

FSM-Based Area Efficient AES-128 Encryption Processor with Pipelined Round Architecture for FPGA Security Applications

Mrs. Alamananda Swapna¹, Yallapu Pravalika²

1. Assistant Professor, Electronics and Communication Engineering Department & Sanketika Vidya Parishad Engineering College (SVPEC), Visakhapatnam, India.

2. M. Tech Student, Electronics and Communication Department & Sanketika Vidya Parishad Engineering College (SVPEC), Visakhapatnam, India

Abstract - The Advanced Encryption Standard (AES) is one of the most widely used symmetric key cryptographic algorithms for secure data communication in modern digital systems. With the rapid growth of Internet of Things (IoT), embedded systems, and FPGA-based applications, there is an increasing demand for high-speed and area-efficient hardware encryption architectures. Traditional software-based encryption methods suffer from higher latency and lower performance, while fully pipelined hardware AES architectures consume significant hardware resources.

This project presents the design and implementation of an FSM-Based Area-Efficient AES-128 Encryption Processor with a partially pipelined round architecture using Verilog HDL. The proposed design utilizes a finite state machine (FSM) to control the AES encryption process efficiently while reusing a single AES round block across multiple rounds to reduce hardware utilization. In addition, partial pipelining is introduced within the AES round operations to improve timing performance and throughput without significantly increasing the silicon area.

The architecture consists of major AES transformation stages including SubBytes, ShiftRows, MixColumns, AddRoundKey, and Key Expansion. A dedicated key expansion module dynamically generates all required round keys for AES-128 encryption. The entire design is verified using standard NIST AES test vectors, and the obtained ciphertext output matches the expected AES encryption results accurately.

Simulation and waveform analysis are carried out using Icarus Verilog and GTKWave tools. The experimental results demonstrate that the proposed architecture achieves a balanced trade-off between hardware area and encryption throughput, making it highly suitable for FPGA-based secure communication systems, embedded platforms, and resource-constrained IoT devices. The modular and scalable structure of the design also enables future enhancement toward AES-192, AES-256, and advanced low-power cryptographic architectures.

1. INTRODUCTION

In the modern digital era, secure communication and data protection have become essential requirements in almost every technological domain, including banking systems, wireless communication, cloud computing,

military applications, healthcare systems, and Internet of Things (IoT) devices. With the rapid increase in cyber threats and unauthorized access to sensitive information, cryptographic techniques play a major role in ensuring confidentiality, integrity, and authenticity of data transmission. Among various cryptographic standards, the Advanced Encryption Standard (AES) is one of the most widely adopted symmetric encryption algorithms due to its strong security, flexibility, and efficient implementation capabilities.

AES was standardized by the National Institute of Standards and Technology (NIST) in 2001 as a replacement for the older Data Encryption Standard (DES). AES operates on fixed 128-bit data blocks and supports key sizes of 128 bits, 192 bits, and 256 bits. Compared to DES, AES provides significantly stronger security with higher resistance against brute-force and cryptanalytic attacks. Due to these advantages, AES is extensively used in secure wireless communication, VPNs, secure file storage, smart cards, and embedded security systems.

Although software implementations of AES are easy to develop and flexible to modify, they often suffer from higher execution time and increased processor overhead, especially in real-time embedded systems. Hardware-based implementations using Very Large Scale Integration (VLSI) technology and FPGA platforms provide improved performance, lower latency, parallel processing capability, and enhanced security. However, designing an efficient AES hardware architecture involves several challenges such as reducing hardware area, minimizing power consumption, and maintaining high throughput.

Traditional fully pipelined AES architectures achieve very high encryption speeds but require a large number of hardware resources, making them unsuitable for low-cost and resource-constrained devices. On the other hand, iterative AES architectures reduce hardware utilization by reusing the same encryption hardware for multiple rounds, but they generally experience lower throughput due to sequential operation. Therefore, achieving an optimal balance between hardware area and performance remains an important research problem in AES hardware design.

This project presents an FSM-Based Area-Efficient AES-128 Encryption Processor with a partially pipelined round architecture implemented using Verilog HDL. The proposed design uses a Finite State Machine (FSM) to

control AES operations efficiently while reusing a single AES round module across all encryption rounds to reduce hardware complexity. In addition, partial pipelining is introduced within the AES round stages to improve timing performance and overall throughput. The architecture includes all major AES transformations such as SubBytes, ShiftRows, MixColumns, AddRoundKey, and Key Expansion.

