IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 12, 2021, accepted November 19, 2021, date of publication December 28, 2021,
date of current version January 7, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3139040

VLSI Architecture of S-Box With High Area Efficiency Based on Composite Field Arithmetic

YOU-TUN TENG[1], WEN-LONG CHIN[1], (Senior Member, IEEE), DENG-KAI CHANG[1], PEI-YIN CHEN[2], (Senior Member, IEEE), AND PIN-WEI CHEN[1]

[1]Department of Engineering Science, National Cheng Kung University, Tainan 70101, Taiwan
[2]Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan 70101, Taiwan

Corresponding author: Wen-Long Chin (wlchin@mail.ncku.edu.tw)

This work was supported in part by the Ministry of Education, Taiwan, under Grant PEE1100748.

ABSTRACT This work aims at optimizing the hardware implementation of the SubBytes and inverse SubBytes operations in the advanced encryption standard (AES). To this, the composite field arithmetic (CFA) is employed to optimize all building blocks in S-box (and inverse S-box) of SubBytes (and inverse SubBytes) transformation. A joint design of S-box and inverse S-box is also proposed to further enhance the area efficiency. Specifically, the area of multiplier in the Galois composite field, GF($(2^2)^2$), is reduced. The squaring and multiplication with constant λ in GF($(2^2)^2$) are combined and optimized as well. Moreover, the multiplicative inversion in GF($(2^2)^2$) is manually optimized. Furthermore, the S-box and inverse S-box are combined and optimized using the pre_processing and post_processing modules. To increase the throughput, a balanced and pipelined architecture is derived. Using the proposed architecture, a throughput of 5.79 Gbps for the S-box can be achieved on Virtex-6 XC6VLX240T and 10% better than the conventional work. According to the ASIC implementation result, the proposed design can still achieve the highest area efficiency and approximately 30% better than conventional works using TSMC 90nm process.

INDEX TERMS Advanced encryption standard (AES), composite field arithmetic (CFA), S-box, VLSI architecture.

I. INTRODUCTION

In 2001, National Institute of Standard and Technology (NIST) invited proposals for new algorithm of the advanced encryption standard (AES) to replace the old data encryption standard (DES). The Rijndael algorithm [1], designed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen, was finally selected as the AES specification and became a FIPS standard [2]. Nowadays, AES algorithm is the most popular symmetric encryption algorithm. With the rapid development of transmission technology in communication networks, the data throughput has increased significantly. Therefore, implementing a low-cost but high-throughput AES engine has become an essential issue.

Compared to the software solution [3], the hardware implementation is more suitable for high-throughput data applications. Among hardware implementations, the nonlinear SubBytes transformation realized using look-up tables (LUTs) [4]–[6] requires a large area compared to those using the composite field arithmetic (CFA) [7]–[20]. The works [7]–[11] studied low-area implementations based on the fully combinational logic. The work [12] presented a S-box based on the multiplexer. The work [13] evaluated 5-, 6-, and 7-stages pipelined S-box based on the CFA. By contrast, the studies, [14] and [15], proposed a 4-stage pipelined S-box. The work [16] adopted the pre-computation technique with three block design of S-box. In [17], modified MUX based S-box was introduced in AES to reduce the area without affecting the throughput. The study [18] proposed a new compact S-box. In [19], a joint AES encryption/decryption with a 7-stage pipeline using the CFA was proposed. The study [20] proposed a parallel pipelined architecture to obtain high data throughput. In [21], a compact AES through optimizing the mix columns and inverse mix columns transformations was obtained. The work [22] shared the circuitry of AES and SHA-3. A 60 Gbps reconfigurable cryptographic processor, including AES and SHA algorithms, was proposed in [23]. The authenticated encryption circuits for the operation modes of AES, AES-CCM and AES-XEX, were considered in [24] and [25], respectively.

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

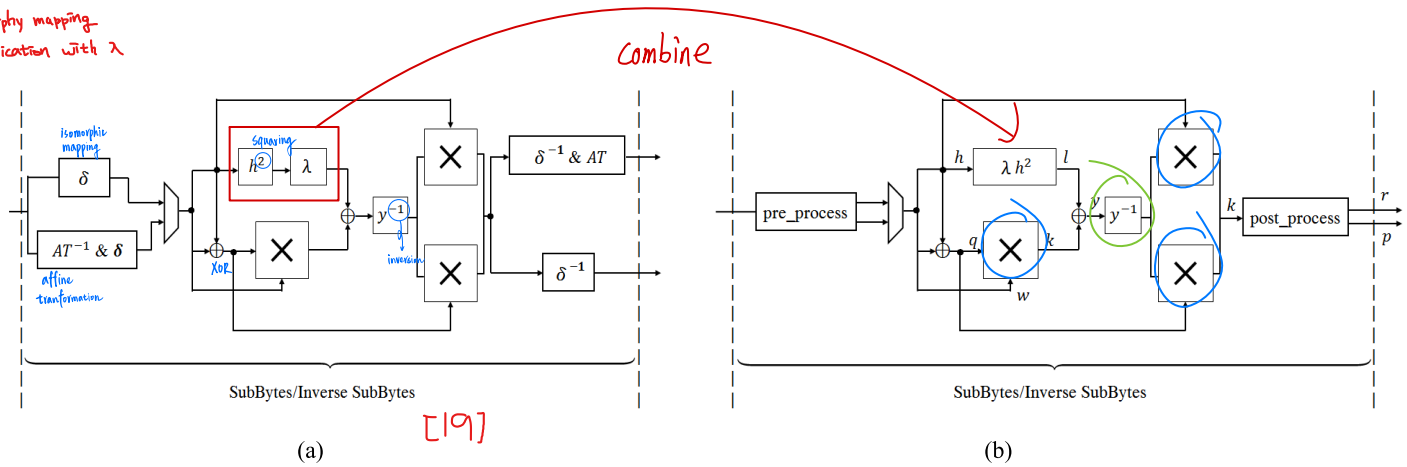


FIGURE 1. Architectures of SubBytes and inverse SubBytes operations, (a) conventional and (b) proposed approaches.

