International Journal of Emerging Science and Engineering (IJESE)
ISSN: 2319–6378 (Online), Volume-13 Issue-3, February 2025

An Efficient AES Design and Implementation Using FPGA

Hasan Abdel Aziz Mohamed, Mohamed A. Yakout

Abstract: ***The more technology develops, the greater the amount of digital information. This requires that information be secure and free from hacking, so we use encryption algorithms. One of the most famous is the Advanced Encryption Standard (AES). This paper deals with the hardware implementation of the AES Rijndael Encryption Algorithm using Xilinx Virtex-6 & Artix-7 FPGA. The work aims for a balanced design between speed, area, and power. The S-Box hardware design is based on pre-calculated look-up tables (LUTs). This method is characterized by less time and less architectural complexity. The mix-column transformations are calculated by shift and XOR methods. The encryption block is efficiently designed using Verilog-HDL and synthesized on a Virtex-6 chip (Target Device) with the help of Xilinx ISE Design Suite 14.7 Tool. The proposed architecture has good results regarding throughput, area, and power.***

Keywords: Advanced Encryption Standard (AES), Rijndael, Encryption, Hardware Description Language (HDL), Field Programmable Gate Array (FPGA).

I. INTRODUCTION

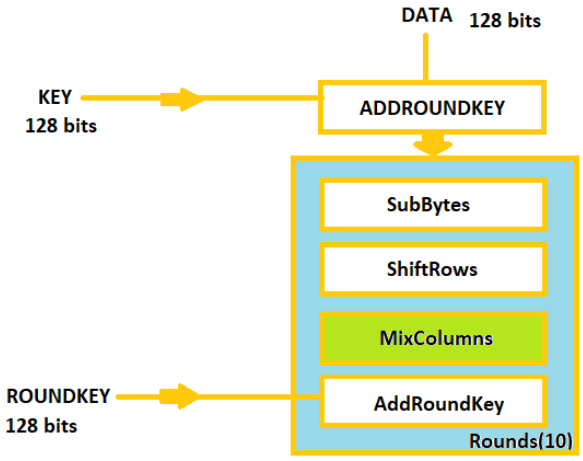
At the beginning of the emergence of communications, encryption was not important because it meant more time and resources. Encryption algorithms consist of sequential operations and their repetition a specified number of times, which puts an excessive load on the hardware (handshaking, more memory, and CPU time) for calculating. We need the fastest possible speed to process huge amounts of information. Software is no longer the appropriate solution because it needs time to execute the code. So, this work attempts to find a better hardware design to solve the previous problem with the help of Verilog code using Xilinx kits [1]. In 2000, the National Institute of Standards and Technology (NIST) selected Rijndael (Daemen and Rijmen) to present the new Advanced Encryption Standard (AES) instead of the Data Encryption Standard (DES). The new algorithm has different sizes for both the key and the encryption stages. The AES standard supports a fixed block size of 128 bits but allows for various key sizes of 128, 192, or 256 bits. additionally, AES can operate in many modes (ECB, CBC, CFB, OFB, CTR) of operation leaving the option to choose the level of encryption required [2].

Manuscript received on 18 March 2023 | First Revised Manuscript received on 02 January 2025 | Second Revised Manuscript received on 18 January 2025 | Manuscript Accepted on 15 February 2025 | Manuscript published on 28 February 2025.

*Correspondence Author(s)
Hasan Abdel Aziz Mohamed*, Senior Student, Department of Electronics and Communication Engineering, Faculty of Engineering, Mansoura University, Egypt. Email ID: Hassanabdalziz1357@mans.edu.eg, ORCID ID: 0000-0002-4945-3735
Mohamed A. Yakout, Associate Professor, Department of Electronics and Communication Engineering, Faculty of Engineering, Mansoura University, Egypt. Email ID: Myakout@mans.edu.eg, ORCID ID: 0009-0000-9788-2772

II. AES ENCRYPTION

This work centers on implementing AES with a 128-bit key size, as it is the most used and versatile for various applications. Our emphasis is on the encryption modules of the algorithm. AES comprises two main operations: encryption and decryption. The encryption process starts with the Add Round Key stage, followed by nine rounds, each consisting of four stages. The process concludes with a tenth round, which includes only three stages. Conversely, the decryption algorithm mirrors the encryption process, with the same number of rounds and stages, but the operations are performed in reverse order.



