Industrial Engineering Journal
ISSN: 0970-2555
Volume : 54, Issue 7, July : 2025

VLSI DESIGN OF A RECONFIGURABLE HARDWARE ARCHITECTURE FOR AES ENCRYPTION WITH SUPPORT FOR 128, 192, AND 256-BIT KEYS

[1]KURUNELLI JYOTHI PRAKASH, [2]T. MOHANDAS,
[1]M. Tech, VLSI System Design, Student,
Department of ECE,
Email: kurunellijyothiprakash@gmail.com,
[2]Assistant Professor(C), Department of ECE,
Email: mohandas.nitdgp@gmail.com,
JNTUH University College of Engineering Sultanpur,
Sangareddy, Telangana, India

ABSTRACT
The Advanced Encryption Standard (AES) is a widely adopted symmetric cryptographic algorithm used for securing digital communication. AES supports three key lengths — 128, 192, and 256 bits — each with varying levels of security and computational complexity. we present the VLSI design of a reconfigurable and unified AES encryption core capable of dynamically supporting all three key lengths within a single hardware architecture. The proposed design is implemented in Verilog HDL and features a pipelined architecture that enhances throughput while maintaining design scalability. Key expansion is handled through a modular approach to generate round keys based on the selected key size. The unified design enables flexible deployment across a range of applications including wireless sensor networks, IoT security modules, and embedded systems, where both performance and area-efficiency are critical. Simulation results demonstrate that the proposed design achieves a good trade-off between hardware utilization, encryption speed, and energy efficiency.

Key Words: AES-128, AES-192, AES-256, Verilog HDL.

I. INTRODUCTION

In today's interconnected digital landscape, safeguarding data as it travels across open and often insecure networks is essential. Cryptography plays a pivotal role in ensuring secure communication by transforming plaintext into ciphertext using mathematical algorithms and cryptographic keys, allowing only authorized entities to access the original information [1]. It underpins security in diverse applications, including online transactions (SSL/TLS), wireless communication (WPA2/WPA3), secure email (PGP), and VPNs (IPSec), and is increasingly vital in resource-constrained environments like the Internet of Things (IoT) and Wireless Sensor Networks (WSNs) [4]. Rooted in mathematical disciplines such as number theory and algebra, and supported by computational complexity theory, modern cryptography balances efficiency and robustness. Symmetric-key systems, which use a single shared key, are computationally efficient but require secure key distribution. Asymmetric systems, on the other hand, use key pairs to simplify key management at the cost of higher computational load. Both types aim to uphold core security principles: confidentiality, integrity, authentication, and availability (CIAA) [2]. Formal verification models such as the “Dolev-Yao” framework have further

strengthened protocol design through provable security [6]. Historically, the Data Encryption Standard (DES), introduced in 1977, was the first widely adopted symmetric block cipher, encrypting 64-bit data blocks using a 56-bit key over 16 rounds [7]. Despite its early success, DES was eventually compromised due to its limited key space. By 1998, dedicated hardware could break DES in under a day, exposing its vulnerability to brute-force attacks and advanced cryptanalysis techniques [9]. In response, Triple DES (3DES) extended security by applying DES thrice, but at the cost of performance and without resolving fundamental design limitations [10]. To address these challenges, NIST initiated a global search for a successor in 1997. The Rijndael algorithm, developed by "Daemen" and "Rijmen", emerged as the winner and was standardized as the Advanced Encryption Standard (AES) in 2001 [11]. AES offers enhanced security, flexibility, and efficiency, making it the prevailing standard for symmetric encryption in modern cryptographic systems.

II. LITERATURE SURVEY

The Advanced Encryption Standard (AES), adopted by NIST in 2001, superseded DES due to its stronger security and efficiency. AES encrypts 128-bit data blocks and supports key sizes of 128, 192, and 256 bits, which correspond to 10, 12, and 14 rounds of encryption, respectively [19]. These variants offer a trade-off between security, hardware cost, and computational efficiency, with AES-128 widely used in general-purpose systems, and AES-192 and AES-256 employed in high-security applications such as government and military systems.

REVIEW ON VARIOUS AES KEY LENGTHS:

- **AES-128:** It is preferred for its balance of security and low hardware complexity, making it ideal for embedded systems and mobile devices. Early designs employed iterative architectures to minimize area and power, but suffered from high latency. Notable work by "Satoh" et al. introduced compact AES cores using optimized S-boxes [13], while "Ichikawa" improved Mix Columns performance through efficient Galois Field operations [14]. To enhance throughput, researchers explored loop unrolling and partial pipelining. "Kuo" and "Verbauwhede's" design partially unrolled AES rounds for performance gains [15], and "McLoone" introduced a pipelined approach balancing area and speed [16]. Power-efficient S-box implementations were also a major focus; Standaert proposed low-power designs using composite fields [17], and Canright's FPGA-optimized S-box supported lightweight applications [18].
- **AES-192:** Although offering stronger security, is less common due to its more complex key expansion and higher latency.