Recently, the research direction of AES towards the countermeasures for the side-channel attack (SCA) [26]–[36]. The side-channel attack is based on the correlation power analysis through monitoring the chip supply current signature or electromagnetic emissions. The work [26] proposed a 16-bit serial AES-128 hardware accelerator with randomized byte-order shuffling through heterogeneous S-boxes. The works [27], [28] adopted a SCA-resistant methodology based on the machine learning. The work [29] proposed to use the random fast voltage dithering against the SCA, while the works, [30] and [31], used the asynchronous logic and memristor, respectively. A masked S-box was devised in [32] to cope with the SCA. The work [33] proposed a fast power leakage simulation method for hardware-implemented cryptographic ICs. The works, [34] and [35], proposed the collision fault information and incremental fault analysis for the attacks on AES, respectively. In [36], the SCA is evaluated for the operation mode of XTS-AES.

The Internet of Things (IoT) paves another way for the AES research. The works, [37]–[40], proposed lightweight or area-efficient AES cryptographic circuits. In [41], lightweight AES, PRESENT, and GIFT ciphers were evaluated on FPGA.

Since there is no network that is immune to attacks, an efficient network security system is essential to protecting client data. Given the rapid growth of networks and data centers, and high demand for the greatest possible bandwidth, a new AES circuit is designed in this study with the highest performances, in terms of throughput and area efficiency, for the core networks and data centers.

The SubBytes and inverse SubBytes operations usually occupy the largest area and exhibit the longest path delay in AES. Thus, this work aims to develop a new VLSI architecture for the joint S-box and inverse S-box with a high area efficiency based on the CFA. More specifically, this study provides several contributions outlined below. (1) The area of multiplier in the Galois composite field, GF($(2^2)^2$), is reduced, where GF(·) denotes the Galois field. (2) The squaring and multiplication with constant λ in GF($(2^2)^2$) are combined and optimized. (3) The multiplicative inversion in GF($(2^2)^2$) is manually optimized. (4) The S-box and inverse S-box are combined and optimized as well. A pre_processing module is proposed to share the resources in isomorphic mapping and inverse affine transformation in tandem with isomorphic mapping of the S-box and inverse S-box, respectively. Likewise, a post_processing is also proposed to combine the inverse isomorphic mapping in tandem with affine transformation and inverse isomorphic mapping of the S-box and inverse S-box, respectively.

The rest of this paper is organized as follows. Section II briefly introduces the AES algorithm. Section III presents the proposed SubBytes and inverse SubBytes operations. Section IV demonstrates the implementation results. Finally, Section V draws conclusions.

II. ADVANCED ENCRYPTION STANDARD

AES [1] is the most popular symmetric encryption algorithm that can process 128 bits at a time. The same key is used for both encryption and decryption. It divides the plaintext into a fixed block size of 128 bits. Several rounds of repeated encryption and decryption processes are performed on the plaintext and ciphertext, respectively.

The AES is an iterative algorithm and uses a round function repeatedly. In encryption, each round is composed of four different byte-oriented processing steps: substitute bytes (SubBytes), shift rows (ShiftRows), mix columns (MixColumns), and add round key (AddRoundKey), while the last round does not contain the MixColumns step. In decryption, each round is also composed of four different byte-oriented processing steps: inverse substitute bytes (InvSubBytes), inverse shift rows (InvShiftRows), inverse mix columns (InvMixColumns), and add round key (AddRoundKey), while the last round does not contain the InvMixColumns step.

SubBytes is designed using the multiplicative inverse of input element over GF(2^8) and then applying an affine transformation on the multiplicative inverse. This step is the only non-linear transformation in AES. For the decryption, on the contrary, InvSubBytes is designed applying an inverse affine transformation firstly. The multiplicative inverse over GF(2^8)

of the output of the inverse affine transformation is the final output of the inverse S-box.

The proposed design can be integrated into AES, which has been included in many communication standards, such as Internet engineering task force (IETF) and requests for comments, international organization for standardization (ISO), third-generation partnership project (3GPP), and IEEE standards. Therefore, it is a popular and secure encryption algorithm for media access control (MAC) layer and higher layers, such as Internet protocol (IP). Actually, AES can be applied in any applications for encrypting the digital contents.

III. PROPOSED SubBytes AND INVERSE SubBytes OPERATIONS

To improve the area efficiency of AES implementation, The architecture of the SubBytes and inverse SubBytes operations is proposed in Fig. 1, where Fig. 1(a) and Fig. 1(b) are conventional [19] and proposed architectures, respectively. In Fig. 1(a), δ, AT, $(\cdot)^{-1}$, $(\cdot)^2$, λ, $\times$, and $\oplus$ denote the isomorphic mapping, affine transformation, inversion, squaring, multiplication with λ, multiplication, and addition, respectively. Proposed sub-blocks are introduced below. Notably, the adder in Galois field corresponds to the XOR operation. Therefore, the adders in Fig. 1 can be simply described in the register-transfer level (RTL) codes using the Verilog XOR operator.

A. MULTIPLICATION IN GF($(2^2)^2$)

An element in GF($(2^2)^2$) is denoted by $k = \{k_3k_2k_1k_0\}_2$, where $k_i \in \{0, 1\}$, $i = 0, 1, 2, 3$. Let $k_H = \{k_3k_2\}_2$ and $k_L = \{k_1k_0\}_2$, then k can be written as

$$k = k_H x + k_L. \tag{1}$$

Let the product $k = qw$, where q and w are also elements in GF($(2^2)^2$). According to the irreducible polynomial, $x^2 + x + \varphi$, where $\varphi = \{10\}_2$, one has

$$\begin{aligned} k &= qw \\ &= (q_H x + q_L)(w_H x + w_L) \\ &= q_H w_H x^2 + (q_H w_L + q_L w_H)x + q_L w_L \\ &= (q_H w_H + q_H w_L + q_L w_H)x + q_H w_H \varphi \\ &\quad + q_L w_L. \end{aligned} \tag{2}$$

