IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

ENHANCED DATA ENCRYPTION USING OPTIMIZED S-BOX TECHNIQUES

[1]Mrs.N.Chamanthi, [2]D. Harini, [3]K.Gowtham, [4]B.Dinesh kumar, [5]K.Bhavya, [6]A.Dhanasekhar

[1]Assistant Professor, [2]Student, [3]Student, [4]Student, [5]Student, [6]Student

[1]ECE(Electronics and communication engineering),

[1]Siddhartha institute of science and technology, Puttur, India

Abstract: The Advanced Encryption Standard (AES) is a widely used symmetric key cryptographic algorithm ensuring secure data transmission. This paper presents an optimized AES implementation using Verilog HDL, enhancing both performance and resource efficiency. The design incorporates fundamental AES operations, including SubBytes, ShiftRows, MixColumns, and AddRoundKey, while employing optimization techniques to reduce latency and hardware utilization. A key enhancement is the reuse of the S-box for both encryption and decryption, with a multiplexer selecting between the two modes. Additionally, the MixColumns and AddRoundKey operations are integrated into a single unit to streamline processing. The proposed design further improves efficiency by reducing S-box instances through optimized key expansion, generating a 128-bit key every five cycles. Security is reinforced by extending the key size to 256 bits, with unique transformations applied across different encryption rounds. Functional simulation and synthesis results validate the correctness and effectiveness of the implementation, demonstrating its suitability for resource-constrained environments such as IoT devices and FPGA-based secure communication systems. This study highlights the advantages of Verilog HDL in cryptographic hardware design, offering scalability and flexibility for future advancements in encryption techniques.

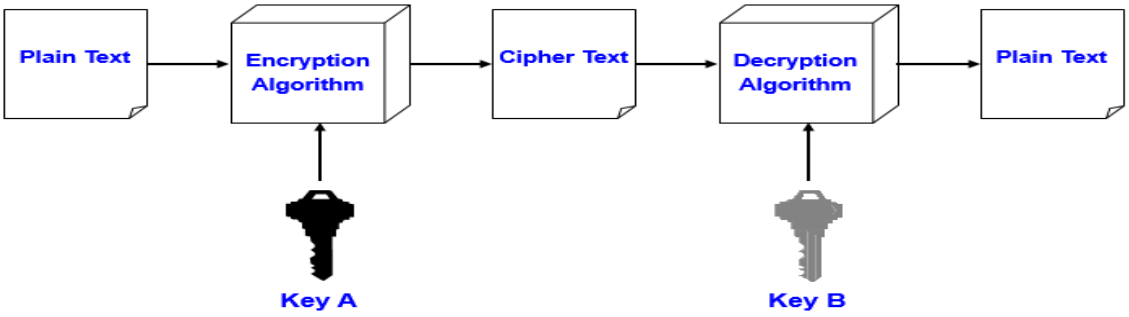
Index Terms- AES, Cryptography, Verilog HDL, S-box Optimization, FPGA**.**

I. INTRODUCTION

The process of transforming plaintext into ciphertext to guarantee confidentiality, authentication, and message integrity is known as cryptography. Asymmetric key cryptography, which employs a public key for encryption and a private key for decryption (like RSA); symmetric key cryptography, which employs the same key for both encryption and decryption (like AES, DES, and Blowfish); and hash functions, which guarantee data integrity (like SHA-256). The non-Feistel block cipher known as the Advanced Encryption Standard (AES), which was established by NIST in 2001, encrypts blocks of 128 bits and supports key sizes of 256 bits (14 rounds), 192 bits (12 rounds), and 128 bits (10 rounds). While stream ciphers encrypt data bit-by-bit or byte-

by-byte, block ciphers, such as AES, encrypt data in fixed-size blocks. The substitution-box (S-Box), an essential part of block ciphers.

A.CRYPTOGRAPHY PROCESS



This paper introduces a novel modular approach to construct highly nonlinear substitution-boxes (S-boxes) to enhance the security of block ciphers. The technique allows for flexible S-box creation with very small parameter changes by integrating permutation, transformation, and modular inverses. When an example S-box is evaluated using key cryptographic criteria, strong security features are shown. Additionally, a picture encryption method with strong statistical and differential encryption capabilities uses the proposed S-box. In the contemporary digital world, safe data communication is essential because private information is vulnerable to assaults. Encryption techniques are crucial for protecting private data since internet data transmission is growing so rapidly. Cryptographic algorithms transform data into an incomprehensible format to maintain confidentiality.

