IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 17 May 2023, accepted 3 June 2023, date of publication 8 June 2023, date of current version 19 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3284142

RESEARCH ARTICLE

Hardware Implementation of High-Throughput S-Box in AES for Information Security

SHIH-HSIANG LIN[1], JUN-YI LEE[2], CHIA-CHOU CHUANG[2], NARN-YIH LEE[3], PEI-YIN CHEN[1], (Senior Member, IEEE), AND WEN-LONG CHIN[4], (Senior Member, IEEE)

[1]Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan 70101, Taiwan
[2]Department of Electrical Engineering, National Cheng Kung University, Tainan 70101, Taiwan
[3]Network and Information Security Division, Computer and Network Center, National Cheng Kung University, Tainan 70101, Taiwan
[4]Department of Engineering Science, National Cheng Kung University, Tainan 70101, Taiwan

Corresponding author: Pei-Yin Chen (pychen@mail.ncku.edu.tw)

This work was supported by the National Science and Technology Council at Taiwan under Grant NSTC 110-2221-E-006-164-MY3.

ABSTRACT The Advanced Encryption Standard (AES) is used for achieving quantum-resistant cryptography when a 256-bit key is applied. This paper presents a high-throughput, seven-stage hardware pipeline architecture for SubByte computations in the AES for information security applications. Composite field arithmetic–based calculations are employed for logic optimization. The proposed architecture includes dedicated multistage multiplication processes based on Galois field polynomials for constants, squaring, and variables; thus, the critical path of SubByte computations is shortened, and the maximum operating frequency is enhanced. The proposed architecture was synthesized using a TSMC 40-nm cell library, and the throughput of the proposed architecture (34.78 Gbps) was observed to be superior to that of an existing state-of-the-art architecture by 43.47%. Moreover, our architecture was noted to consume lower dynamic power for combinational logic circuits, indicating that the proposed architecture has greater computational logic optimization than existing designs. The proposed architecture is feasible for communication security applications in the Internet of Things systems because of its high throughput and area efficiency.

INDEX TERMS AES, hardware, high-throughput, information security, S-box.

I. INTRODUCTION

Information security is critical for communication network systems, particularly for Internet of Things (IoT) applications and smart city systems [1]. IoT systems must be equipped with high-throughput and low-latency features to ensure high service quality [2]. To ensure communication security, [3] proposed an elliptic curve cryptography (ECC)-based routing protocol for vehicle-to-vehicle and vehicle-to-infrastructure communications in transportation systems. Considering potential quantum computing attacks, [4] presented a detailed comparison of quantum-resistant cryptosystems. In addition, several cryptography techniques have been introduced, and a comprehensive analysis of future challenges and possible attacks from quantum computing has been provided. The ECC-based protocol proposed in [3] affords remarkable performance in terms of throughput and delay. Nevertheless, the overall comparison of cryptosystems in [4] revealed that ECC is potentially vulnerable to quantum computing attacks; the comparisons also revealed that Advanced Encryption Standard-256 (AES-256) is among the most potent symmetric cryptosystems and is considered to be quantum resistant: quantum computers are not expected to reduce the attack time effectively if the encryption key is sufficiently large.

The AES [5] is a commonly used specification for data encryption. AES-based encryption involves the following sequence of operations: SubBytes, ShiftRows, MixColumns, and AddRoundKey [6]. The decryption process involves a similar sequence of operations and similar functions; the only difference between the operations is that decryption involves an inverse calculation process. If 128-, 192-, and 256-bit key lengths are used in AES, the four aforementioned operations would require 10, 12, and 14 transformation rounds, respectively. Despite the emergence of potential threats posed by quantum computing, the AES

The associate editor coordinating the review of this manuscript and approving it for publication was Sedat Akleylek.

continues to offer powerful data protection if a 256-bit key is used [4].

Developing an efficient cryptographic accelerator to improve the performance of secure applications in IoT systems is essential; however, SubBytes and inverse SubBytes are the most complex computational operations in the AES and have the longest computation time. These byte-oriented operations can be designed as substitution-box (S-Box) modules; in such modules, an 8-bit input is transformed into another output value with identical bit-width through a series of nonlinear operations comprising multiplication and multiplicative inverse calculations in the Galois field $GF(2^8)$. Therefore, hardware accelerators—rather than software implemented on general-purpose processors—are commonly used to improve the performance of AES processes to meet real-time processing requirements [7]. For efficient data transfer and storage, AES-based systems must exhibit high throughput [8], which can be achieved using field-programmable gate arrays [9], [10], [11] or application-specific integrated circuits (ASICs) [12], [13], [14], [15], [16]. Accordingly, to meet the requirement of high-throughput performance for communication security applications, the present study presents a high-throughput, seven-stage hardware pipeline architecture for SubByte computations in the AES for information security applications. The contributions of this study are outlined as follows:

1) A seven-stage architecture for SubByte and inverse SubByte computations is proposed for AES encryption. The computation time of each stage is balanced and shortened to obtain a higher operating frequency.
2) An analysis of throughput and latency among different implementations, including pipeline and non-pipeline architectures, is presented. In addition, a comparison of area–throughput efficiency is presented to demonstrate the performance of the proposed design.
3) As demonstrated by the hardware implementation results, the proposed architecture has the lowest latency despite the increase in the cell area for pipeline registers. It is superior to state-of-the-art designs in terms of maximum throughput.

II. RELATED WORK

S-Box transformation can be directly implemented using look-up tables [17]; however, implementing such tables using hardware functions involves a substantial area cost. Accordingly, studies have proposed composite field arithmetic methods for reducing the cost of AES-based S-Box transformation processes implemented using hardware functions. For example, [18] presented an algorithm for computing SubBytes and inverse SubBytes in AES processes; the algorithm uses multiplicative inverse and affine transformation operations in $GF((2^4)^2)$ and $GF(((2^2)^2)^2)$ rather than directly using complete representations in $GF(2^8)$ because the calculation of multiplicative inverse operations in $GF(2^8)$ requires much more cell area when implementing in hardware. For this