Comparing (1) and (2), one has

$$\begin{cases} k_H = q_H w_H + q_H w_L + q_L w_H \\ k_L = q_H w_H \varphi + q_L w_L. \end{cases} \tag{3}$$

According to the irreducible polynomial, $x^2 + x + 1$, in GF(2^2), one can further reduce (3) by

$$\begin{aligned} q_H w_H &= (q_3 x + q_2)(w_3 x + w_2) \\ &= (q_3 w_3 + q_3 w_2 + q_2 w_3)x + q_3 w_3 + q_2 w_2. \end{aligned} \tag{4}$$

Similarly, it can be shown that

$$q_H w_L = (q_3 w_1 + q_2 w_1 + q_3 w_0)x + q_3 w_1 + q_2 w_0, \tag{5}$$

$$q_L w_H = (q_1 w_3 + q_1 w_2 + q_0 w_3)x + q_1 w_3 + q_0 w_2, \tag{6}$$

$$\begin{aligned} q_H w_H \varphi &= (q_2 w_2 + q_2 w_3 + q_3 w_2)x + q_3 w_3 + q_2 w_3 \\ &\quad + q_3 w_2, \end{aligned} \tag{7}$$

$$q_L w_L = (q_1 w_1 + q_0 w_1 + q_1 w_0)x + q_1 w_1 + q_0 w_0. \tag{8}$$

Since $k_H = k_3 x + k_2$ and $k_L = k_1 x + k_0$, substituting (4), (5), (6), (7), (8) into (3) and extracting common factors in k_3, k_2, k_1, k_0, one finally has

$$\begin{cases} k_3 = (q_3 + q_2)(w_3 + w_2) + (q_3 + q_2)(w_1 + w_0) \\ \quad + (q_1 + q_0)(w_3 + w_2) + q_2 w_0 + q_0 w_2 + q_2 w_2 \\ k_2 = q_3 w_3 + q_3 w_1 + q_1 w_3 + q_2 w_2 + q_2 w_0 + q_0 w_2 \\ k_1 = (q_3 + q_2)(w_3 + w_2) + q_3 w_3 \\ \quad + (q_1 + q_0)(w_1 + w_0) + q_0 w_0 \\ k_0 = (q_3 + q_2)(w_3 + w_2) + q_2 w_2 + q_1 w_1 + q_0 w_0. \end{cases} \tag{9}$$

It must be emphasized here that + in Galois field corresponds to the bitwise XOR operation. As presented in (9), only $(q_3 + q_2)$, $(q_1 + q_0)$, $(w_3 + w_2)$, $(w_1 + w_0)$, q_3w_3, q_2w_2, q_1w_1, q_0w_0, and $(q_2w_0 + q_0w_2)$ are needed to implement the multiplication in GF($(2^2)^2$). As displayed in Fig. 2, the proposed multiplication requires 18 XOR and 12 AND gates. Its critical path has 4 XOR and 1 AND gates. In comparison, the multiplier in GF($(2^2)^2$) of [19] requires an area of 21 XOR and 9 AND gates, and critical path of 4 XOR and 1 AND gates.

B. SQUARING AND MULTIPLICATION WITH CONSTANT λ IN GF($(2^2)^2$)

In this section, two submodules, i.e., squaring and multiplication with constant $\lambda = \{1100\}_2$ in GF($(2^2)^2$), are combined and optimized to improve the area efficiency. Let h be an element in GF($(2^2)^2$), its squaring and multiplication with constant λ can be written as

$$l = \lambda h^2. \tag{10}$$

After mathematical derivations (shown in Appendix), one finally has

$$\begin{cases} l_3 = h_2 + h_1 + h_0 \\ l_2 = h_3 + h_0 \\ l_1 = h_3 \\ l_0 = h_2 + h_3 \end{cases} \tag{11}$$

where l_i and h_i, $i = 0, 1, 2, 3$, are bits of elements, l and h, in GF($(2^2)^2$), respectively. As displayed in Fig. 3, the joint squaring and multiplication with constant λ requires 4 XOR gates, and its critical path has 2 XOR gates. The circuit with the same function in [19] requires an area and critical path of 7 XOR and 4 XOR gates, respectively.

C. MULTIPLICATIVE INVERSION IN GF($(2^2)^2$)

The multiplicative inversion of y in GF($(2^2)^2$), denoted by $y^{-1} = \{y_3^{-1}y_2^{-1}y_1^{-1}y_0^{-1}\}$, is listed in Table 1.

There are basically three distinct ways to implement the inversion. 1) Method A: According to [19], the mathematical

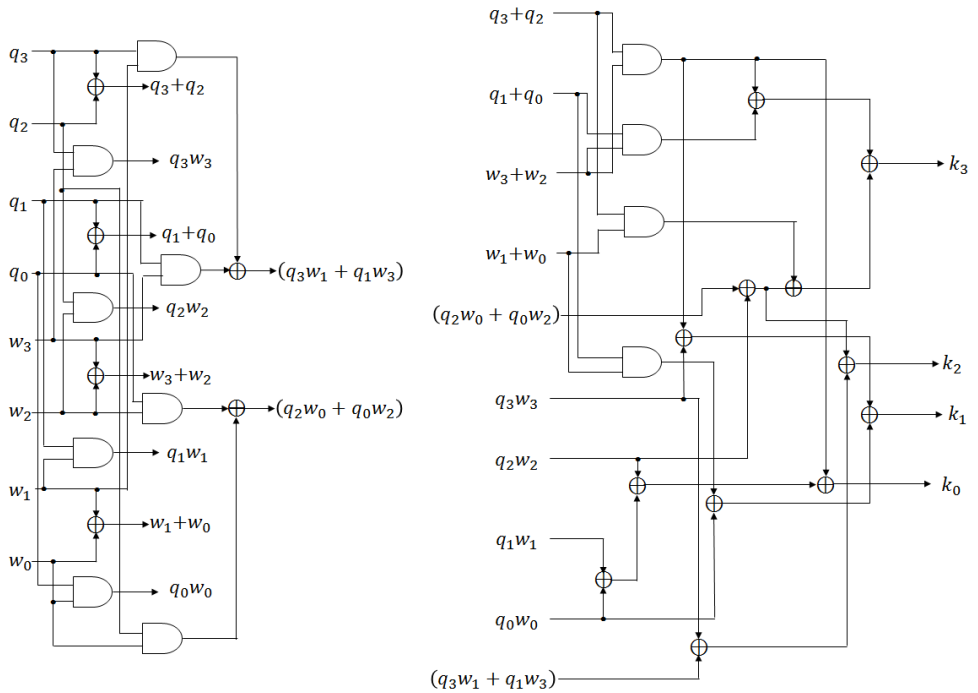


FIGURE 2. The schematic of proposed multiplication in GF($(2^2)^2$).

