schematic implementation of the normal processing of the AES encryption system.

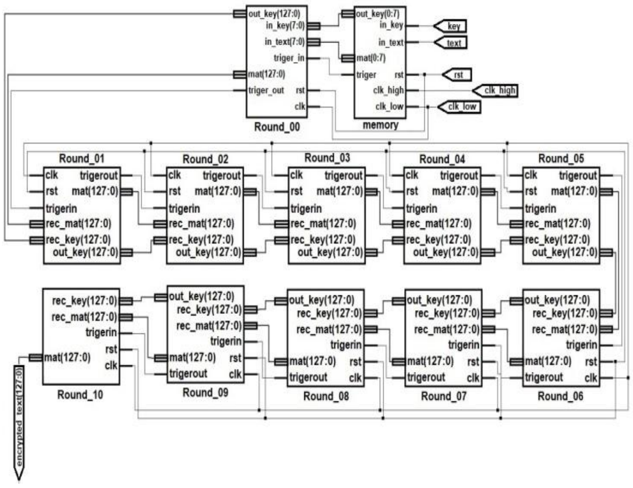


Fig 9: The Normal Schematic Implementation of the AES

The total number of the rounds is 11 rounds which consume 11 clocks for processing; meanwhile, the inputs of the system are the text matrix and the key which consist of 16 elements. Hence the input process consumes 16 clocks to complete the data in the text and the key matrix. The first round started in the clock of the last input element; hence, the total number of the clocks is 26 clocks as shown in Fig. 12 [2]. Assume the input text and key are as indicated in the matrix below. Therefore, so the encrypted text will be performed. The input text and key matrices are in the hexadecimal form as well as the output also.

$$\text{Text} = \begin{bmatrix} 328831E0 \\ 435A3137 \\ F6309807 \\ A88DA234 \end{bmatrix} \text{Key} = \begin{bmatrix} 2B28AB09 \\ 7EAEF7CF \\ 15D2154F \\ 16A6883C \end{bmatrix}$$

The simulation shows the output of each round and the final encrypted text. This simulation helps the user to debug the project in case of any problem that may occur as it shows the encryption process round by round [5].

$$\text{Encrypted Text} = \begin{bmatrix} 39 & 02 & DC & 19 \\ 25 & DC & 11 & 6A \\ 84 & 09 & 85 & 0B \\ 1D & FB & 97 & 32 \end{bmatrix}$$

The pipelined process speeds up the normal process as it consumes only 12 clocks; the schematic implementation of the pipelined is shown in Fig. 10.

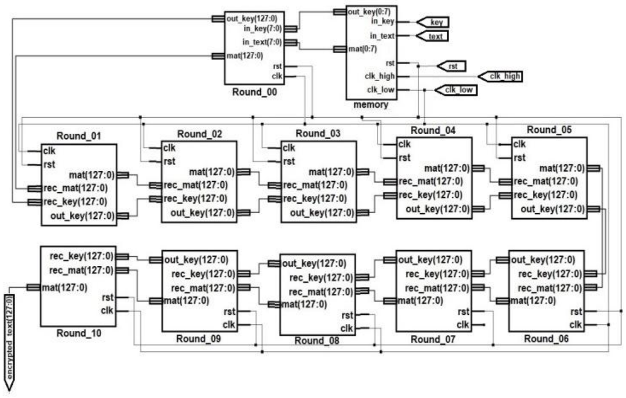


Fig. 10. The Pipelined Schematic Implementation of the AES

The ModelSim simulation and the results of the pipelined implementation are shown in Fig. 13. The simulation shows that if the input text and the input key are similar to the inputs of the normal implementation, the output will appear after 12 clocks. This algorithm saves 14 clock and speed up the encryption process [23].

The improved performance algorithm needs more resources to speed up the encryption process. This is shown in the device utilization summary for the normal and the pipelined processes as shown in Table I and Table II.

The pipelined performance is shown in Fig. 14 using the chipscope (Xilinx Chipscope Pro tool inserts the logic analyzer and virtual I/O directly into the design which allows the view of any internal signal or node) to confirm the design after being downloaded on the used FPGA kit to make sure the behavior of the pipelined algorithm.

V. SIMULATIONS AND RESULTS

Fig. 11 shows the consumed clocks against data with various lengths.as we seen, the pipelined process speeds up the normal process because it consume less numbers of clocks so it will finish the encryption process before the normal. The Matlab graph show the consumed clocks in the both cases for the same data and for the same length to make sure the results.

