High Throughput/Gate AES Hardware Architectures Based on Datapath Compression

Rei Ueno, *Member, IEEE*, Sumio Morioka, *Member, IEEE*, Noriyuki Miura, *Member, IEEE*, Kohei Matsuda, *Member, IEEE*, Makoto Nagata, *Senior Member, IEEE*, Shivam Bhasin, *Member, IEEE*, Yves Mathieu, Tarik Graba, Jean-Luc Danger, *Member, IEEE*, and Naofumi Homma, *Senior Member, IEEE*

Abstract—This article proposes highly efficient Advanced Encryption Standard (AES) hardware architectures that support encryption and both encryption and decryption. New operation-reordering and register-retiming techniques presented in this article allow us to unify the inversion circuits in SubBytes and InvSubBytes without any delay overhead. In addition, a new optimization technique for minimizing linear mappings, named multiplicative-offset, further enhances the hardware efficiency. We also present a shared key scheduling datapath that can work on-the-fly in the proposed architecture. To the best of our knowledge, the proposed architecture has the shortest critical path delay and is the most efficient in terms of throughput per area among conventional AES encryption/decryption and encryption architectures with tower-field S-boxes. The proposed round-based architecture can perform AES encryption where block-wise parallelism is unavailable (e.g., cipher block chaining (CBC) mode); thus, our techniques can be globally applied to any type of architecture including pipelined ones. We evaluated the performance of the proposed and some conventional datapaths by logic synthesis with the NanGate 45-nm open-cell library. As a result, we can confirm that our proposed architectures achieve approximately 51–64 percent higher efficiency (i.e., higher bps/GE) and lower power/energy consumption than the other conventional counterparts.

Index Terms—AES, hardware architectures, round-based encryption architecture, unified encryption/decryption architecture

1 Introduction

Cryptographic applications are essential for many systems that rely on secure communications, authentication, and digital signatures. In accordance with the rapid increase in Internet of Things (IoT) applications, numerous cryptographic algorithms are now required to be implemented in resource-constrained devices and embedded systems with high throughput and efficiency. Advanced Encryption Standard (AES) is an ISO/IEC 18033 standard symmetric cipher that is one of the most widely used ciphers around the world. Since the publication of AES at 2001, many hardware implementations for AES have been proposed and evaluated for CMOS logic technologies. Studies of AES design and implementation are important from both practical and academic perspectives since AES employs a substitution permutation network (SPN) structure and the major subfunctions, which are followed by many other security primitives.

AES encryption and decryption are frequently used in block-chaining modes of operation, such as cipher block chaining (CBC), cipher-based message authentication code (CMAC), and counter with CBC-MAC (CCM), which are used in, for example, IEEE802.11 wireless LAN and IEEE802.15.4 wireless sensor networks. Therefore, AES architectures that efficiently perform both encryption and decryption in block-chaining modes are in high demand. Here, we cannot exploit the tradeoff of pipelining between throughput and area for encryption in block-chaining modes, although many conventional architectures employ pipelining techniques [1], [2], [3]. This raises the importance of datapath optimization for a higher throughput and efficiency. In addition, on-the-fly key scheduling should be implemented in resource-constrained devices because offline key scheduling implementation requires additional memory to store expanded round keys. Moreover, on-the-fly key scheduling is sometimes more important when implementing block cipher with a tweak [4] (e.g., used in authenticated encryption), in some of which temporal key is generated using a master key and a tweak unique for each block [5], [6]. Thus, it would be valuable to develop efficient AES architectures with on-the-fly key scheduling without block-wise pipelining techniques.

- *R. Ueno is with the Tohoku University, 2–1–1 Katahira, Aoba-ku, Sendaishi 980–8579, Japan, and also with JST PRESTO4-1-8 Honcho, Kawaguchi, Saitama 332-0012, Japan. E-mail: ueno@riec.tohoku.ac.jp.*
- *N. Homma is with the Tohoku University, 2–1–1 Katahira, Aoba-ku, Sendaishi 980–8579, Japan. E-mail: homma@riec.tohoku.ac.jp.*
- *S. Morioka is with the Interstellar Technologies Inc., 690-4 Memu, Taiki, Hiroo-gun, Hokkaido 089-2113, Japan. E-mail: morioka@fb3.so-net.ne.jp.*
- *N. Miura, K. Matsuda, and M. Nagata are with the Kobe University, 1–1 Rokkodai-machi, Nada-ku, Kobe-shi 657–8501, Japan. E-mail: {miura, matsuda, nagata}@cs.kobe-u.ac.jp.*
- *S. Bhasin is with the Nanyang Technological University, 50 Nanyang Drive, Research Techno Plaza, BorderX Block, 9th Storey, Singapore 637553. E-mail: sbhasin@ntu.edu.sg.*
- *Y. Mathieu, T. Graba, and J.-L. Danger are with the Télécom ParisTech, 46, rue Barrault, 75013 Paris, France. E-mail: {yves.mathieu, tarik.graba, jean-luc.danger}@telecom-paristech.fr.*

Manuscript received 24 Feb. 2019; revised 6 Nov. 2019; accepted 26 Nov. 2019. Date of publication 4 Dec. 2019; date of current version 10 Mar. 2020. (Corresponding author: Rei Ueno.)
Recommended for acceptance by P. Gratz.
Digital Object Identifier no. 10.1109/TC.2019.2957355

In this paper, we present new round-based AES architectures for encryption only and both encryption and decryption with on-the-fly key scheduling. The proposed architectures achieve the lowest critical path delay (the least number of serially connected gates in the critical path) with less area overhead compared to conventional architectures with tower-field S-boxes. Our architectures employ new operation-reordering and register-retiming techniques to unify the inversion circuits without any selectors. These techniques also make it possible to unify the affine transformation and linear mappings (i.e., the isomorphism and constant multiplications) to reduce the total number of logic gates. The proposed and conventional AES datapaths were synthesized and evaluated with an open-cell library. The evaluation results show that our two architectures can perform encryption and both encryption and decryption more area-time efficiently. In particular, the throughput per gates of the proposed architectures are 51–64 percent larger than those of the corresponding conventional best architectures.

