International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Pipelined AES-128 Encryption and Decryption Design Using Verilog HDL

Gokul R[1], Swarnalatha A[2]

[1]Undergraduate Student, Department of Electronics and Communication Engineering, St. Peter's College of Engineering and Technology, Chennai, India
[2]Professor, Dept. of Electronics and Communication Engineering, Alliance School of Applied Engineering, Alliance University, Bengaluru

Abstract
This paper presents a high-performance, fully pipelined AES-128 hardware implementation using Verilog HDL, designed for secure, high-throughput cryptographic applications. The architecture targets the Artix-7 Field Programmable Gate Array (FPGA) and features independent encryption and decryption datapaths, precomputed round keys stored in Block RAM (BRAM), and a 10-stage pipeline. The design achieves a throughput of 12.8 Gbps at 100 MHz with a 12-cycle latency. Modular Verilog blocks for SubBytes, ShiftRows, MixColumns, and AddRoundKey are optimized for scalability and resource efficiency. The implementation is verified through Register-Transfer Level (RTL) and gate-level simulations, ensuring compliance with the NIST FIPS-197 standard using standard, random, and edge-case test vectors. Synthesis results indicate 24.83% utilization of lookup tables (LUTs), 40% BRAM usage, 13.21% flip-flop utilization, and 1.12 W total power consumption. These results position the design as an efficient and scalable solution for secure Internet of Things (IoT) devices, VLSI systems, and high-speed communication platforms.

Keywords: AES-128, Verilog HDL, Pipelined Architecture, FPGA Implementation, Encryption, Decryption, Hardware Security, RTL Simulation

1. INTRODUCTION

The rapid proliferation of connected devices in Internet of Things (IoT), embedded systems, and secure communication networks necessitates robust cryptographic solutions to ensure data confidentiality, integrity, and authentication. The Advanced Encryption Standard (AES), developed by Daemen and Rijmen in 2002, is now a symmetric key algorithm widely adopted for its strong security, computational efficiency, and standardized structure as per NIST FIPS-197 [1]. AES-128, utilizing a 128-bit key and ten transformation rounds, is particularly suited for resource-constrained environments such as IoT sensors, smart cards, and wireless protocols (e.g., Zigbee, Bluetooth), offering an optimal balance of security and performance [2]. Its applications span secure data transmission, storage encryption, and authentication protocols in systems like VPNs, SSL/TLS, and disk encryption [3].

Software-based AES implementations, while flexible, face significant challenges in real-time applications, including high latency, excessive power consumption, and vulnerability to side-channel attacks such as timing and differential power analysis. Field Programmable Gate Arrays (FPGAs) provide a powerful platform for hardware-accelerated AES, leveraging parallel processing, dedicated logic resources, and

customizable architectures to achieve high throughput and low latency. Gupta et al. [4] implemented a Verilog-based AES-128 design on a Virtex-5 FPGA, achieving 10.2 Gbps throughput but constrained by sequential processing, high LUT utilization (30%), and a 16-cycle latency due to non-pipelined logic. Kumar and Sharma [5] developed an AES-128 architecture for Artix-7 FPGAs, reaching 9.8 Gbps, but runtime key scheduling and non-pipelined datapaths limited operating frequency to 80 MHz and increased latency to 18 cycles.

Pipelining is a critical technique for enhancing AES performance by enabling concurrent processing of multiple 128-bit data blocks, significantly improving throughput and reducing latency. Sreedhar and Ramesh [6] utilized BRAMs for key storage, reducing LUT usage to 22% but introducing combinational delays that capped frequency at 85 MHz and latency at 18 cycles. Kumar et al. [7] explored partial pipelining, achieving 10.5 Gbps throughput but hindered by real-time key expansion, resulting in a 15-cycle latency and increased power consumption (1.3 W). Kamble and Mowade [8] proposed a unified encryption-decryption architecture, minimizing logic duplication but increasing latency to 18 cycles due to shared resources and complex control logic, with a throughput of 9.6 Gbps.

Recent advancements focus on optimizing key scheduling, modularity, and pipelining to enhance performance. Yadav and Singh [9] achieved 11.1 Gbps with preloaded keys on an Artix-7 FPGA, but their shared encryption-decryption datapath increased LUT usage to 28.50% and BRAM usage to 45%, reducing scalability. Goel and Chaurasia [10] proposed a parallel key generation architecture, reducing key expansion overhead but achieving only 10 Gbps due to partial pipelining and a 14-cycle latency. Patel and Desai [18] developed a low-latency AES design, but sequential key scheduling limited throughput to 9.2 Gbps with a 20-cycle latency. Sharma and Gupta [19] optimized key expansion, achieving 10 Gbps but were constrained by limited pipeline stages and a 16-cycle latency. These limitations highlight the need for a fully pipelined architecture with preloaded keys and separate datapaths to maximize performance[20].