[Fig.1: AES Encryption Process]

The four stages are as follows in Fig. 1:

Sub Bytes, Shift Rows, Mix Columns, Add Round Key The tenth round leaves out the Mix Columns stage.

A. Sub Bytes

This stage is a matrix of 16x16 from pre-calculated values according to specific mathematical equations. This matrix is a table lookup of 16×16-byte values called an s-box. The s-box contains all the possible values of an 8-bit sequence ($2^8 = 16 \times 16 = 256$), but as mentioned before these values are not random [2]. We can give examples of how it works as follows, if s1,1 ={83}, then the substitution value would be determined by the intersection of the row with index '8' and the column with index '3' as shown in Fig. 2. the result would be {ec} [3].

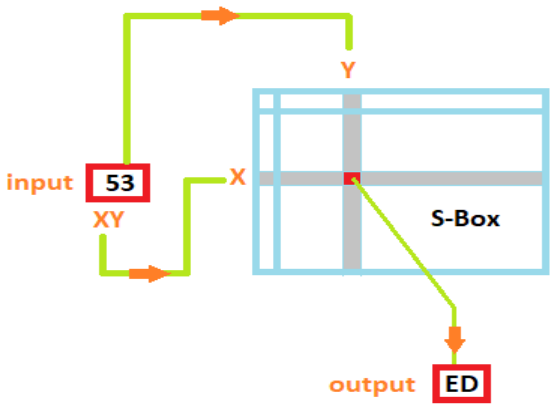
B. Shift Row

This stage is about changing

Retrieval Number:100.1/ijese.E95060412523
DOI:10.35940/ijese.E9506.13030225
Journal Website: www.ijese.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

bytes places in a specific way. It works as follows: In The first row, there is no change, but in the second row there is a shift by one byte to the left in



[Fig.2: Sub Bytes in AES]

D4	E0	B8	1e
27	bf	B4	41
11	98	5d	52
ae	F1	E5	30

D4	E0	B8	1e
bf	B4	41	27
5d	52	11	98
30	ae	F1	E5

[Fig.3: Shift Rows in AES]

In a circular manner, there is a shift of 2 bytes in the third row, and in the fourth, there is a shift of 3 bytes in the same direction as shown in Fig. 3 [2].

C. Mix Columns

This stage can be explained as multiplying two 4x4 matrices. A column is transferred from the data matrix to a new column in the matrix of the product of multiplication. Every input column is considered a polynomial vector above GF (2^8). This process is done four times, noting that the multiplication matrix (fixed matrix) is constant each time as shown in Fig. 4 [2].

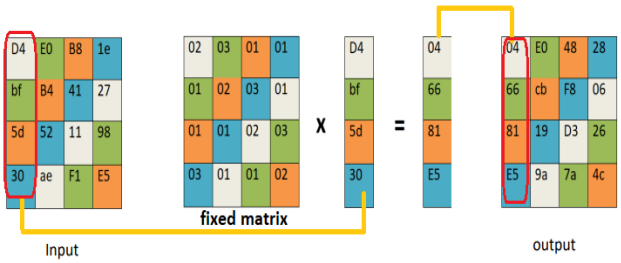
D. ADD Round Key

This stage consists of one simple operation, all we need to do is bitwise XORed each bit from the 128 bits of the state matrix with each corresponding bit from the 128 bits of the round key. An organized way to complete the process is viewed as a column-wise operation between one word (4 bytes) of the state matrix and the corresponding word of the round key [2].

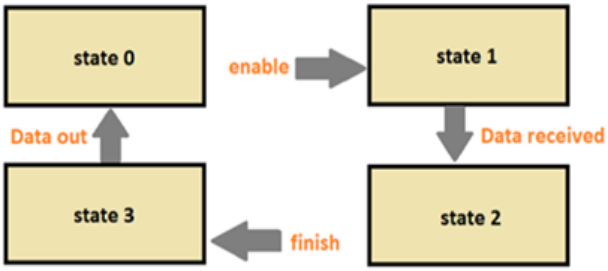
E. Key Expansion

The AES algorithm takes two basic inputs, 128 bits for data and 128 bits for key. This key consists of four words changed every round to produce a linear array of 44 words. Each round needs 4 words from the previous array. The first 4 words are the same as the input to the algorithm without any change. The process of expansion round key consists of the following 3 functions [2]:

1) RotWord is a one-byte circular left shift on a word. For example, if the input word is [b0, b1, b2, b3], the output word will be [b1, b2, b3, b0].
2) SubWord replaces every byte in a word with the equivalent byte of s-box.
3) The result of steps 1 and 2 is XORed with round constant, Rcon[j].