While the basic concepts and preliminary evaluations of our AES encryption/decryption architecture were resented in previous work [7], this paper newly presents more efficient AES architectures with threefold novel contributions. First, we propose a new optimization technique for minimizing linear operations named multiplicative-offset. The multiplicative-offset provides a larger variety of matrix constructions for linear mappings without any additional block nor overhead, which can lead to more compact and/or lower-latency implementation. The proposed multiplicative-offset is given as an extension of a previous method [8] to round-based architectures on the basis of operation-reordering and register-retiming. Thanks to the new method, the proposed architecture achieves a further 7–9 percent higher efficiency than that in the previous version [7]. Second, we newly present a high throughput/gate AES encryption hardware design based on the proposed concept/technique (i.e., tower-field arithmetic, register-retiming, unification of linear operations, and multiplicative-offset). In particular, we show that the combination of unification and multiplicative-offset techniques can significantly reduce the logic depth (i.e., critical delay) of the encryption datapath. As a result, the proposed AES encryption architecture achieves 58–64 percent higher throughput/gate efficiency than other conventional ones. Third, while the previous study did not include any reports on power consumption, we describe a power consumption estimations based on a Monte-Carlo gate-level timing simulations with back-annotation, in which the effects of glitches were considered. The results clearly show the advantage of the proposed architectures in terms of power/energy consumption.

The rest of this paper is organized as follows. Section 2 introduces related works on AES hardware architectures. Section 3 presents a new AES encryption/decryption hardware architecture based on our operation-reordering, register-retiming, unification of linear mappings, and multiplicative-offset. Section 3 also presents an evaluation of the proposed datapath by the logic synthesis and gate-level timing simulations in comparison with conventional round-based datapaths. Section 4 proposes an AES encryption hardware architecture based on the techniques presented in Section 3, and we evaluate the proposed architecture in the same manner as Section 3. Section 5 discusses variations of the proposed architectures. Finally, Section 6 contains our conclusion.

2 Related Works

In this paper, we briefly describe the related works. See the previous version [7] for more detail.

2.1 Unified AES Datapath for Encryption and Decryption

Architectures that perform one round of encryption or decryption per clock cycle without pipelining are the most typical for AES design and are called round-based architectures in this paper. Round-based architectures can be implemented more efficiently in terms of throughput per area than other architectures by utilizing the inherent parallelism of symmetric key ciphers.

To design such round-based encryption/decryption architectures in an efficient manner, we consider how to unify the resource-consuming components such as the inversion circuits in SubBytes/InvSubBytes for the encryption and decryption datapaths. There are two conventional approaches for designing such unified datapaths. The first approach is to place two distinct datapaths for encryption and decryption and select one of the datapaths with multiplexers as in [1]. In the architecture, the intermediate value is stored in a register after InvMixColumns instead of AddRoundKey. Such register-retiming is suitable for pipelined architectures. The second approach is to unify the circuits of the functions SubBytes, ShiftRows, and MixColumns with their inverse functions, respectively (e.g., [2], [9]). The order of the decryption operations was changed to be the same as that of the encryption operations. The main drawbacks of conventional architectures are the false critical path delay and the required area and delay overheads caused by several multiplexers. This false critical path reduces the maximum operation frequency owing to logic synthesis due to the false longest logic chain. The overhead caused by the multiplexers is also nonnegligible for common standard-cell-based designs. In addition, the second approach sometimes requires an additional InvMixColumns required for the above reordering [9], which is also considered as an overhead.

2.2 Inversion Circuit Design and Tower-Field Arithmetic

The design of the inversion circuit used in (Inv)SubBytes has a significant impact on the performance of AES implementations. There are two major approaches for its design, namely, direct mapping and tower-field arithmetic. Inversion circuits based on direct mapping such as table-lookup, binary decision diagram (BDD), and positive-polarity Reed-Muller (PPRM) transforms [1], [10], [11] are faster but larger than those based on tower field. On the other hand, tower-field arithmetic enables us to design more compact and area-time efficient inversion circuits in comparison with direct mapping [2], [8], [9], [12], [13], [14], [15], [16], [17], [18]. Therefore, we focus on inversion circuits based on tower-field arithmetic in this paper.

To embed such a tower-field-based inversion circuit in AES hardware, isomorphic mapping between the AES field and

the tower field is required because the inversion and MixColumns are performed over the AES field (i.e., PB-based $GF(2^8)$ with an irreducible polynomial $x^8+x^4+x^3+x+1$). Typically, the input into the inversion circuit (in the AES field) is initially mapped to the tower field by the isomorphic mapping. After the inversion operation over the tower field, inverse isomorphic mapping prior to affine transformation is applied [9]. On the other hand, some architectures perform all of the AES subfunctions (i.e., SubBytes as well as ShiftRows, MixColumns, and AddRoundKey) over the tower field, where isomorphic mapping and its inverse mappings are performed at the timings of the data (i.e., plaintext and ciphertext) input and output, respectively [2], [19]. In other words, the cost of field conversion is suppressed when the conversion is performed only once during encryption or decryption. However, the cost of constant multiplications in MixColumns over a tower field is worse than that over the AES field while inversion is efficiently performed over the tower field. More precisely, in tower-field architectures, such linear mappings including constant multiplications usually require a $3T_{\mathrm{XOR}}$ delay, where T_{XOR} indicates the delay of an XOR gate [20]. The XOR gate count used in (Inv)MixColumns over a tower field is also worse than that over the AES field.

3 Proposed Encryption/Decryption Architecture

This section presents a new round-based AES architecture that unifies the encryption and decryption paths in an efficient manner. The key ideas for reducing the critical path delay are summarized as follows: (a) to merge linear mappings such as MixColumns and isomorphic mappings as much as possible by reordering subfunctions, (b) to minimize the number of selectors to unify the encryption and decryption paths by the above merging and a register-retiming, and (c) to perform isomorphic mapping and its inverse mappings only once in the pre- and post-round datapaths. We can reduce the number of linear mappings to at most one for each round operation as the effect of (a). Moreover, we can reduce the number of selectors to only one (4-to-1 multiplexer) in the unified datapath as the effect of (b) while the inversion circuit is shared by the encryption and decryption paths. From the idea of (c), we can remove the isomorphic mapping and its inverse mappings from the critical path. Fig. 1 shows the overall architecture that consists of the round function and key scheduling parts. Our architecture performs all of the subfunctions over a tower field for both the round function and key scheduling parts and therefore applies isomorphic mappings between the AES and tower fields in the datapaths of the pre- and post-round operations, which are represented as the blocks "Pre-round datapath" and "Post-round datapath" in Fig. 1. "Round datapath" performs one round operation for either encryption or decryption.