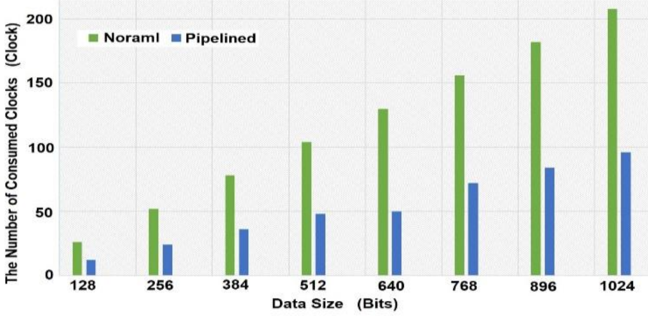
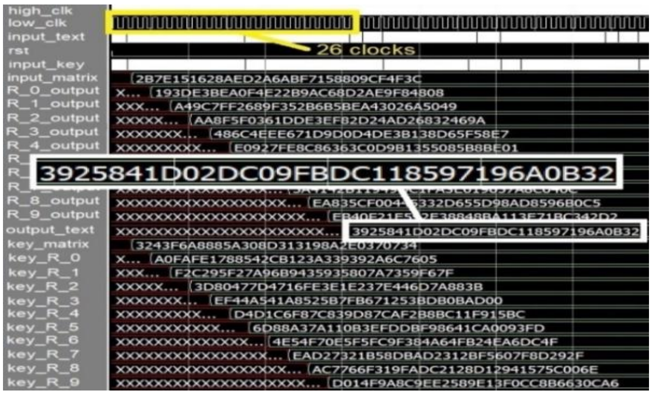


Fig. 11. The Consumed Numbers of Clocks

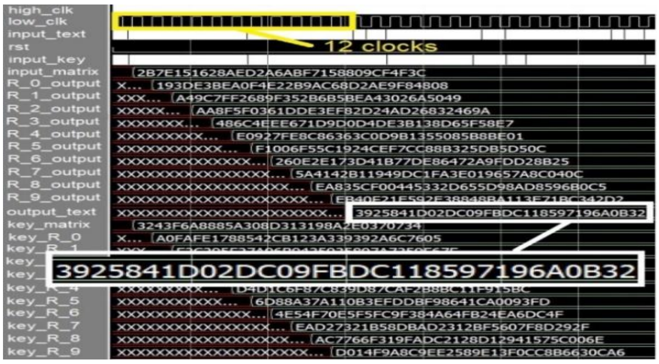
Fig. 12 shows the ModelSim simulation of the normal AES processing, as shown, the output will appear after 26 clocks. This performance will be improved in the pipelined because which will decrease these numbers to speed the encryption process.



```
high_clk
low_clk
input_text                26 clocks
rst
input_key
input_matrix   2B7E151628AED2A6ABF7158809CF4F3C
R_0_output     X...  193DE3BEA0F4E22B9AC68D2AE9F84808
R_1_output     XXX...  A49C7FF2689F352B6B5BEA43026A5049
R_2_output     XXXXX...  AA8F5F0361DDE3EFB2D24AD26832469A
R_3_output     XXXXXXX...  486C4EEE671D9D0D4DE3B138D65F58E7
R_4_output     XXXXXXXXX...  E0927FE8C86363C0D9B1355085B8BE01
R_
R_   3925841D02DC09FBDC118597196A0B32
R_
R_8_output     XXXXXXXXXXXXXXXXX...  EA835CF00445332D655D98AD8596B0C5
R_9_output     XXXXXXXXXXXXXXXXXXX...  EB40E21F592F38848BA113E71BC342D2
output_text    XXXXXXXXXXXXXXXXXXXXX...  3925841D02DC09FBDC118597196A0B32
key_matrix     3243F6A8885A308D313198A2E0370734
key_R_0        X...  A0FAFE1788542CB123A339392A6C7605
key_R_1        XXX...  F2C295F27A96B9435935807A7359F67F
key_R_2        XXXXX...  3D80477D4716FE3E1E237E446D7A883B
key_R_3        XXXXXXX...  EF44A541A8525B7FB671253BDB0BAD00
key_R_4        XXXXXXXXX...  D4D1C6F87C839D87CAF2B8BC11F915BC
key_R_5        XXXXXXXXXXX...  6D88A37A110B3EFDDBF98641CA0093FD
key_R_6        XXXXXXXXXXXXX...  4E54F70E5F5FC9F384A64FB24EA6DC4F
key_R_7        XXXXXXXXXXXXXXX...  EAD27321B58DBAD2312BF5607F8D292F
key_R_8        XXXXXXXXXXXXXXXXX...  AC7766F319FADC2128D12941575C006E
key_R_9        XXXXXXXXXXXXXXXXXXX...  D014F9A8C9EE2589E13F0CC8B6630CA6
```