II. BACKGROUND

The Advanced Encryption Standard (AES) is a symmetric key cryptographic algorithm renowned for its robust security and flexibility, supporting key sizes of 128, 192, or 256 bits. AES has replaced older encryption methods such as DES, providing efficiency and adaptability in secure communication systems ([1]).

Recent advancements in encryption focus on hybrid techniques and domain-specific optimization. For instance, adaptive encryption systems like GCN2 integrate chaotic neural networks and genetic algorithms to enhance security and performance, though they pose challenges in practical implementation ([2]). For IoT applications, lightweight and hybrid encryption schemes address the need for resource efficiency while maintaining strong security ([3], [4]).

Techniques that combine cryptography and steganography, such as AES with LSB steganography, improve data hiding capabilities, while lightweight multi-level encryption frameworks enable scalability for resource-constrained environments ([5]). Additionally, AES implementation using Verilog HDL optimizes hardware usage and achieves low latency, making it ideal for FPGA-based systems ([1]).These developments demonstrate AES's adaptability in ensuring secure data transmission across diverse fields.

III. PROPOSED SYSTEM

A .PLATFORM

Xilinx Vivado is a powerful FPGA and SoC design platform that helps engineers create and test digital circuits. It supports both traditional hardware languages like VHDL and Verilog, as well as High-Level Synthesis (HLS) using C/C++. Vivado includes a large library of pre-built IP cores and a user-friendly IP Integrator for easy system design. It also offers built-in simulation and debugging tools, such as the Vivado Simulator and Integrated Logic Analyzer (ILA), for testing and troubleshooting designs.

For synthesis and implementation, Vivado uses advanced optimization techniques to improve performance and resource usage. It supports automation through TCL scripting, making design processes more efficient. The platform also provides hardware debugging and prototyping features, allowing users to test their designs on Xilinx development boards. Compared to the older ISE Design Suite, Vivado is faster, more efficient, and includes better debugging tools, making it the preferred choice for modern FPGA development.

using selective transformation over input text and improves security by extending the key size to 256 bits, with different transformation blocks applied for each round range. This approach enhances both efficiency and security while maintaining the integrity of AES encryption and decryption.

B. AES WITH S -B0X OPTIMIZATION

The proposed AES implementation optimizes hardware efficiency by reusing the same S-box for both encryption and decryption, with the only difference being the Affine transform. A multiplexer selects between the S-box and inverse S-box based on the operation mode. The design also enhances performance by reusing S-box and Mix Column blocks, integrating "Mix Column" and "Add Round Key" into a single Mix Block. Key expansion is optimized by generating a 128-bit key every five cycles, reducing the required S-box instances from eight to four per cycle. The keys generated in earlier cycles are strategically used in later cycles to streamline processing. Additionally, the implementation reduces computational complexity using selective transformation over input text and improves security by extending the key size to 256 bits, with different transformation blocks applied for each round range. This approach enhances both efficiency and security while maintaining the integrity of AES encryption and decryption. Fig1: Pipelined structure of proposed method. the pipeline structure of the proposed design, where each color represents different round as follows, Mix – round 0 Mix – round 1 Mix – round 2 Mix – round 3 and Mix – round 14 Each word having a size of 32 bits. In cycle 1, we are doing Mix operation of word 0.

The pipeline structure of the proposed design, where each color represents different round as follows,

Mix – round 0

Mix – round 1

Mix – round 2

Mix – round 3 and Mix – round 14

Each word having a size of 32 bits. In cycle 1, we are doing Mix operation of word 0 (mix_0). We can denote this as cycle1 [round0 (mix_0)].

S box	Shift	Mix	Cycle
		Mix_0	1
Sub_0	-	Mix_1	2
Sub_1	Shift_0	Mix_2	3
Sub_2	Shift_1	Mix_3	4
Sub_3	Shift_2	-	5
Key_2	Shift_3	Mix_0	6
Sub_0	-	Mix_1	7
Sub_1	Shift_0	Mix_2	8
Sub_2	Shift_1	Mix_3	9
Sub_3	Shift_2	-	10
Key_3	Shift_3	Mix_0	11
Sub_0	-	Mix_1	12
Sub_1	Shift_0	Mix_2	13
Sub_2	Shift_1	Mix_3	14
Sub_3	Shift_2	-	15
-	Shift_3	Mix_0	16
.	.	Mix_1	17
Key_14	.	Mix_2	18
Sub_0	-	Mix_3	19
Sub_1	Shift_0	.	.
Sub_2	Shift_1	.	.
Sub_3	Shift_2	-	.
-	Shift_3	Mix_0	71
		Mix_1	72
		Mix_2	73
		Mix_3	74

Figure 1: Pipelined structure of proposed method

1. **Sub-bytes:**

Each byte in the state array undergoes a non-linear substitution, where a new byte replaces the existing one. This substitution is based on a predefined SBOX, a lookup table containing 256 hexadecimal values. The original data in the state matrix is replaced by locating a specific value in the SBOX using one byte for the column and another for the row. By selecting the appropriate row and column, the state matrix's data is transformed accordingly. This process continues until all bytes in the state matrix are updated. Lookup tables are used to minimize hardware complexity and reduce computation time. The original state matrix values, represented as [b'(i,j)].