3.1 Round Function Part

The proposed architecture employs a datapath for encryption and decryption where inversion is unified and applies new operation-reordering and register-retiming techniques

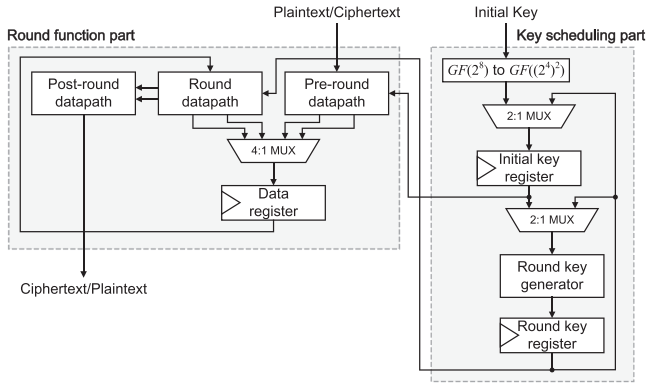

Fig. 1. Overall architecture of proposed AES encryption/decryption hardware.

to address the conventional issues of a false critical path and additional multiplexers. By using our operation-reordering technique and then merging linear mappings, we can reduce the number of linear mappings on the critical path of the round datapath to at most one. Our reordering technique also allows for the unification of the linear mappings and affine transformation in a round. The unification of these mappings can drastically reduce the critical path delay and the XOR-gate count of linear mappings in a tower-field architecture.

The new operation reordering is derived as follows. First, the operational round operation of AES encryption is represented by the following equation:

$$\begin{aligned} m_{i,j}^{(r+1)} =& u_{-i}S(m_{0,i+j}^{(r)}) + u_{1-i}S(m_{1,i+j}^{(r)}) \\ &+ u_{2-i}S(m_{2,i+j}^{(r)}) + u_{3-i}S(m_{3,i+j}^{(r)}) + k_{i,j}^{(r)} \\ =& \sum_{e=0}^{3}(u_{e-i}S(m_{e,i+j}^{(r)})) + k_{i,j}^{(r)}, \end{aligned} \tag{1}$$

where $m_{i,j}^{(r)}$ and $k_{i,j}^{(r)}$ are the ith row and jth column intermediate value at the rth round input in encryption and the rth round key, respectively, except for the final round. Note that the subscripts of each variable are a member of $\mathbb{Z}/4\mathbb{Z}$. The function S indicates the 8-bit S-box, and u_0, u_1, u_2, and u_3 are the coefficients of the matrix of MixColumns, which are respectively given by β, $\beta+1$, 1, and 1, where β is the indeterminate of $GF(2^8)$ satisfying $\beta^8+\beta^4+\beta^3+\beta+1=0$. We can rewrite Eq. (1) by decomposing S into inversion and affine transformation as follows:

$$m_{i,j}^{(r+1)} = \sum_{e=0}^{3}\left(u_{e-i}\left(A\left(\left(m_{e,i+j}^{(r)}\right)^{-1}\right)+c\right)\right) + k_{i,j}^{(r)}, \tag{2}$$

where A is the linear mapping of the affine transformation, and c $(=\beta^6+\beta^5+\beta+1)$ is a constant. In the case of tower-field architectures, Eq. (2) is represented by

$$m_{i,j}^{(r+1)} = \sum_{e=0}^{3}(u_{e-i}(A(\Delta'(\left(\Delta(m_{e,i+j}^{(r)})\right)^{-1}))+c)) + k_{i,j}^{(r)}, \tag{3}$$

where Δ is the isomorphic mapping from the AES field to a tower field, and Δ' is the inverse isomorphic mapping.

The linear mappings, which include an isomorphism and constant multiplications over the GF, are performed by the constant multiplication of the corresponding matrix over $GF(2)$. Therefore, we can merge such mappings to reduce the critical path delay and the number of XOR gates. To unify all linear mappings on one round as at most one mapping, we first apply Δ to both sides of Eq. (3) as follows:

$$\Delta(m_{i,j}^{(r+1)}) = \Delta\left(\sum_{e=0}^{3}(u_{e-i}(A(\Delta'(\left(\Delta(m_{e,i+j}^{(r)})\right)^{-1})) + c)\right) + k_{i,j}^{(r)}), \tag{4}$$

which is followed by

$$\Delta(m_{i,j}^{(r+1)}) = \sum_{e=0}^{3}\Delta(u_{e-i}(A(\Delta'(\left(\Delta(m_{e,i+j}^{(r)})\right)^{-1})))) + \Delta(c) + \Delta(k_{i,j}^{(r)}), \tag{5}$$

because an arbitrary linear mapping Lin satisfies $Lin(a+b) = Lin(a) + Lin(b)$, $\sum(a+b) = \sum a + \sum b$, and $\sum_{e=0}^{3} u_{e-i}c = c + c\beta + c(\beta+1) + c = c$. Then, we consider the variable $d_{i,j}^{(r)}$ of the tower field derived from $m_{i,j}^{(r)}$ (i.e., $d_{i,j}^{(r)} = \Delta(m_{i,j}^{(r)})$). By substituting $d_{i,j}^{(r)}$ to $\Delta(m_{i,j}^{(r)})$ we can merge the linear mappings as follows:

$$d_{i,j}^{(r+1)} = \sum_{e=0}^{3}\left(U_{e-i}\left(\left(d_{e,i+j}^{(r)}\right)^{-1}\right)\right) + \Delta(c) + \Delta(k_{i,j}^{(r)}), \tag{6}$$

where $U_{e-i}(x) = \Delta(u_{e-i}(A(\Delta'(x))))$. Note that $U_2(x) = U_3(x)$ is equal to the affine transformation over the tower field denoted as $\Psi(x) = \Delta(A(\Delta'(x)))$. Thus, the linear mappings of a round in Eq. (6) can be merged into at most one, even with a tower-field S-box, while the linear mappings in Eq. (3) cannot be.