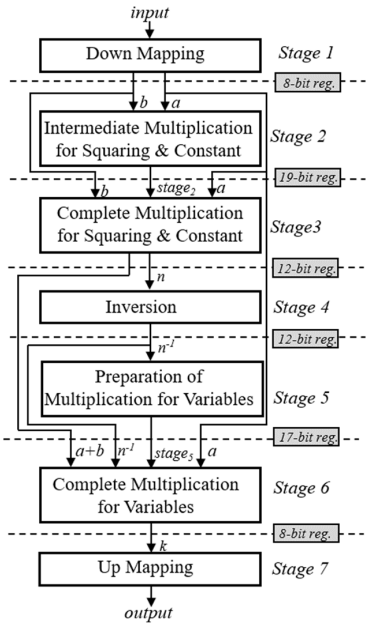

FIGURE 1. High throughput design of proposed combined S-Box architecture.

reason, it is area-efficient if mapping elements in $GF(2^8)$ into $GF(2^4)$ or further into $GF(2^2)$ by isomorphic mapping. The multiplicative inverse operations which referenced from [18] are expressed in (1) and (2), where y represents a polynomial and ν represents a constant decimal value 12 in the GF. In addition, [19] proposed an optimization technique for enhancing hardware efficiency for linear mappings in the composite field.

$$G = (ay + b)^{-1} = an^{-1}y + (a + b)n^{-1} \tag{1}$$

$$n = \nu a^2 + ab + b^2 \tag{2}$$

Reyhani et al. [12], [13] have developed a series of approaches for optimizing S-Box cost efficiency. However, their methods entail executing computations in one clock cycle, resulting in a lengthy data path and a slower operating frequency for encryption and decryption functions. Moreover, [14], [15], and [16] have presented pipeline designs involving various stages for operating frequency improvement. In particular, the design presented in [16] involves five stages and is the most cost-efficient scheme with the highest throughput; nevertheless, the maximum operating frequency of the design is limited by its multiplication computation, which is a critical problem that requires addressing.

$$ENC_{stage\ 1} = IM(in) = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} in_7 \\ in_6 \\ in_5 \\ in_4 \\ in_3 \\ in_2 \\ in_1 \\ in_0 \end{bmatrix} \tag{3}$$

$$DEC_{stage\ 1} = IM\left(AF^{-1}(in)\right)$$
$$= \begin{bmatrix} 1&0&1&0&0&0&0&0\\ 1&1&0&1&1&1&1&0\\ 1&0&1&0&1&1&0&0\\ 1&0&1&0&1&1&1&0\\ 1&1&0&0&0&1&1&0\\ 1&0&0&1&1&1&1&0\\ 0&1&0&1&0&0&1&0\\ 0&1&0&0&0&0&1&1 \end{bmatrix}$$
$$\cdot \left(\begin{bmatrix} 0&1&0&1&0&0&1&0\\ 0&0&1&0&1&0&0&1\\ 1&0&0&1&0&1&0&0\\ 0&1&0&0&1&0&1&0\\ 0&0&1&0&0&1&0&1\\ 1&0&0&1&0&0&1&0\\ 0&1&0&0&1&0&0&1\\ 1&0&1&0&0&1&0&0 \end{bmatrix} \cdot \begin{bmatrix} in_7\\ in_6\\ in_5\\ in_4\\ in_3\\ in_2\\ in_1\\ in_0 \end{bmatrix} + \begin{bmatrix} 0\\0\\0\\0\\0\\1\\0\\1 \end{bmatrix} \right)$$
$$= \begin{bmatrix} 1&1&0&0&0&1&1&0\\ 1&1&0&0&1&1&1&1\\ 0&1&1&1&0&0&0&1\\ 0&0&1&1&1&0&0&0\\ 1&0&1&0&0&0&0&0\\ 1&1&1&0&0&1&1&0\\ 0&0&1&0&1&0&1&0\\ 1&1&0&0&0&1&0&0 \end{bmatrix} * \begin{bmatrix} in_7\\ in_6\\ in_5\\ in_4\\ in_3\\ in_2\\ in_1\\ in_0 \end{bmatrix} + \begin{bmatrix} 0\\1\\1\\1\\1\\1\\0\\1 \end{bmatrix} \quad (4)$$

Besides, we can realize from [6] that S-Box is the most crucial block of AES to protect the plaintext, used for substituting the plaintext bytes. It is a non-linear function that maps inputs to different outputs, increasing the complexity of the ciphertext, which can enhance resistance against attacks from hackers. People who know the cipher key can decrypt the ciphertext quickly. But, for the hackers, decrypting the ciphertext is a challenging task. Therefore, more and more research focus on the S-Box to enhance security strength, as mentioned by [20], [21], and [22], which proposed different S-Box with reliable security but lower computational complexity compared to the one used in AES. However, AES is the standard of the U.S. National Institute of Standards and Technology (NIST), as mentioned by [23]. For this reason, in our proposed S-Box design, we follow the same functionality as the S-Box used in AES and optimize the architecture to improve throughput.

III. HIGH-THROUGHPUT S-BOX ARCHITECTURE

This section provides a detailed description of the proposed seven-stage architecture for combined S-Box computations. In addition to seven-stage pipeline, the proposed architecture also optimizes the mathematical equations to reduce the combination circuit cost of each stage. The optimized mathematical equations for each stage are described in (3) – (15). Fig. 1 illustrates an overview of the proposed S-Box architecture based on a reformulation of (1) and (2) under the consideration of requirements pertaining to hardware friendliness and high throughput. Stages 1 and 7 involve isomorphic mapping and affine transformation operations. Stages 2–6 involve multiplicative inversion operations for $ay + b$ in (1); the multiplication operations are implemented separately to simplify the mathematical processes and thus improve system performance. Two different strategies are provided according to the features observed during the execution of various types of multiplication, which are presented in stages 2–3 and 5-6. Under the consideration of area and cost overheads, the use of excessive pipeline stages in the proposed architecture (e.g., more than seven stages) would require additional registers without yielding apparent improvements in performance, which is inappropriate.