[Fig.4: Mix-Columns Operation]



[Fig.5: 1st FSM for Control Input & Output Process]

III. DESIGN

The design has been divided into individual blocks. All the individual blocks are coded separately with Verilog and tested for functionality. Finally, the top module representing the Encryption block is designed by instantiating all the individual blocks. AES needs 10 rounds which means 10 blocks if we use instantiation but, in our design, we use one block and Finite State machines (FSM) as shown in Fig. 6 which reduce area, power, and get acceptable speed compared to modified designs. However, we implemented S-Box using the Lookup Table method increasing the computation cost.

A. First Finite State Machine

The first FSM implements the input and output to the system, we want to input 128 bits byte by byte, and then the encryption process will start. If encryption is complete, the final process gets output byte by byte. The first FSM is divided into four states:

State 0: The system remains inactive while waiting for the enable signal.

State 1: The system is active and receiving data byte by byte.

State 2: The received data is fully loaded, and the system has entered the encryption process.

State 3: Encryption is complete, and the system outs data byte by byte as shown in Fig.5.

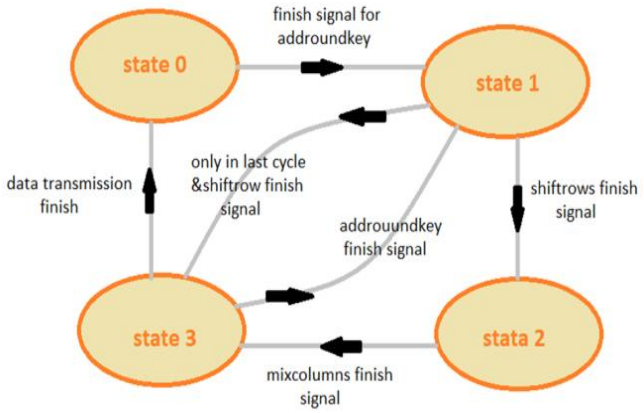
B. Seconed Finite State Machine

The second FSM begins operation by waiting for the data-loading process to complete. Once the loading is finished, it transitions to the first step of the encryption process. Each step in the encryption corresponds to a specific state within the FSM. When the operation starts, the FSM activates the module by setting the enable

Retrieval Number:100.1/ijese.E95060412523
DOI:10.35940/ijese.E9506.13030225
Journal Website: www.ijese.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

signal. It then waits for the module's done signal before transitioning to the next state. This process continues step by step through the encryption cycles. Furthermore, the FSM sets the finish flag and returns to the initial state (state0). The second FSM is divided into four states:



[Fig.6: 2nd FSM to Control Encryption Process]

State 0: Add Round key 0, State 1: Shift Rows, State 2: Mix Columns, State 3: Add Round key. As shown in Fig. 6.

C. Mix Columns Method

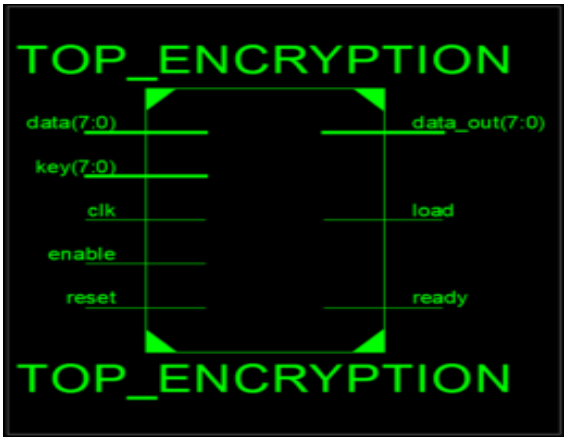
In this work, we implement the Mix Columns operation using shift and XOR operations, achieving the required transformations by multiplying bytes by 2 and 3. If we were multiplying by {02} then the process could be done as follows, a 1-bit left shift followed by a conditional bitwise XOR with (00011011) if the leftmost bit of the original value (before shifting) was 1. Multiplication by (03) is a multiplication by (02) bitwise XORed with the original value. This method is less computationally complex than GF (2^8) multiplications [2].