Its 192-bit key requires 12 rounds and generates 52 words, adding control complexity. “Elbirt” et al. showed that AES-192 consumed more area and cycles than AES-128 [20]. Parameterized designs by “Sharma” and “Mathew” supported AES-128/192/256 in unified architectures [21], while others highlighted AES-192’s poor area-performance trade-off in multi-mode implementations due to conditional FSMs and memory overheads [22].

- **AES-256:** It delivers the highest security level, often used in classified communication and secure cloud services. It demands 14 rounds and a 256-bit key, increasing design complexity and hardware usage. “Hodjat” et al. noted a 1.7x area and 2x latency increase over AES-128 in their extended design [23]. “Canright” and “Batina” highlighted the challenges of computing 60 key words, leading to timing issues in the key scheduler [24]. “Alrabaee” et al. introduced shared logic designs to reduce area while preserving performance [25], and pipelined AES-256 designs have been proposed for high-throughput systems, though with significant hardware duplication [26].

III. PROPOSED METHODOLOGY

The proposed design introduces a **Unified AES Encryption Core** capable of handling all three AES key lengths—128, 192, and 256 bits—within a single hardware module. This unified architecture eliminates the need for separate implementations, thereby optimizing area, reducing power consumption, and streamlining development. Key features include configurable control logic for selecting the appropriate number of encryption rounds (10, 12, or 14), dynamic key expansion, and adaptive round key management. The design maintains modularity and supports pipelining to ensure high throughput, making it well-suited for resource-constrained applications such as Wireless Sensor Networks (WSNs), IoT nodes, and real-time secure communication systems.

A. BLOCK DIAGRAM

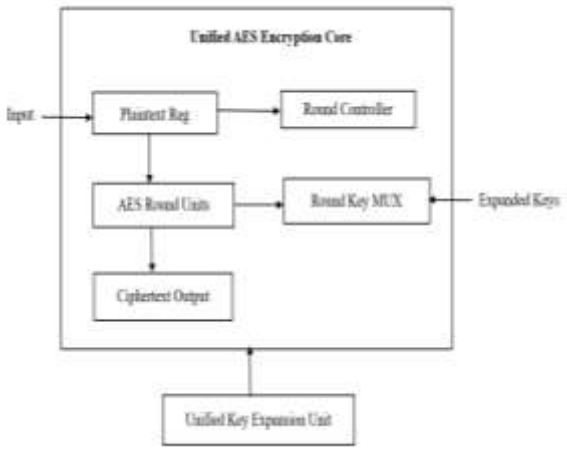

Fig. 1 Block diagram of Unified AES

The proposed Unified AES Encryption Core enables encryption for all three AES variants (128, 192, and 256-bit keys) within a single hardware module. The design is optimized for low-area, high-speed applications such as IoT, WSNs, and embedded security systems. Figure 1 illustrates the architecture and data flow across core components.

The Block diagram consist of:

- **Input & Plaintext Register:** The plaintext input is first latched into a buffer register to synchronize with the system clock and enable pipelined execution.
- **Round Controller**: This unit determines the total number of encryption rounds based on the selected AES mode. It generates control signals to manage round progression, terminate encryption after the final round, and bypass MixColumns in the last stage, as required by AES specifications.
- **AES Round Unit**: Each round performs four operations—SubBytes, ShiftRows, MixColumns, and AddRoundKey. These are implemented as separate Verilog modules for modularity and pipeline integration.
    - **SubBytes:** Applies a non-linear substitution using an S-box implemented as a LUT for low-latency lookups.
    - **ShiftRows:** Rearranges bytes in the AES state to ensure diffusion. Implemented via byte-wise reordering logic.
    - **MixColumns:** Performs matrix multiplication in GF(2^8) using efficient shift-XOR circuits for speed and compactness.
    - **AddRoundKey:** Applies a bitwise XOR between the AES state and the current round key—lightweight and suitable for pipeline stages.

  The final round omits MixColumns, adhering to the AES standard.
- **Unified Key Expansion Unit**: This module supports dynamic key expansion for all three AES key sizes. It generates the required number of round keys based on the selected mode, with logic to manage 44 (AES-128), 52 (AES-192), or 60 (AES-256) 32-bit key words. A 2-bit control input (key_size_sel) configures the AES mode:

Key size Select	AES Mode	Key Length	Rounds
2'b00	AES-128	128 bits	10
2'b01	AES-192	192 bits	12
2'b10	AES-256	256 bits	14

- Dynamic internal logic adjusts the number of rounds, round keys, and FSM control accordingly.
- **Round Key MUX**: A multiplexer selects the appropriate round key at each stage based on round count and provides it to the AES round unit.
- **Ciphertext Output**: After completion of all rounds, the encrypted data is made available as ciphertext output.
- **Expanded Keys**: The key expansion module outputs all required round keys, which are then used sequentially by the round controller and MUX logic for encryption.