On the other hand, the equations for AES decryption corresponding to Eqs. (3) and (6) are respectively given by

$$m_{i,j}^{\prime(r-1)} = \sum_{e=0}^{3}\left(v_{e-i}\left(\Delta'\left(\left(\Delta(A'(m_{e,j-i}^{\prime(r)}) + c')\right)^{-1}\right) + k_{e,j}^{(r)}\right)\right), \tag{7}$$

$$d_{i,j}^{(r-1)} = \sum_{e=0}^{3}(\Delta(v_{e-i}(\Delta'(\left(\Delta(A'(\Delta'(d_{e,j-i}^{(r)}))) + \Delta(c')\right)^{-1} + \Delta(k_{e,j}^{(r)}))))), \tag{8}$$

where $m_{i,j}^{\prime(r)}$ denotes the ith row and jth column intermediate value at the rth round input at decryption, and A' indicates the linear mapping of the inverse affine transformation. The coefficients v_0, v_1, v_2, and v_3 are respectively given by $\beta^3+\beta^2+\beta$, $\beta^3+\beta+1$, $\beta^3+\beta^2+1$, and β^3+1, and c' $(=\beta^2+1)$ is a constant. Here, the linear mappings cannot be merged into one because they are performed both before and after the inversion operation. In addition, if we construct an encryption/decryption datapath based on Eqs. (6) and (8), the inversion circuit cannot be shared by encryption and decryption without a selector because the timings of the inversion operations are different from each other. Therefore, we consider a register retiming to store the intermediate value $s_{i,j}^{(r)}$ given after the inverse affine transformation over the tower-field. Here, $s_{i,j}^{(r)}$ is given by $s_{i,j}^{(r)} = \Delta(A'(\Delta'(d_{i,j}^{(r)}))) + \Delta(c')$. In the decryption, we store $s_{i,j}^{(r)}$ in the data register instead of $d_{i,j}^{(r)}$. By using $s_{i,j}^{(r)}$ and $s_{i,j}^{(r-1)}$, we rewrite Eq. (8) as follows:

$$s_{i,j}^{(r-1)} = \sum_{e=0}^{3}\left(V_{e-i}\left(\left(s_{e,j-i}^{(r)}\right)^{-1} + \Delta(k_{e,j}^{(r)})\right)\right) + \Delta(c'), \tag{9}$$

where $V_{e-i}(x) = \Delta(A'(v_{e-i}(\Delta'(x))))$.

Our round datapath is constructed with a minimal critical path delay according to Eqs. (6) and (9). Fig. 2 shows the proposed reordering technique. In the proposed flow, we perform isomorphic mapping from/to $GF(2^8)$ to/from $GF((2^4)^2)$ at the data input/output. We first decompose SubBytes into the inversion and (Inv)Affine. In the encryption, Affine, MixColumns, and AddRoundKey can be merged by exchanging Affine and ShiftRows. In the decryption, the inversion circuit is located at the beginning of the round by exchanging the inversion and InvShiftRows. Note that all the operations excluding isomorphic mappings are performed over the tower field as mentioned before. Thus, additional selectors for sharing the inversion circuit are not required thanks to the operation-reordering and register-retiming techniques. This is because both inversion operations are performed at the beginning of the round, which means that the data register output can be directly connected to the inversion circuit.

Fig. 3 illustrates the proposed round function datapath with the unification of linear mappings. Our architecture employs only one 128-bit 4-in-1 multiplexer, whereas conventional ones employ several 128-bit multiplexers. For example, the datapath in [21] employs seven 128-bit multiplexers.[1] Fewer selectors can reduce the critical path delay and circuit area and solve the false critical path problem. Unified affine and Unified affine^{-1} in Fig. 3 perform the unified linear mappings (i.e., $U_0, \ldots, U_3$ and $V_0, \ldots, V_3$) and constant addition. The number of linear mappings on the critical path is at most one in our architecture, whereas that of the conventional architectures is not. We can also suppress the overhead of constant multiplication over the tower field by the unification. Adder arrays in Fig. 3 consist of four 4-input 8-bit adders in MixColumns or InvMixColumns. In the encryption, the factoring technique for MixColumns and AddRoundKey [20] is available for Unified affine as described in Section 4.1 in detail, which makes the circuit area smaller without a delay overhead. As a result, the data width between Unified affine and Adder array in Encryption path is reduced from 512 to 256 bits because the calculations of U_1 and U_3 are not performed in the Encryption path. In addition, Adder array and AddRoundKey are unified in the Encryption path because both of them are composed of 8-bit adders.[2] On the other hand, since there is no factoring technique for InvMixColumns

1. The selectors in SubBytes/InvSubBytes are included in the seven multiplexers.

2. Some architectures such as [9], [21] unify AddInitialKey and AddRoundKeys. We did not unify them to avoid increasing the number of selectors.

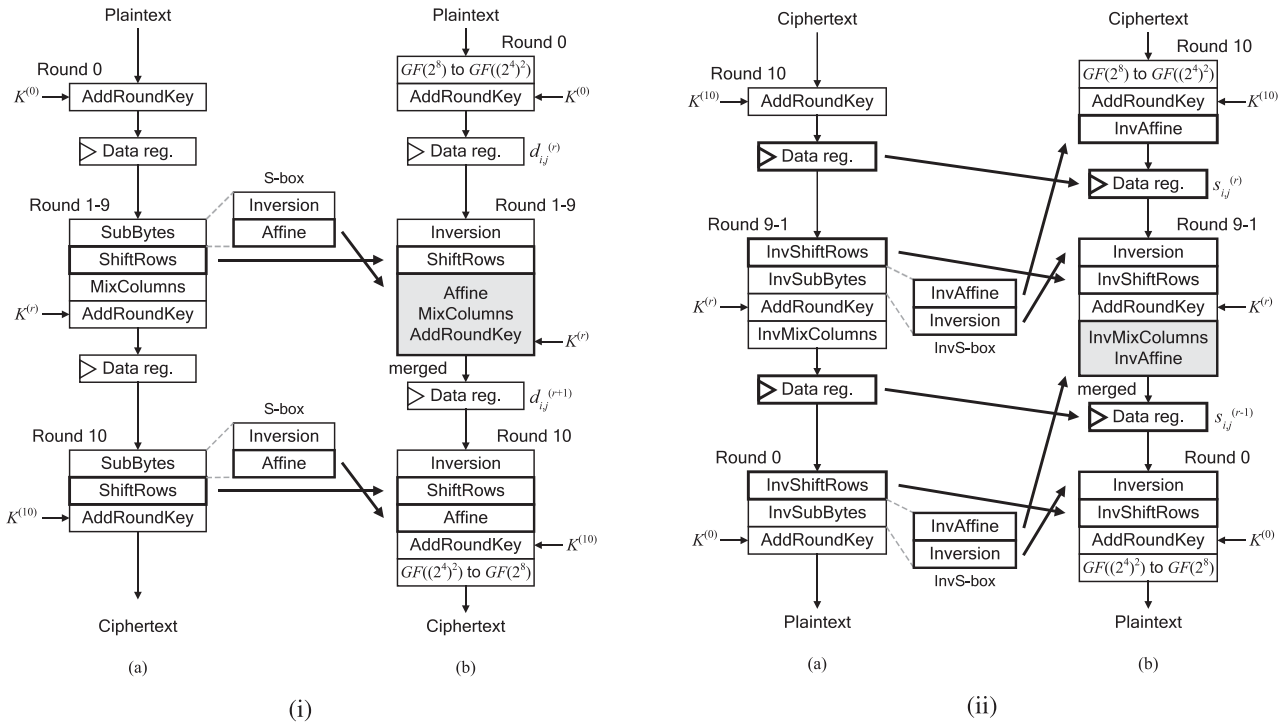

Fig. 2. Proposed (i) encryption and (ii) decryption flows (a) before and (b) after reordering and register-retiming.