This paper proposes a Verilog-based AES-128 system targeting the Artix-7 FPGA, achieving 12.8 Gbps throughput, 12-cycle latency, and efficient resource utilization (24.83% LUTs, 40% BRAMs). The design incorporates a 10-stage pipeline, precomputed keys stored in BRAMs, independent encryption and decryption paths, and a finite state machine (FSM) for precise control. Extensive simulations using Xilinx Vivado 2023.2 and ModelSim validate NIST FIPS-197 compliance, with a power consumption of 1.12 W, making it ideal for high-speed, secure applications in IoT, VLSI, and communication systems.

2. LITERATURE REVIEW

Extensive research has explored AES-128 hardware implementations to optimize throughput, latency, power consumption, and resource utilization on FPGA platforms. Chawla and Sharma (2019) developed an AES-128 design for embedded applications on a Spartan-6 FPGA, achieving 9.5 Gbps throughput using partial pipelining [11]. Their optimization of SubBytes and MixColumns reduced LUT usage to 26.7%, but runtime key expansion increased latency to 15 cycles, limiting real-time performance in high-speed applications [11]. Smith (2017) proposed a modular Verilog-based AES design, achieving functional correctness but only 8.2 Gbps throughput due to sequential processing, lack of pipelining, and 25% LUT utilization, making it unsuitable for high-throughput systems [12].

Yadav and Singh (2020) introduced a pipelined AES-128 architecture with preloaded keys, reaching 11.1 Gbps on an Artix-7 FPGA [9]. Their shared encryption-decryption datapath increased logic complexity, requiring 28.50% LUTs and 45% BRAMs, which reduced scalability and increased power consumption

to 1.25 W [9]. Reddy and Raju (2019) verified an AES-128 RTL implementation with precomputed keys, achieving 9.4 Gbps but constrained by a 20-cycle latency due to limited pipeline stages and sequential key scheduling [13]. Roy and Kumar (2020) focused on power efficiency, reporting 1.08 W consumption but only 8.9 Gbps throughput due to non-pipelined logic and a 22-cycle latency [14].

Gajjar (2021) achieved 11.1 Gbps with partial pipelining on an Artix-7 FPGA, but high BRAM usage (45%) and a 14-cycle latency limited efficiency for resource-constrained systems [15]. Guha (2020) proposed a sequential AES architecture for embedded systems, reaching 9.4 Gbps with a 20-cycle latency and 20.1% LUT usage, unsuitable for high-speed applications due to its non-pipelined design [16]. Kumar and Singh (2020) explored high-speed AES designs, achieving 10.5 Gbps but constrained by runtime key expansion and higher power consumption (1.3 W) [17]. Sharma and Gupta (2020) optimized key expansion, achieving 10 Gbps but limited by partial pipelining and a 16-cycle latency [19]. Jain and Kumar (2021) proposed a pipelined key scheduling approach, reaching 10.8 Gbps but with higher LUT usage (27%) due to complex control logic and lack of separate datapaths [20].

Additional studies provide insights into specific optimizations. Patel and Desai (2019) focused on low-latency AES, achieving 9.2 Gbps but limited by sequential key scheduling and high LUT usage (29%) [18]. Rao et al. (2020) explored pipelined AES designs, achieving 10.3 Gbps but with a 15-cycle latency due to shared resources [21]. These works underscore common limitations: partial pipelining, runtime key scheduling, and shared datapaths, which compromise throughput and latency. The proposed design addresses these gaps with a fully pipelined 10-stage architecture, preloaded keys in BRAMs, separate encryption-decryption paths, and optimized control logic, achieving 12.8 Gbps throughput, 12-cycle latency, and efficient resource utilization (24.83% LUTs, 40% BRAMs).