$$1 = 2^0 = \begin{bmatrix} 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&1 \end{bmatrix} \quad (5)$$

$$2 = 2^1 = \begin{bmatrix} 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 1&0&0&0&1&0&0&0\\ 1&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&1&0\\ 1&0&0&0&0&0&0&1\\ 1&0&0&0&0&0&0&0 \end{bmatrix} \quad (6)$$

$$ENC_{stage1} = IM\ (C\_en \cdot \text{in})$$
$$= \begin{bmatrix} a_3\\a_2\\a_1\\a_0\\b_3\\b_2\\b_1\\b_0 \end{bmatrix} = \begin{bmatrix} 1&0&1&0&0&0&0&0\\ 1&1&0&1&1&1&1&0\\ 1&0&1&0&1&1&0&0\\ 1&0&1&0&1&1&1&0\\ 1&1&0&0&0&1&1&0\\ 1&0&0&1&1&1&1&0\\ 0&1&0&1&0&0&1&0\\ 0&1&0&0&0&0&1&1 \end{bmatrix}$$
$$\cdot \left(\begin{bmatrix} 1&1&1&1&1&0&1&0\\ 1&1&1&1&1&1&0&1\\ 1&1&1&1&1&1&1&0\\ 1&1&1&1&1&1&1&1\\ 1&0&0&0&0&1&0&1\\ 0&0&1&1&1&0&0&0\\ 0&0&0&1&1&1&0&0\\ 1&1&1&1&0&1&0&0 \end{bmatrix} \cdot \begin{bmatrix} in_7\\ in_6\\ in_5\\ in_4\\ in_3\\ in_2\\ in_1\\ in_0 \end{bmatrix} \right)$$
$$= \begin{bmatrix} 0&0&0&0&0&1&0&0\\ 0&1&0&1&1&0&0&1\\ 1&0&1&1&1&0&0&1\\ 1&0&1&0&0&1&0&1\\ 0&0&1&0&0&0&1&1\\ 1&0&1&0&0&1&0&0\\ 0&0&0&1&1&1&1&0\\ 0&0&0&1&0&1&0&1 \end{bmatrix} \cdot \begin{bmatrix} in_7\\ in_6\\ in_5\\ in_4\\ in_3\\ in_2\\ in_1\\ in_0 \end{bmatrix} \quad (7)$$

$$DEC_{stage1} = IM\left(C\_de \cdot AF^{-1}(in)\right)$$

$$= \begin{bmatrix} a_3 \\ a_2 \\ a_1 \\ a_0 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} 1&0&1&0&0&0&0&0 \\ 1&1&0&1&1&1&1&0 \\ 1&0&1&0&1&1&0&0 \\ 1&0&1&0&1&1&1&0 \\ 1&1&0&0&0&1&1&0 \\ 1&0&0&1&1&1&1&0 \\ 0&1&0&1&0&0&1&0 \\ 0&1&0&0&0&0&1&1 \end{bmatrix}$$

$$\cdot \left(\begin{bmatrix} 0&0&0&0&1&1&0&0 \\ 1&0&0&0&0&1&1&0 \\ 1&1&0&0&0&0&1&1 \\ 0&1&1&0&0&0&0&1 \\ 1&0&1&1&1&1&0&0 \\ 0&1&0&1&0&0&1&0 \\ 0&0&1&0&1&0&0&1 \\ 0&0&0&1&1&0&0&0 \end{bmatrix} \right.$$

$$\left. \cdot \left(\begin{bmatrix} 0&1&0&1&0&0&1&0 \\ 0&0&1&0&1&0&0&1 \\ 1&0&0&1&0&1&0&0 \\ 0&1&0&0&1&0&1&0 \\ 0&0&1&0&0&1&0&1 \\ 1&0&0&1&0&0&1&0 \\ 0&1&0&0&1&0&0&1 \\ 1&0&1&0&0&1&0&0 \end{bmatrix} \cdot \begin{bmatrix} in_7 \\ in_6 \\ in_5 \\ in_4 \\ in_3 \\ in_2 \\ in_1 \\ in_0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \right) \right)$$

$$= \begin{bmatrix} 0&0&1&0&0&0&0&1 \\ 0&0&1&0&0&0&1&1 \\ 0&0&1&1&0&0&0&0 \\ 0&0&1&0&0&1&0&1 \\ 0&0&0&0&0&0&0&1 \\ 1&0&1&0&1&0&1&0 \\ 1&0&0&0&0&1&0&1 \\ 1&1&1&1&0&0&1&1 \end{bmatrix} * \begin{bmatrix} in_7 \\ in_6 \\ in_5 \\ in_4 \\ in_3 \\ in_2 \\ in_1 \\ in_0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad (8)$$

In the proposed architecture, single-bit addition and multiplication operations can be directly achieved using *XOR* and *AND* logic gates, respectively.

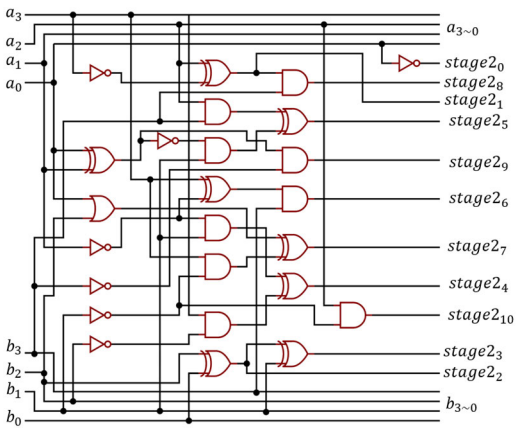

FIGURE 2. Schematic of stage 2.

A. DOWN MAPPING

In the first stage of the architecture, an 8-bit value is received as the input data for the S-Box. A control signal that indicates whether an encryption or decryption process should be performed is also received. In this stage, encryption is executed through isomorphic mapping operations (3), and decryption is accomplished through a combination of isomorphic mapping and inverse affine transformation operations (4), where *IM* denotes isomorphic mapping and AF^{-1} denotes inverse affine transformation. The "+" operator and "+1" computations are implemented using *XOR* and *NOT* logic gates, respectively; hence, the resource cost would be 27 XOR gates and 6 NOT gates, according to the derivations in (3) and (4).

Multiplicative offsets are adopted in our architecture to create matrix computations that require fewer resources. The input data are multiplied initially by constant offsets, denoted as *C_en* and *C_de* for encryption and decryption, respectively. The same offsets are applied in the final stage of our architecture. The offsets can be represented by 8 × 8 matrices, and (5) and (6) present examples for constant values (i.e., 1 and 2) derived through composite field arithmetic computations. Encryption and decryption offsets that afford the highest resource efficiency can be derived through an exhaustive search for all possible values in the range from 0 to 255; thus, the *C_en* value can be derived as 88 (computed by $2^6 + 2^4 + 2^3$), and the *C_de* value can be derived as 50 (calculated by $2^5 + 2^4 + 2^1$). These constants can be transformed into 8 × 8 matrix forms based on (5) and (6) for further derivation through composite

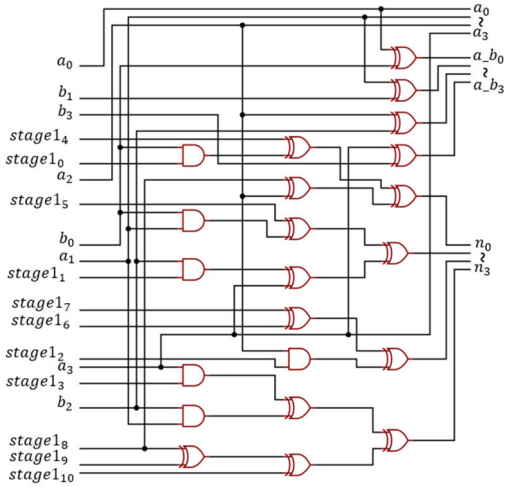

FIGURE 3. Schematic of stage 3.

