International Journal of Advanced Research and Review

www.ijarr.org
https://doi.org/10.70914/ijarr.2026.v11.i04.pp108-116

Low Power Crypto-chip Design for IoT Applications

K.G. Venkata Krishna[1], Madasu Rama Tulasi[2], Yeswanth[3], Kunaparedy Mounica Sai Durga[4], Danda Bathina Ajay Babu[5]

[1]*Assistant Professor, Department of Electronics and Communication Engineering*

[2,3,4,5]*Department of Electronics and Communication Engineering*

Krishna University College of Engineering and Technology, Machilipatnam-521001, India

ABSTRACT

Cryptography is fundamental to ensuring data security and integrity in modern digital communications. While numerous algorithms have been developed for data encoding and decoding, the Advanced Encryption Standard (AES) emerged as a robust solution for securing large and confidential datasets where existing algorithms proved inadequate. Initially designed to protect highly sensitive information, AES has become widely adopted in networking applications as the standard for data protection. This paper presents a low-power crypto-chip design for IoT applications implementing the AES algorithm using Verilog HDL. AES operates on 16-byte blocks with variable key sizes ranging from 128 to 256 bits. The implementation leverages Verilog's advantages over standard VHDL, offering significantly reduced operation time and propagation delay for encoding and decoding operations. Unlike its predecessor DES (Data Encryption Standard), which suffered from a fixed 56-bit key size limitation, AES provides flexibility through variable key sizes, enhancing security while maintaining computational efficiency. The proposed design achieves 4635 LUTs with a delay of 6.548 ns, demonstrating the feasibility of hardware-accelerated cryptographic operations for resource-constrained IoT devices.

Keywords: *Advanced Encryption Standard (AES), Cryptography, Verilog HDL, IoT Security, Low Power Design, Hardware Implementation, FPGA*

1. INTRODUCTION

The Advanced Encryption Standard (AES) represents a significant milestone in cryptographic security, published by the National Institute of Standards and Technology (NIST) in December 2001. As a symmetric key block cipher, AES processes fixed data blocks of 128 bits through a non-Feistel structure, distinguishing it from its predecessor, the Data Encryption Standard (DES). The algorithm supports three different key lengths (128, 192, and 256 bits) with corresponding encryption/decryption processing rounds of 10, 12, and 14 respectively.

Cryptography, the science of secret writing, encompasses the process of converting ordinary plaintext into unintelligible ciphertext and vice versa. Modern cryptographic techniques are categorized into three primary types: symmetric key cryptography, hash functions, and public key cryptography. Symmetric key algorithms, including AES and DES, utilize identical keys for both encryption and decryption processes, offering advantages in computational speed, implementation simplicity, and reduced processing power requirements compared to asymmetric alternatives.

1.1 Core Components of Cryptography

Encryption: The practice of concealing messages to ensure readability only by intended recipients, protecting data confidentiality during transmission and storage.

Authentication and Integrity: Mechanisms ensuring that users of data and resources are verified as claimed entities, and that messages remain unaltered during transmission, whether through malicious intervention or accidental corruption.

1.2 Requirements of Secure Communication

Modern secure communication systems must satisfy three fundamental requirements:

1. Secrecy: Ensuring that only the intended receiver can comprehend the transmitted message.
2. Authentication: Enabling sender and receiver to confirm each other's identities.
3. Message Integrity: Guaranteeing that communications remain unaltered during transmission.

1.3 Symmetric vs. Asymmetric Cryptography

Symmetric key cryptography employs identical keys for encryption and decryption operations. These algorithms are further classified into block ciphers, which encrypt data in fixed-size blocks

(typically 64 or 128 bits) suitable for single messages, and stream ciphers, which process data bit-by-bit or byte-by-byte for continuous information streams. In contrast, asymmetric cryptography utilizes key pairs consisting of public keys for encryption and private keys for decryption, eliminating the need for secure key transmission while enabling broader distribution of public encryption keys.