The complete design is simulated and verified using Icarus Verilog and GTKWave tools with standard NIST AES test vectors. The obtained results confirm correct encryption functionality and demonstrate that the proposed architecture provides a balanced trade-off between area efficiency and encryption performance. Due to its modular and scalable structure, the proposed system is highly suitable for FPGA-based security applications, embedded systems, and resource-constrained IoT environments.

2. Body of Paper

AES Operations:

1. SubBytes :

SubBytes is a nonlinear byte substitution operation in which each byte of the input state matrix is replaced with another byte using a predefined substitution table called the S-box. The AES S-box is designed using multiplicative inverse operations in Galois Field arithmetic followed by affine transformation.

The main purpose of SubBytes is to provide nonlinearity and confusion in the encryption process, thereby increasing resistance against linear and differential cryptanalysis attacks.

2. ShiftRows:

ShiftRows is a permutation operation performed on the AES state matrix. In this operation, each row of the state matrix is cyclically shifted toward the left by a certain number of byte positions.

The shifting pattern is as follows:

- First row: No shift
- Second row: Shift by 1 byte
- Third row: Shift by 2 bytes
- Fourth row: Shift by 3 bytes

The purpose of ShiftRows is to provide diffusion by spreading byte values across multiple columns, thereby increasing encryption complexity.

The ShiftRows operation rearranges the byte positions without modifying the actual byte values.

3. MixColumns :

MixColumns is a column-wise transformation performed using Galois Field arithmetic. Each column of the AES state matrix is multiplied with a fixed polynomial matrix in $(GF(2^8))$.

The transformation matrix is given as:

```
[
\begin{bmatrix}
02 & 03 & 01 & 01 \\
01 & 02 & 03 & 01 \\
01 & 01 & 02 & 03 \\
03 & 01 & 01 & 02
\end{bmatrix}
```

The purpose of MixColumns is to provide strong diffusion by combining all four bytes within each column. A small change in one byte affects the entire column after transformation.

In the proposed design, MixColumns is implemented using XOR operations and finite field multiplication to achieve efficient hardware realization.

4. AddRoundKey :

AddRoundKey is the final operation in each AES round where the current state matrix is XORed with the corresponding round key generated by the Key Expansion unit.

The operation is represented as:

```
[
State_{out} = State_{in} \oplus RoundKey
]
```

where:

- $(State_{in})$ = input AES state
- $(RoundKey)$ = generated round key
- $(\oplus)$ = XOR operation

This transformation integrates the secret encryption key into the AES encryption process, ensuring cryptographic security.

In the proposed architecture, AddRoundKey is implemented using parallel XOR logic for efficient hardware execution.