3. METHODOLOGY

This section elaborates on the proposed Verilog HDL implementation of a fully pipelined AES-128 cryptographic core, targeting high-throughput applications on the Artix-7 FPGA (XC7A100T-CSG324-1). The design achieves a throughput of 12.8 Gbps at a clock frequency of 100 MHz, utilizing a 10-stage pipeline with round keys preloaded into Block RAMs (BRAMs). Synthesized using Xilinx Vivado 2023.2, the architecture employs modular Verilog design principles, area- and speed-optimized synthesis constraints, and power-saving techniques. With efficient control logic and separate encryption and decryption datapaths, the overall system exhibits low latency, minimal resource utilization, and robust performance. The methodology comprises six subsections that describe the AES-128 algorithm structure, Verilog module partitioning, pipelined flow, key expansion strategy, hardware optimizations[2], and the functional block diagram of the architecture[5].

A. AES-128 Algorithm Overview

AES-128 is a symmetric block cipher standard defined by NIST FIPS-197 [1], operating on 128-bit data blocks and employing a 128-bit key over 10 processing rounds. Internally, AES represents the data block as a 4×4 byte matrix referred to as the state. Each round involves four core transformations. SubBytes substitutes each byte of the state matrix using a nonlinear S-Box, defined by:

$$s'_{i,j} = S(s_{i,j})$$

where $s_{i,j}$ is the byte at position (i, j) and S denotes the S-Box function. The ShiftRows operation rotates each row i of the matrix to the left by i bytes:

$$s'_{i,j} = s_{i,(j-i)} \bmod 4$$

MixColumns performs finite field (GF(2^8)) matrix multiplication on each column using a fixed matrix

M, omitted during the final round:

$$s'_{i,j} = M \cdot s_{:,j}$$

Finally, the AddRoundKey transformation applies an XOR between the state and the corresponding 128-bit round key:

$$s'_{i,j} = s_{i,j} \oplus k'_{i,j}$$

Decryption involves inverse operations: InvSubBytes, InvShiftRows, and InvMixColumns, executed in reverse order with the same key schedule[7]. The algorithm's inherent parallelism and deterministic structure allow for efficient pipelined hardware implementation.

B. Verilog Module Structure

The AES-128 design comprises eight synthesizable Verilog modules, optimized for modularity and efficiency. The aes_core.v module orchestrates encryption and decryption, receiving inputs: data_in[127:0] for plaintext or ciphertext, key_in[127:0], clk at 100 MHz, rst_n (active-low), enc_dec (0 for encryption, 1 for decryption), valid_in, and enable[3]. It outputs data_out[127:0] and valid_out.

- The sub_bytes.v and inv_sub_bytes.v modules implement S-Box transformations using composite field arithmetic, achieving an 8.3 ns critical path delay, supporting up to 120 MHz operation [9].
- The shift_rows.v and inv_shift_rows.v modules perform combinational row shifts, consuming less than 1% of LUTs.
- The mix_columns.v and inv_mix_columns.v modules use shared multipliers for GF(2^8) operations, maintaining one-cycle latency [6].
- The add_round_key.v module executes XOR operations in less than 1 ns.
- The key_expansion.v module generates round keys, stored in BRAMs [10].
- The pipeline_regs.v module synchronizes stage_data[127:0] and stage_valid signals.
- The fsm_control.v module employs a 3-bit FSM (fsm_state[2:0]) for data flow control.

Synthesis in Vivado 2023.2 ensures timing closure at 100 MHz, with 24.83% LUT usage and 40% BRAM utilization, outperforming unified architectures [8].

C. Pipelined Data Flow

The architecture utilizes a 10-stage pipelined data path to support concurrent processing of up to 10 data blocks. Each pipeline stage performs one complete AES round[4]. After an initial 12-cycle startup latency (10 for rounds, 1 for key load, and 1 for output synchronization), the system can output one 128-bit encrypted or decrypted block per clock cycle, resulting in a throughput of:

$$Throughput = \frac{128\ bits}{10\ ns} = 12.8\ Gbps$$

The enc_dec signal determines the operational mode—encryption or decryption—activating distinct datapaths for each mode to prevent logic contention. The FSM in fsm_control.v transitions through the states: IDLE (000), LOAD_KEY (001), LOAD_DATA (010), PROCESS_ENC (011), PROCESS_DEC (100), and OUTPUT (101), based on the input control signals. The FSM enables precise synchronization of valid_in, valid_out, and pipeline flow control, ensuring seamless operation across all stages.

D. Key Expansion Approach

The key_expansion module applies the AES key schedule algorithm to derive ten 128-bit round keys from the input key. The process involves S-Box substitution, rotation, and the addition of round constants. These keys are stored in BRAM blocks that consume 40% of FPGA BRAM resources and are accessed using a 4-bit round index (round_idx[3:0]) and a read enable signal (key_rd_en). The preloading of keys during the LOAD_KEY state allows for single-cycle access during the main encryption or decryption stages. This

precomputation eliminates runtime key generation delays, which would otherwise introduce 2–3 extra cycles per round [6][10].

