Penerbit Akademia Baru

Journal of Advanced Research Design

Journal homepage:
https://akademiabaru.com/submit/index.php/ard
ISSN: 2289-7984

Design of Advanced Encryption System (AES) Algorithm Using ASIC Implementation for Internet of Things (IoT) Devices

Nabihah Ahmad[1,2,*], Muhammad Hafiz Zamri[1], Salman Ahmed[1,3], Ardhi Wijayanto[1,4], Suhaila Isaak[5]

[1] Faculty of Electrical and Electronics Engineering, Universiti Tun Hussein Onn Malysia, Parit Raja, 86400, Johor Malaysia
[2] VLSI and Embedded Technology (VEST) Focus group, Universiti Tun Hussein Onn Malaysia, Parit Raja, 86400, Johor Malaysia
[3] Department of Electrical Engineering, Sukkur IBA University, Sukkur, 65200, Sindh, Pakistan
[4] Universitas Sebelas Maret, Indonesia
[5] Faculty of Electrical Engineering Universiti Teknologi Malaysia, 81310 Skudai, Johor Malaysia

ARTICLE INFO

Article history:
Received 14 April 2024
Received in revised form 18 November 2024
Accepted 16 December 2024
Available online 31 December 2024

Keywords:
AES-128; IoT; S-Box; CMOS technology

ABSTRACT

This paper presents an AES-128 configuration designed specifically for IoT devices, utilizing 90nm CMOS technology. The architecture, developed in Verilog HDL and executed through Synopsys tools, incorporates a modified S-Box aimed at enhancing performance, area efficiency, and throughput. The motivation stems from the increasing security requirements of IoT applications, highlighting the need for robust data protection for resource-constrained devices. Versatile for IoT applications, this design handles standard input data block sizes of 128 bits. It stands out as a purpose-built solution for securing digital communications in the IoT, overcoming unique challenges such as limited resources and fluctuating communication environments. The modified S-Box The altered S-Box bolsters security, aids in space optimization, and enhances efficiency compared to prevailing solutions. By using techniques to carefully modify replacement boxes, these designs deliver optimized performance from both a safety and space utilization perspective. Extensive validation work, including Synopsys tool testing and simulation, ensures the reliability of the proposed AES-128 design. It achieves a throughput of 14.54Mbps at a clock frequency of 100MHz while maintaining a compact footprint of 0.4324mm^2 to meet the constraints of IoT devices. The practical implications of this design lie in the balance between performance and resource utilization, making it suitable for real-world IoT implementations. The utilization of a 128-bit key length augments security, rendering the proposed AES-128 an ideal choice for safeguarding data across diverse IoT applications.

1. Introduction

The National Institute of Standards and Technology (NIST) published the Advanced Encryption Standard (AES) proposal call in 1997. The Rijndael algorithm, which adjusts the number of rounds required for each key size, was chosen as the AES algorithm.

* *Corresponding author.*
E-mail address: nabihah@uthm.edu.my

https://doi.org/10.37934/ard.123.1.131145

It was created by scientists John Daemen and Vincent Rijmen to replace the encryption standards of the time, 3DES (Triple Data Encryption Standard), and IDES (International Data Encryption Algorithm) [1]. Phil Zimmermann states that cryptography is the mathematical encrypting and decrypting of data science. Cryptography is related to encryption, the change of information and data into a structure that is unusable by a non-authorized individual to access the data. Today, cryptography is more advanced than ever before. There are three types of cryptography; symmetric, asymmetric, and HASH function [1]. Symmetric cryptography, known as secret-key cryptography or conventional cryptography, is an encryption system in which the sender and receiver of a message share a single, common key to encrypt and decrypt the message. Symmetric cryptography is fast and efficient but requires the key to be shared securely between the sender and receiver [2].

Examples of symmetric-key cryptography is Advanced Encryption System (AES), Data Encryption Standard (DES) and Fast Data Encipherment Algorithm (FEAL). Second is asymmetric cryptography, also known as public-key cryptography. It refers to cryptography that requires two separate keys, which is private and public keys. The public key is used to encrypt the message, and the private key is used to decrypt the message. It is more secure than symmetric cryptographic; however, it is slower and more computationally expensive [1]. An example of asymmetric cryptography is the RSA algorithm (Rivest-Shamir-Adleman). Last, the HASH function is another type of cryptography used for data integrity. It takes a message and produces fixed-size output, and its application is Secure Hash Algorithm (SHA) [3].

Advanced Encryption System (AES) is a widely used symmetric-key encryption for Internet of Things (IoT) applications due to its simplicity and low energy consumption [1]. IoT is mainly focused on low-power implementation of security algorithms because most are related to autonomous applications such as ultrathin sensory systems. These applications' power budgets are minimal to extend battery life [1]. For secure data transfer, the current IoT protocols IEEE 802.15.4, Low Power Wide Area Network (Lora WAN), and Sigfox use the AES algorithm. In the context of the IoT, machine-to-machine (M2M), smart city, and industrial applications, Lora WAN is a Low Power Wide Area Network (LPWAN) protocol that enables affordable, mobile, and secure bi-directional communication [4]. Lora WAN stands out because it is one of the few IoT technologies that supports end-to-end encryption. The Lora WAN security primitives should meet the following requirements: high scalability, low power consumption, low cost, and low implementation complexity.

In addition, the protocol provides mutual authentication, integrity protection, and confidentiality as security services. The AES-128 algorithm, employed to provide these cryptographic services, is run in The Counter (CTR) mode for encryption and Cipher-based message authentication (CMAC) mode for integrity protection. The protocol's key characteristics include a data rate between 0.3 to 50 kbps, a maximum operating frequency of Ultrasonic Machining (USM) bands of 902 to 928 MHz and a bandwidth of around 125 kHz [5]. AES is more powerful, safe, and secure than DES because it uses a 128, 192, or 256-bit key block cipher algorithm to convert a block of 128-bit message into ciphertext. The number of rounds for encryption will depend on the key length. As an example, the 128-bit key AES employs ten rounds of encryption. Plaintext is transformed into ciphertext throughout encryption and decryption operations and vice versa during the latter. Four distinct processes had to be taken for encryption and decryption [6]. Protection against unauthorized persons is the primary goal of encryption and decryption. Encryption protects information stored on a server or transmitted over the internet. Authorized users could access the data through decryption, which required the right key to reverse the encryption and retrieve the original data [6].