without delay overheads, the data width from Unified affine^{-1} to Adder array in the Decryption path is 512 bits. Finally, an inactive path can be disabled by using a demultiplexer since our datapath is fully parallel after the inversion circuit. Thanks to the disabling, a multiplexer and AddRoundKey are unified as Bit-parallel XOR. (The addition of $\Delta(c)$ in Unified affine should be active only during encryption.) In addition, the demultiplexer can suppress power consumption due to a dynamic hazard. Although tower-field inversion circuits are known to be power-consuming owing to dynamic hazards [11], these hazards can be terminated at the input of the inactive path.

Our datapath employs the inversion circuit presented in [8], [15] because it has the highest area-time efficiency among inversion circuits applicable to the tower-field architectures and/or both encryption and decryption. We can merge the isomorphic mappings in order to reduce the linear function on the round datapath to only one, even if the inversion circuit has different GF representations at the input and output. Since the output is given by an RRB, the data width from Inversion to Unified affine (or Unified affine^{-1}) is given by 160 bits. However, AddRoundKey in the decryption path and Bit-parallel XOR in the post-round datapath are implemented respectively by only 128 XOR gates because the NB used as the input is equal to the reduced version of the RRB. Note here that, while the path through the Encryption path and Post-round datapath contains two linear mappings (i.e., Unified affine and $GF((2^4)^2)$ to $GF(2^8)$), this path would not be a critical path because the Adder array (including AddRoundKey) and 4-in-1 multiplexer basically has a longer delay than $GF((2^4)^2)$ to $GF(2^8)$. In addition, a 1:2 DeMUX is implemented with NOR gates thanks to the redundancy, whereas nonredundant representations require AND gates.

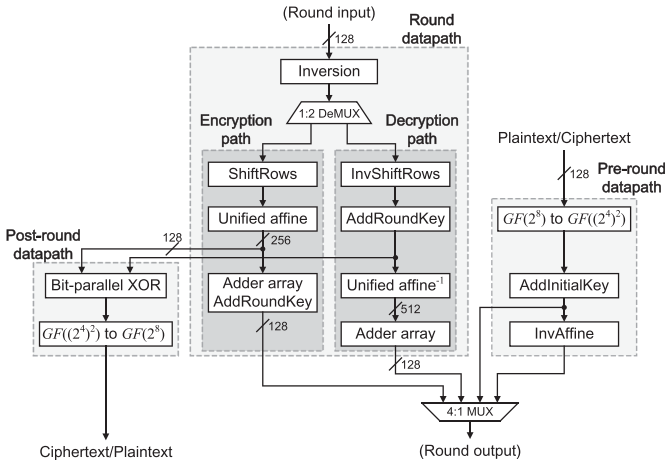

Fig. 3. Proposed round function part for encryption/decryption hardware.

3.2 Key Scheduling Part

The on-the-fly key scheduling part is shared by the encryption and decryption processes. For the encryption, the key scheduling part first stores the initial key in the Initial key register in Fig. 1 and then generates the round keys during the following clock cycles. For the decryption, the final round key should be calculated from the initial key and stored in Initial key register in advance. The key scheduling part then generates the round keys in the reverse order by Round key generator in Fig. 1. However, conventional key scheduling datapaths such those as in [9], [21] are not applicable to our architecture because they have a loop with a false path and/or a longer true critical path than our optimized round datapath.

To address the above issue, we introduce a new architecture for the key scheduling datapath. For on-the-fly implementation,

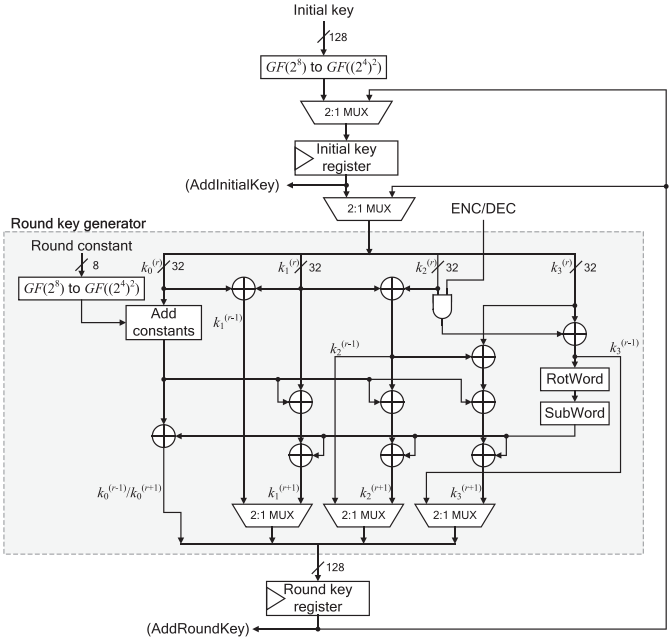

Fig. 4. Proposed key scheduling part for encryption/decryption hardware.

the subkeys are calculated for each of the four subkeys (i.e., 128 bits) in a clock cycle. Therefore, the on-the-fly key scheduling for the encryption over a tower field is expressed as

$$\begin{cases} k_0^{(r+1)} &= k_0^{(r)} + \mathrm{KeyEx}(k_3^{(r)}) \\ k_1^{(r+1)} &= k_0^{(r)} + k_1^{(r)} + \mathrm{KeyEx}(k_3^{(r)}) \\ k_2^{(r+1)} &= k_0^{(r)} + k_1^{(r)} + k_2^{(r)} + \mathrm{KeyEx}(k_3^{(r)}) \\ k_3^{(r+1)} &= k_0^{(r)} + k_1^{(r)} + k_2^{(r)} + k_3^{(r)} + \mathrm{KeyEx}(k_3^{(r)}) \end{cases}, \tag{10}$$

where $k_0^{(r)}$, $k_1^{(r)}$, $k_2^{(r)}$, and $k_3^{(r)}$ are the 32-bit subkeys at the rth round over the tower field, and KeyEx is the key expansion function that consists of a round-constant addition, RotWord, and SubWord over the tower field. The inverse key scheduling for the decryption over the tower field is represented by

$$\begin{cases} k_0^{(r-1)} &= k_0^{(r)} + \mathrm{KeyEx}(k_2^{(r)} + k_3^{(r)}) \\ k_1^{(r-1)} &= k_0^{(r)} + k_1^{(r)} \\ k_2^{(r-1)} &= k_1^{(r)} + k_2^{(r)} \\ k_3^{(r-1)} &= k_2^{(r)} + k_3^{(r)} \end{cases}. \tag{11}$$