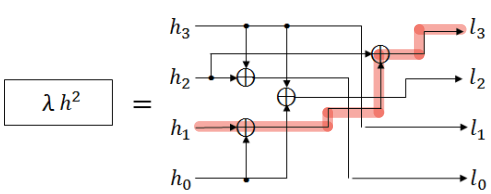


FIGURE 3. The schematic of proposed joint squaring and multiplication with constant λ in GF($(2^2)^2$).

TABLE 1. The multiplicative inversion in GF($(2^2)^2$).

Input $\{y_3y_2y_1y_0\}$	Output $\{y_3^{-1}y_2^{-1}y_1^{-1}y_0^{-1}\}$
0000	0000
0001	0001
0010	0011
0011	0010
0100	1111
0101	1100
0110	1001
0111	1011
1000	1010
1001	0110
1010	1000
1011	0111
1100	0101
1101	1110
1110	1101
1111	0100

equations of the inversion in GF($(2^2)^2$) can be expressed as

$$\begin{cases} y_3^{-1} = y_3 + y_3y_2y_1 + y_3y_0 + y_2 \\ y_2^{-1} = y_3y_2y_1 + y_3y_2y_0 + y_3y_0 + y_2 + y_2y_1 \\ y_1^{-1} = y_3 + y_3y_2y_1 + y_3y_1y_0 + y_2 + y_2y_0 + y_1 \\ y_0^{-1} = y_3y_2y_1 + y_3y_2y_0 + y_3y_1 + y_3y_1y_0 + \\ \quad y_3y_0 + y_2 + y_2y_1 + y_2y_1y_0 + y_1 + y_0 \end{cases} \quad (12)$$

where + in Galois field corresponds to the bitwise XOR operation. 2) Method B: The inversion can intuitively be implemented using a table, such as being inferred using the **case** statement in Verilog hardware description language (HDL). 3) Method C: The Boolean equations can be manually derived using the Karnaugh map (omitted here), and they are

$$\begin{cases} y_3^{-1} = \overline{y_3}y_2 + y_2\overline{y_1}y_0 + y_2y_1\overline{y_0} + y_3\overline{y_2}\,\overline{y_0} \\ y_2^{-1} = \overline{y_3}y_2\overline{y_1} + y_3y_2 + y_3\overline{y_2}y_0 \\ y_1^{-1} = \overline{y_1}\,\overline{y_3}y_2\overline{y_0} + \overline{y_1}y_3\overline{y_2} + y_3y_0\overline{y_1} + y_3y_0\overline{y_2} \\ \quad + \overline{y_3}y_1\overline{y_2} + \overline{y_3}y_1y_0 \\ y_0^{-1} = \overline{y_3}y_1\overline{y_0} + \overline{y_3}y_1y_2 + \overline{y_2}y_0y_3y_1 + \overline{y_2}y_0\overline{y_3}\,\overline{y_1} \\ \quad + y_2\overline{y_0} \end{cases} \quad (13)$$

where the $\overline{(\cdot)}$ and + respectively denote the bitwise NOT and OR. In contrast to conventional approach [19] directly derived from the mathematical equations using XOR gates, i.e., "+" in GF($(2^2)^2$), one can manually optimize the inversion using simpler gates such as AND, OR, and NOT gates. Apparently, (13) is better than (12) because "+" in (13) is an OR gate, while that in (12) is a XOR gate, and the number of "+" operators in (13) is less than that in (12).

The implementation results of three methods are analyzed in Table 2, where the gate and critical path equivalents are represented by NAND and NOT gates. As presented, Method C has the best area efficiency.

D. Pre_processing AND Post_processing DESIGNS

The post_processing can output the results of inverse isomorphic mapping in tandem with affine transformation for the S-box, and inverse isomorphic mapping for the

TABLE 2. Total gates, critical path, and their equivalents for implementing multiplicative inversion in GF($(2^2)^2$) of three methods.

	Total gates	Total gates equivalent	Critical path	Critical path equivalent
Method A	14 XOR +9 AND	65 NAND + 9 INV	3 XOR +2 AND	11 NAND +2 INV
Method B	60 MUX	180 NAND +60 INV	4 MUX	8 NAND +4 INV
Method C	5 INV + 7 OR + 2 AND + 9 NAND + 1 NOR + 1 XOR + 1 XNOR	27 NAND +16 INV	1 XNOR + 1 OR + 2 INV + 1 NAND +1 NOR +1 AND	7 NAND +5 INV

inverse S-box. It can be observed that the input of the inverse isomorphic mapping and the inverse isomorphic mapping in tandem with affine transformation are the same, and the outputs of them require additions in GF($(2^2)^2$). By sharing the resources involved in the S-box and inverse S-box, the output, $p = \{p_7p_6p_5p_4p_3p_2p_1p_0\}_2$, of the inverse isomorphic mapping and the output, $r = \{r_7r_6r_5r_4r_3r_2r_1r_0\}_2$, of the inverse isomorphic mapping in tandem with affine transformation can be respectively rewritten as

$$\begin{cases} p_7 = t_5 + k_5 + k_1 \\ p_6 = k_6 + k_2 \\ p_5 = t_4 + k_1 \\ p_4 = t_4 + k_4 + t_1 \\ p_3 = t_3 + k_3 + t_1 \\ p_2 = k_7 + t_2 + t_1 \\ p_1 = t_3 \\ p_0 = t_8 + k_4 \end{cases} \tag{14}$$