Application-Specific Integrated Circuit (ASIC) and Field Programmable Gate Arrays (FPGA) are two possible hardware platforms for the implementation of AES [7]. Numerous universal reconfigurable logic blocks coupled by reconfigurable interconnects, and switches make up an FPGA. Additionally,

modern FPGA contains embedded higher-level components. The performance characteristics for both ASIC and FPGA is almost identical and can make full use of parallel processing, pipelining, and operating on arbitrary size words. The main performance attribute that sets FPGA apart from ASIC is its slower speed, which is brought on by delays in the circuitry needed for reconfiguration. Both AES and FPGA are very similar in their hardware implementation approaches. The main distinction between the two is that FPGA does not require a physical layout, and the design cycle is quicker and less expensive as compared to using ASIC [8].

The research article contributes in designing a modified AES-128 S-box architecture for resource constraints IoT devices aimed at enhancing performance, area, throughput and efficiency. The implementation in 90nm CMOS technology. The architecture is developed and executed through Synopsys EDA tools. The safety of data is ensured using 128-bit key in addition to space utilization perspective. The architecture maintains a foot print of 0.4324mm2 at 100MHz clock frequency and 14.54 Mbps throughput.

2. Methodology

The project was conducted in three phases: Phase 1 involved research on the proposed project, defining suitable specifications for AES implementation in IoT, and developing a Verilog code for the AES architecture design. Phase 2 included the design simulation using Synopsys Tools, synthesis of the Verilog code, and static timing analysis. Phase 3 covered the layout design and design verification of the AES implementation. The process is illustrated in Figure 1.

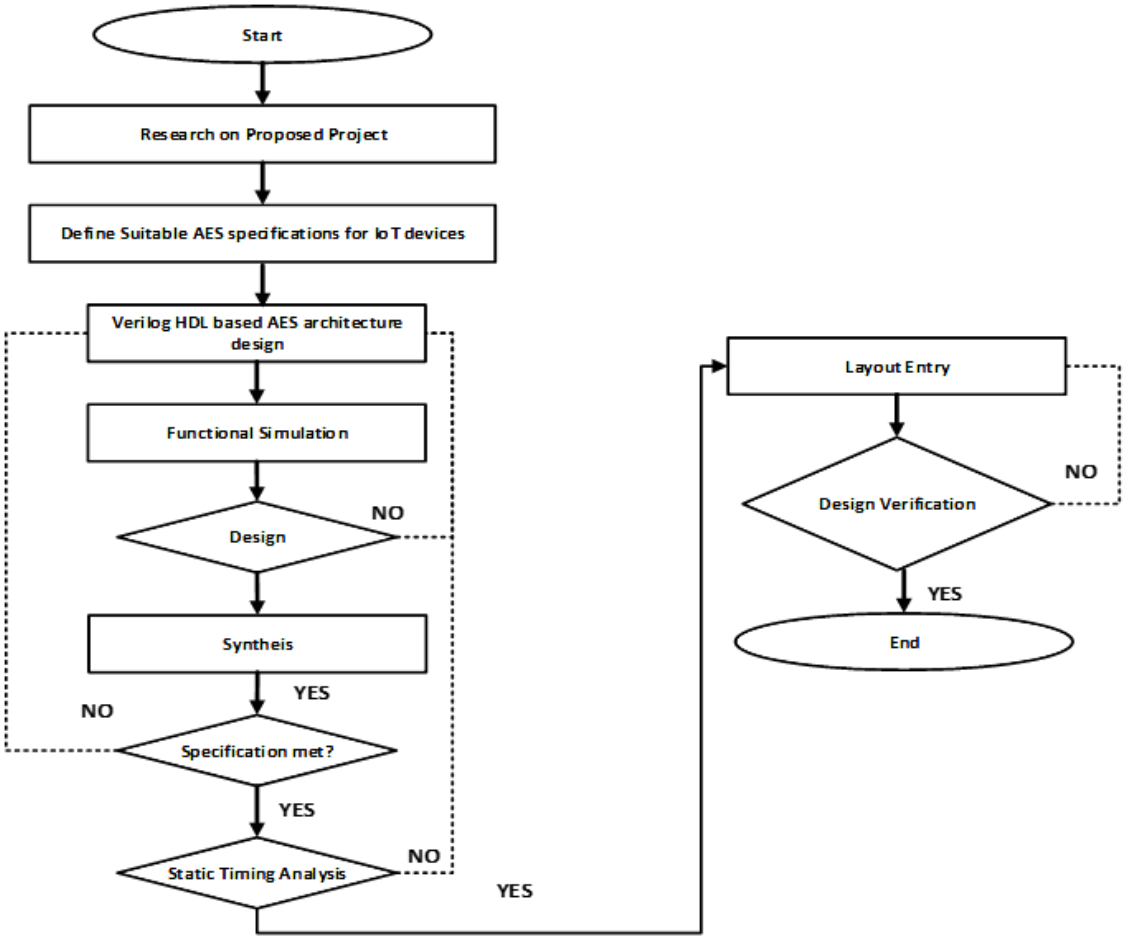

Fig. 1. Process of project

2.1 Architecture of S-Box

The Verilog Code for the S-Box design in / is crafted by transitioning from a full custom schematic entry approach, wherein the original work employed meticulous manual placement and routing, to a comprehensive implementation in Verilog HDL for this project. Using a circuit reduction technique, the S-box design in Figure 2, which consists of Stage 1, inversion, and combination of multiplication in GF(2^4), merges the sub-component of the standard multiplicative inverse. Figure 2 illustrates this design. The hardware complexity of the circuit is optimized and reduced. Stage 1 comprises a single circuit addition, squaring in GF(2^4), multiplication in GF(2^4), and multiplication with constant. After multiplicative inversion, CombineXAXB is reduced for multiplication in GF(2^4). When compared to a normal circuit using a typical composite field architecture, this novel architecture has fewer gates. In this architecture it can perform both encryption and decryption. AND gates, multiplexer and XOR gates are implement in this architecture to achieve a compact AES implementation. The transformation of the composite field can be represented as $b \xrightarrow{ISO} q \xrightarrow{MINV} q' \xrightarrow{INVISO} b' \xrightarrow{AFFINE} b''$ where, b are byte elements form state matrix, q are multiplicate inverse from isomorphic state, b' elements after inverse isomorphic mapping and b'' are element after affine transformation. As for the InvSubBytes is vice versa transformation can be represented as shown $b'' \xrightarrow{ISO} q'' \xrightarrow{MINV} q' \xrightarrow{INVISO} q \xrightarrow{AFFINE} b$.