2. LITERATURE REVIEW

Recent advances in AES implementation have focused on optimizing hardware complexity and resource utilization. Rao et al. (2017) presented a combinational memory-less architecture implementing both S-Box and inverse S-Box transformations on shared hardware for ByteSub and InvByteSub operations. Their approach utilized composite field arithmetic in finite fields GF(2^8), demonstrating advantages over lookup table (LUT) implementations in terms of hardware complexity. The resource sharing of the multiplicative inverse module between S-Box and inverse S-Box operations resulted in significant reductions in gate count, area, and power consumption. This work highlighted the suitability of AES for diverse applications including banking systems, digital video recorders, web servers, automated teller machines, and cellular communications [1].

Iyer et al. presented compact architectures for AES MixColumn and inverse MixColumn transformations based on byte and bit-level decomposition. Their proposed design demonstrated deeper resource sharing both within bytes and between bytes, achieving a 40% reduction in reconfigurable logic area compared to separate implementations of MixColumn and inverse MixColumn operations, along with a 9% reduction in path delay. The research emphasized the advantages of FPGA-based implementations over ASIC approaches, particularly regarding dynamic reconfiguration capabilities and rapid response to emerging security threats [2].

Zhang and Wang (2010) introduced an outer-round pipelined architecture for FPGA-based AES-128 encryption processors. Their design utilized Block RAM for S-box value storage and exploited dual Block RAM types to reduce critical delay through combined round operations. The hardware-based implementation addressed the limitations of software approaches in meeting gigabit-per-second network transmission requirements, offering improved throughput, reduced key generation time, and enhanced physical security through on-chip packaging of cryptographic algorithms [3].

3. METHODOLOGY

3.1 Existing Method: Data Encryption Standard (DES)

The Data Encryption Standard (DES), developed by IBM in the early 1970s, served as the foundational symmetric-key cryptographic method for decades. Utilizing a 56-bit key length, DES was widely implemented in virtual private networks, email encryption, and electronic payment systems. However, advances in processing power exposed vulnerabilities in DES security. Research conducted in the late 1990s demonstrated that specialized hardware could break DES encryption within hours, necessitating the development of Triple DES, which employs multiple encryption cycles with different keys, achieving an effective key length of 168 bits through keying option 2.

Despite the availability of Triple DES and more advanced encryption methods, DES remains in use in certain legacy systems. However, NIST has discouraged DES usage in government applications since 2005 due to its relatively weak security compared to modern alternatives such as AES and Elliptic Curve Cryptography (ECC). The primary limitations of DES include its short key length, susceptibility to brute-force attacks, 64-bit block size, and utilization of a Feistel network structure that processes input halves independently.

Parameter	DES	AES
Key Size	56 bits	128, 192, 256 bits
Block Size	64 bits	128 bits
Structure	Feistel Network	Substitution-Permutation
Security Level	Vulnerable to brute-force	Highly secure

Table 1: Comparison of DES and AES Characteristics

3.2 Proposed Method: AES Implementation

The proposed AES implementation utilizes a 128-bit architecture where input blocks, output blocks, and state arrays maintain consistent 128-bit lengths, represented as four 32-bit words (Nb = 4). The number of 32-bit words in the cipher key is denoted by Nk, accepting values of 4, 6, or 8 for key sizes of 128, 192, and 256 bits respectively. The number of encryption rounds (Nr) is determined by the key size: Nr = 10 for Nk = 4, Nr = 12 for Nk = 6, and Nr = 14 for Nk = 8.

Key Size (bits)	Nk (32-bit words)	Nr (Rounds)
128	4	10
192	6	12
256	8	14

Table 2: AES Key Size and Round Configuration

3.2.1 AES Round Structure

Each AES round comprises four fundamental transformations: SubBytes, ShiftRows, MixColumns, and AddRoundKey. For the 128-bit key implementation (Nr = 10), each round utilizes a unique round key generated through the key expansion module from the original cipher key. The input plaintext undergoes conversion to hexadecimal format and arrangement into a 4×4 state matrix, upon which all transformations operate sequentially. Following completion of all rounds, the final state matrix is converted back to text format to produce the ciphertext.

3.2.2 SubBytes Transformation

The SubBytes transformation applies non-linear byte substitution independently to each byte in the state array. This operation utilizes a predefined substitution table (S-Box) containing 256 hexadecimal values. For each byte in the state matrix, the first four bits determine the row index and the last four bits determine the column index in the S-Box lookup table. The original byte value is then replaced with the corresponding S-Box entry. Lookup table implementation reduces hardware complexity and computational time compared to real-time calculation approaches.