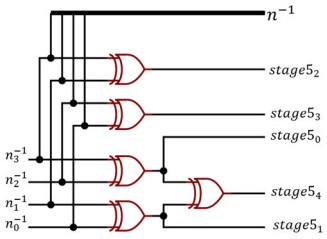

FIGURE 4. Schematic of stage 5.

field arithmetic computations implemented as polynomial operations. Rewriting (3) and (4) can yield (7) and (8), representing revised mapping computations. Hence, according to the derivations in (7) and (8), the resource cost in this stage would be reduced to 24 *XOR* gates and 4 *NOT* gates.

The output is divided into two 4-bit values, represented by the coefficients a and b in (1), where a and b are the most and least significant 4 bits of the output, respectively. The subscripts in (7) and (8) denote the corresponding bit positions of the data, where 0 is the least significant bit.

B. INTERMEDIATE MULTIPLICATION FOR SQUARING AND CONSTANT

Stage 2 and 3 generate the result of $va^2 + ab + b^2$ in (2). This stage involves deriving 11-bit intermediate results for calculating (2), as shown in equation (9). In the calculation, the coefficient a is squared and multiplied by the constant v. The coefficients a and b are also multiplied. These steps generate only temporary values, and complete multiplication is performed in the subsequent stage. In the current stage, a and b, both 4-bit values, are the input. Because the two coefficients must be transmitted to later stages, an additional 8-bit register is required for storage. Therefore, a 19-bit register is needed. Fig. 2 displays a schematic of this stage. The critical path of this stage is as follows: 1 *NOT* + 1 *AND* + 2 *XOR*.

C. COMPLETE MULTIPLICATION FOR SQUARING AND CONSTANT

In stage 3, the intermediate results derived from stage 2 and two coefficients are used as the input. Subsequently, the complete result of $va^2 + b(a + b)$, reformulated from (2), is obtained as shown in equation (10). This stage also involves the addition computation a + b implemented using the *XOR* operator. The output of this stage comprises a 4-bit value n obtained from the multiplication, a 4-bit value obtained from the summation of a and b (denoted as a_b in Fig. 4), and a 4-bit coefficient a. Only the coefficient a is required after this stage. Fig. 3 illustrates a schematic of the stage. The critical path of this stage is 3 *XOR*.

D. INVERSION

Stage 4 involves the multiplicative inversion operation of $GF((2^2)^2)$. The input in this stage is the 4-bit n derived from stage 3, and the output is the inversion result. This stage also involves a 12-bit register for storage because the 4-bit value obtained from the summation $a + b$ and the 4-bit coefficient a must be stored, as shown in Fig. 1. The coefficients are bypassed directly without additional computation in this stage. The inversion equation is presented in (11), where n_0' denotes the logic *NOT* operation for n_0; the logic *AND* operation are used for the bitwise multiplication process. The critical path of this stage is as follows: 1 *XOR* + 2 *INV* + 1 *NAND* + 1 *OR* + 1 *AND*.

$$\begin{cases} stage_1 = b_3\left(a_3' + a_2\right) \\ stage2_2 = (b_2 + b_1 + b_0) \\ stage2_3 = b_3'\left(a_1 + a_0\right) \\ stage2_4 = \left(a_2 b_1'\right) \\ stage2_5 = b_3\left(a_1' + a_3\right) \\ stage_6 = \left(a_0 \mid b_2 + a_3 b_1'\right) \\ stage2_7 = (b_2 + b_0) \\ stage2_8 = \left(a_3' + a_2\right) \\ stage2_9 = \left(b_2' a_3\right) \\ stage2_{10} = \left(b_1 a_1'\right) \\ stage2_{11} = a_0' \end{cases} \tag{9}$$

$$\begin{cases} n_3 = \text{stage2}_1 + \text{a}_3\,\text{stage2}_2 + \text{stage2}_3 + \text{b}_2\text{a}_1 + \text{stage2}_4 \\ n_2 = \text{stage2}_5 + \text{stage2}_6 + \text{a}_2\,\text{stage2}_7 \\ n_1 = \text{b}_2\text{stage2}_8 + \text{b}_3\text{a}_2 + \text{b}_1\,(\text{a}_1 + \text{a}_0)' + \text{b}_0\text{a}_1 + \text{a}_3 \\ \text{n}_0 = \text{stage2}_1 + \text{stage2}_9 + \text{stage2}_{10} + \text{b}_0\,\text{stage2}_{11} + \text{a}_2 \end{cases} \tag{10}$$

$$\begin{cases} n_3^{-1} = n_3' n_2 + n_2 n_1' n_0 + n_2 n_1 n_0' + n_3 n_2' n_0' \\ n_2^{-1} = n_3' n_2 n_1' + n_3 n_2 + n_3 n_2' n_0 \\ n_1^{-1} = n_1' n_3' n_2 n_0' + n_1' n_3 n_2' + n_3 n_0 n_1' + n_3 n_0 n_2' + n_3' n_1 n_2' \\ \qquad + n_3' n_1 n_0 \\ n_0^{-1} = n_3' n_1 n_0' + n_3' n_1 n_2 + n_2' n_0 n_3 n_1 + n_1' n_3' n_0 n_2' + n_0' n_2 \end{cases} \tag{11}$$

$$\begin{cases} stage5_0 = n_3^{-1} + n_2^{-1} \\ stage5_1 = n_1^{-1} + n_0^{-1} \\ stage5_2 = n_3^{-1} + n_1^{-1} \\ stage5_3 = n_2^{-1} + n_0^{-1} \\ stage5_4 = n_3^{-1} + n_2^{-1} + n_1^{-1} + n_0^{-1} \end{cases} \tag{12}$$