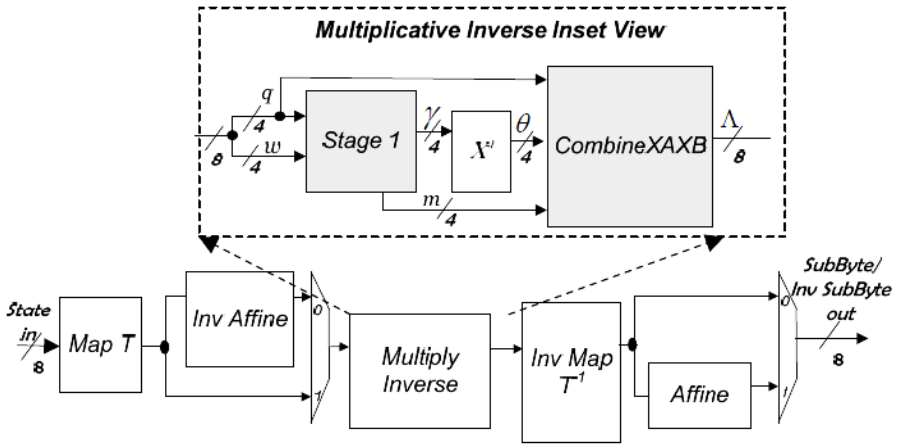

Fig. 2. Proposed multiplicative inverse in GF(2^8) architecture

The Affine transformation, AT operates on multiplicative inverse of $\mathrm{GF}(2^8)$ bytes on, b of the State matrix represented by b', while the inverse-affine transformation AT^{-1} operates on isomorphic affine transformed $\mathrm{GF}(2^8)$ multiplicate inverse of same bytes, b represented by q''. The resultant matrix operation of AT and AT^{-1} is shown from Eq. (1) and (2) and can be translated into logical implementations using 12 XOR gates for each transformation.

$$[AT(b')] = \begin{bmatrix} b''0 \\ b''1 \\ b''2 \\ b''3 \\ b''4 \\ b''5 \\ b''6 \\ b''7 \end{bmatrix} = \begin{bmatrix} \overline{b'0 \oplus b'4 \oplus b'5 \oplus b'6 \oplus b'7} \\ \overline{b'0 \oplus b'1 \oplus b'5 \oplus b'6 \oplus b'7} \\ b'0 \oplus b'1 \oplus b'2 \oplus b'6 \oplus b'7 \\ b'0 \oplus b'1 \oplus b'2 \oplus b'3 \oplus b'7 \\ b'0 \oplus b'1 \oplus b'2 \oplus b'3 \oplus b'4 \\ \overline{b'1 \oplus b'2 \oplus b'3 \oplus b'4 \oplus b'5} \\ \overline{b'2 \oplus b'3 \oplus b'4 \oplus b'5 \oplus b'6} \\ b'3 \oplus b'4 \oplus b'5 \oplus b'6 \oplus b'7 \end{bmatrix} \quad (1)$$

$$[AT^{-1}(q'')] = \begin{bmatrix} q'0 \\ q'1 \\ q'2 \\ q'3 \\ q'4 \\ q'5 \\ q'6 \\ q'7 \end{bmatrix} = \begin{bmatrix} \overline{q''2 \oplus q''5 \oplus q''7} \\ q''0 \oplus q''3 \oplus q''6 \\ \overline{q''1 \oplus q''4 \oplus q''7} \\ q''0 \oplus q''2 \oplus q''5 \\ q''1 \oplus q''3 \oplus q''6 \\ q''2 \oplus q''4 \oplus q''7 \\ \overline{q''0 \oplus q''3 \oplus q''5} \\ q''1 \oplus q''4 \oplus q''6 \end{bmatrix} \quad (2)$$

Next, a new SubBytes and InvSubBytes proposed merging the sub-components in order to reduce the hardware complexity. In this architecture, it consists of Stage 1 and CombineXAXB circuit. Stage 1 is the inversion and combination of multiplication in $\mathrm{GF}(2^4)$ and the CombineXAXB represents two multiplications in $\mathrm{GF}(2^4)$ after the Stage 1. Stage 1 is an optimizes block logic for transformation of multiplication in $\mathrm{GF}(2^4)$, multiplication with constant lambda, squaring in $\mathrm{GF}(2^4)$, and module-2addition merged in one circuit. Input for Stage 1 is $w = \{w_3\, w_2 w_1, w_0\}_2$ and $q = \{q_3 q_2 q_1 q_0\}_2$. The output is $y = \{y_3 y_2 y_1 y_0\}_2$ and implemented by $y = \{\rightarrow \text{multiplication in } \mathrm{GF}(2^4) \rightarrow \text{multiplication with } \lambda \rightarrow \text{squaring in } \mathrm{GF}(2^4) \rightarrow \text{modulo} - 2 \text{ Addition}$. Low complexity logic expression obtained for output; y of Stage 1 is in Eq. (3).

$$\begin{aligned} y_3 &= q_3 q_0 m_0 w_3 m_1 \partial m_2 \chi m_3 Q \\ y_2 &= k q_1 m_0 w_2 m_1 w_3 m_2 v m_3 \chi \\ y_1 &= k m_0 w_1 m_1 x m_2 \partial m_3 w_2 \\ y_0 &= q_2 m_0 w_0 m_1 w_1 m_2 w_3 m_3 \partial \end{aligned} \quad (3)$$

Where, $\chi = w_3 w_1, v = w_2 w_0, \partial = w_3 w_2, x = w_0 w_1, Q = \chi v$ and $k = q_3 q_2, m_3 = q_3 w_3, m_2 = q_2 w_2, m_1 = q_1 w_1$ and $m_0 = q_0 w_0$. For the X module between Stage 1 and CombineXAXB, the input of the module is $y = \{y_3\, y_2\, y_1\, y_0\}_2$ and the output is $\theta = \{\theta_3\, \theta_2\, \theta_1\, \theta_0\}_2$. The formulation result is shown in Eq. (4).

$$\begin{aligned} \theta_3 &= y_3 \bar{y}_0 + y_2(\overline{y_3 y_1}) \\ \theta_2 &= y_3(y_0 \cup y_2) + y_2 \overline{y_1} \\ \theta_1 &= y_1 \overline{y_3 y_2} + y_3 \overline{y_1 y_0} + y_2 \overline{y_0} \\ \theta_0 &= (y_0 + y_1)\overline{y_3 y_2} + y_2(y_0 \cup \overline{y_1}) \end{aligned} \quad (4)$$

Where ∪ and + are OR gate and XOR gate implementation.