and

$$\begin{cases} r_7 = t_6 + k_3 \\ r_6 = t_5 + t_3 + 1 \\ r_5 = t_6 + 1 \\ r_4 = t_7 + k_4 + k_1 \\ r_3 = t_1 + k_0 \\ r_2 = t_8 + t_2 \\ r_1 = t_7 + 1 \\ r_0 = t_5 + t_1 + k_0 + 1 \end{cases} \tag{15}$$

where "+" is the addition in GF($(2^2)^2$), $k = \{k_7k_6k_5k_4k_3k_2k_1k_0\}_2$ denotes the input of post_processing, $t_8 = t_4 + t_0$, $t_7 = k_7 + k_0$, $t_6 = k_7 + k_2$, $t_5 = k_7 + k_6$, $t_4 = k_6 + k_5$, $t_3 = k_5 + k_4$, $t_2 = k_4 + k_3$, $t_1 = k_2 + k_1$, and $t_0 = k_2 + k_0$. The proposed post_processing requires 25 XOR gates, and its critical path has 3 XOR gates. Compared to the separate circuits of inverse isomorphic mapping and inverse isomorphic mapping in tandem with affine transformation in [19], which requires 32 XOR gates with critical path of 4 XOR gates, the optimized circuit needs less area and shorter critical path.

The pre_processing can be similarly designed and omitted here. It has 22 XOR gates with critical path of 3 XOR gates. By contrast, the conventional design requires 32 XOR gates with critical path of 4 XOR gates.

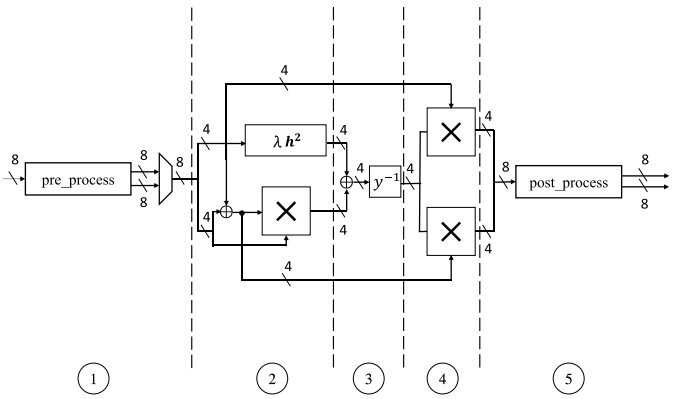


FIGURE 4. Five-stage pipelined joint S-box and inverse S-box.

E. DESIGN CHARACTERISTICS OF SUB-BLOCKS AND PIPELINED DESIGN

To obtain a balanced design in different pipelining stages without inserting too many pipelined registers, a 5-stage pipelined joint S-box and inverse S-box is devised, as that shown in Fig. 4. Dashed lines represent the positions registers are placed. Hence, there are 11 4-bit registers inserted in the pipelined architecture. In this design, the critical path with 4 XOR and 1 AND gates is in the second stage.

IV. IMPLEMENTATION RESULTS

First, the proposed architecture of S-box using various FPGA devices is evaluated, such as Virtex-6 XC6VLX240T, Virtex-5 XC5VLX20T, Virtex-4 XC4VF100, and Spartan-3 XC3S200, that were adopted in conventional pipelined designs in literature. The proposed design is described using the Verilog HDL. The development system is Xilinx ISE Design Suite 14.7.

The implementation results about the FPGA implementation are displayed in Table 3. As presented, the proposed architecture of S-box has the highest throughput and area efficiency. Using the proposed architecture on Virtex-6, a throughput of 5.79 Gbps for the S-box can be achieved. Compared to that of [14], the area efficiency of proposed implementation can increase 10% on Virtex-6 xc6vlx240t. Moreover, it should be emphasized here that conventional designs focus only on pure S-box. Therefore, only the S-box is assessed in Table 3. Beyond that, the proposed VLSI architecture can share the resources in S-box and inverse S-box, i.e., isomorphic mapping and inverse affine transformation in pre_processing, and inverse isomorphic mapping and affine transformation in post_processing.

TABLE 3. Comparison of different architectures on Xilinx FPGAs.

	Devices	Slices	Max. Frequency (Mhz)	Throughput (Gbps)	Throughput/Area (Mbps/Slice)
[14]	Virtex-6 xc6vlx240t	32	696.37	5.57	170
This work	Virtex-6 xc6vlx240t	31	724.638	5.79	187
[12]	Virtex-5 xc5vlx20t	36	571.91	4.57	126
[16]	Virtex-5 xc5vlx50	31	512.821	4.102	132
[17](non pipeline)	Virtex-5 xc5vlx20t	34	303.79	2.430	71
[17](pipeline)	Virtex-5 xc5vlx20t	37	571.91	4.575	124
[18](non pipeline)	Virtex-5 xc5vlx20t	32	523.56	4.188	131
[18](pipeline)	Virtex-5 xc5vlx20t	35	617.25	4.938	141
This work	Virtex-5 xc5vlx20t	17	644.330	5.154	303
[15]	Virtex-4 xc4vf100	45	209.61	1.68	37
This work	Virtex-4 xc4vf100	48	549.753	4.39	91
[13]	Spartan-3 xc3s200	69	327.22	2.62	38
This work	Spartan-3 xc3s200	63	338.295	2.71	43

TABLE 4. Comparison of different architectures using ASICs, where Enc represents the realization of only S-box and Enc&Dec represents the realization of joint S-box and inverse S-box.