IV. HARDWARE IMPLEMENTATION

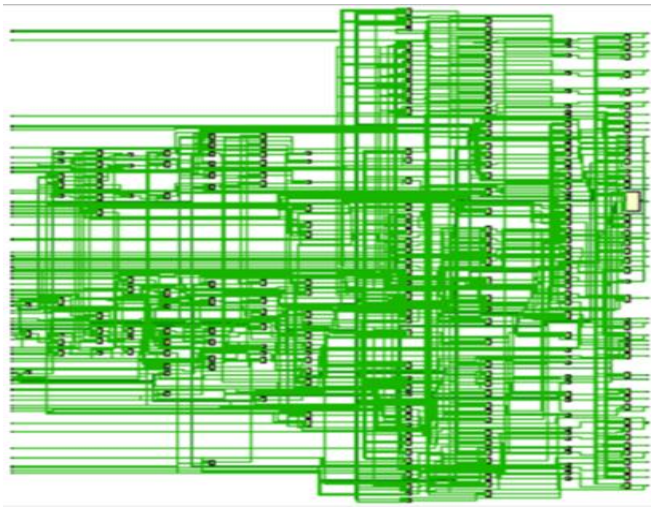
We implement the design using FPGAs (Virtex-6: device XC6VLX75T, Artix-7: device A7100TCSG324-3) as shown in Fig. 7 and utilize the Xilinx ISE 14.7 tool for RTL synthesis, placement, and routing.

A. On ARTIX-7 Family

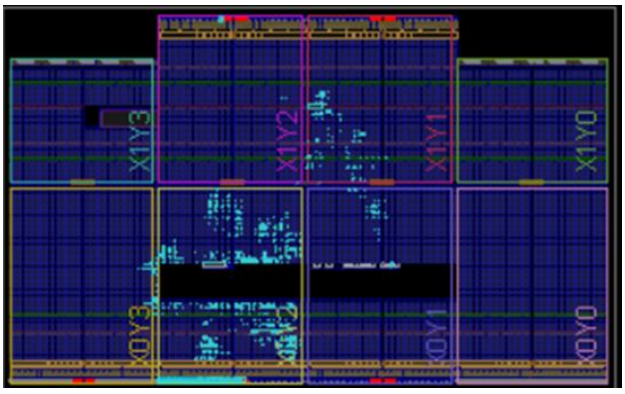
Verilog code is transferred into RTL by synthesizing the compiler in ISE and implemented in ARTIX 7 kit (this family targets low power and area) as shown in Fig. 8 and Fig. 9.



[Fig.7: Synthesis of AES Encryption]



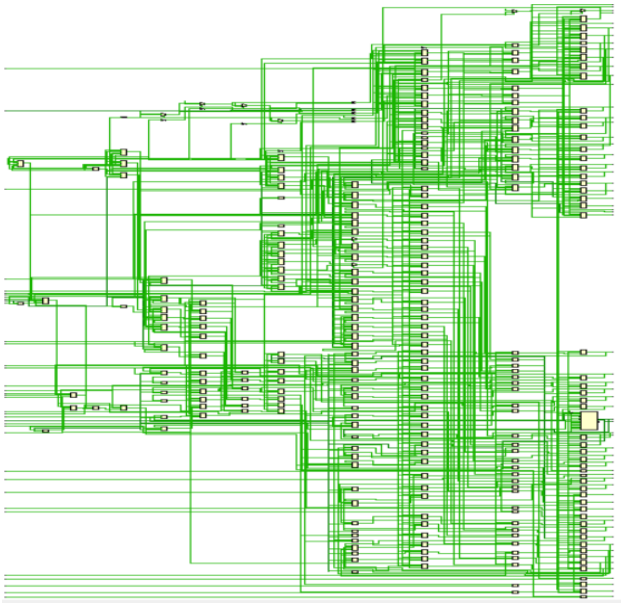
[Fig.8 RTL View of Artix-7]