Fig. 12. The ModelSim Simulation of the Normal AES processing

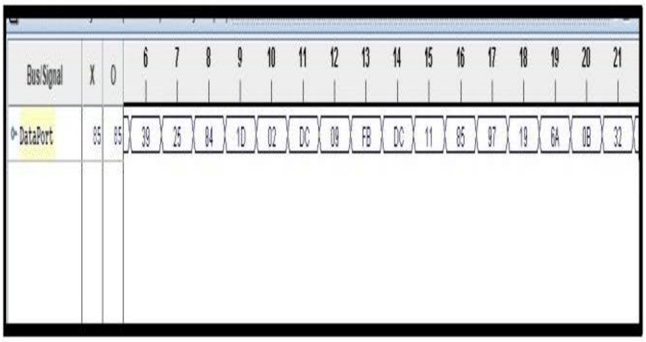
Fig. 13 shows the ModelSim simulation of the pipelined AES processing, as shown the output appears after 12 clocks. The simulation shows that if the input text and the input key are similar to the inputs of the normal implementation, the output will appear after 12 clocks. This algorithm saves 14 clock and speed up the encryption process.



```
high_clk
low_clk
input_text                12 clocks
rst
input_key
input_matrix   2B7E151628AED2A6ABF7158809CF4F3C
R_0_output     X...  193DE3BEA0F4E22B9AC68D2AE9F84808
R_1_output     XXX...  A49C7FF2689F352B6B5BEA43026A5049
R_2_output     XXXXX...  AA8F5F0361DDE3EFB2D24AD26832469A
R_3_output     XXXXXXX...  486C4EEE671D9D0D4DE3B138D65F58E7
R_4_output     XXXXXXXXX...  E0927FE8C86363C0D9B1355085B8BE01
R_5_output     XXXXXXXXXXX...  F1006F55C1924CEF7CC88B325DB5D50C
R_6_output     XXXXXXXXXXXXX...  260E2E173D41B77DE86472A9FDD28B25
R_7_output     XXXXXXXXXXXXXXX...  5A4142B11949DC1FA3E019657A8C040C
R_8_output     XXXXXXXXXXXXXXXXX...  EA835CF00445332D655D98AD8596B0C5
R_9_output     XXXXXXXXXXXXXXXXXXX...  EB40E21F592F38848BA113E71BC342D2
output_text    XXXXXXXXXXXXXXXXXXXXX...  3925841D02DC09FBDC118597196A0B32
key_matrix     3243F6A8885A308D313198A2E0370734
key_R_0        X...  A0FAFE1788542CB123A339392A6C7605
key_
key_   3925841D02DC09FBDC118597196A0B32
key_
key_R_5        XXXXXXXXXXX...  6D88A37A110B3EFDDBF98641CA0093FD
key_R_6        XXXXXXXXXXXXX...  4E54F70E5F5FC9F384A64FB24EA6DC4F
key_R_7        XXXXXXXXXXXXXXX...  EAD27321B58DBAD2312BF5607F8D292F
key_R_8        XXXXXXXXXXXXXXXXX...  AC7766F319FADC2128D12941575C006E
key_R_9        XXXXXXXXXXXXXXXXXXX...  D014F9A8C9EE2589E13F0CC8B6630CA6
```

Fig. 13. The ModelSim Simulation of the Pipelined AES processing

Fig. 14 shows the Chipscope Signals, the chipscope makes sure that the implementation of the pipelined can be used in the real cases in all the communication systems that need encryption techniques for saving the data and avoid the hacking methods.



```
Bus/Signal  X   O   6  7  8  9  10 11 12 13 14 15 16 17 18 19 20 21
DataPort    85  85  39 25 84 1D 02 DC 09 FB DC 11 85 97 19 6A 0B 32
```

Fig. 14. The Chipscope Signals

Table I shows the device utilization summary of the normal process while Table II shows the summary of the pipelined process. As shown, the improved performance algorithm needs more resources to speed up the encryption process. These increased resources are the price of the fast performance.