The architecture of CombineXAXB is the merging of two multiplications in GF(2^4). This architecture helps to achieve low gate count for this architecture. The output is $A = \{A_7\, A_6\, A_5\, A_4\, A_3\, A_2\, A_1\, A_0\}_2$ and the input is $\theta = \{\theta_3\, \theta_2\, \theta_1\, \theta_0\}_2$, $m = \{m_3\, m_2\, m_1\, m_0\}_2$ and $q = \{q_3\, q_2\, q_1\, q_0\}_2$. The output equation for A is as follows in Eq. (5) to (7).

$$\begin{aligned} A_7 &= q_0\theta_3 + q_1(\theta_3 + \theta_2) + q_2(\theta_3 + \theta_2 + \theta_1 + \theta_0) \\ A_6 &= q_0\theta_2 + q_1\theta_3 + q_2(\theta_2 + \theta_0) + q_3(\theta_3 + \theta_1) \\ A_5 &= q_0\theta_1 + q_1(\theta_1 + \theta_0) + q_2(\theta_3 + \theta_2) + q_3\theta_2 \\ A_4 &= q_0\theta_0 + q_1\theta_1 + q_2\theta_3 + q_3(\theta_3 + \theta_2) \\ A_3 &= m_0\theta_3 + m_1(\theta_3 + \theta_2) + m_2(\theta_3 + \theta_1) + m_3(\theta_3 + \theta_2 + \theta_1 + \theta_0) \\ A_2 &= m_0\theta_2 + m_1\theta_3 + m_2(\theta_2 + \theta_0) + m_3(\theta_3 + \theta_1) \end{aligned}$$

$$A_1 = m_0\theta_1 + m_1(\theta_1 + \theta_0) + m_2(\theta_3 + \theta_2) + m_3\theta_2$$
$$A_0 = m_0\theta_0 + m_1\theta_1 + m_2\theta_3 + m_3(\theta_3 + \theta_2) \quad (5)$$

2.2 Architecture of AES

AES block diagrams are frequently utilized in symmetric encryption algorithms that work with data blocks of a predetermined size. Many essential components, including ShiftRows, MixColumns, AddRound keys, rounds with SubBytes, key expansion, and AES output, are usually included in the block diagram for AES implementation [9]. AES offers several key lengths (128, 192, or 256 bits). In order to be used in further encryption and decryption rounds, the key expands the original key into a collection of round keys [10]. Depending on the key size, the AES consists of 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. Figure 3 displays the architecture of AES.

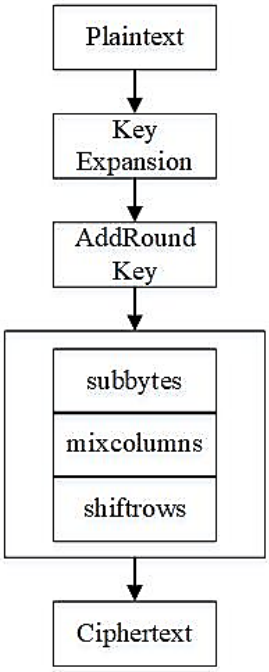

Fig. 3. Architecture of AES

2.3 Designs Constraints

For the proposed AES design with 128-bit key length and 128-bit data path, certain design constraints are paramount for successful implementation. Using 90nm CMOS technology and Synopsys tools adds complexity and requires careful consideration of resource usage and performance. As described in the Verilog code, the use of a modified S-Box introduces important design constraints aimed at achieving area requirements of less than 1 mm^2. This limitation highlights the need to maintain the robustness of the modified S-Box and optimize the implementation for space efficiency while preserving the security properties of the original AES algorithm. At the same time, the 10 Mbps minimum throughput requirement sets a performance threshold, forcing developers to strike a delicate balance between security and operational speed. Meeting these limitations requires a careful approach that includes thorough analysis, synthesis, and simulation using Synopsys tools to ensure that the proposed AES design strikes the desired balance between security, space utilization, and throughput.

2.4 RTL Description

In the register transfer level (RTL) description of the proposed AES design, Verilog code defines the complex details of the system in detail. This architecture follows the Advanced Encryption Standard (AES) with a 128-bit key length and 128-bit data path. The design flow performed by Synopsys tools using 90nm CMOS technology is subject to certain design constraints. The key expansion module is carefully designed to generate the round keys required for subsequent rounds of encryption and decryption. Within a round, the algorithm implements the operations SubBytes, ShiftRows, and MixColumns according to the number of rounds specified by the 128-bit key length. The modified S-Box, which is structured in Verilog code and introduces innovative elements to achieve area sizes of less than 1 mm. This design choice focuses on space optimization without compromising the inherent security of the AES algorithm. Additionally, the system is designed to meet a minimum throughput requirement of 10 Mbps, highlighting the need for a harmonious balance between robust security and operational speed. The RTL description, synthesized and simulated using Synopsys tools, summarizes the complex interplay of cryptographic operations, technical considerations, and innovative changes within AES.

3. Results

This section provides an in-depth analysis of the proposed S-box architecture and its integration into the AES-128 encryption scheme. We begin by examining simulation results of the S-box and its sub-models, followed by an evaluation of the performance metrics specific to the proposed S-box. Subsequently, we discuss the impact of this S-box within the AES-128 structure and present design synthesis results to demonstrate feasibility. Finally, the section concludes with a comprehensive performance assessment of the full AES encryption scheme using the proposed S-box.

3.1 Simulation Results of S-Box Architecture

In the Verilog code provided to implement the AES, the top module serves as the top-level wrapper. It takes a clock signal ‘clk’ as input and produces a 128-bit ‘finalout’ output signal. Inside this module, an instance of the encryption module, named ‘u1’, is instantiated. Initialization includes specific constant values for the clock, 128-bit input data ‘datain’, and a 128-bit encryption key ‘key’. The output of the encryption module is connected to an internal wire ‘tempout’, and the lower 128 bits of this wire are assigned to the final output. A test bench is provided to verify the functionality of the AES top module by simulation. The test bench initiates the clock signal ‘clk’, monitors the final output signal, and applies excitation to the system. The RTL design for proposed S-Box is shown in Figure 4. It includes clock initialization, global reset, stimulus application, and a monitoring block to observe behavior during the simulation. The simulation setup uses a time scale of 1 nanosecond per unit of simulation time and 1 picosecond per unit of temporal precision. Although Verilog code focuses on a simulation and functional verification environment, it does not explicitly specify parameters such as voltage levels or frequencies, which are important considerations when implementing real hardware implementation.

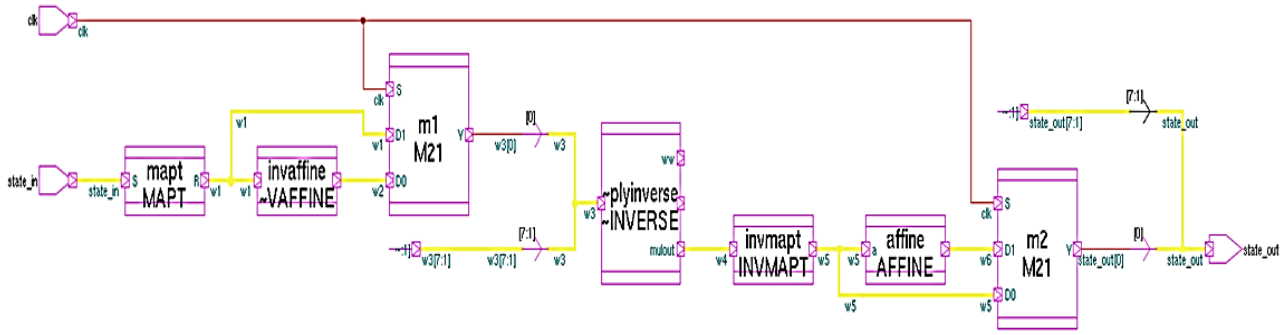

Fig. 4. RTL design for proposed S-Box

3.1.1 Isomorphic mapping

These submodules are constructed with relation of input and output to create the proposed S-Box using their logical implementation for each submodule. Figure 5 shows the simulation result for Map T (isomorphic mapping).