3.2.3 ShiftRows Transformation

The ShiftRows transformation performs cyclical left rotation of bytes within each row of the state matrix. The number of shift positions depends on the row index: row 0 remains unchanged, row 1 undergoes one-byte cyclic shift, row 2 shifts two bytes, and row 3 shifts three bytes. This transformation provides diffusion by distributing byte positions across columns, ensuring that each column in subsequent rounds receives data from different original columns.

3.2.4 MixColumns Transformation

The MixColumns transformation treats each column of the state matrix as a polynomial over GF(2^8) and multiplies it with a fixed polynomial a(x). This operation resembles conventional matrix multiplication, except that addition operations are replaced with XOR operations. Each byte of the resulting column is computed by multiplying corresponding bytes from the input column

with predefined matrix coefficients and combining results through XOR operations. This transformation provides diffusion at the column level, ensuring that changes in individual bytes propagate throughout the entire column.

3.2.5 AddRoundKey Transformation

The AddRoundKey transformation combines the state matrix with the round key through bitwise XOR operations. Since both the state matrix and round key are 128 bits in size, the XOR operation is performed on corresponding bytes in each column. Round 0 utilizes the original cipher key, while subsequent rounds employ unique round keys generated through the key expansion algorithm. This transformation incorporates the secret key material into the encryption process, providing the fundamental security mechanism.

3.2.6 Key Expansion

The key expansion algorithm generates (Nr + 1) round keys of 128 bits each from the original cipher key. Implementation in Verilog HDL employs combinational and sequential logic circuits. The 128-bit initial key is stored in a register and undergoes a series of transformations to produce the required round keys. Each round key is derived through operations including word rotation, S-Box substitution, and XOR with round constants. The generated round keys maintain cryptographic strength while ensuring that compromise of any single round key does not reveal the original cipher key or other round keys.

4. RESULTS AND DISCUSSION

The proposed AES implementation was synthesized and implemented using Xilinx Vivado Design Suite targeting Virtex-7 FPGA architecture. The design was verified through comprehensive simulation testing covering various plaintext-key combinations to ensure correct encryption and decryption functionality. RTL synthesis results demonstrate the feasibility of the proposed architecture for resource-constrained IoT applications.

4.1 Hardware Resource Utilization

The implementation achieves efficient hardware resource utilization, requiring 4,635 lookup tables (LUTs) for the complete AES encryption and decryption datapath. This moderate resource footprint makes the design suitable for integration into resource-constrained IoT devices where area and power consumption are critical considerations. The LUT count represents an optimized implementation that balances hardware complexity with cryptographic functionality.

4.2 Timing Performance

The critical path delay measures 6.548 nanoseconds, enabling a maximum operating frequency of approximately 152.7 MHz. This timing performance allows the design to achieve throughput suitable for IoT security applications where real-time encryption and decryption are required. The propagation delay is significantly lower than VHDL-based implementations, demonstrating the efficiency advantages of Verilog HDL for cryptographic hardware design.

Performance Metric	Measured Value
Area (LUTs)	4,635
Critical Path Delay	6.548 ns
Maximum Frequency	~152.7 MHz

Table 3: Implementation Performance Metrics

4.3 Advantages and Applications

The proposed implementation offers several advantages for IoT security applications:

4. Low Complexity: Optimized hardware architecture minimizes resource utilization while maintaining cryptographic security.
5. High Security: AES algorithm provides robust protection against cryptanalytic attacks through multiple rounds of transformations.
6. Reduced Power Consumption: Fewer clock cycles and optimized datapath design minimize dynamic power consumption.
7. Hardware Acceleration: FPGA implementation provides superior performance compared to software-based cryptography on embedded processors.

Primary application domains include wireless security for IoT sensor networks, processor security for embedded systems, and file encryption for edge computing devices. The design's efficiency makes it particularly suitable for battery-powered IoT devices where energy consumption directly impacts operational lifetime.