[Fig.9: Design on Floor Plane of ARTIX 7]

B. On VIRTEX-6 Family

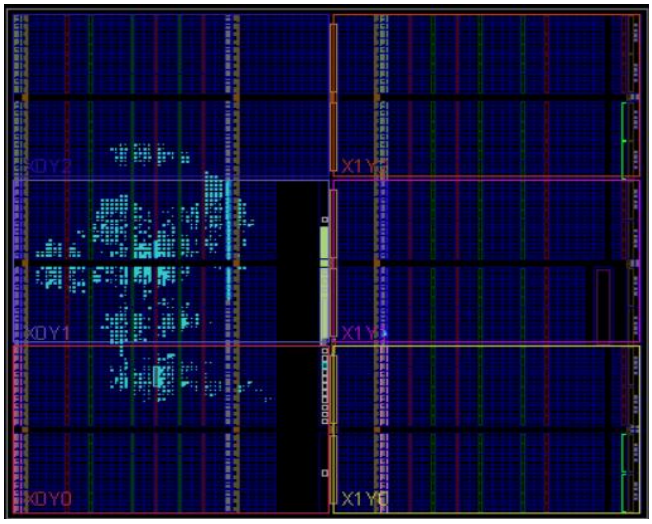
Similarly to the ARTIX 7 kit, the RTL synthesis and implementation in the VIRTEX 6 kit (This family targets high performance) are shown in Fig. 10 and Fig. 11.



[Fig.10: RTL View of VIRTEX-6]

Retrieval Number:100.1/ijese.E95060412523
DOI:10.35940/ijese.E9506.13030225
Journal Website: www.ijese.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)



[Fig.11: Design on Floor Plane of VIRTEX 6]

V. SIMULATION AND RESULTS

A. Utilization and Power Analysis

Area and utilization are important parameters in modern digital designs and many applications that require low power and small areas like IoT applications with acceptable speed. Our design emphasizes minimal resource utilization and eliminates complex computations that increase power consumption. The AES encryption in the ARTIX 7 FPGA achieves little power, the total consumed power is (.82W) as shown in Table 1. The design is implemented in a small area which reduces costs. The area numbers are 1191 slice registers, and 2074 slice LUTs (3%) in utilization as shown in Table 2. The same small numbers are achieved in the VIRTEX 6 FPGA our design consumed (1.29W) and used 1517 slice registers, and 2210 slice LUTs (4%) in utilization as shown in Table 3, Table 4.

Table 1: Power Analysis Report on ARTIX 7 Kit

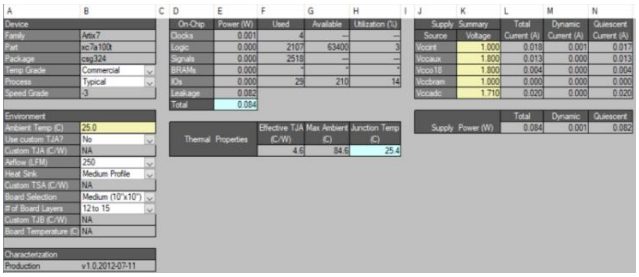


Table 2: Utilization of AES-Encryption on ARTIX 7 Kit

```
Device utilization summary:
---------------------------

Selected Device : 7a100tcsg324-3


Slice Logic Utilization:
 Number of Slice Registers:             1191  out of  126800     0%
 Number of Slice LUTs:                  2074  out of  63400      3%
    Number used as Logic:               2074  out of  63400      3%

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:    2312
   Number with an unused Flip Flop:     1121  out of   2312     48%
   Number with an unused LUT:            238  out of   2312     10%
   Number of fully used LUT-FF pairs:    953  out of   2312     41%
   Number of unique control sets:         21

IO Utilization:
 Number of IOs:                           29
 Number of bonded IOBs:                   29  out of    210    13%

Specific Feature Utilization:
 Number of Block RAM/FIFO:                 1  out of    135     0%
    Number using Block RAM only:           1
 Number of BUFG/BUFGCTRLs:                 1  out of     32     3%
```