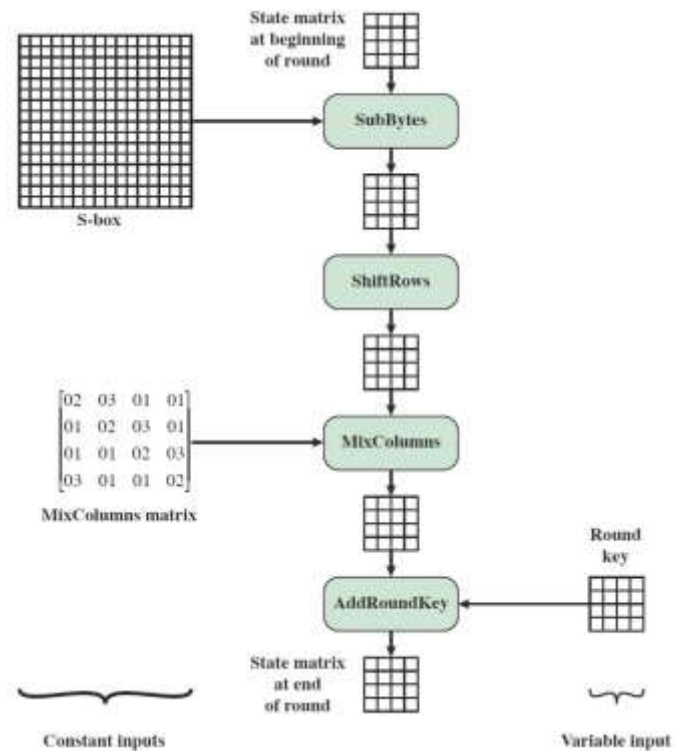
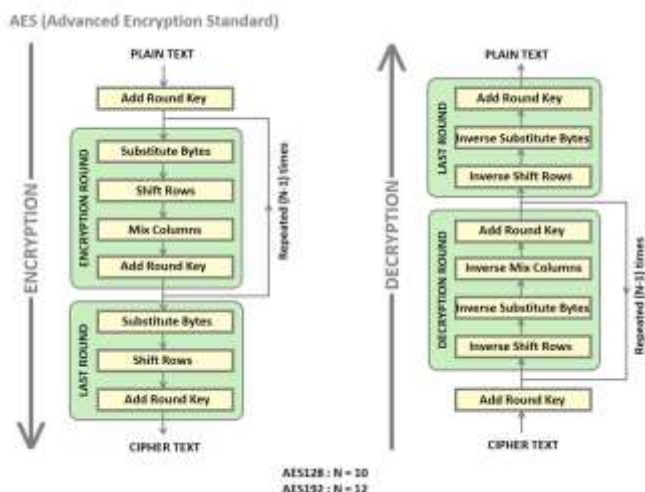
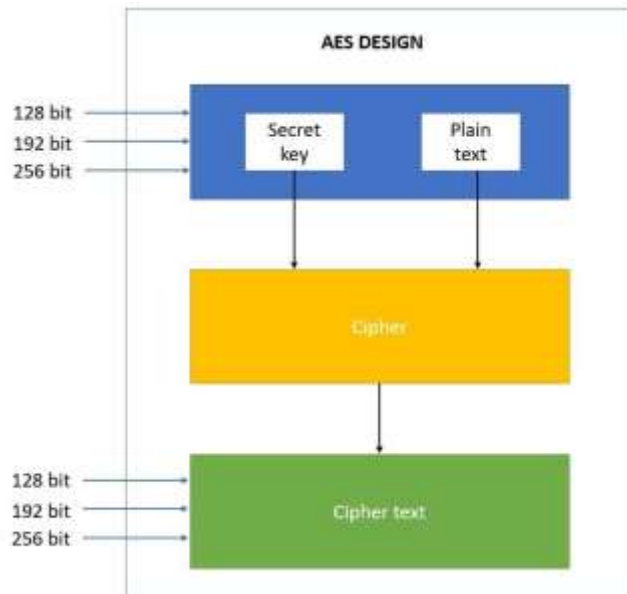
FSM States:

- IDLE
- KEYGEN
- LOAD

- ROUND
- DONE

Key Features:

- Single AES round reused iteratively
- Dynamic key expansion
- Partial pipelining for timing improvement

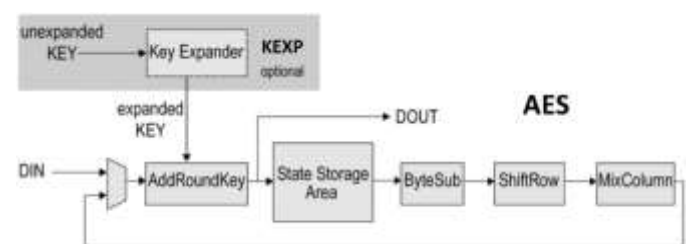


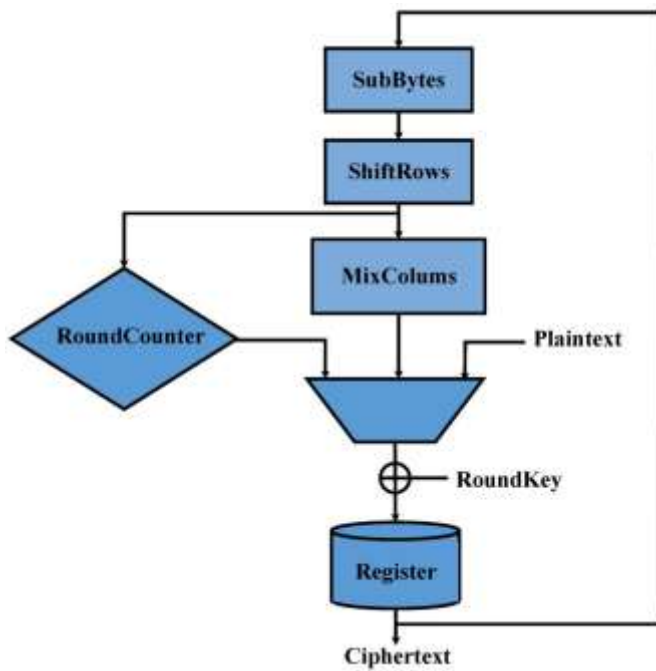
SYSTEM ARCHITECTURE

Main Blocks:

- Input Interface
- Key Expansion Unit
- FSM Controller
- AES Round Module
- Output Unit

Data Flow:





RESULTS

Input:

Plaintext:

3243f6a8885a308d313198a2e0370734

Key:

2b7e151628aed2a6abf7158809cf4f3c

Output:

Ciphertext:

3925841d02dc09fbdc118597196a0b32

Performance:

- Frequency $\approx$ 100 MHz
- Latency = 11 cycles
- Throughput $\approx$ 1.16 Gbps

Observation:

- Correct AES encryption verified
- FSM transitions validated
- Balanced area-throughput performance achieved

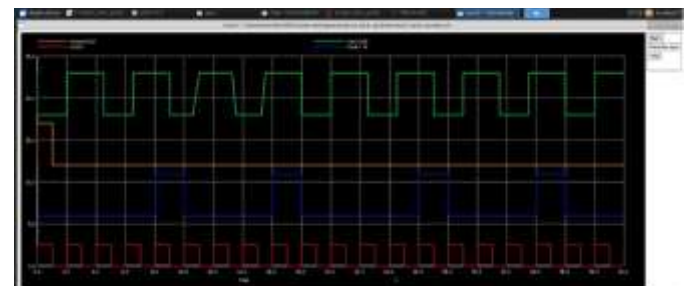
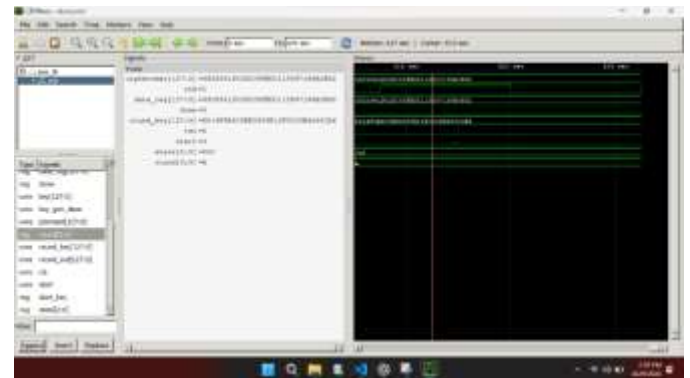
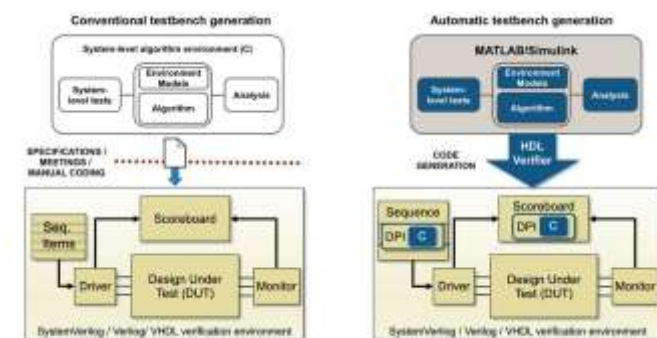
EXPERIMENTAL SETUP / SIMULATION

Tools Used:

- Icarus Verilog
- GTKWave

Verification:

- Standard NIST AES test vectors
- Waveform analysis for FSM and data flow verification



3. CONCLUSION

- Successfully implemented AES-128 encryption processor.
- FSM architecture reduced hardware area.
- Partial pipelining improved performance.

- Verified using NIST test vectors.
- Suitable for FPGA-based embedded security applications.

4.ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my project guide and faculty members for their valuable guidance, continuous support, and encouragement throughout the completion of this project. Their technical suggestions and motivation played a major role in the successful implementation of this work.

I also extend my heartfelt thanks to our institution and department for providing the necessary laboratory facilities, software tools, and academic environment required for carrying out this project effectively.

I would like to thank my friends and classmates for their support, discussions, and assistance during the development and testing phases of the project.

Finally, I express my gratitude to my family members for their constant encouragement and moral support throughout the course of this work.

REFERENCES

- [1] A. Hodjat and I. Verbauwhede, "A 21.54 Gbits/s Fully Pipelined AES Processor on FPGA," Proc. 12th IEEE Symp. on Field-Programmable Custom Computing Machines (FCCM), Napa, CA, USA, Apr. 2004, pp. 308–309.
- [2] National Institute of Standards and Technology (NIST), "Advanced Encryption Standard (AES)," FIPS PUB 197, Nov. 2001.
- [3] M. Dworkin, M. Sonmez Turan, and N. Mouha, "Advanced Encryption Standard (AES)," FIPS 197-upd1, NIST, May 2023.
- [4] J. T. Grycel and R. J. Walls, "DRAB-LOCUS: An Area-Efficient AES Architecture for Hardware Accelerator Co-Location on FPGAs," arXiv:1911.04378, 2019.