B. Control FSM for Unified AES Core

The Finite State Machine (FSM) governs the control flow of the unified AES encryption process, ensuring proper sequencing of key expansion and round operations based on the

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 54, Issue 7, July : 2025

selected key length (128, 192, or 256 bits). The FSM comprises four primary states: Idle, Key Expansion, Initial Round, and Rounds, as shown in Figure 2.

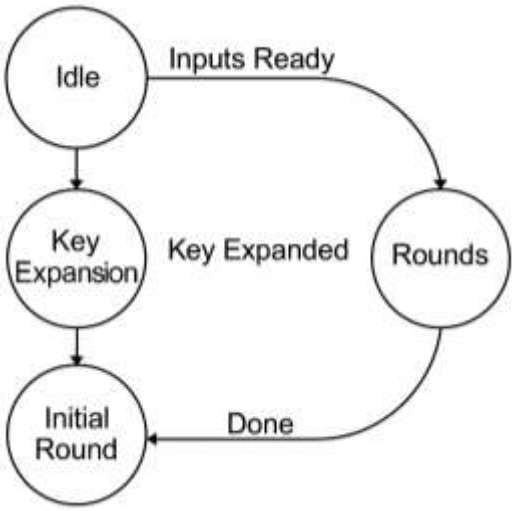

Figure 2. AES Unified State Diagram

- **Idle:** The core remains in a low-power standby mode until a valid encryption request is detected. Transition to the next state occurs when the start signal is asserted and both plaintext and key inputs are available. This state ensures minimal power usage and prepares the control logic for operation.
- **Key Expansion:** In this state, the key expansion unit generates all required round keys based on the selected AES mode. Once key generation is complete and signalled via key_expansion_done, the FSM advances to the initial round.
- **Initial Round:** This stage applies the initial AddRoundKey operation, combining the plaintext with the first round key using XOR. It sets up the AES state matrix for the main encryption rounds.
- **Rounds:** The core performs iterative AES rounds—9 for AES-128, 11 for AES-192, and 13 for AES-256—followed by a final round. Each round includes SubBytes, ShiftRows, MixColumns (skipped in the final round), and AddRoundKey transformations. A round counter tracks progress, and upon completion of the final round, the FSM asserts the done signal and returns to the Idle state.

This FSM ensures synchronized operation and dynamic adjustment of the AES core's behavior depending on the key size, maintaining compliance with AES standards while supporting efficient pipelined execution.

IV. RESULTS:

The waveform simulations validate the correct functionality of the unified AES encryption core across different key configurations. Each simulation captures key operational stages, including reset, key and plaintext loading, encryption rounds, and output generation. The clk and rst_n signals manage synchronous operation and initialization, while the start signal triggers the encryption process. Based on the key_size_sel input (00 for AES-128, 01 for AES-192, 10 for AES-256), the core dynamically adjusts the number of rounds and selects the appropriate key schedule through internal control logic. Upon completion, the

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 54, Issue 7, July : 2025

data_out signal presents the final ciphertext, and the done flag confirms successful encryption. These results confirm the core's adaptability to varying AES modes without requiring hardware modifications.