Table I: The device utilization summary for normal process

Logic Utilization	Used
Number of Slice Flip Flops	5546
Number of 4 inputs LUTs	2200
Number of Occupied Slices	45368
Number of Slices containing only related logic	22798
Total Number 4 Inputs LUTs	45368
Number of Bonded IOBs	133
Number of BUFGMUXs	2
Number of RAMB16BWEs	8

Table II: The device utilization summary for pipelined process

Logic Utilization	Used
Number of Slice Flip Flops	5810
Number of 4 inputs LUTs	2800
Number of Occupied Slices	46745
Number of Slices containing only related logic	22963
Total Number 4 Inputs LUTs	46745
Number of Bonded IOBs	147
Number of BUFGMUXs	2
Number of RAMB16BWEs	8

VI. CONCLUSION

The high speed of the communication made the security systems researchers are trying to improve the encryption systems to be suitable for using at such high speed systems. The goals of this article can provide significant effects on the AES encryption systems to be suitable for using in the high rate data transmission.

This paper found a new algorithm that can be used to speed up the encryption process of the AES encryption system based on the pipelined processing technique. The pipelined processing makes the consumed time less than the time consumed during the processing of the normal algorithm as every clock anew encryption process is being started. Also, the algorithm depends on the concept of making the rounds of different process to be performed at the same time instead of waiting each process to be completed.

The implementation of the normal and the pipelined processes are being provided to compare between the two algorithms and to prove the pipelined concept. The proposed algorithm consumes more resources to speed up the AES encryption process and improves the performance speed.

REFERENCES

1. Joshi, A., P.K. Dakhole, and A. Thatere. Implementation of S-Box for Advanced Encryption Standard. in 2015 IEEE International Conference on Engineering and Technology (ICETECH). 2015.
2. T, A., et al. Implementation Of Aes Coprocessor For Wireless Sensor Networks. in 2018 International Conference on Applied Smart Systems (ICASS). 2018.
3. Xu, J., et al. Differential Power Analysis of 8-Bit Datapath AES for IoT Applications. in 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE). 2018.
4. Gandh, D.R., et al. FPGA implementation of enhanced key expansion algorithm for Advanced Encryption Standard. in 2014 International Conference on Contemporary Computing and Informatics (IC3I). 2014.
5. Sung, B., K. Kim, and K. Shin. An AES-GCM authenticated encryption crypto-core for IoT security. in 2018 International Conference on Electronics, Information, and Communication (ICEIC). 2018.
6. Nayak, P., S.K. Nayak, and S. Das. A Secure and Efficient Color Image Encryption Scheme based on Two Chaotic Systems and Advanced Encryption Standard. in 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI). 2018.
7. Gaur, N., A. Mehra, and P. Kumar. Enhanced AES Architecture using Extended Set ALU at 28nm FPGA. in 2018 5th International Conference on Signal Processing and Integrated Networks (SPIN). 2018.
8. Yu, L., et al. AES Design Improvements Towards Information Security Considering Scan Attack. in 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE). 2018.
9. Lee, B., E.K. Dewi, and M.F. Wajdi. Data security in cloud computing using AES under HEROKU cloud. in 2018 27th Wireless and Optical Communication Conference (WOCC). 2018.
10. Kaur, G.R., Dr. Dheerendra Singh, Amanpreet, Multi Round Selective Encryption using AES over Storage Cloud. Global Journal of Computer Science and Technology, 2013.
11. Petrzela, J. Chaotic Oscillator Based on Mathematical Model of Multiple-Valued Memory Cell. in 2018 International Conference on Applied Electronics (AE). 2018.
12. Jishamol, T.K. and K. Rahimunnisa. Low power and low area design for advanced encryption standard and fault detection scheme for secret communications. in 2013 International Conference on Communication and Signal Processing. 2013.
13. Bulu A., B.E., Cipher with AES. 2018 3rd International Conference on Computer Science and Engineering (UBMK), 2018: p. 27-30.
14. Yuan, Y., et al. A High Performance Encryption System Based on AES Algorithm with Novel Hardware Implementation. in 2018 IEEE International Conference on Electron Devices and Solid State Circuits (EDSSC). 2018.
15. Yiqun, Z., et al. A compact 446 Gbps/W AES accelerator for mobile SoC and IoT in 40nm. in 2016 IEEE Symposium on VLSI Circuits (VLSI-Circuits). 2016.
16. Sadkhan, S.B. and A.O. Salman. Fuzzy Logic for Performance Analysis of AES and Lightweight AES. in 2018 International Conference on Advanced Science and Engineering (ICOASE). 2018.
17. Jean, J., et al. Bit-Sliding: A Generic Technique for Bit-Serial Implementations of SPN-based Primitives. 2017. Cham: Springer International Publishing.
18. Kumar, L.P. and A.K. Gupta. Implementation of speech encryption and decryption using advanced encryption standard. in 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT). 2016.
19. D’souza, F.J. and D. Panchal. Design and Implementation of AES using Hybrid Approach. in 2018 International Conference on Power Energy, Environment and Intelligent Control (PEEIC). 2018.
20. Zhang, Q. and Q. Ding. Digital Image Encryption Based on Advanced Encryption Standard (AES). in 2015 Fifth International Conference on Instrumentation and Measurement, Computer, Communication and Control (IMCCC). 2015.
21. Charot, F., E. Yahya, and C. Wagner. Efficient Modular-Pipelined AES Implementation in Counter Mode on ALTERA FPGA. in Field Programmable Logic and Application. 2003. Berlin, Heidelberg: Springer Berlin Heidelberg.