	Architecture	Technology	Area (μm^2)	Max. Frequency (Mhz)	Throughput (Gbps)	Throughput/Area (Mbps/μm^2)
[8]	Enc	TSMC 90nm	2149.26	454.54	3.636	1.69
[8]	Enc&Dec	TSMC 90nm	2747.61	333.33	2.666	0.971
[9]	Enc	TSMC 90nm	1932.64	370.37	2.962	1.533
[10]	Enc	TSMC 90nm	2016.60	344.83	2.758	1.367
[7]	Enc	TSMC 90nm	2299.55	285.71	2.285	0.994
[7]	Enc&Dec	TSMC 90nm	2978.34	263.16	2.105	0.707
[11]	Enc&Dec	TSMC 90nm	3451.08	281.69	2.222	0.652
[19]	Enc	TSMC 90nm	3007.27	1000.00	8.000	2.660
[19]	Enc&Dec	TSMC 90nm	3363.59	909.09	7.272	2.162
This work	Enc	TSMC 90nm	2769.48	1204.82	9.639	3.480
This work	Enc&Dec	TSMC 90nm	3067.95	1075.27	8.602	2.804
[19]	Enc	TSMC 40nm	567.91	2857.14	22.857	40.248
[19]	Enc&Dec	TSMC 40nm	638.44	2631.58	21.053	32.975
This work	Enc	TSMC 40nm	528.44	3333.33	26.667	50.463
This work	Enc&Dec	TSMC 40nm	593.31	3125.00	25.000	42.136

Next, regarding the ASIC implementation in Table 4 based on the TSMC 90nm cell library and the synthesis tool of Synopsys Design Compiler, at the maximum achievable clock rate for each architectures, the proposed design can still achieve the highest area efficiency and approximately 30% better than conventional works using TSMC 90nm process. Notably, the works [7]–[11] proposed fully combinational circuits for the S-box (and inverse S-box). Therefore, they can achieve smaller areas than this work and [19]. To further compare this work and [19], they are also synthesized using TSMC 40nm in Table 4. As presented, in terms of the area efficiency, this work is still 25.5% better than [19] using TSMC 40nm process.

The comparative algorithms are described in brief here. The works [7], [8], and [10] use the normal basis to establish the tower field and then map the elements of the Galois field GF(2^8) to this field. By contrast, the proposed work, [9], [11], and [19] use the composite field representation of GF((2^4)^2) to construct the S-box. However, different optimization approaches are adopted to implement the circuits. The work [9] proposed logic-minimization algorithm and searching for optimum transformation matrices. By contrast, our approach is to combine and optimize each modules through mathematical reduction. The work [11] proposed to combine the S-box and the inverse S-box through multiplexers. However, this work proposes the pre_processing module to optimize the inverse affine transformation and isomorphic mapping, and the post_processing to integrate the affine transformation and inverse isomorphic mapping. The work [19] proposed an efficient S-box using pipelining. While this work further optimizes all building blocks of S-box and inverse S-box, such as multiplication, squaring and multiplication with constant λ, multiplicative inversion, and pre_processing and post_processing. Moreover, the places registers are inserted in this work are different from those of [19]. Therefore, the number of required registers of this work can be reduced.

TABLE 5. Detailed analytical results of [19].

Module	Total gates	Total gates equivalent	Critical path	Critical path equivalent
δ	12 XOR	48 NAND	4 XOR	12 NAND
h^2	4 XOR	16 NAND	2 XOR	6 NAND
λ	3 XOR	12 NAND	2 XOR	6 NAND
$\times$	21 XOR+ 9 AND	93 NAND+ 9 INV	4 XOR+ 1 AND	13 NAND+ 1 INV
y^{-1}	14 XOR+ 9 AND	65 NAND+ 9 INV	3 XOR+ 2 AND	11 NAND+ 2 INV
δ^{-1}	13 XOR	52 NAND	4 XOR	12 NAND
$\delta^{-1}\&AT$	19 XOR	76 NAND	3 XOR	12 NAND
$AT^{-1}\&\delta$	18 XOR	72 NAND	4 XOR	12 NAND
S-box	154 XOR+ 36 AND+ 8 MUX	676 NAND+ 44 INV	21 XOR+ 4 AND+ 1 MUX	69 NAND+ 5 INV

TABLE 6. Detailed analytical results of proposed design.

Module	Total gates	Total gates equivalent	Critical path	Critical path equivalent
pre_process	22 XOR	88 NAND	4 XOR	12 NAND
λh^2	4 XOR	16 NAND	2 XOR	6 NAND
$\times$	18 XOR+	84 NAND	4 XOR	13 NAND +
	12 AND	12 INV	1 AND	1 INV
y^{-1}	1 XOR+	27 NAND+	1 XNOR+	7 NAND +
	1 XNOR+	16 INV	2 INV+	5 INV
	5 INV+		1 AND+	
	2 AND+		1 OR+	
	7 OR+		1 NAND+	
	9 NAND+		1 NOR	
	1 NOR			
post_process	25 XOR	100 NAND	4 XOR	12 NAND
S-box	106 XOR+	527 NAND+	19 XOR +	65 NAND +
	1 XNOR+	52 INV	3 AND +	8 INV
	38 AND+		1 OR +	
	7 OR+		2 MUX	
	5 INV+			
	9 NAND+			
	1 NOR+			
	8 MUX			

To analyze the proposed design and [19] with the highest throughput and area efficiency in literature, the detailed time complexity and the data in the table are given in new Tables 5 and 6. As presented, the area and timing of the proposed design are better than those in [19].

V. CONCLUSION AND OUTLOOKS

A new pipelined VLSI architecture for the joint S-box and inverse S-box using the CFA in GF($(2^2)^2$) is proposed. First, excluding the adder, all building blocks in the SubBytes and inverse SubBytes, such as multiplier, multiplicative inversion, squaring and multiplication with constant λ, isomorphic function and inverse affine transformation, and inverse isomorphic function and affine transformation, are optimized from the algorithm aspect. Next, to focus on VLSI implementation, the schematics of the multiplier and squaring and multiplication with constant λ are plotted in Figs. 2 and 3, respectively. Besides, the Boolean equations of the multiplicative inversion in GF($(2^2)^2$) and post_processing are written as (13) and (14) (and (15)), respectively. Therefore, the proposed VLSI design can be simply implemented and replicated. Finally, the superiority of the proposed design is validated using both FPGA and ASIC implementations. Our future work will integrate the proposed S-box and inverse S-box into the whole AES circuit.