Fig. 4 shows the proposed key scheduling datapath architecture, where the KeyEx components (i.e., RotWord, SubWord, and Add constants) are unified for encryption and decryption. Here, the input key is initially mapped to the tower field, and all of the computations (including AddRoundKey) are performed over the tower field. The upper 2-in-1 multiplexer selects an initial key or a final round key as the input to the Initial key register, the middle 2-in-1 multiplexer selects a key stored in Initial key register or a round key as the input to the Round key generator, and the lower 2-in-1 multiplexers select an encryption or a decryption path. Importantly, most of the adders (i.e., XOR gates) for computing $k_1^{(r+1)}$, $k_2^{(r+1)}$, and $k_3^{(r+1)}$ should be nonintegrated in order to make the critical path shorter than that of the round function part. In addition, the ENC/DEC signal controls the input to RotWord and SubWord by using a 32-bit AND gate. Such an usage of AND gate is useful for shortening critical path of key scheduling datapath compared to conventional ones which employ only multiplexers. Moreover, the round constant addition is performed separately from RotWord and SubWord to reduce the critical path delay. As a result, the critical path delay of the key scheduling part becomes shorter than that of our optimized round function part.

3.3 Optimization of Linear Mappings based on Multiplicative-Offset

Linear mappings (i.e., isomorphic mapping, the linear part of unified affine, and constant multiplications in MixColumns/InvMixColumns) are realized as XOR matrix operations, whose construction are determined by the defining polynomials of the tower field in the case of tower-field implementation. The construction of XOR matrices (especially, Hamming weights of matrix) has an impact on the performance of AES hardware. In this subsection, we newly present a method named multiplicative-offset for increasing the variety of constructions of linear mappings, in order to find conversion matrices with less Hamming weights. Although the basic idea is similar to the method for optimizing tower-field S-box implementation proposed in [8], which uses a fixed multiplicative mask for the inversion, we extend and generalize it in order to optimize the whole tower-field AES encryption/decryption architecture on the basis of proposed register-retiming and operation-reordering.

Fig. 5 illustrates the proposed (a) encryption and (b) decryption flows with multiplicative-offset, where $w_{i,j}^{(r)}$ and $t_{i,j}^{(r)}$ denote the ith column and jth row intermediate bytes at the rth round for encryption and decryption with multiplicative-offset, respectively. In addition, $L^{(r)}$ denotes the rth round key with the multiplicative-offset. Here, as well as Fig. 2, all operations excluding the first and final operations (i.e., "Multiply $\gamma/GF(2^8)$ to $GF((2^4)^2)$" and "$GF(2^8)$ to $GF((2^4)^2)$/Multiply γ^{-1}," respectively) should be performed over the tower field. The basic idea of multiplicative-offset is to initially multiply all bytes of the input data by a constant value γ as an offset, which is a non-zero element of the PB-based $GF(2^8)$, and then, to multiply the intermediate bytes at each round by γ^2 to correct the offset, and finally to multiply γ^{-1} before the data output to remove the offset. Note that, in decryption, the offset value should be given by γ^{-1} (and γ^{-2} should be multiplied in a round) in order to share pre- and post-round datapaths. Since the multiplication over a GF is a type of linear mapping over the GF, this multiplication can be merged to isomorphic mapping or unified affine with further operation-reordering and register-retiming. Thus, we can increase the variety of conversion matrices by 255 times without any overhead because γ can take a value from 255 candidates.

Importantly, each isomorphic mapping with multiplicative-offset at Round 0 and 10 and round key is identical in the cases of encryption and decryption, which indicates that we can still unify the pre- and post-round datapaths and key scheduling part even with the multiplicative-offset. In addition, since the number of merged linear operations are same as that without multiplicative-offset in Fig. 2, we can confirm that the

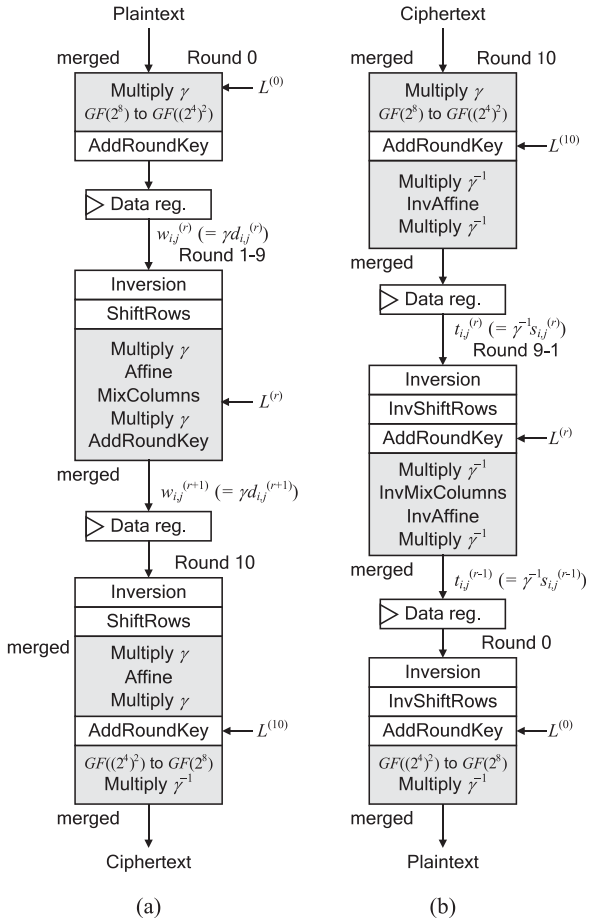

Fig. 5. Proposed tower-field (a) encryption and (b) decryption flows based on operation-reordering, register-retiming, and multiplicative-offset.

multiplicative-offset requires no overhead. Note that, while the conventional optimization method in [8] requires to multiply γ to both input and output, the proposed multiplicative-offset modifies the timing of multiplications with register-retiming such that all the linear operations per round can be unified.