$$\begin{cases} k_7 = a_3 stage5_4 + a_2 stage5_2 + a_1 stage5_0 + a_0 n_3 \\ k_6 = a_3 stage5_2 + a_1 n_3 + a_2 stage5_3 + a_0 n_2 \\ k_5 = a_2 stage5_0 + a_3 n_2 + a_1 stage5_1 + a_0 n_1 \\ k_4 = a_3 stage5_0 + a_2 n_3 + a_1 n_1 + a_0 n_0 \\ k_3 = a\_b_3 stage5_4 + a\_b_2 stage5_2 + a\_b_1 stage5_0 \\ \qquad + a\_b_0 n_3 \\ k_2 = a\_b_3 stage5_2 + a\_b_1 n_3 + a\_b_2 stage5_3 + a\_b_0 n_2 \\ k_1 = a\_b_2 stage5_0 + a\_b_3 n_2 + a\_b_1 stage5_1 + a\_b_0 n_1 \\ k_0 = a\_b_3 stage5_0 + a\_b_2 n_3 + a\_b_1 n_1 + a\_b_0 n_0 \end{cases} \tag{13}$$

$$ENC_{stage7} = \begin{bmatrix} out\_EN_7 \\ out\_EN_6 \\ out\_EN_5 \\ out\_EN_4 \\ out\_EN_3 \\ out\_EN_2 \\ out\_EN_1 \\ out\_EN_0 \end{bmatrix} = AF \cdot \mathrm{C}_{en} \cdot \left(IM^{-1}(in)\right)$$

$$= \begin{bmatrix} 1&1&1&1&1&0&0&0 \\ 0&1&1&1&1&1&0&0 \\ 0&0&1&1&1&1&1&0 \\ 0&0&0&1&1&1&1&1 \\ 1&0&0&0&1&1&1&1 \\ 1&1&0&0&0&1&1&1 \\ 1&1&1&0&0&0&1&1 \\ 1&1&1&1&0&0&0&1 \end{bmatrix} \cdot \left(\begin{bmatrix} 1&1&1&1&1&0&1&0 \\ 1&1&1&1&1&1&0&1 \\ 1&1&1&1&1&1&1&0 \\ 1&1&1&1&1&1&1&1 \\ 1&0&0&0&0&1&0&1 \\ 0&0&1&1&1&0&0&0 \\ 0&0&0&1&1&1&0&0 \\ 1&1&1&1&0&1&0&0 \end{bmatrix} \cdot \begin{bmatrix} 1&1&1&0&0&0&1&0 \\ 0&1&0&0&0&1&0&0 \\ 0&1&1&0&0&0&1&0 \\ 0&1&1&1&0&1&1&0 \\ 0&0&1&1&1&1&1&0 \\ 1&0&0&1&1&1&1&0 \\ 0&0&1&1&0&0&0&0 \\ 0&1&1&1&0&1&0&1 \end{bmatrix} \cdot \begin{bmatrix} k_7 \\ k_6 \\ k_5 \\ k_4 \\ k_3 \\ k_2 \\ k_1 \\ k_0 \end{bmatrix} \right) + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1&0&1&0&0&1&1&1 \\ 0&0&1&1&0&0&0&1 \\ 1&0&0&0&0&0&0&0 \\ 1&0&0&0&1&1&1&0 \\ 0&1&1&0&0&1&0&1 \\ 0&0&0&0&1&0&1&1 \\ 0&0&0&0&0&0&1&1 \\ 1&0&0&0&0&0&1&0 \end{bmatrix} * \begin{bmatrix} k_7 \\ k_6 \\ k_5 \\ k_4 \\ k_3 \\ k_2 \\ k_1 \\ k_0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \tag{14}$$

$$DEC_{stage7} = C\_de \cdot IM^{-1}(k)$$

$$= \begin{bmatrix} out\_DE_7 \\ out\_DE_6 \\ out\_DE_5 \\ out\_DE_4 \\ out\_DE_3 \\ out\_DE_2 \\ out\_DE_1 \\ out\_DE_0 \end{bmatrix} = \begin{bmatrix} 0&0&0&0&1&1&0&0 \\ 1&0&0&0&0&1&1&0 \\ 1&1&0&0&0&0&1&1 \\ 0&1&1&0&0&0&0&1 \\ 1&0&1&1&1&1&0&0 \\ 0&1&0&1&0&0&1&0 \\ 0&0&1&0&1&0&0&1 \\ 0&0&0&1&1&0&0&0 \end{bmatrix}$$

$$\cdot \left(\begin{bmatrix} 1&1&1&0&0&0&1&0 \\ 0&1&0&0&0&1&0&0 \\ 0&1&1&0&0&0&1&0 \\ 0&1&1&1&0&1&1&0 \\ 0&0&1&1&1&1&1&0 \\ 1&0&0&1&1&1&1&0 \\ 0&0&1&1&0&0&0&0 \\ 0&1&1&1&0&1&0&1 \end{bmatrix} \cdot \begin{bmatrix} k_7 \\ k_6 \\ k_5 \\ k_4 \\ k_3 \\ k_2 \\ k_1 \\ k_0 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 1&0&1&0&0&0&0&0 \\ 0&1&0&0&1&1&0&0 \\ 1&1&1&0&0&0&1&1 \\ 0&1&0&1&0&0&1&1 \\ 0&1&0&1&0&1&1&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&1&0&1&0&0&1 \\ 0&1&0&0&1&0&0&0 \end{bmatrix} \cdot \begin{bmatrix} k_7 \\ k_6 \\ k_5 \\ k_4 \\ k_3 \\ k_2 \\ k_1 \\ k_0 \end{bmatrix} \tag{15}$$