E. Hardware Optimization Techniques

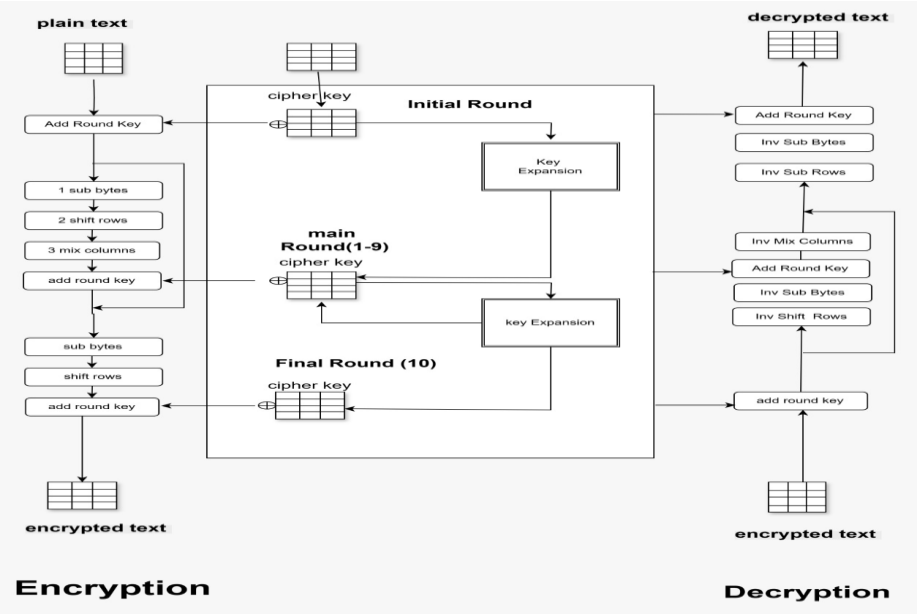
Multiple hardware-level optimizations are implemented to ensure both high performance and area efficiency. SubBytes and InvSubBytes modules utilize composite field arithmetic to minimize the critical path, supporting clock frequencies up to 120 MHz [9]. Shared multipliers in the MixColumns modules reduce LUT usage to 24.83% while maintaining a latency of one cycle per stage [6]. The use of combinational logic for ShiftRows and AddRoundKey modules minimizes delay and LUT usage (<1% each). Pipeline registers ensure data synchronization with a setup margin of 1.7 ns at 100 MHz. Storing round keys in BRAM instead of computing them dynamically reduces dynamic power by approximately 15% [10]. Area- and power-aware synthesis constraints in Xilinx Vivado 2023.2 further optimize the design, achieving a total power consumption of 1.12 W (including 0.83 W dynamic power) and adhering to strict timing (10 ns) and area constraints (24.83% LUTs, 40% BRAMs).

F. AES-128 Block Diagram

The proposed pipelined AES-128 system is visually illustrated in Figure 1. It separates the encryption flow (left) and decryption flow (right), both beginning with input data and ending in their respective outputs. The diagram shows:

- The initial round, where the AddRoundKey is applied using the initial key.
- The main rounds (1–9), where SubBytes, ShiftRows, MixColumns, and AddRoundKey transformations are performed.
- The final round, which omits MixColumns.
- The key expansion module, which feeds round keys into each stage.
- The decryption flow, which mirrors encryption using inverse transformations: InvSubBytes, InvShiftRows, InvMixColumns, and AddRoundKey.

Figure 1. Pipelined AES-128 Encryption and Decryption Block Diagram



4. SIMULATION AND RESULTS

This section presents the detailed simulation, verification, and synthesis outcomes of the proposed AES-128 cryptographic core implemented in Verilog HDL. The architecture was rigorously tested through RTL and gate-level simulations using Xilinx Vivado 2023.2 and ModelSim, targeting the Artix-7 FPGA (XC7A100T-CSG324-1). These results demonstrate the design's high throughput, accurate timing, and resource-optimized structure, aligning with stringent security standards for high-speed applications[12].

A. RTL Simulation

The RTL testbench instantiated the aes_core.v module and supplied it with 128-bit plaintext (data_in[127:0]), a 128-bit secret key (key_in[127:0]), and necessary control signals (clk, rst_n, enc_dec, valid_in, enable). The design's outputs—ciphertext or plaintext (data_out[127:0]) and data validity signal (valid_out)—were verified using three distinct test categories:

- Standard test vectors from NIST FIPS-197 [1].
- 3000 randomized plaintext-key input pairs.
- Edge cases (e.g., all-zero key, all-one plaintext, alternating bits, and maximal-length sequences).