More precisely, we first multiply γ to $m_{i,j}^{(r)}$ and the results of inversion in Eq. (3) and we derive

$$m_{i,j}^{(r+1)} = \sum_{e=0}^{3}(u_{e-i}(A(\gamma(\Delta'(\left(\Delta(\gamma m_{e,i+j}^{(r)})\right)^{-1}))) + c)) + k_{i,j}^{(r)}. \tag{12}$$

This is so because the offset γ of $m_{i,j}^{(r)}$ is canceled by the multiplication with γ after the inversion. However, we cannot implement the AES round function based on Eq. (12) in an efficient round-based manner because the $m_{i,j}^{(r+1)}$ is not offset by γ, which indicates that we should store $m_{i,j}^{(r+1)}$ in the register as an intermediate value and should perform the multiplication by γ prior to inversion. Therefore, we multiply γ to both sides of Eq. (12) to correct the offset as follows:

$$\gamma m_{i,j}^{(r+1)} = \textstyle\sum_{e=0}^{3} \gamma(u_{e-i}(A(\gamma(\Delta'(\left(\Delta(\gamma m_{e,i+j}^{(r)})\right)^{-1}))) + c)) + \gamma k_{i,j}^{(r)}. \tag{13}$$

Thus, the intermediate bytes and round key bytes at every round have the same offset γ. Let $w_{i,j}^{(r)}$ be the intermediate byte with the offset over the tower field (i.e., $\Delta(\gamma m_{i,j}^{(r)})$), which is the intermediate value stored in the register. We can merge all the linear mappings in Eq. (13) as well as Eq. (3) by resister-retiming, and the equation for the round datapath of encryption is derived as follows:

$$w_{i,j}^{(r+1)} = \sum_{e=0}^{3}\left(\Omega_{e-i}\left(\left(w_{e,i+j}^{(r)}\right)^{-1}\right)\right) + \Delta(\gamma c) + \Delta(\gamma k_{i,j}^{(r)}), \tag{14}$$

where $\Omega_{e-i}(x) = \Delta(\gamma(u_{e-i}(A(\gamma(\Delta'(x))))))$, which denotes the unified linear transformation with the offset correction in the encryption. Note that γc is a constant value for a fixed γ, and it should be embedded in the datapath. In the pre-round datapath, plaintext is not only mapped to the tower field, but is also offset by γ at $\Delta_\gamma = \Delta \circ \gamma$, which denotes the merged linear mapping for Δ and the constant multiplication of γ. In the post-round datapath, the ciphertext is mapped to the AES field, and the offset is removed at $\Delta'_{\gamma^{-1}} = \gamma^{-1} \circ \Delta'$, which denotes the merged linear mapping for Δ' and the constant multiplication of γ^{-1}.

On the other hand, in decryption, we multiply γ^{-1} to $s_{i,j}^{(r)}$ and the result of inversion in Eq. (8), adjust c', and multiply γ^{-1} to both sides of Eq. (8). As well as the encryption, by using the register-retiming and operation-reordering techniques, we then derive the multiplicative-offset version of Eq. (9) as follows:

$$t_{i,j}^{(r-1)} = \sum_{e=0}^{3}(\Lambda_{e-i}(\left(t_{e,j-i}^{(r)}\right)^{-1} + \Delta(\gamma k_{e,j}^{(r)}))) + \Delta(\gamma c'), \tag{15}$$

where $\Lambda_{e-i}(x) = \Delta(\gamma^{-1}(A'(v_{e-i}(\gamma^{-1}(\Delta'(x))))))$ and $t_{i,j}^{(r)} = \Delta(\gamma^{-1}(\Delta'(s_{i,j}^{(r)})))$, which denote the unified linear transformation with offset correction and the intermediate value with the proposed register-retiming and multiplicative-offset in the decryption, respectively. Note here that the offset value is given by γ^{-1} instead of γ because of the register-retiming. In other words, while the ciphertext bytes are initially offset by γ at Δ_γ as same as encryption, the first intermediate bytes are computed as $t_{i,j}^{(1)} = \Delta(\gamma^{-1}(A'(\gamma^{-1}(\Delta'(\gamma z_{i,j})))))$, where $z_{i,j}$ denotes the ith row and jth column byte of ciphertext (i.e., input data at decryption). Moreover, we remove the offset at $\Delta'_{\gamma^{-1}}$ in the post-round datapath as in the case of encryption. Whereas the offset value is γ^{-1} at the register, the offset is correctly removed by multiply γ^{-1} at Δ'_γ because the offset value is inverted at the inversion as $\gamma(t_{i,j}^{(0)})^{-1} = (\gamma^{-1}t_{i,j}^{(0)})^{-1}$.

Since the offset of a round key is given by γ in both cases of encryption and decryption, we can still use the unified key scheduling part. Let $l_{i,j}^{(r)}$ be the ith row and jth column of the rth round key with the offset over the tower field (i.e., $\Delta(\gamma k_{i,j}^{(r)})$ $(= \Delta_\gamma(k_{i,j}^{(r)}))$), and let $l_i^{(r)}$ be the ith 32-bit word of the rth round key with the offset over the tower field. We first map the initial key $k_{i,j}^{(0)}$ to $l_{i,j}^{(0)}$ by using Δ_γ. Then, the key scheduling with the offset for encryption is expressed as

$$\begin{cases} l_0^{(r+1)} &= l_0^{(r)} + \mathrm{KeyEx}_{\mathrm{offset}}(l_3^{(r)}) \\ l_1^{(r+1)} &= l_0^{(r)} + l_1^{(r)} + \mathrm{KeyEx}_{\mathrm{offset}}(l_3^{(r)}) \\ l_2^{(r+1)} &= l_0^{(r)} + l_1^{(r)} + l_2^{(r)} + \mathrm{KeyEx}_{\mathrm{offset}}(l_3^{(r)}) \\ l_3^{(r+1)} &= l_0^{(r)} + l_1^{(r)} + l_2^{(r)} + l_3^{(r)} + \mathrm{KeyEx}_{\mathrm{offset}}(l_3^{(r)}) \end{cases}, \tag{16}$$

TABLE 1
Synthesis Results for Proposed and Conventional AES Hardware Architectures With Area Optimization