2. **Shift rows**

During this process, the data in the state matrix is cyclically shifted to the left. The number of shifts depends on the data's position within the matrix. Shifting occurs row by row. In the first row, no changes are made since indexing starts from zero. In the second row, the last column's data is shifted left once. In the third row, shifting occurs twice, and so on. This ensures a structured transformation of the state matrix. The notation [bl,0] represents elements belonging to the same row in the matrix.

3. **Mix Columns**

A matrix containing rows of shifted data is used as input in this step. After that, this matrix is multiplied by another matrix made up of predetermined values, which is obtained by using a standard polynomial a(X). Matrix multiplication is essentially the same as regular matrix multiplication, with the exception that in this process, the corresponding data in a row is multiplied by the entire column, and then the XOR operation is performed in place of the addition operation. The state matrix's corresponding hexadecimal data is denoted by [b(i,j)]. Following the transformation, the data is represented by [b'(i,j)].

4. **Adding round key**

Since the key is also 128 bits in size, the data in the state matrix that is produced from the previous procedure is XORed with each of the state matrix's columns in this step. During round 0 of the encryption process, the first key is added. The initial key used in round 0 is the source of the unique keys required for each round. The key expansion module generates this collection of keys; the quantity of keys produced is contingent upon the number of algorithmic rounds. The state matrix's corresponding hexadecimal data is denoted by [b(i,j)]. Following the transformation, the data is represented by [b'(i,j)].

5.**Key expansion**

AES employs a key expansion algorithm to generate a series of round keys for both encryption and decryption. The algorithm takes the initial key as input and produces multiple round keys through a series of transformations. A total of (N+1) 128-bit round keys are derived, where N represents the number of AES rounds. In Verilog HDL, the key expansion algorithm can be implemented using combinational and sequential logic circuits. The 128-bit initial key, stored in a register, serves as the input. Through successive transformations, the algorithm generates the required round keys. The key expansion process in Verilog HDL typically follows specific stages to ensure proper key generation for AES operations.

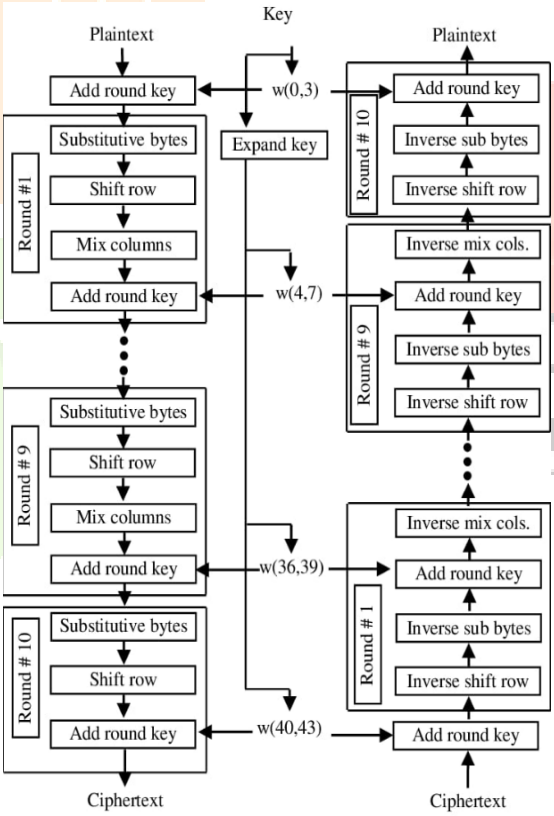


Figure2: Architecture of 256 AES Algorithm

IV. DESIGN PROCEDURE

1. Create Verilog design input file(s) using template driven editor.

2. Compile and implement the Verilog design file(s).

3. Create the test-vectors and simulate the design (functional simulation) without using a PLD (FPGA or CPLD).

4. Assign input/output pins to implement the design on a target device.

5. Download bitstream to an FPGA or CPLD device.

6. Test design on FPGA/CPLD device

A Verilog input file in the Xilinx software environment consists of the following segments:

Header: module name, list of input and output ports.