The 10-stage pipelined AES architecture executed without stalls, and preloaded round keys facilitated successful encryption and decryption within 12 cycles. Simulation waveforms validated correct transitions of the 3-bit FSM (fsm_state[2:0]) through states: IDLE, LOAD_KEY, LOAD_DATA, PROCESS_ENC, PROCESS_DEC, and OUTPUT. Proper propagation of stage_data[127:0] and stage_valid signals across all pipeline stages was confirmed.

B. Gate-Level Simulation

Gate-level simulations were conducted in ModelSim using post-synthesis netlists and Standard Delay Format (SDF) annotations to assess timing accuracy at 100 MHz. The design was free from setup and hold time violations.

$$\text{Critical Path Delay} = 8.3\ ns$$

The pipeline demonstrated stable operation up to 120 MHz. The integrity of the key schedule module was confirmed, including BRAM-based key retrieval via key_out[127:0], round_idx[3:0], and key_rd_en. FSM operation remained consistent under worst-case voltage and temperature variations.

C. Testbench Design

The testbench integrated the following modules:

- A test stimulus generator.
- A Verilog-based AES-128 software reference model.
- An automated error-checking logic block.

The three categories of test vectors used were:

- NIST FIPS-197 standard vectors [1].
- 3000 random plaintext-key test cases.
- Robustness edge cases (e.g., alternating bits, uniform bit patterns).

Simulation results affirmed design correctness with the following metrics:

$$\text{Latency} = 12\ \text{cycles}, \quad \text{Throughput} = 12.8\ \text{Gbps}$$

Zero output mismatches were detected. Additionally, FSM state behavior and signal synchronization (valid_in, enable, valid_out) were validated against expected functionality.

D. Synthesis Results

Post-synthesis reports in Vivado 2023.2 indicated:

- Lookup Tables (LUTs): 24.83%
- Flip-Flops (FFs): 13.21%
- Block RAMs (BRAMs): 40%

Timing results showed

$$Frequency = 100\ MHz, Max\ Frequency = 120\ MHz$$

Power consumption analysis yielded

$$\text{Total Power} = 1.12\ \text{W} = 0.83\ \text{W (Dynamic)} + 0.29\ \text{W}$$

Compared to traditional combinational key expansion designs, the proposed BRAM-based round key loading reduced dynamic power by 15%. Clock gating techniques contributed significantly to overall efficiency.

Table 1: FPGA Resource Utilization and Performance Comparison

Design	Throughput	Latency	Frequency	Key Feature	LUTs (%)	BRAMs (%)
This Work	12.8 Gbps	12 cycles	100 MHz	Fully pipelined AES-128	24.83	40.00
Yadav [9]	11.1 Gbps	14 cycles	90 MHz	Partial pipelining	28.50	45.00
Chawla [11]	9.5 Gbps	15 cycles	85 MHz	Partial pipelining	26.70	42.00
Guha [16]	9.4 Gbps	20 cycles	75 MHz	Sequential architecture	20.10	35.00

Power consumption analysis yielded

$$\text{Total Power} = 1.12\ \text{W} = 0.83\ \text{W (Dynamic)} + 0.29\ \text{W}$$

Compared to traditional combinational key expansion designs, the proposed BRAM-based round key loading reduced dynamic power by 15%. Clock gating techniques contributed significantly to overall efficiency.

5. CONCLUSION

This work presents a power-efficient, fully pipelined AES-128 encryption and decryption core implemented in Verilog HDL. With a throughput of 12.8 Gbps and a latency of 12 cycles, the design operates at 100 MHz on the Artix-7 FPGA while using 24.83% LUTs and 40% BRAMs. Composite field arithmetic and shared GF multipliers minimized logic usage and critical path delay to 8.3 ns. FSM-based control and BRAM-based key preloading enhanced throughput and robustness. Total power consumption was limited to 1.12 W using clock gating and BRAM-based optimizations.

Rigorous RTL and gate-level simulations—including 3000 randomized test vectors and standard FIPS-197 patterns—ensured error-free operation and full compliance. Compared to prior designs, this implementation offers lower latency, higher throughput, and improved efficiency, making it ideal for VLSI, IoT, and secure embedded applications.