Table 3: Power Analysis Report on VIRTEX 6 Kit

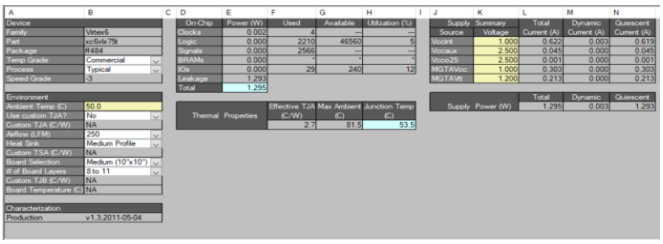


Table 4: Utilization of AES-Encryption on VIRTEX 6

```
Device Utilization Summary:

Slice Logic Utilization:
  Number of Slice Registers:                1,517 out of  93,120    1%
    Number used as Flip Flops:              1,516
    Number used as Latches:                     1
    Number used as Latch-thrus:                 0
    Number used as AND/OR logics:               0
  Number of Slice LUTs:                     2,210 out of  46,560    4%
    Number used as logic:                   2,085 out of  46,560    4%
      Number using O6 output only:          1,801
      Number using O5 output only:             52
      Number using O5 and O6:                 232
      Number used as ROM:                       0
    Number used as Memory:                     75 out of  16,720    1%
      Number used as Dual Port RAM:             0
      Number used as Single Port RAM:           0
      Number used as Shift Register:           75
        Number using O6 output only:           65
        Number using O5 output only:            1
        Number using O5 and O6:                 9
    Number used exclusively as route-thrus:    50
      Number with same-slice register load:    43
      Number with same-slice carry load:        7
      Number with other load:                   0

Slice Logic Distribution:
  Number of occupied Slices:                  780 out of  11,640    6%
  Number of LUT Flip Flop pairs used:       2,490
    Number with an unused Flip Flop:        1,129 out of   2,490   45%
    Number with an unused LUT:                280 out of   2,490   11%
    Number of fully used LUT-FF pairs:      1,081 out of   2,490   43%
    Number of slice register sites lost
      to control set restrictions:              0 out of  93,120    0%
```

Table 5: VIRTEX 6 FPGA (Period:4.46 ns, Max Delay:1.4 ns)

```
Timing summary:
---------------

Timing errors: 0  Score: 0  (Setup/Max: 0, Hold: 0)

Constraints cover 3773 paths, 0 nets, and 988 connections

Design statistics:
   Minimum period:   4.465ns{1}   (Maximum frequency: 223.964MHz)
   Maximum path delay from/to any node:   1.414ns
```

Table. 6: ARTIX 7 FPGA (Period:6 ns, Max Delay:1.97 ns)

```
Timing summary:
---------------

Timing errors: 0  Score: 0  (Setup/Max: 0, Hold: 0)

Constraints cover 3778 paths, 0 nets, and 995 connections

Design statistics:
   Minimum period:   6.008ns{1}   (Maximum frequency: 166.445MHz)
   Maximum path delay from/to any node:   1.976ns
```

B. Timing Reports

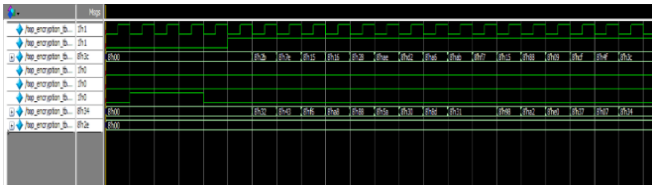
The proposed design achieves fast speed, with a maximum path delay of 1.4 ns on the VIRTEX 6, as shown in Table 5. The ARTIX 7, with lower performance, has a higher maximum path delay 1.97 ns as shown in Table 6**.**

C. Waveform Results

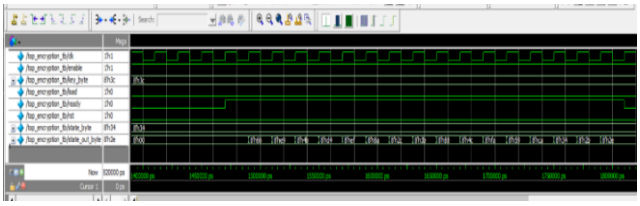
In AES Encryption 128, 128-bit plain text and 128-bit encryption keys are given as inputs, and 128-bit cipher text is output. The simulation waveform using Questasim is shown in Fig. 12 and Fig. 13.