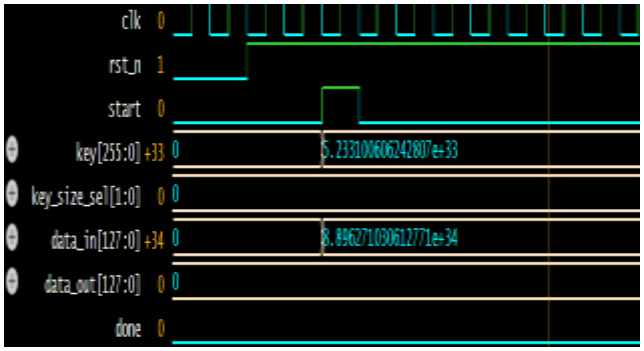

Fig. 3 Result-1

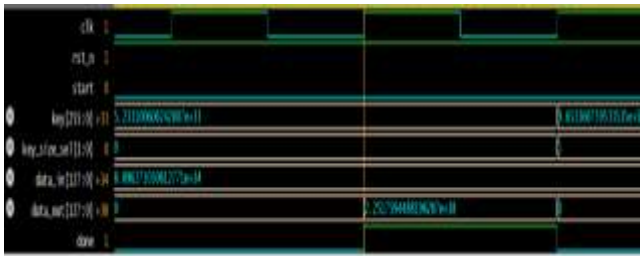

Fig. 4 Result-2

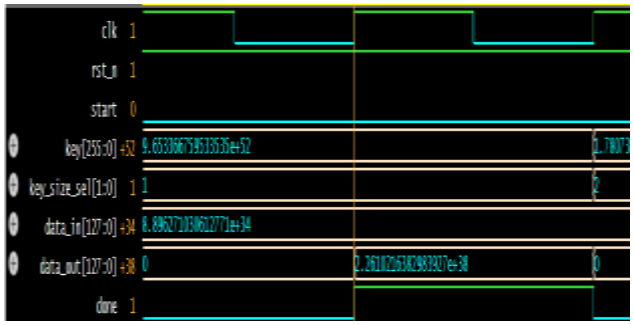

Fig. 5 Result-3

Fig. 6 Result-4

V. CONCLUSION

In conclusion, the design, simulation, and verification of a unified AES encryption core capable of handling 128-, 192-, and 256-bit keys using Verilog HDL. A key selection mechanism (key_size_sel) allows dynamic configuration without modifying the hardware, enabling a single architecture to support all AES modes. Core AES functions—SubBytes, ShiftRows, MixColumns, AddRoundKey, and Key Expansion—were modularly implemented for reusability and scalability. Pipelining and FSM-based control logic were incorporated to enhance throughput and flexibility. Simulation results using EDA Playground confirm correct operation across all key configurations, demonstrating seamless data flow from plaintext input to ciphertext output. The architecture achieves a balanced trade-off between area, performance, and adaptability, making it ideal for secure embedded and VLSI applications.

VI. FUTURE SCOPE

Although the current design is efficient and functionally robust, several enhancements are proposed for future development:

- Power Optimization: Integration of clock gating and low-power techniques to reduce energy consumption.
- Decryption and AES Modes: Extending support to decryption and modes like CBC, OFB, CFB, and CTR.
- Side-Channel Security: Hardening against attacks like DPA and timing analysis using masking or randomized logic.
- SoC Integration: Embedding the core into secure SoCs for WSNs, IoT, and embedded systems.

REFERENCES

JOURNALS:

[1]. C. Elbirt et al., "A comparison of the hardware implementations of the AES algorithms," in IEEE Transactions on Computers, vol. 52, no. 4, pp. 449–472, 2003.
[2]. E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," J. Cryptology, vol. 4, no. 1, pp. 3–72, 1991.
[3]. G. Gaubatz and B. Sunar, "High-speed and low-power AES engines for ubiquitous computing," in IEEE Transactions on Computers, vol. 55, no. 4, pp. 366–372, 2006.
[4]. NIST, "Data Encryption Standard (DES)," FIPS PUB 46, U.S. Dept. of Commerce, Jan. 1977.
[5]. M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in ACM CCS, 1993.
[6]. NIST, "FIPS PUB 197: Advanced Encryption Standard (AES)," U.S. Department of Commerce, 2001.
[7]. P. Rogaway and T. Shrimpton, "Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance," in FSE, 2004.

CONFERENCES:

[8]. A. Hodjat and I. Verbauwhede, "Area-throughput trade-offs for fully pipelined 30 to 70 Gbps AES processors," in Proc. IEEE ISCAS, 2005.
[9]. A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A compact Rijndael hardware architecture with S-box optimization," in ASIACRYPT, 2001, pp. 239–254.
[10]. B. Canright, "A very compact S-box for AES," in CHES, 2005, pp. 441–455.
[11]. B. Canright and L. Batina, "A Very Compact 'Perfectly Masked' S-Box for AES," in ACNS, 2008.
[12]. D. Mukhopadhyay et al., "A novel unified architecture for AES encryption and decryption," in ISCAS, 2005.
[13]. F.-X. Standaert, G. Rouvroy, J.-J. Quisquater, and J.-D. Legat, "Efficient Implementation of Rijndael Encryption in Reconfigurable Hardware: Improvements and Design Tradeoffs," in CHES, 2003.
[14]. H. Kuo and I. Verbauwhede, "Architectural Optimization for a 1.82 Gbps AES Processing," in CHES, 2002, pp. 51–64.
[15]. M. McLoone and J. V. McCanny, "High Performance Single-Chip FPGA Rijndael Algorithm Implementations," in Proc. IEEE FPL, 2001.
[16]. R. Sharma and V. Mathew, "Design of parameterized AES encryption and decryption core using Verilog," in ICCIDS, 2017.

BOOKS:

[17]. B. Schneier, Applied Cryptography, 2nd ed., Wiley, 1996.
[18]. J. Daemen and V. Rijmen, "The Design of Rijndael: AES — The Advanced Encryption Standard," Springer, 2002.
[19]. W. Stallings, Cryptography and Network Security: Principles and Practice, 7th ed., Pearson, 2017.
[20]. W. Trappe and L. Washington, Introduction to Cryptography with Coding Theory, 2nd ed., Pearson, 2005.