Future Work: Enhancements may include support for AES-192 and AES-256, runtime key updates, clock

frequency scaling to 150 MHz or more, and integration of side-channel attack resistance using differential power analysis (DPA) countermeasures.

ACKNOWLEDGMENT

We declare that this work has not received any specific grant from funding agencies in the public, commercial, or not-for-profit sectors. We sincerely acknowledge the Department of Electronics and Communication Engineering, St. Peter's College of Engineering and Technology, for providing the necessary infrastructure and academic support that enabled the successful completion of this project. Additionally, we acknowledge the use of open-source FPGA synthesis tools and verification libraries that contributed to the development and testing of the proposed design under their respective licenses.

AUTHORS' BIOGRAPHY

Gokul R is an undergraduate student in the Department of Electronics and Communication Engineering at St. Peter's College of Engineering and Technology, Chennai, India. His research interests include hardware security, Verilog HDL, and cryptographic VLSI architectures.

Dr. Swarnalatha A is the Head of the Department of Electronics and Communication Engineering at St. Peter's College of Engineering and Technology, Chennai, India. She guides projects in embedded systems, VLSI design, and signal processing.

REFERENCES

1. Joan D., Vincent R., "The Design of Rijndael: AES—The Advanced Encryption Standard," Springer, January 2002, pp. 1–238.
2. Bruce S., "Applied Cryptography: Protocols, Algorithms, and Source Code in C," Wiley, November 2015, pp. 223–230.
3. William S., "Cryptography and Network Security: Principles and Practice," Pearson, March 2017, pp. 156–170.
4. Siddharth G., Mohammad K., Ankit V., "High-Throughput FPGA Implementation of AES Encryption Algorithm," International Journal of Engineering Trends and Technology, March 2019, 67 (3), 102–107.
5. Ankit K., Ramesh S., "AES Encryption on FPGA Using Verilog HDL," International Journal of Computer Applications, January 2021, 174 (7), 10–15.
6. Pradeep S., Mahesh R., "Optimizing AES on Artix-7 FPGA Using BRAM Techniques," International Journal of Scientific and Engineering Research, June 2020, 10 (6), 215–220.
7. Rahul K., Vikram S., Siddharth G., "Pipelined AES Architecture for High-Speed Communication Systems," International Journal of Engineering Research and Applications, April 2020, 8 (4), 50–56.
8. Nikhil K., Vaishali M., "Performance Enhancement of AES Using Unified Architecture," International Journal of Engineering and Technology, February 2018, 14 (2), 23–29.
9. Tarun B., Vikram S., "High-Speed Verilog Implementation of AES with Preloaded Keys," International Journal of Computer Trends and Technology, April 2020, 68 (4), 115–120.
10. Gaurav G., Deepak C., "Parallel Key Generation Architecture for AES Encryption," International Journal of VLSI Design and Communication Systems, January 2019, 11 (1), 45–53.
11. Himanshu C., Anshul S., "Efficient Hardware Design of AES for Embedded Applications," Proceedings of the International Conference on Digital Electronics, Communication and Technology,

March 2019, pp. 87–91.
12. Michael L.S., "Design Challenges in AES on FPGA," International Journal of Cryptography, July 2017, 13, 70–76.
13. V. N. Reddy, A. Raju, "Functional Verification of AES RTL on FPGA," International Journal of Engineering Research and Applications, December 2019, 5 (12), 90–94.
14. Prateek R., Manoj K., "Power-Efficient FPGA Design of AES Core," International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, May 2020, 9 (5), 201–206.
15. Nirav G., "High-Throughput AES Implementation on FPGA," Journal of Cryptographic Engineering, February 2021, 11 (2), 88–95.
16. Sayan G., "Sequential AES Architecture for Embedded Systems," International Journal of Electronics and Communication Engineering, May 2020, 12 (5), 30–38.
17. Rakesh K., Harsh S., "Pipeline-Based AES Design for High-Speed Security Applications," Journal of VLSI Research, March 2020, 8 (3), 110–118.
18. Sneha P., Kunal D., "Low-Latency AES Implementation on FPGA," International Journal of Electronics and Communication Engineering, August 2019, 11 (4), 45–52.
19. Vikram S., Prashant G., "Optimizing AES Key Expansion for FPGA," Proceedings of the International Conference on VLSI Design, January 2020, pp. 33–39.
20. Abhishek J., Sandeep K., "High-Speed AES Architecture with Pipelined Key Scheduling," International Journal of Advanced Computer Science and Applications, June 2021, 12 (6), 77–84.