22. N. Iyer, P.V.A., D. V. Poornaiah, and V. D. Kulkarni, Efficient Hardware Architectures for AES on FPGA. Computational Intelligence and Information Technology, 2014: p. 249-257.
23. Dao, M., et al. An Energy Efficient AES Encryption Core for Hardware Security Implementation in IoT Systems. in 2018 International Conference on Advanced Technologies for Communications (ATC). 2018.
24. Labbé, A. and A. Pérez. AES Implementation on FPGA: Time - Flexibility Tradeoff. in Field-Programmable Logic and Applications: Reconfigurable Computing Is Going Mainstream. 2002. Berlin, Heidelberg: Springer Berlin Heidelberg.
25. Opritoiu, F. and M. Vladutiu. Offline self-test architecture for the inversion operation of advanced encryption standard. in 2014 IEEE 20th International Symposium for Design and Technology in Electronic Packaging (SIITME). 2014.
26. Priya, S.S., et al., An Efficient Hardware Architecture for High Throughput AES Encryptor Using MUX Based Sub Pipelined S-Box. Wireless Personal Communications, 2017. 94(4): p. 2259-2273.
27. Liu, Y., W. Gong, and W. Fan. Application of AES and RSA Hybrid Algorithm in E-mail. in 2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS). 2018.
28. He, Y., et al. Dynamic Bandwidth Scheduling Algorithm for Space Applications in FC-AE-1553 Switching Network. in 2018 Asia Communications and Photonics Conference (ACP). 2018.
29. Sapna Kumari, C. and K.V. Prasad. FPGA Implementation of AES Algorithm for Image, Audio, and Video Signal. in International Conference on Intelligent Computing and Applications. 2018. Singapore: Springer Singapore.
30. Xinmiao, Z. and K.K. Parhi, Implementation approaches for the Advanced Encryption Standard algorithm. IEEE Circuits and Systems Magazine, 2002. 2(4): p. 24-46.
31. Bu, A., et al. Correlation-Based Electromagnetic Analysis Attack Using Haar Wavelet Reconstruction with Low-Pass Filtering on an FPGA Implementaion of AES. in 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE). 2018.
32. He, F. and D. Gao. Two kinds of correlation analysis method attack on implementations of Advanced Encryption Standard software running inside STC89C52 microprocessor. in 2016 2nd IEEE International Conference on Computer and Communications (ICCC). 2016.

AUTHOR PROFILE

Mohamed Nabil, received the BE degree of Electronics and Communications in 2013. He received master's degree in Electronics and Communications in 2018 from Arab Academy for Science, Technology and Maritime Transport. Currently, He is working towards PhD degree at Minia University at Electronics and Communications Department. E-mail: mohammednabel5050@gmail.com.

Ashraf A. M. Khalaf (PhD) received his B.Sc. and M.Sc. degrees in electrical engineering from Minia University, Egypt, in 1989 and 1994 respectively. He received his Ph.D. in electrical engineering from Graduate School of Natural Science and Technology, Kanazawa University, Japan, in March 2000. He is currently associate professor at electronics and communications engineering Department, Faculty of Eng., Minia University, Egypt. E-mail: ashkhalaf@yahoo.com

Sara M. Hassan received the B.Sc. degree in Electrical Engineering from Modern academy for engineering and technology, Cairo, Egypt, in 2007, the M. Sc. degree from Ain Shams University, Cairo, Egypt, in 2013, and the Ph.D. from Ain Shams University, Cairo, Egypt, in 2017. She is currently a staff member at Modern academy for engineering and technology, Cairo, Egypt. Her fields of interest include electrical, electronics, design and implementation for communication systems. E-mail: Sara.Hassan@eng.modern-academy.edu.eg