Input	55
Output	0e

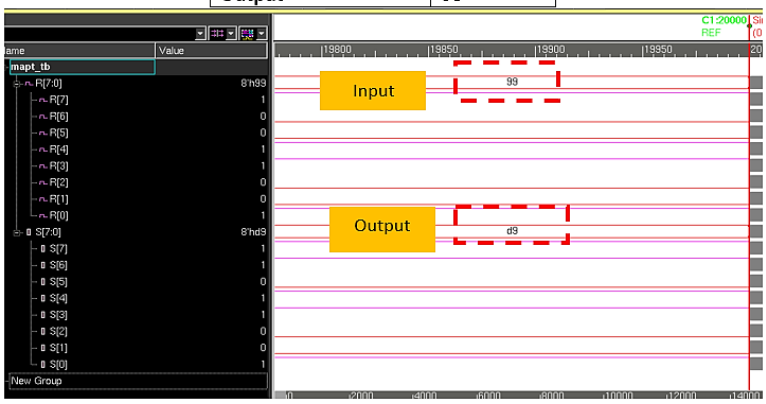

Fig. 5. Output waveform for Map T (isomorphic mapping)

3.1.2 Inverse isomorphic mapping

Inverse Map T brings the same function as the Map T submodules however operates vice versa. Figures 6 show the simulation result for inverse Map T.

Input	55
Output	66

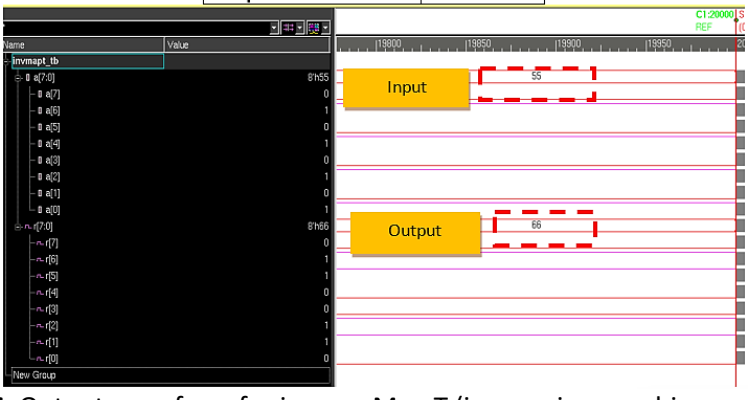

Fig. 6. Output waveform for inverse Map T (inverse isomorphic mapping)

3.1.3 Affine transformation

The Affine transformation operates on isomorphic affine transformed GF (2^8) multiplicate inverse of the same bytes. Figure 7 shows the result output generated from Affine operation.

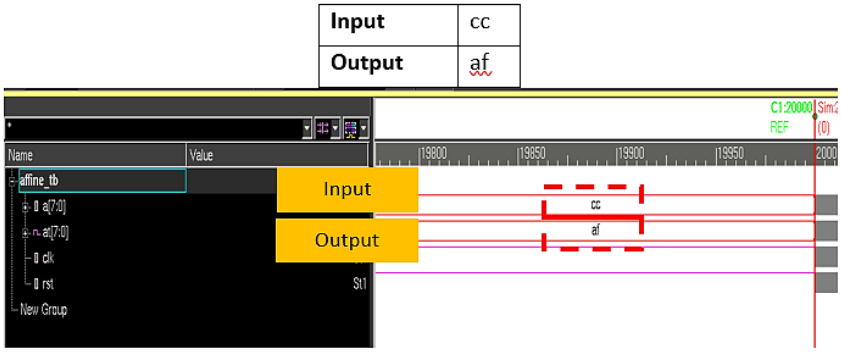

Fig. 7. Output waveform for affine transformation

3.1.4 Inverse Affine transformation

The inverse Affine transformation operates vice-versa with the Affine. Figure 8 shows the result output generated from Inverse Affine operation.

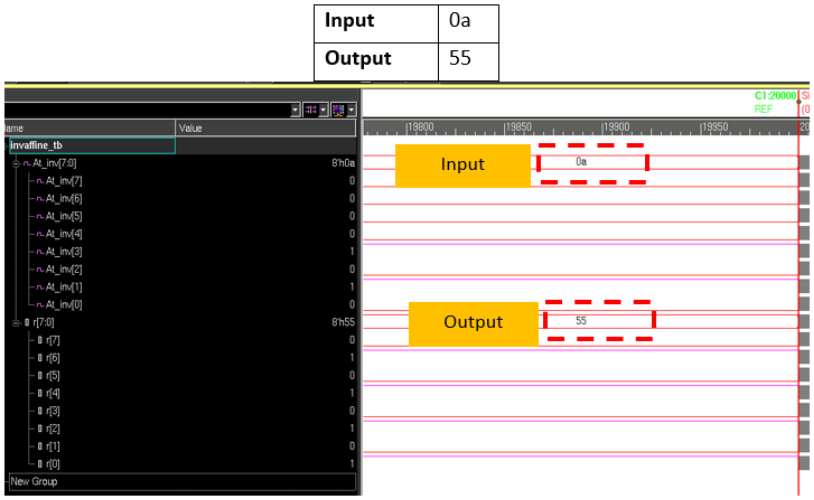

Fig. 8. Output waveform for inverse Affine transformation

3.1.5 Stage 1

Stage 1 includes logic optimization, multiplication with constant, squaring and addition included in one submodule. Figures 9 show simulation results for Stage 1.