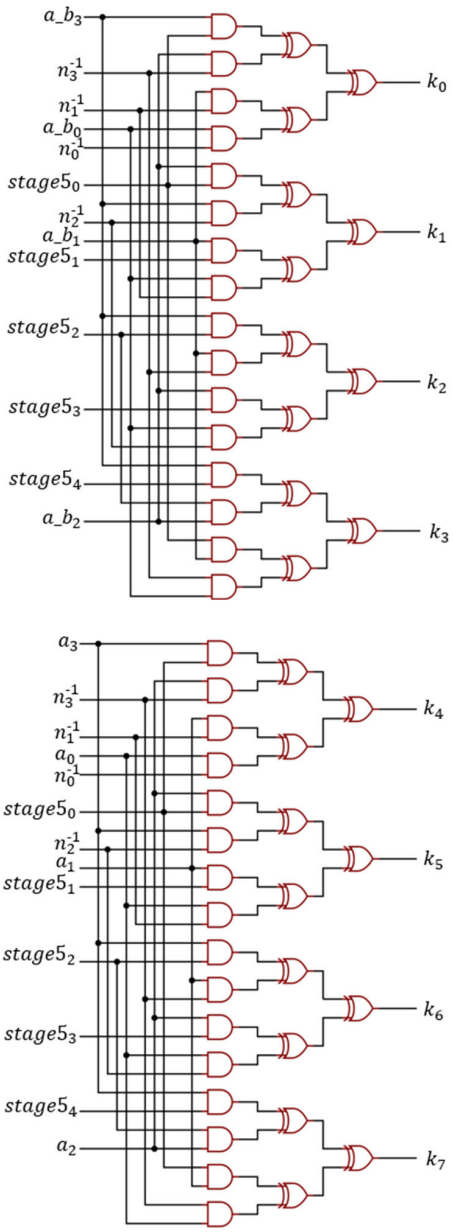

FIGURE 5. Schematic of stage 6.

E. PREPARATION OF MULTIPLICATION FOR VARIABLES

Stages 5 and 6 generate the result of an^{-1} and $(a+b)n^{-1}$ in (1). Because these variables are typically derived through multiplication operations, a simplification strategy by sharing identical calculation procedures—rather than using two separate processes— is a more optimal approach. In contrast to the output of stage 2, the production of this stage comprises 5-bit common terms that are required for deriving (1). The computations in this stage require numerous input variables, including the inversion result n^{-1} of $GF((2^2)^2)$, the coefficient a, and the summation of $a+b$ from preceding stages. Accordingly, this stage could be considered a preparation step for reusable calculation; thus, the critical path required for

generating complete multiplication results can be shortened. The equation of this stage is expressed in (12), and the corresponding schematic is presented in Fig. 4. Along with coefficients and inversion results, 12-bit data from the previous phase, a 17-bit register is required for pipeline storage, as shown in Fig. 1.

F. COMPLETE MULTIPLICATION FOR VARIABLE

Stage 6 generates 8-bit complete multiplication results, denoted as k in (13). As the preparation materials from the preceding stage are received, results of an^{-1} and $(a+b)n^{-1}$ in (1) are rendered. The equation for this stage is presented in (11), and the corresponding schematic is illustrated in Fig. 5. Notably, this stage entails the use of the same combinational logic but different inputs to compute an^{-1} and $(a+b)n^{-1}$ as the four most and four least significant bits for k, respectively. The critical path of this stage is as follows: 1 *AND* + 2 *XOR*.

G. UP MAPPING

Use one space after periods and (4). Similar to stage 1, stage 7 involves a control signal indicating whether an encryption or decryption process should be performed. The encryption process in this stage consists of inverse isomorphic mapping and affine transformation, and the decryption process involves inverse isomorphic mapping. The same multiplicative offset derived in stage 1 is adopted in this stage for encryption and decryption, where *C_en* is 88, and *C_de* is 50. The equations are expressed in (14) and (15), where IM^{-1} denotes inverse isomorphic mapping, and *AF* denotes affine transformation. The inverse mapping function in this stage requires 21 *XOR* and 4 *NOT* gates, and this stage also uses a multiplexer for selection after computation operations.

IV. EXPERIMENTAL RESULTS

The combined S-Box architecture proposed in this study was implemented using the Verilog hardware description language and synthesized using the Synopsys Design Compiler (Version Q-2019.12) with the TSMC 40-nm cell library following the standard design flow specified for ASICs. After gate-level synthesis, the power consumption of our proposed architecture was measured by the same tool (the result of report_power). The results of our proposed architecture were also verified by checking whether the output value was the same as the lookup table of all S-box 256 input values used in AES. All mentioned above were implemented under Linux.

Our proposed architecture involved non-pipeline and pipeline architectures, where the non-pipeline design could be achieved by removing registers in Fig.1. Simulations were conducted to compare the proposed architecture with the combined S-Box architectures presented in [13] and [16] in terms of cell area, maximum operating speed, and power consumption. In [13], a low-area, low-latency design without a pipeline architecture was presented, and in [16], a five-stage pipeline architecture with a higher operating speed than that of [13] was introduced. For the comparison of resource usage between the various architectures, the gate count (GC) in the 40-nm library was defined as cell area(μm^2)/0.68, where 0.68 is the cell area of a two-input *NAND* gate. The latency of a non-pipeline S-Box is one clock cycle; by contrast, the latency of a pipeline design is determined by the number of stages. Because the S-Box output is an 8-bit value, the corresponding throughput can be defined as *operating frequency* (*MHz*) × 8 (in Mbps). The ratio of throughput efficiency to area cost (also denoted as throughput–area efficiency), calculated as *throughput*(*Mbps*)/*gate count* (*GC*), was also assessed to evaluate the performance of the various architectures.

TABLE 1. Comparison of nonpipeline architecture for proposed S-box and design in [13] using TSMC 40-nm cell library when operating at 800 MHz.

	[13]	our design
frequency (MHz)	800	800
throughput (Gbps)	6.4	6.4
cell area (μm^2)	615.76	462.45
gate counts (GC)	905	679.67
latency (clock cycle)	1	1
latency (ns)	1.25	1.25
power(mW)	0.3895	0.2996
area-throughput efficiency (Mbps/GC)	7.07	9.42

TABLE 2. Comparison of nonpipeline architecture for proposed S-box and design in [13] using TSMC 40-nm cell library when operating at maximum frequency.

	[13]	our design
max. frequency (MHz)	1000	1250
throughput (Gbps)	8	10
cell area (μm^2)	850.95	832.13
gate counts (GC)	1250.67	1223
latency (clock cycle)	1	1
latency (ns)	1	0.8
power(mW)	0.6416	0.7149
area-throughput efficiency (Mbps/GC)	6.4	8.18

The experimental results of area complexity, power consumption and latency are listed in Table 1 to Table 4. Table 1 presents the experimental results regarding the performance of our S-Box architecture and that of the architecture presented in [13] when operated at the same operating frequency (800 MHz). The results revealed that our architecture exhibited a higher throughput–area efficiency, smaller area cost, and less power dissipation than the architecture presented in [13]. The lower power dissipation demonstrates the effectiveness of the logic optimization implemented in the proposed

TABLE 3. Comparison of pipeline architecture for proposed S-box and design in [16] using TSMC 40-nm cell library when operating at 2500 MHz.