APPENDIX

By the multiplication (2) in GF($(2^2)^2$), and knowing that $h = h_H x + h_L$ and $\lambda = \lambda_H x + \lambda_L$, (10) can be written as

$$\begin{aligned} l &= (\lambda_H x + \lambda_L)(h_H x + h_L)(h_H x + h_L) \\ &= (\lambda_H x + \lambda_L)(h_H^2 x + h_H^2 \varphi + h_L^2) \\ &= (\lambda_H h_H^2 + \lambda_H h_H^2 \varphi + \lambda_H h_L^2 + \lambda_L h_H^2)x \\ &\quad + (\lambda_H h_H^2 \varphi + \lambda_L h_H^2 \varphi + \lambda_L h_L^2). \end{aligned} \tag{16}$$

Therefore,

$$\begin{cases} l_H = \lambda_H h_H^2 + \lambda_H h_H^2 \varphi + \lambda_H h_L^2 + \lambda_L h_H^2 \\ l_L = \lambda_H h_H^2 \varphi + \lambda_L h_H^2 \varphi + \lambda_L h_L^2. \end{cases} \tag{17}$$

According to the irreducible polynomial, x^2+x+1, in GF(2^2) and $\lambda = \{1100\}_2$, one can further reduce (17) and derive that in (11). The proof follows. ∎

REFERENCES

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

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

[3] A. Kchaou, W. E. H. Youssef, and R. Tourki, "Software implementation of AES algorithm on leon3 processor," in *Proc. 15th Int. Conf. Sci. Techn. Autom. Control Comput. Eng. (STA)*, Dec. 2014, pp. 237–242.

[4] R. Santhosh, R. Shashidhar, M. Mahalingaswamy, S. Praveen, and M. Roopa, "Design of high speed AES system for efficient data encryption and decryption system using FPGA," in *Proc. Int. Conf. Electr., Electron., Commun., Comput., Optim. Techn. (ICEECCOT)*, Dec. 2018, pp. 1279–1282.

[5] P. V. S. Shastry, N. Somani, A. Gadre, B. Vispute, and M. S. Sutaone, "Rolled architecture based implementation of AES using T-box," in *Proc. IEEE 55th Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Aug. 2012, pp. 626–630.

[6] B. C. Manjith, "Improving overall parallelism in AES accelerator using BRAM and multiple input blocks," in *Proc. Innov. Power Adv. Comput. Technol. (i-PACT)*, Mar. 2019, pp. 1–5.

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

[8] 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.

[9] A. Reyhani-Masoleh, M. Taha, and D. Ashmawy, "Smashing the implementation records of AES S-box," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2018, pp. 298–336, May 2018.

[10] Y. Liu, N. Wu, X. Zhang, F. Zhou, and F. Ge, "A compact implementation of AES S-box using evolutionary algorithm," *Chin. J. Electron.*, vol. 26, no. 4, pp. 688–695, Jul. 2017.

[11] J. Wolkerstorfer, E. Oswald, and M. Lamberger, "An ASIC implementation of the AES Sboxes," in *Proc. CT-RSA*, 2002, pp. 67–78.

[12] S. S. Priya, K. G. Das, N. M. SivaMangai, and P. K. Kumar, "Multiplexer based high throughput S-box for AES application," in *Proc. 2nd Int. Conf. Electron. Commun. Syst. (ICECS)*, Feb. 2015, pp. 242–247.

[13] 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.

[14] 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.

[15] B. Rashidi and B. Rashidi, "Implementation of an optimized and pipelined combinational logic Rijndael S-box on FPGA," *Int. J. Comput. Netw. Inf. Secur.*, vol. 5, no. 1, pp. 41–48, Jan. 2013.

[16] R. R. Rachh, P. V. AnandaMohan, and B. S. Anami, "High speed S-box architecture for advanced encryption standard," in *Proc. IEEE 5th Int. Conf. Internet Multimedia Syst. Archit. Appl.*, Dec. 2011, pp. 1–6.

[17] S. S. Priya, P. Karthigaikumar, N. M. S. Mangai, P. Kirti, and G. Das, "An efficient hardware architecture for high throughput AES encryptor using MUX based sub pipelined S-box," *Wireless Pers. Commun.*, vol. 94, no. 4, pp. 2259–2273, Jun. 2017.

[18] C. A. Murugan, P. Karthigaikumar, and S. S. Priya, "FPGA implementation of hardware architecture with AES encryptor using sub-pipelined S-box techniques for compact applications," *Automatika*, vol. 61, no. 4, pp. 682–693, Sep. 2020.

[19] X. Zhang and K. K. Parhi, "High-speed VLSI architectures for the AES algorithm," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, no. 9, pp. 957–967, Sep. 2004.

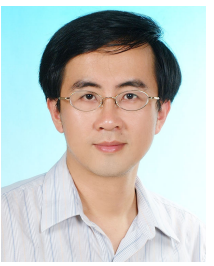
[20] K. Rahimunnisa, P. Karthigaikumar, N. Christy, S. Kumar, and J. Jayakumar, "PSP: Parallel sub-pipelined architecture for high throughput AES on FPGA and ASIC," *Open Comput. Sci.*, vol. 3, no. 4, pp. 173–186, Jan. 2013.