Input		Output	
w	**q**	**y**	**m**
a	d	0	8

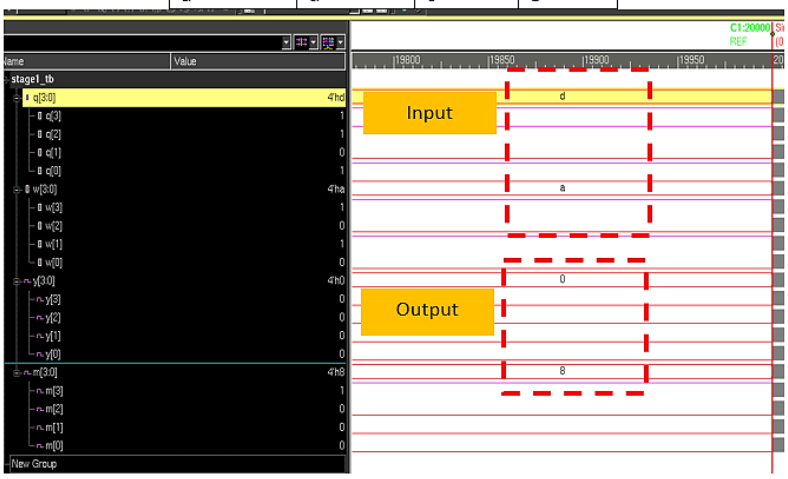

Fig. 9. Output result for Stage 1 module

3.1.6 Combine XAXB

Combine XAXB is to minimize the multiplication in GF(2^4) after multiplicative inversion in GF(2^4). Figure 10 shows the simulation result for CombineXAXB.

Input			Output
w	**q**	**theta**	**A**
d	a	6	47

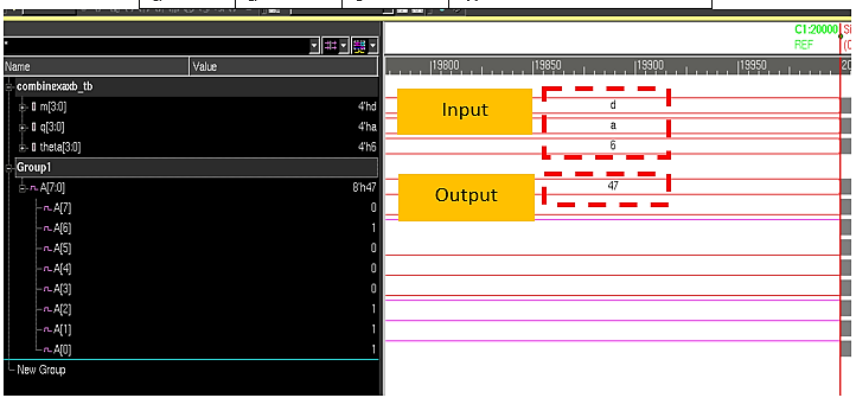

Fig. 10. Output waveform for combine XAXB module

3.1.7 Simplification of multiplicative inverse of nibble in GF (2^4)

The results obtained from the simulation of the X module, which operates as an intermediary component between Stage 1 and the CombineXAXB module, are comprehensively illustrated in Figure 11. This figure provides a detailed depiction of the module's behavior, showcasing how the X module contributes to the overall processing flow between these stages.

Input	d
Output	f

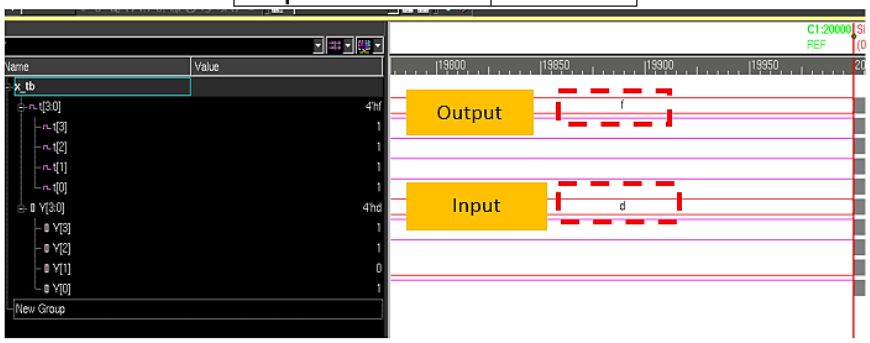

Fig. 11. Output waveform for multiplicative inverse of nibble

3.1.8 S-Box

Throughput for an S-box refers to how quickly it can process data, typically measured in bits per second (bps). Higher throughput means the S-box can handle more data efficiently, impacting the speed and responsiveness of cryptographic operations in a system. Achieving optimal throughput often involves trade-offs with factors like power consumption and implementation size. Figure 12 is the simulation result, and Eq. (6) is the throughput for the modified S-Box.

Input	d9
Output	08

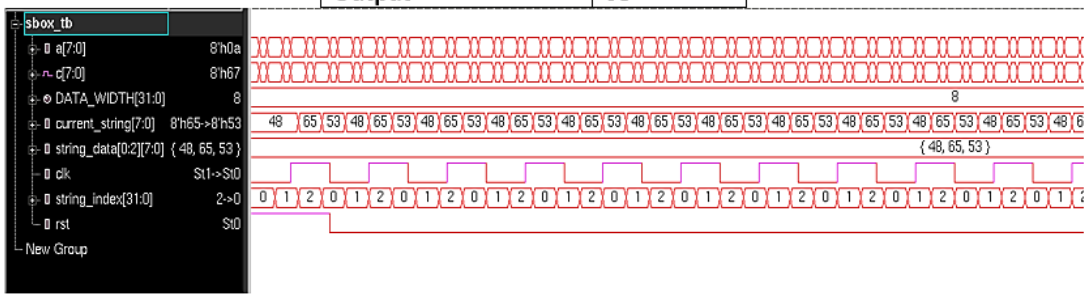

Fig. 12. Output waveform for S-Box

$$Throughput = \frac{128bit}{latency} = \frac{128bit}{5000\ ps} = 3.2\ Mbps \quad (6)$$

3.2 Performance of Proposed S-Box

In comparing various S-box implementations in Table 1, the proposed design on a 90nm process supports decryption with an operating frequency of 100MHz, a latency of 5ns, and a throughput of 0.0256Gbps. Among the existing works, [11] in 2021 on a 40nm process stands out with the highest throughput of 26.667Gbps, but it does not support decryption. Jagata *et al.,* [12] achieves a high throughput of 61.863Gbps but at a relatively low operating frequency of 25MHz. Works Lin *et al.,* [13], Ahmad and Rezaul [7], and Teng *et al.,* [11] strike a balance between throughput, power consumption, and area, with notable variations in operating frequency, latency, and supported features. Work Teng *et al.,* [11] in 2021 on a 90 nm process achieves a high throughput of 9.639 Gbps but lacks decryption support. Ultimately, the choice between these designs should be made considering specific application requirements, including power constraints, area considerations, and the need for decryption functionality.