	[16]	our design
frequency (MHz)	2500	2500
throughput (Gbps)	20	20
cell area (μm^2)	551.57	569.26
gate counts (GC)	810.67	836.67
latency (clock cycle)	5	7
latency (ns)	2	2.8
power(mW)	0.9034	1.1028
area-throughput efficiency (Mbps/GC)	24.67	23.9

TABLE 4. Comparison of pipeline architecture for proposed S-box and design in [16] using TSMC 40-nm cell library when operating at maximum frequency.

	[16]	our design
max. frequency (MHz)	3030.3	4347.83
throughput (Gbps)	24.24	34.78
cell area (μm^2)	632.77	702.17
gate counts (GC)	930	1031.99
latency (clock cycle)	5	7
latency (ns)	1.65	1.61
power(mW)	1.1897	2.0889
area-throughput efficiency (Mbps/GC)	26.07	33.70

architecture. As indicated in Table 2, the operating speed of the proposed architecture was higher than that of the architecture in [13] by 12.5%. The table also indicates that our architecture exhibited greater power consumption, which can be attributed to its higher frequency according to the formula $P_{dynamic} = V_{dd}^2 * C_L * frequency$.

The proposed S-Box architecture was also compared with that in [16], which is a state-of-the-art design in terms of maximum frequency and area efficiency. Table 3 shows the experimental results for both architectures when operated using identical timing constraints and the same operating frequency (2500 MHz). The two architectures exhibited similar performance levels, including performance in area cost and throughput efficiency. As presented in Table 4, the maximum frequency of the proposed architecture was higher than that of the architecture in [16] by 43.47%; this is because the proposed architecture reduces the critical path of the multiplication computations. Our architecture achieved a throughput of 34.78 Gbps when operated at the highest speed.

Moreover, 44 bits registers are used in [16], and 76 bits registers are used in the proposed design. The main difference between the two designs is the two more pipeline stages. Because of the usage of additional pipeline registers, the total cell area observed for our architecture was greater than that of the architecture in [16] by 10.97%, which proposed circuits consumed more power. Nevertheless, our architecture performed better in terms of throughput–area efficiency (9.42 Mbps/GC) than did the architecture in [16] (7.07 Mbps/GC). The proposed architecture required seven clock cycles to generate calculation results after receiving input data, representing greater latency. However, because the proposed architecture has a greater operating speed, the nanosecond latency (1.61 ns) was shorter than that of the architecture in [16] (1.65 ns). Because the operating speed of the proposed architecture can reach 4347.83 MHz, which indicates that the delay of each cycle is 0.23 ns, the 1.61-ns latency of our architecture can be calculated as 0.23(ns) $\times$ 7(cycles). The combined throughput–area efficiency and low latency demonstrate that the proposed architecture can feasibly meet the requirements of IoT systems.

Detailed comparisons of power consumption of the pipeline architectures are described in brief here; the comparisons were made in terms of the power consumption of the registers and combinational logic circuits. Internal cell power, switching power, and leakage power were also measured. Our architecture exhibited higher power consumption levels at maximum frequency, which can be attributed to its higher operating frequency. Comparing our architecture with the combined S-Box architecture in [16] when operated at the same frequency revealed that our architecture exhibited greater internal cell power consumption in the register group. This result is theoretically reasonable owing to the additional two stages in the pipeline. However, the combinational power group of the proposed S-Box architecture was smaller than that of the architecture in [16], indicating that the combinational logic design in our architecture is superior to that of the architecture in [16].

V. CONCLUSION

The S-Box, a critical component in AES, plays a vital role in ensuring the security of the encryption process. This paper presents a novel and efficient seven-stage S-Box pipeline architecture design to enhance the performance of the AES for high-throughput applications. The proposed architecture focuses on reducing the critical path of S-Box computations, which involve complex operations on GF polynomials, through advanced logic optimization techniques.

In the proposed architecture, dedicated multi-stage multiplications are employed for constants, squaring, and variables, utilizing tailored strategies. Simulation results demonstrate that the proposed architecture achieves an impressive throughput of 34.78 Gbps when operating at maximum speed. Despite the slight increase in cell area cost, the results also indicate higher area-throughput efficiency, measured in Mbps/GC, emphasizing the effectiveness of architecture. Moreover, the proposed architecture exhibits significant advantages in terms of reduced dynamic power consumption, validating the efficiency of the logic optimization process. These results underscore the suitability of the architectural

design for communication security applications, where high throughput and low latency are paramount.

Future research endeavors will focus on further reducing the power consumption of the S-Box by conducting an in-depth analysis of data transmission characteristics. Additionally, integrating the proposed architecture into a comprehensive AES design holds the potential for addressing security system performance challenges, such as bandwidth limitations during data access with memory, and improving the area complexity of the encryption and decryption process.

ACKNOWLEDGMENT

The authors would like to express their gratitude to the anonymous reviewers for their valuable comments and editorial suggestions, which improved the comprehension of the manuscript.

REFERENCES

[1] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying IoT security: An exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2702–2733, 3rd Quart., 2019, doi: 10.1109/COMST.2019.2910750.

[2] C. Liu, Y. Zhang, T. Zhang, X. Wu, L. Gao, and Q. Zhang, "High throughput vehicle coordination strategies at road intersections," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14341–14354, Dec. 2020, doi: 10.1109/TVT.2020.3029933.

[3] S. Safavat and D. B. Rawat, "On the elliptic curve cryptography for privacy-aware secure ACO-AODV routing in intent-based Internet of Vehicles for smart cities," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 5050–5059, Aug. 2021, doi: 10.1109/TITS.2020.3008361.

[4] T. M. Fernández-Caramés, "From pre-quantum to post-quantum IoT security: A survey on quantum-resistant cryptosystems for the Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6457–6480, Jul. 2020, doi: 10.1109/JIOT.2019.2958788.

[5] J. Daemen and V. Rijmen, "The Rijndael block cipher: AES proposal," in *Proc. 1st Candidate Conf. (AES)*, 1999, pp. 343–348.

[6] J. Daemen and V. Rijmen, *The Design of Rijndaels: AES—The Advanced Encryption Standard*. New York, NY, USA: Springer, 2002.