Retrieval Number:100.1/ijese.E95060412523
DOI:10.35940/ijese.E9506.13030225
Journal Website: www.ijese.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)



[Fig12: The Waveform of Input 128-Bit Plain Text and 128-Bit Encryption Key]



[Fig.13: The Waveform of Output 128-Bit Cipher Text]

D. Performance

Table 7: Performance Compassion of AES on Different Architecture – Virtex-6 Family

Design	Device	Slices	Freq. (MHZ)	Throughput (Gbps)	Power (w)	Throughputs/slice (Mbps/slice)
Chellam and Natarajan (2018) [4]	6Virtex	2537	740.7	94.81	-	37.37
Wang and Ha (2013) [5]	Virtex 6	9071	319.29	40.86	11.02	4.51
Oukili and Bri (2017) [6]	Virtex 6	5759	849.18	108.69	-	17.08
Shanthi Rekha and Saravanan (2019) [7]	Virtex 6	1626	166.66	0.24	3.87	0.15
Rajaseker and Mangalam 2020 [8]	Virtex 6	1551	190.658	0.56	0.815	0.361

Note: all designs use the XC6VLX240T device from the VIRTEX 6 family, but we use the XC6VLX75T.

Compared with other designs implemented in the VIRTEX 6 kit we have low utilization (1517 slice registers) around the same as Rajaseker and Mangalam 2020 [8], we consumed more power (+40%) but we got more efficient throughput (90.52) (1) at a frequency (223.964 MHZ) [9]. The other designs have more utilization, some of them operate in high frequencies that cause more violations as shown in Table 7 [10]. This design does not have the fastest speed. However, its throughput is high, at the same, the area and power are low which means more efficiency [11].

$$\text{Throughput} = \frac{\text{Number of processed bits}}{\text{Critical path delay}} \quad \dots \quad (1)$$

Number of processed bits= 128 bits

VI. CONCLUSION

Encryption is indispensable as an essential element in the security of information and data transmission. Both fields have an increasing number of applications over time, requiring different speed/area trade-offs while considering power consumption. Our work aims to implement a low area, power, and high-speed AES encryption. The S-Box using the look-up table method gives less complex architecture and saves processing time by Fetching the values directly from locations in memory. Fetching values from memory locations is generally faster than executing complex computation operations. Using FSMs in the encryption process can greatly reduce area and power, making designs more efficient. In addition, many designs reduce area and power, but they are inefficient in speed and throughput. We perform Mix columns using a shift and an XOR operation. This method has less computational complexity than GF (2^8) multiplications. This Proposed design has total throughput (90.52) at frequency (223.964 MHZ), utilization (1517 slice registers), power (1.295 w), and the maximum path delay is 1.4 ns in VIRTEX 6 FPGA by using Verilog end ISE tool.

DECLARATION STATEMENT

After aggregating input from all authors, I must verify the accuracy of the following information as the article's author.

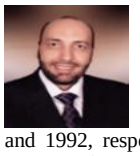
- **Conflicts of Interest/ Competing Interests:** Based on my understanding, this article has no conflicts of interest.
- **Funding Support:** This article has not been sponsored or funded by any organization or agency. The independence of this research is a crucial factor in affirming its impartiality, as it has been conducted without any external sway.
- **Ethical Approval and Consent to Participate:** The data provided in this article is exempt from the requirement for ethical approval or participant consent.
- **Data Access Statement and Material Availability:** The adequate resources of this article are publicly accessible.
- **Authors Contributions:** The authorship of this article is contributed equally to all participating individuals.