Declarations: input and output ports, registers and wires.

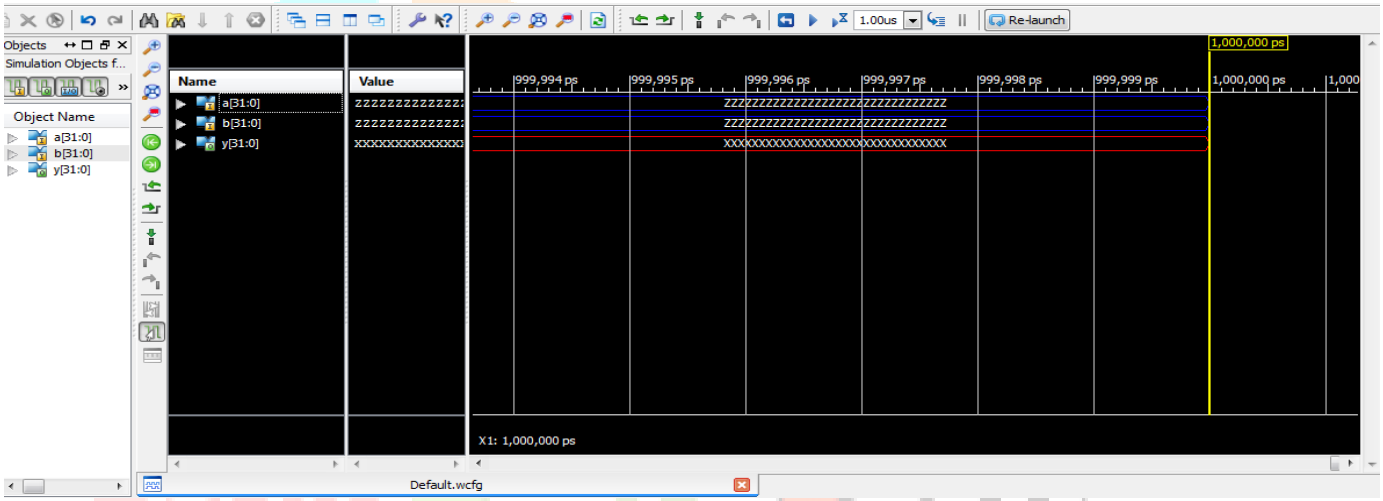
Logic Descriptions: equations, state machines and logic functions.

End: end module.

a). XILINX VIVADO SIMULATION PROCEDURE

After completion of synthesis we will go simulation in order to verify the functionality of the implemented design.

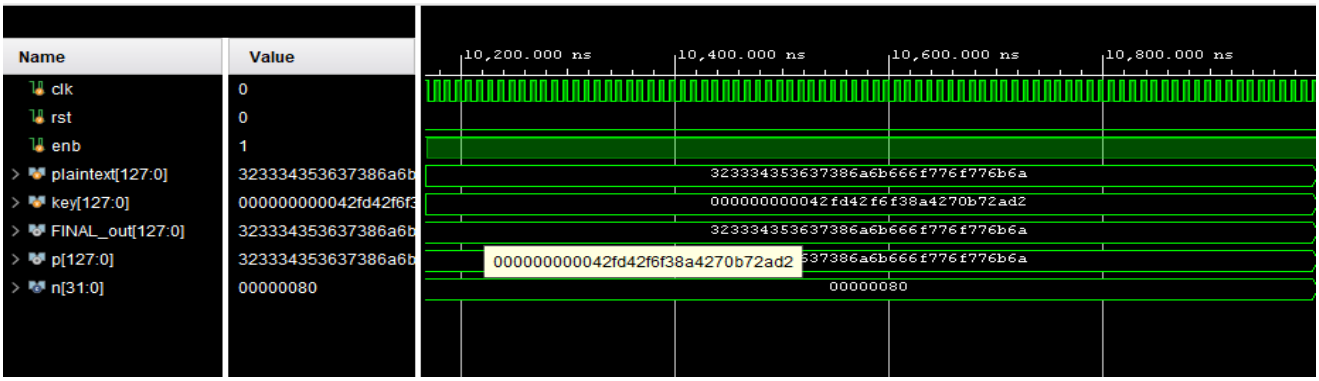
- Click on **Run Simulation** and set the module that is need to Run
- Next **double click on Run Behavioral Simulation** to check the errors. If no errors are found then double click on simulate behavioral model to get the output waveforms.
- After clicking on **simulate behavioral model**, the following window appears



- the simulation widow will appear pass the input values by making force constant and if it is clock by making force clock. Mention the simulation period and run for certain time and results will appear as shown in following window. Verify the results to the given input values

V. RESULTS AND DISCUSSION

A. Simulation



Simulation Waveform Results:

clk (clock): 0 → Repeated toggling observed.

rst (reset): 0 → No reset signal activated.

enb (enable): 1 → Enable signal active.