5. CONCLUSION

This paper presented a low-power crypto-chip design implementing the AES encryption algorithm using Verilog HDL for IoT applications. The proposed implementation achieves efficient

hardware resource utilization with 4,635 LUTs and a critical path delay of 6.548 ns, demonstrating feasibility for resource-constrained embedded systems. Verilog-based implementation offers significant advantages over VHDL in terms of reduced operation time and propagation delay, directly translating to lower power consumption through decreased clock cycle requirements.

The superior security characteristics of AES compared to DES stem from increased round operations and variable key sizes ranging from 128 to 256 bits, providing robust protection against brute-force attacks. Hardware implementation on FPGA platforms offers substantial benefits for high-speed real-time applications, including enhanced physical security through on-chip algorithm implementation and greater flexibility for adaptation to evolving security requirements.

Future work may explore SystemVerilog implementations to leverage advanced verification features and investigate power optimization techniques such as clock gating and voltage scaling. Additionally, integration with lightweight authentication protocols and evaluation of side-channel attack resistance would enhance the design's suitability for security-critical IoT deployments. The demonstrated performance characteristics validate the viability of FPGA-based AES implementations as practical solutions for securing Internet of Things ecosystems.

REFERENCES

[1] M. Rajeswara Rao and R. K. Sharma, "FPGA Implementation of combined S box and Inv S box of AES," in *2017 4th International Conference on Signal Processing and Integrated Networks (SPIN)*, Noida, India, 2017, pp. 483-487, doi: 10.1109/SPIN.2017.8049996.

[2] N. C. Iyer, Deepa, P. V. Anandmohan, and D. V. Poornaiah, "Mix/InvMixColumn decomposition and resource sharing in AES," in *2016 International Conference on Emerging Trends in Engineering, Technology and Science (ICETETS)*, Pudukkottai, India, 2016, pp. 1-6, doi: 10.1109/ICETETS.2016.7603066.

[3] Y. Zhang and X. Wang, "Pipelined implementation of AES encryption based on FPGA," in *2010 International Conference on Information Theory and Information Security*, Beijing, China, 2010, pp. 514-517, doi: 10.1109/ICITIS.2010.5689467.

[4] J. Daemen and V. Rijmen, "The block cipher Rijndael," in *Smart Card Research and Applications, LNCS 1820*, Springer-Verlag, 2000, pp. 288-296.

[5] S. M. Soliman, B. Magdy, and M. A. AbdElGhany, "Efficient implementation of the AES algorithm for security applications," in *2016 IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, Monte Carlo, Monaco, 2016, pp. 400-403.

[6] A. Kumar and R. Gupta, "Design and implementation of AES algorithm in Verilog," *International Journal of Engineering Research and Technology*, vol. 5, no. 4, pp. 217-220, 2016.

[7] Z. Kouser, M. Singhal, and A. M. Joshi, "FPGA implementation of Advanced encryption standard algorithm," in *2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, Jaipur, India, 2016, pp. 1-5.

[8] K. Jamal, P. Srihari, K. M. Chari, and B. Sabitha, "Low power test pattern generation using test-per-scan technique for BIST implementation," *ARPN Journal of Engineering and Applied Sciences*, vol. 13, no. 6, pp. 2096-2101, 2018.

[9] M. Mohurle and V. V. Panchbhai, "Review on realization of AES encryption and decryption with power and area optimization," in *2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)*, Delhi, India, 2016, pp. 1-6.

[10] Y. A. Nasser, M. A. Bazzoun, and S. Abdul Nabi, "AES algorithm implementation for a simple low cost portable 8-bit microcontroller," in *2016 3rd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA)*, Beirut, Lebanon, 2016, pp. 142-147.

[11] J. Orlin Grabbe, "The DES algorithm illustrated," *Laissez Faire City Times*, vol. 2, no. 28, 1998.

[12] K. Jamal, K. M. Chari, and P. Srihari, "Test pattern generation using thermometer code counter in TPC technique for BIST implementation," *Microprocessors and Microsystems*, vol. 71, article 102890, 2019.

[13] K. Jamal, P. Srihari, and G. Kanakasri, "Test Vector Generation using Genetic Algorithm for Fault Tolerant Systems," *International Journal of Control Theory and Applications (IJCTA)*, vol. 9, no. 12, pp. 5591-5598, 2016.