REFERENCES

1. JNNCE Journal of Engineering & Management, Volume 4, No.1, January – June 2020 -Implementation of AES Algorithm Using Verilog Ganesh Gopal Shet, Jamuna V, Shravani S, Nayana H G, Pramod Kumar S J N N College of Engineering, Shimoga. https://typeset.io/pdf/implementation-of-aes-algorithm-using-verilog-1uuuwecb45.pdf
2. B. A. Forouzan, "Advanced encryption techniques," in *Cryptography and Network Security*, 2015, pp. 191–220, McGraw-Hill Education. https://almuhammadi.com/sultan/books_2020/Forouzan.pdf
3. National Institute of Standards and Technology, "Announcing the Advanced Encryption Standard (AES) (FIPS Publication 197)," U.S. Department of Commerce, 2001. https://csrc.nist.gov/pubs/fips/197/final
4. Chellam, M.B. and Natarajan, R. (2018), "AES hardware accelerator on FPGA with improved throughput and resource efficiency", Arabian Journal for Science and Engineering, Vol. 43 No. 12, pp. 6873-6890. DOI: https://doi.org/10.1007/s13369-017-2925-0
5. Wang, Y. and Ha, Y. (2013), "FPGA-based 40.9-gbits/s masked AES with area optimization for storage area network", IEEE Trans. Circ. Syst. II:

Express Briefs, Vol. 60 No. 1, pp. 36-40. DOI: https://ieeexplore.ieee.org/abstract/document/6472789
6. Oukili, S. and Bri, S. (2017), "Hardware implementation of AES algorithm with logic S-box", J. Circuits, Systems, and Computers, Vol. 26 No. 9. DOI: https://doi.org/10.1142/S0218126617501419
7. Shanthi Rekha, S. and Saravanan, P. (2019), "Low-Cost AES-128 implementation for edge devices in IoT applications", Journal of Circuits, Systems, and Computers, Vol. 28 No. 4, p. 1950062. DOI: https://doi.org/10.1142/S0218126619500622
8. P., R. and H., M. (2021), "Design and implementation of power and area optimized AES architecture on FPGA for IoT application", Circuit World, Vol. 47 No. 2, pp. 153-163. DOI: https://www.emerald.com/insight/content/doi/10.1108/cw-04-2019-0039/full/pdf
9. Lakshminarayana, G., Deshpande, Dr. A. A., & Babu, Dr. M. G. (2019). Design of High Speed Advanced Encryption Standard using PPA and PPM. In International Journal of Innovative Technology and Exploring Engineering (Vol. 8, Issue 9, pp. 2679–2684). DOI: https://doi.org/10.35940/ijitee.i8989.078919
10. Satyanarayana*, B., & Srinivasan, D. M. (2019). Advanced Encryption Standard for Data Encryption using EDK Environment in FPGA. In International Journal of Recent Technology and Engineering (IJRTE) (Vol. 8, Issue 4, pp. 11969–11972). DOI: https://doi.org/10.35940/ijrte.d9920.118419
11. Murugan M, S., & Sasilatha, Dr. T. (2019). Hardware Implementation of Hybrid Model Encryption Algorithm for Secure Transmission of Medical Images. In International Journal of Engineering and Advanced Technology (Vol. 9, Issue 1s, pp. 164–167). DOI: https://doi.org/10.35940/ijeat.a1042.1091s19

AUTHOR'S PROFILE



Dr. Mohamed Abdel-Alim Yakout, born in Cairo, Egypt, is an Associate Professor in the Electronics and Electrical Communications Engineering Department at the Faculty of Engineering, Mansoura University (MU), Egypt. He earned his B.Sc. with honors and M.Sc. degrees from MU in 1986 and 1992, respectively. He completed his Ph.D. at MU in 1997, with a portion of his research conducted as a visiting scholar at the Technical University of Nova Scotia (TUNS), Canada, between 1993 and 1996. His Ph.D. thesis was recognized with the MU Best Ph.D. Thesis Award for 1997/1998. Since 2001, he has also served as an Assistant Professor in the Electronics Department at the College of Technological Studies (CTS), PAAET, Kuwait. His research interests span VLSI, current-mode circuits, artificial neural networks (ANN), RF circuits, and nanoelectronics.

Hasan Abdel Aziz Mohamed. Earned his Bachelor of Engineering (BE) degree in Electronics and Communication Engineering from the Faculty of Engineering, Mansoura University, Egypt, in 2022. He is currently pursuing a Master of Engineering (ME) degree at the American University in Cairo (AUC), where he also works as a teaching assistant. In addition, he serves as a member of the technical staff specializing in ASIC and FPGA design at the Center of Nanoelectronics and Devices. Hasan's research interests include lightweight cryptography algorithms, hardware design for deep learning, and AI accelerators. His work focuses on low-power VLSI design, cryptography, and AI applications.