Plaintext[127:0]: 3233343536373836ab666f72676b6a6a

key[127:0]: 00000000042fd42f638a4270b72ad2

FINAL out[127:0]: 3233343536373836ab666f72676b6a6a (matches plaintext output).

p[127:0]: 3233343536373836ab666f72676b6a6a

n [31:0]: 00000080

Resource Utilization:

Design Name: Top

Slice LUTs Used: 4635

Total Slice LUTs Available: 134600

B. Area

Name	Slice LUTs (134600)
top	4635

C. Delay

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay
Path 1	∞	9	9	15	e1/ins2/u0/w_reg[3][21]/C	e1/ins2/u0/w_reg[1][30]/D	6.548

VI. CONCLUSION AND FUTURE SCOPE

The current method utilizes DES and Triple-DES, both of which are traditional symmetric key cryptographic techniques. Although these methods were widely used in the past, they have security weaknesses due to their relatively small key sizes (56-bit for DES and multiple iterations for Triple-DES), making them vulnerable to attacks with modern computational power. Additionally, the standard AES implementation follows a conventional structure without optimization, leading to higher computational costs, particularly in S-box and Mix Column operations.

Future Scope: Ongoing research in cryptanalysis may lead to new attacks or vulnerabilities against AES-128. The cryptographic community will continue to analyze the algorithm for potential weaknesses, and if any are found, modifications or adjustments to AES-128's design may be proposed to enhance its security.

REFERENCES

[1] B. Swayam Prakash et al., "Design of Advanced Encryption Standard using Verilog HDL," 2023 7th International Conference on Trends in Electronics and Informatics (ICOEI), 2023. DOI: 10.1109/ICOEI56765.2023.10125765

[2] K. S. Madhuri and J. Mungara, "Data Security using Integrated GCN2 Encryption Algorithm," 2020 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT), 2020. DOI: 10.1109/RTEICT49044.2020.9315534

[3] P. Kumar and B. Deshpande, "Ensuring of Secure Data Transmission by Modified Encryption and Decryption Method in IoT," 2023 IEEE 4th Annual Flagship India Council International Subsections Conference (INDISCON), 2023. DOI: 10.1109/INDISCON58499.2023.10270161

[4] S. Mishra, D. Singh, D. Pant, and A. Rawat, "Secure Data Communication Using Information Hiding and Encryption Algorithms," 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS), 2022. DOI: 10.1109/ICAIS53314.2022.9743004

[5] P. Chakrabarty et al., "Enhanced Data Security Framework Using Lightweight Cryptography and Multi-Level Encryption," 2024 International Conference on Communication, Computer Sciences and Engineering (IC3SE), 2024. DOI: 10.1109/IC3SE62002.2024.10593191

[6] S. Najam, M. Ur Rehman, and J. Ahmed, "Data Encryption Scheme Based On Adaptive System," 2020 Global Conference on Wireless and Optical Technologies (GCWOT), 2020. DOI: 10.1109/GCWOT49901.2020.9391622

[7] Haider, M. A. Qureshi, A. Saeed and M. A. Haider, "Enhanced Model for Data Security in Cyber Space Using Combined Steganographic and Encryption Techniques," 2023 IEEE International Conference on Emerging Trends in Engineering, Sciences and Technology (ICES&T), Bahawalpur, Pakistan, 2023, pp. 1-6, doi: 10.1109/ICEST56843.2023.10138827

[8] Y. Wei, B. Li, B. Zhang, Y. Yan and Q. Zhou, "High-performance Data Hybrid Encryption Scheme Based on Mimic Defense," 2023 3rd International Symposium on Computer Technology and Information Science (ISCTIS), Chengdu, China, 2023, pp. 114-121, DOI: 10.1109/ISCTIS58954.2023.10213016.

[9] K. I. Masud, M. R. Hasan, M. M. Hoque, U. D. Nath and M. O. Rahman, "A New Approach of Cryptography for Data Encryption and Decryption," 2022 5th International Conference on Computing and Informatics (ICCI), New Cairo, Cairo, Egypt, 2022, pp. 234-239, DOI: 10.1109/ICCI54321.2022.9756078.