[7] W. Jiang, Z. Guo, Y. Ma, and N. Sang, "Measurement-based research on cryptographic algorithms for embedded real-time systems," *J. Syst. Archit.*, vol. 59, no. 10, pp. 1394–1404, Nov. 2013.

[8] J. Juremi, C. N. Mahendran, M. V. Naseri, and S. Sulaiman, "FlashSafe: USB flash drives encryption tool with AES algorithm," in *Proc. 14th Int. Conf. Develop. eSyst. Eng. (DeSE)*, Dec. 2021, pp. 537–540, doi: 10.1109/DeSE54285.2021.9719365.

[9] K. Thongkhome, C. Thanavijitpun, and S. Choomchuay, "A FPGA design of AES core architecture for portable hard disk," in *Proc. 8th Int. Joint Conf. Comput. Sci. Softw. Eng. (JCSSE)*, May 2011, pp. 223–228, doi: 10.1109/JCSSE.2011.5930124.

[10] A. Gielata, P. Russek, and K. Wiatr, "AES hardware implementation in FPGA for algorithm acceleration purpose," in *Proc. Int. Conf. Signals Electron. Syst.*, 2008, pp. 137–140, doi: 10.1109/ICSES.2008.4673377.

[11] C. Savalam and P. Korapati, "Implementation and design of AES S-box on FPGA," *Int. J. Res. Eng. Sci.*, vol. 3, pp. 9–14, Jan. 2015.

[12] A. Reyhani-Masoleh, M. Taha, and D. Ashmawy, "Smashing the implementation records of AES S-box," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2018, no. 2, pp. 298–336, May 2018.

[13] A. Reyhani-Masoleh, M. Taha, and D. Ashmawy, "New low-area designs for the AES forward, inverse and combined S-boxes," *IEEE Trans. Comput.*, vol. 69, no. 12, pp. 1757–1773, Dec. 2020.

[14] N. D. Parmar and P. Kadam, "Pipelined implementation of dynamic Rijndael S-box," *Int. J. Comput. Appl.*, vol. 111, no. 10, pp. 36–38, Feb. 2015.

[15] K.-L. Tsai, Y. -L. Huang, F.-Y. Leu, I. You, Y.-L. Huang, and C.-H. Tsai, "AES-128 based secure low power communication for LoRaWAN IoT environments," *IEEE Access*, vol. 6, pp. 45325–45334, 2018, doi: 10.1109/ACCESS.2018.2852563.

[16] Y. T. Teng, W. L. Chin, D. K. Chang, P. Y. Chen, and P. W. Chen, "VLSI architecture of S-box with high area efficiency based on composite field arithmetic," *IEEE Access*, vol. 10, pp. 2721–2728, 2022, doi: 10.1109/ACCESS.2021.3139040.

[17] X. Gao, E. Lu, L. Li, and K. Lang, "LUT-based FPGA implementation of SMS4/AES/Camellia," in *Proc. 5th IEEE Int. Symp. Embedded Comput.*, Oct. 2008, pp. 73–76, doi: 10.1109/SEC.2008.43.

[18] D. Canright, *A Very Compact S-Box for AES*, vol. 3659. Cham, Switzerland: Springer, 2005, pp. 441–455.

[19] R. Ueno, N. Homma, Y. Nogami, and T. Aoki, "Highly efficient GF(2^8) inversion circuit based on hybrid GF representations," *J. Cryptograph. Eng.*, vol. 9, no. 2, pp. 101–113, 2019.

[20] B. Rashidi, "Lightweight cryptographic S-boxes based on efficient hardware structures for block ciphers," *ISC Int. J. Inf. Secur.*, vol. 15, no. 1, pp. 137–151, 2023.

[21] B. Rashidi, "Lightweight 8-bit S-box and combined S-box/S-box^{-1} for cryptographic applications," *Int. J. Circuit Theory Appl.*, vol. 49, no. 8, pp. 2348–2362, 2021.

[22] B. Rashidi, "Compact and efficient structure of 8-bit S-box for lightweight cryptography," *Integr., VLSI J.*, vol. 76, pp. 172–182, Jan. 2021.

[23] *Specification for the Advanced Encryption Standard (AES)*, Standard FIPS PUB197, National Institute of Standards and Technology, Nov. 2001.

SHIH-HSIANG LIN received the B.S. and Ph.D. degrees in computer science and information engineering from the National Cheng Kung University, Tainan, Taiwan, in 2014 and 2018, respectively. His current research interests include image processing, very large-scale integrated chip design, and embedded systems.

JUN-YI LEE received the B.S. degree from the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, in 2021. His current research interests include information security, very large-scale integrated chip design, and embedded systems.

CHIA-CHOU CHUANG received the B.S. degree from the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, in 2022, where he is currently pursuing the Ph.D. degree in program on integrated circuit design of AISSM. His current research interests include information security, very large-scale integrated chip design, and embedded systems.

NARN-YIH LEE received the Ph.D. degree in information engineering from the National Cheng Kung University, Taiwan, in 1996. He is currently a Professor with the Computer and Network Center, National Cheng Kung University. His research interests include cryptography, blockchain, network security, post quantum cryptography, and chip security.

PEI-YIN CHEN (Senior Member, IEEE) received the B.S. degree from the National Cheng Kung University, Tainan, Taiwan, in 1986, the M.S. degree from Pennsylvania State University, University Park, PA, USA, in 1990, and the Ph.D. degree from the National Cheng Kung University, in 1999, all in electrical engineering. He is currently a Professor with the Department of Computer Science and Information Engineering, National Cheng Kung University. His research interests include very large-scale integration chip design, video compression, fuzzy logic control, and gray prediction.

WEN-LONG CHIN (Senior Member, IEEE) received the B.S. and Ph.D. degrees in electronics engineering from the National Chiao Tung University, Hsinchu, Taiwan, and the M.S. degree in electrical engineering from the National Taiwan University, Taipei, Taiwan. He is currently a Professor with the Department of Engineering Science, National Cheng Kung University. Before holding the faculty position, he was with Hsinchu Science Park, Taiwan, for over 11 years, leading communication and network ASIC designs. His research interests include ASIC design and DSP for communications and networking. He served as an Associate Editor for IEEE Access and *EURASIP Journal on Wireless Communications and Networking*. He currently serves as a Technical Editor for *IEEE Wireless Communications Magazine*.