Table 1
Performance of S-Box

	Proposed S-box	[13]	[7]	[11]	[11]	[11]	[11]	[12]
Year	2023	2023	2021	2021	2021	2021	2021	2020
Technology (nm)	90	40	130	90	90	40	40	-
Voltage (V)	-	-	0.8	-	-	-	-	-
Decryption	yes	yes	yes	no	yes	no	yes	yes
Latency (ns)	5	0.8	-	-	-	-	-	80
Frequency (MHz)	100	800	100	1024.82	1075.27	3333.33	3125	25
Power (uW)	-	299.6	7.33	-	-	-	-	2.69
Throughput (Gbps)	0.0256	6.4	0.8	9.639	8.602	26.667	25	61.863
Area (um^2)	-	462.45	39.44	2769.84	3067.95	528.44	593.31	566.6

3.3 AES-128

In this section, the simulation of AES-128 encryption was successfully completed. In Figure 13, the result of AES-128 encryption with the following input is shown below. AES encryption will complete the encryption in 1100000 ps obtained from Figure 14. The performance of AES is analyzed based on equation on Eq. (7). The time to complete the encryption is 1100000 ps, thus the throughput is 11.63 Mbps.

Input	
Plaintext	00000000000000000000000000000000
Key	80000000000000000000000000000000
Output	
Ciphertext	0edd33d3c621e546455bd8ba1418bec8

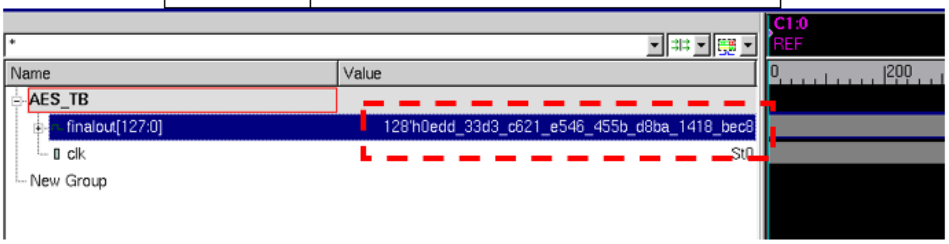

Fig. 13. Simulation results of AES-128

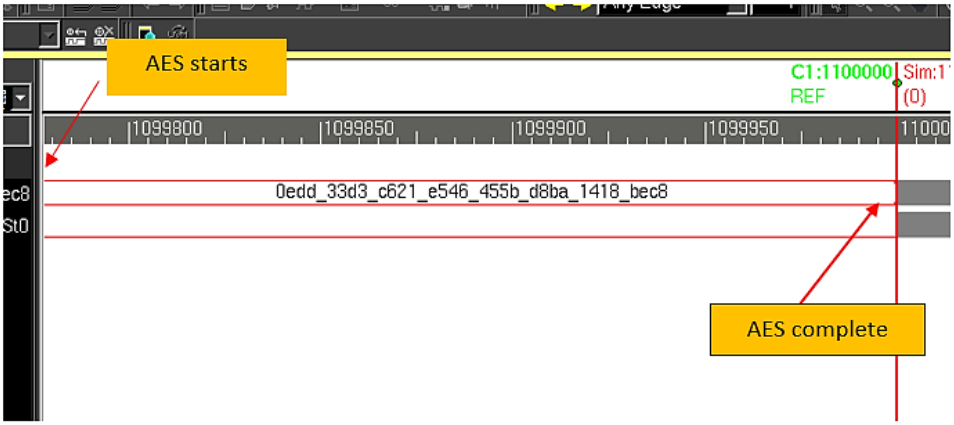

Fig. 14. AES-128 complete encryption

$$Throughput = \frac{128bit}{latency} = \frac{128bit}{1100000\ ps} = 14.54\ MBps \quad (7)$$

3.4 Design Synthesis

The error-free top-level module is loaded into Synopsys Design Compiler to undergo compilation along with design constraints, resulting in the generation of a gate-level netlist. This netlist is then exported in the .ddc file format named AES1.ddc. To assess the performance of the design, timing, area, and power analyses were conducted. The outcomes of these analyses are summarized in Table 2, which presents the performance results of the design system. The results indicate that no timing violations were reported, as the timing slack exhibited a positive value. This indicates that the data required time exceeds the data arrival time, ensuring proper functionality. The latency depends on the design of AES-128. The result of latency obtained is 110000 ps. The throughput achieved from proposed design is 14.54 MBps, utilized 432490.54 um^2. The proposed work is 78.21% slower than the reference work.

Table 2
Results summaries of timing analysis

Design matrix	Logical synthesis
Timing slack (ns)	0
Total area (um^2)	432490.54

3.5 Performance of Proposed AES

The designed system, implemented in Verilog HDL, has been subjected to simulation and synthesis using Synopsys. Distinguishing itself with a 128-bit key length in comparison to prior efforts, Table 3 presents a comprehensive performance evaluation, encompassing throughput, frequency, key length, and area utilization for both the proposed design and preceding works. The proposed work, based on 90nm technology with a key length of 128-bit, exhibits a promising performance metric compared to various references in the field. It achieves a throughput of 0.01163Gbps with an area of 0.4324 mm^2.

Table 3
Performance of proposed AES

Design	Tech (nm)	Key	Frequency (MHz)	Area (mm^2)	Throughput (Gbps)
Proposed work	90	128	100	0.4324	0.01163
[14]	90	128	100	22.724	0.0256
[15]	45	128	870	0.13	111.3
[16]	28	128	50	0.0028	0.03
[17]	28	128	10	-	0.028
[18]	65	128	130.9	0.014	-
[18]	65	128	127.2	0.013	-
[18]	65	128	10	0.0037	0.0165
[19]	130	128	10	0.148	-
[20]	45	128	100	6.32	128

When compared to work Ali *et al.,* [18], which shares the same technology node, key length and frequency, the proposed work lags behind in throughput but showcases efficiency in the terms of area utilization. Furthermore, against work [11], despite larger area utilization with 45 nm technology node, the proposed work demonstrated a superior throughput. In comparison to work [12],

Bahnasawi *et al.,* [19] and Shanthi and Saravanan [5], the proposed work excels in throughput while maintaining competitive or smaller area utilization. However, consideration of trade-offs is necessary especially when assessing alternatives such as work [12], where technology nodes and areas present a nuanced comparison in terms of throughput and area efficiency, offering competitive edge in the domain of design tech. In conclusion, the proposed work holds promise in terms of throughput and area efficiency, though careful consideration of specific application requirements is advisable.

5. Conclusions

In conclusion, this study successfully achieved the objectives of designing, optimizing, and verifying AES algorithm using ASIC implementation. The AES was designed in 90 nm CMOS technology, utilizing the Synopsys Design Tool as a platform and Verilog HDL for system design. In AES encryption and decryption process, the standard length of input and output block is 128-bit, and the length of the key is proposed for 128-bit. The AES contains modified S-Box employs combinational logic using composite field arithmetic. The proposed S-Box in the AES managed to achieve a balance between low area, high performance and maintaining the requisite level of security. This optimization contributes to the overall efficiency of the AES architecture, ensuring the operations are performed with minimal resources utilization.