[21] C. Equihua, E. Anides, J. L. Garcia, E. Vazquez, G. Sanchez, J.-G. Avalos, and G. Sanchez, "A low-cost and highly compact FPGA-based encryption/decryption architecture for AES algorithm," *IEEE Latin Amer. Trans.*, vol. 19, no. 9, pp. 1443–1450, Sep. 2021.
[22] D.-E.-S. Kundi, A. Khalid, A. Aziz, C. Wang, M. O'Neill, and W. Liu, "Resource-shared crypto-coprocessor of AES Enc/Dec with SHA-3," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 12, pp. 4869–4882, Dec. 2020.
[23] C. Deng, B. Wang, L. Liu, M. Zhu, Y. Wu, H. Li, S. Yin, and S. Wei, "A 60 Gb/s-level coarse-grained reconfigurable cryptographic processor with less than 1-W power," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 2, pp. 375–379, Feb. 2020.
[24] A. A. Pammu, W.-G. Ho, N. K. Z. Lwin, K.-S. Chong, and B.-H. Gwee, "A high throughput and secure authentication-encryption AES-CCM algorithm on asynchronous multicore processor," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 4, pp. 1023–1036, Apr. 2019.
[25] S. Sawataishi, R. Ueno, and N. Homma, "Unified hardware for high-throughput AES-based authenticated encryptions," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 9, pp. 1604–1608, Sep. 2020.
[26] R. Kumar, V. Suresh, M. Kar, S. Satpathy, M. A. Anders, H. Kaul, A. Agarwal, S. Hsu, G. K. Chen, R. K. Krishnamurthy, and V. De, "A 4900-μm^2 839-Mb/s side-channel attack-resistant AES-128 in 14-nm CMOS with heterogeneous Sboxes, linear masked MixColumns, and dual-rail key addition," *IEEE J. Solid-State Circuits*, vol. 55, no. 4, pp. 945–955, Apr. 2020.
[27] W. Shan, S. Zhang, J. Xu, M. Lu, L. Shi, and J. Yang, "Machine learning assisted side-channel-attack countermeasure and its application on a 28-nm AES circuit," *IEEE J. Solid-State Circuits*, vol. 55, no. 3, pp. 794–804, Mar. 2020.
[28] F. Kenarangi and I. Partin-Vaisband, "Exploiting machine learning against on-chip power analysis attacks: Tradeoffs and design considerations," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 2, pp. 769–781, Feb. 2019.
[29] A. Singh, M. Kar, S. K. Mathew, A. Rajan, V. De, and S. Mukhopadhyay, "Improved power/EM side-channel attack resistance of 128-bit AES engines with random fast voltage dithering," *IEEE J. Solid-State Circuits*, vol. 54, no. 2, pp. 569–583, Feb. 2019.
[30] K.-S. Chong, J.-S. Ng, J. Chen, N. K. Z. Lwin, N. A. Kyaw, W.-G. Ho, J. Chang, and B.-H. Gwee, "Dual-hiding side-channel-attack resistant FPGA-based asynchronous-logic AES: Design, countermeasures and evaluation," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 11, no. 2, pp. 343–356, Jun. 2021.
[31] M. Masoumi, "Novel hybrid CMOS/Memristor implementation of the AES algorithm robust against differential power analysis attack," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 7, pp. 1314–1318, Jul. 2020, doi: 10.1109/TCSII.2019.2932337.
[32] J. Zeng, Y. Wang, C. Xu, and R. Li, "Improvement on masked S-box hardware implementation," in *Proc. Int. Conf. Innov. Inf. Technol. (IIT)*, Mar. 2012, pp. 113–116.
[33] A. Tsukioka, K. Srinivasan, S. Wan, L. Lin, Y. S. Li, N. Chang, and M. Nagata, "A fast side-channel leakage simulation technique based on IC chip power modeling," *IEEE Lett. Electromagn. Compat. Pract. Appl.*, vol. 1, no. 4, pp. 83–87, Dec. 2019.
[34] S. Zheng, X. Liu, S. Zang, Y. Deng, D. Huang, and C. Ou, "A persistent fault-based collision analysis against the advanced encryption standard," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 40, no. 6, pp. 1117–1129, Jun. 2021.
[35] T. E. Pogue and N. Nicolici, "Incremental fault analysis: Relaxing the fault model of differential fault attacks," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 3, pp. 750–763, Mar. 2020.
[36] C. Luo, Y. Fei, A. A. Ding, and P. Closas, "Comprehensive side-channel power analysis of XTS-AES," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 12, pp. 2191–2200, Dec. 2019.
[37] K. Shahbazi and S.-B. Ko, "Area-efficient nano-AES implementation for Internet-of-Things devices," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 29, no. 1, pp. 136–148, Jan. 2021.
[38] S. Mathew, S. Satpathy, V. Suresh, M. Anders, H. Kaul, A. Agarwal, S. Hsu, G. Chen, and R. Krishnamurthy, "340 mV–1.1 V, 289 Gbps/W, 2090-gate nanoAES hardware accelerator with area-optimized encrypt/decrypt GF$(2^4)^2$ polynomials in 22 nm tri-gate CMOS," *IEEE J. Solid-State Circuits*, vol. 50, no. 4, pp. 1048–1058, Apr. 2015.
[39] U. Banerjee, A. Wright, C. Juvekar, M. Waller, and A. P. Chandrakasan, "An energy-efficient reconfigurable DTLS cryptographic engine for securing Internet-of-Things applications," *IEEE J. Solid-State Circuits*, vol. 54, no. 8, pp. 2339–2352, Aug. 2019.
[40] S. N. Dhanuskodi, S. Allen, and D. E. Holcomb, "Efficient register renaming architectures for 8-bit AES datapath at 0.55 pJ/bit in 16-nm FinFET," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 8, pp. 1807–1820, Aug. 2020.
[41] L. Dalmasso, F. Bruguier, P. Benoit, and L. Torres, "Evaluation of SPN-based lightweight crypto-ciphers," *IEEE Access*, vol. 7, pp. 10559–10567, 2019, doi: 10.1109/ACCESS.2018.2889790.

YOU-TUN TENG received the M.S. degree from the Department of Engineering Science, National Cheng Kung University, Tainan, Taiwan. His research interest includes VLSI design.

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 worked at 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.

DENG-KAI CHANG is currently pursuing the M.S. degree with the Department of Engineering Science, National Cheng Kung University, Tainan, Taiwan. His research interest includes VLSI design.



PEI-YIN CHEN (Senior Member, IEEE) received the B.S. and Ph.D. degrees in electrical engineering from the National Cheng Kung University, Tainan, Taiwan, in 1986 and 1999, respectively, and the M.S. degree in electrical engineering from the Pennsylvania State University, University Park, PA, USA, in 1990. He is currently a Distinguished 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.

PIN-WEI CHEN is currently pursuing the M.S. degree with the Department of Engineering Science, National Cheng Kung University, Tainan, Taiwan. His research interest includes VLSI design.