However, the throughput achieved from proposed design is 11.63 Mbps, utilized 432490.54 um^2. The proposed work is 78.21% slower than the reference work. The verification process for AES architecture encompassed functionality, timing, area, and performance. Rigorous testing using NIST sample vectors validated the accuracy of the output results, demonstrating flawless performance with minimal delays. It ensures that the ASIC implementation meets the requirement and provides a reliable and efficient cryptography operations.

Acknowledgement

This research was supported by Ministry of Higher Education (MOHE) through Fundamental Research Grant Scheme (FRGS/1/2022/TK07/UTHM/02/20).

References

[1] Liu, Shilin, Yongzhen Li, and Zhexue Jin. "Research on Enhanced AES Algorithm Based on Key Operations." In *2023 IEEE 5th International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, p. 318-322. IEEE, 2023. https://doi.org/10.1109/ICCASIT58768.2023.10351719

[2] Sandhya Rani, M., R. Rekha, and K. V. N. Sunitha. "Secure Group Key Exchange and Encryption Mechanism in MANETs." In *Innovations in Computer Science and Engineering: Proceedings of the Fifth ICICSE 2017*, p. 383-390. Springer Singapore, 2019. https://doi.org/10.1007/978-981-10-8201-6_43

[3] Cobb, Michael. "RSA algorithm (Rivest-Shamir-Adleman)." *TechTarget and Informa,* 2021.

[4] Rajasekar, P., and H. Mangalam. "Design and implementation of power and area optimized AES architecture on FPGA for IoT application." *Circuit World* 47, no. 2 (2020): 153-163. https://doi.org/10.1108/CW-04-2019-0039

[5] Shanthi Rekha, S., and P. Saravanan. "Low-cost AES-128 implementation for edge devices in IoT applications." *Journal of Circuits, Systems and Computers* 28, no. 04 (2019): 1950062. https://doi.org/10.1142/S0218126619500622

[6] Sousi, Ahmad-Loay, Dalia Yehya, and Mohamad Joudi. "Aes encryption: Study & evaluation." *CCEE552: Cryptography and Network Security* (2020).

[7] Ahmad, Nabihah, and SM Rezaul Hasan. "A new ASIC implementation of an advanced encryption standard (AES) crypto-hardware accelerator." *Microelectronics Journal* 117 (2021): 105255. https://doi.org/10.1016/j.mejo.2021.105255

[8] Nabil, Mohamed, Ashraf AM Khalaf, and Sara M. Hassan. "Design and implementation of pipelined and parallel AES encryption systems using FPGA." *Indonesian Journal of Electrical Engineering and Computer Science* 20, no. 1 (2020): 287-299. https://doi.org/10.11591/ijeecs.v20.i1.pp287-299

[9] Hamzah, Hidayarni, Nabihah Ahmad, and Siti Hawa Ruslan. "The 128-bit AES design by using FPGA." In *Journal of Physics: Conference Series*, vol. 1529, no. 2, p. 022059. IOP Publishing, 2020. https://doi.org/10.1088/1742-6596/1529/2/022059

[10] Hamzah, Hidayarni, Nabihah Ahmad, M. Hairol Jabbar, and Chin Fhong Soon. "AES S-Box/Inv S-Box Optimization Using FPGA Implementation." *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)* 9, no. 3-8 (2017): 133-136.

[11] Teng, You-Tun, Wen-Long Chin, Deng-Kai Chang, Pei-Yin Chen, and Pin-Wei Chen. "VLSI architecture of S-box with high area efficiency based on composite field arithmetic." *IEEE Access* 10 (2021): 2721-2728. https://doi.org/10.1109/ACCESS.2021.3139040

[12] Jagata, Sridevi, Ganapathi Hegde, and N. S. Murty. "High Throughput Pipelined S-Boxes for Encryption and Watermarking Applications." In *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, p. 710-715. IEEE, 2020. https://doi.org/10.1109/ICOSEC49089.2020.9215252

[13] Lin, Shih-Hsiang, Jun-Yi Lee, Chia-Chou Chuang, Narn-Yih Lee, Pei-Yin Chen, and Wen-Long Chin. "Hardware Implementation of High-Throughput S-Box in AES for Information Security." *IEEE Access* 11 (2023): 59049-59058. https://doi.org/10.1109/ACCESS.2023.3284142

[14] Noor, Safwat Mostafa, and Eugene B. John. "Resource shared galois field computation for energy efficient AES/CRC in IoT applications." *IEEE Transactions on Sustainable Computing* 4, no. 4 (2019): 340-348. https://doi.org/10.1109/TSUSC.2019.2943878

[15] Dong, Pham-Khoi, Hung K. Nguyen, and Xuan-Tu Tran. "A 45nm high-throughput and low latency aes encryption for real-time applications." In *2019 19th International Symposium on Communications and Information Technologies (ISCIT)*, pp. 196-200. IEEE, 2019. https://doi.org/10.1109/ISCIT.2019.8905235

[16] Lu, Minyi, Ao Fan, Jiaming Xu, and Weiwei Shan. "A compact, lightweight and low-cost 8-bit datapath AES circuit for IOT applications in 28nm CMOS." In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE)*, pp. 1464-1469. IEEE, 2018. https://doi.org/10.1109/TrustCom/BigDataSE.2018.00204

[17] Bui, Duy-Hieu, Diego Puschini, Simone Bacles-Min, Edith Beigné, and Xuan-Tu Tran. "AES datapath optimization strategies for low-power low-energy multisecurity-level internet-of-things applications." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 25, no. 12 (2017): 3281-3290. https://doi.org/10.1109/TVLSI.2017.2716386

[18] Ali, Liakot, Ishak Aris, Fakir Sharif Hossain, and Niranjan Roy. "Design of an ultra high speed AES processor for next generation IT security." *Computers & Electrical Engineering* 37, no. 6 (2011): 1160-1170. https://doi.org/10.1016/j.compeleceng.2011.06.003

[19] Bahnasawi, Mohamed A., Khalid Ibrahim, Ahmed Mohamed, Mohamed Khalifa Mohamed, Ahmed Moustafa, Kareem Abdelmonem, Yehea Ismail, and Hassan Mostafa. "ASIC-oriented comparative review of hardware security algorithms for internet of things applications." In *2016 28th International Conference on Microelectronics (ICM)*, p. 285-288. IEEE, 2016. https://doi.org/10.1109/ICM.2016.7847871

[20] Rashidi, Bahram. "High-throughput and lightweight hardware structures of HIGHT and PRESENT block ciphers." *Microelectronics Journal* 90 (2019): 232-252. https://doi.org/10.1016/j.mejo.2019.06.012