	Area (GE)	Latency (ns)	Max. freq. (MHz)	Throughput (Gbps)		Power@100MHz (uW)		Efficiency (Kbps/GE)		PL product	
				10 cycles	11 cycles	Enc.	Dec.	10 cycles	11 cycles	Enc.	Dec.
Satoh *et al.* [6]	15,269.67	32.56	337.84	4.32	3.69	849	820	273.20	257.45	27,643	26,699
Lutz *et al.* [1]	26,113.67	23.00	434.78	5.57	N/A	759	833	213.12	N/A	17,457	19,159
Liu *et al.* [18]	13,760.33	37.62	292.40	3.74	3.40	1,020	1,030	271.99	247.26	38,372	38,749
Mathew *et al.* [2]	18,576.33	43.34	253.81	3.25	2.95	1,150	1,640	174.89	158.99	49.841	71,078
Previous version [4]	16,428.33	21.01	523.56	6.70	6.09	582	512	407.93	370.84	12,228	10,757
This work	16,418.33	19.25	571.43	7.31	6.65	490	511	445.50	405.00	9,433	9,837

TABLE 2
Synthesis Results for Proposed and Conventional AES Hardware Architectures With Area-Speed Optimization

	Area (GE)	Latency (ns)	Max. freq. (MHz)	Throughput (Gbps)		Power@100MHz (uW)		Efficiency (Kbps/GE)		PL product	
				10 cycles	11 cycles	Enc.	Dec.	10 cycles	11 cycles	Enc.	Dec.
Satoh *et al.* [6]	16,628.67	24.97	440.53	5.64	5.13	902	868	339.10	308.27	22,523	21,674
Lutz *et al.* [1]	28,301.33	16.20	617.28	7.90	N/A	735	843	279.18	N/A	11,907	13,657
Liu *et al.* [18]	15,335.67	29.70	380.37	4.74	4.31	1,010	1,050	309.13	281.03	29,997	31,185
Mathew *et al.* [2]	21,429.33	30.80	357.14	4.57	4.16	1,390	1,850	213.33	193.93	42,812	56,980
Previous version [4]	18,013.00	16.28	675.68	8.65	7.86	569	507	480.13	436.49	9,263	8,254
This work	17,368.67	15.84	694.44	8.89	8.08	465	484	511.78	465.25	7,366	7,667

where $\mathrm{KeyEx}_{\mathrm{offset}}$ denotes KeyEx with the offset correction, in which the linear mapping of the affine transformation in SubWord over the tower field (i.e., $\Psi = \Delta(A(\Delta'(x)))$) is replaced with $\Phi(x) = \Delta(\gamma(A(\gamma(\Delta'(x)))))$. Similarly, the inverse key scheduling with the offset for decryption is represented by

$$\begin{cases} l_0^{(r-1)} &= l_0^{(r)} + \mathrm{KeyEx}_{\mathrm{offset}}(l_2^{(r)} + l_3^{(r)}) \\ l_1^{(r-1)} &= l_0^{(r)} + l_1^{(r)} \\ l_2^{(r-1)} &= l_1^{(r)} + l_2^{(r)} \\ l_3^{(r-1)} &= l_2^{(r)} + l_3^{(r)} \end{cases}. \tag{17}$$

Thus, we can perform AddRoundKey (and AddInitialKey) with the offset by replacing $k_{i,j}^{(r)}$ and KeyEx in Eqs. (14) and (15) with $l_{i,j}^{(r)}$ and $\mathrm{KeyEx}_{\mathrm{offset}}$, respectively.

The proposed encryption/decryption hardware with the multiplicative-offset can be realized by replacing the operations in Figs. 1, 3, and 4 with the corresponding ones for multiplicative-offset. Since we use common Δ_γ and $\Delta'_{\gamma^{-1}}$ for encryption and decryption as shown in Fig. 5 thanks to the offset correction at each round, the multiplicative-offset can be applied to our architecture in Figs. 1, 3, and 4 without any overhead. Thus, we can increase the variety of conversion matrices by 255 times because γ can take a value from 255 candidates. The optimal conversion matrices can be searched in an exhaustive manner. Consequently, we found a set of conversion matrices with a Hamming weight of 4,016 in total while we found no conversion matrices set with Hamming weight of less than 4,416 without multiplicative-offset when we used the state-of-the-art tower-field inversion in [8]. Roughly speaking, the multiplicative-offset method reduces the circuit area for linear mappings by approximately 9 percent without any overhead. Note that a set of conversion matrices with a low Hamming weight is useful for reducing the fan-out of each XOR gate, which is related to the latency and power consumption.

3.4 Performance Evaluation

Tables 1 and 2 summarize the synthesis results of the proposed AES encryption/decryption architecture by the Synopsys Design Compiler (Version D2010-3) with the NanGate 45 nm open-cell library [22] under the worst-case conditions, where Area indicates a two-way NAND equivalent gate size (i.e., gate equivalents (GEs)); Latency indicates the latency for one block encryption/decryption, which is estimated from the circuit path delay of the datapath under the worst-low condition; Max. freq. indicates the maximum operation frequency obtained from the critical path delay; Throughput indicates the throughput at the maximum operation frequency in which 10 cycles and 11 cycles indicate the throughput respectively, if and unless the plaintext and ciphertext blocks are input and output simultaneously in one clock cycle (see the next paragraph); Power@100 MHz indicates the power consumption estimated by a Monte-Carlo gate-level timing simulation with back-annotation, in which Enc. and Dec. indicates that for encryption and decryption, respectively; Efficiency indicates the throughput per area; and PL product indicates the power-latency product. To conduct a practical performance comparison, an area optimization (which maximizes the effort of minimizing the number of gates without flattening the description) was applied in Table 1, and an area-speed optimization (where an asymptotical search with a set of timing constraints was performed after the area optimization) was applied in Table 2. Fig. 6 also shows the latency and area of the synthesized proposed AES encryption/decryption architecture.

In these tables and figures, the conventional representative datapaths [1], [2], [9], [21] and that in the previous version [7] were also synthesized by using the same conditions. The source codes for these syntheses were described by the authors referring to [1], [2], [9], [21],[3] except for the source codes of Satoh's and Canright's S-boxes in [9], [12] that can be obtained from their websites [23], [24]. To fairly evaluate and compare the performance of datapaths without the

3. According to [2], the $GF(2^4)$ inversion in the $GF((2^4)^2)$ inversion circuit in [2] can be implemented with a $T_{\mathrm{XOR}} + 3T_{\mathrm{NAND}}$ delay, where T_{NAND} denotes the delay of the NAND gates. However, there is no detailed description to realize such a circuit. Therefore, we described the circuit by a direct mapping based on the PPRM expansion, which is an algebraic normal form frequently used to design GF arithmetic circuits [11], [25].

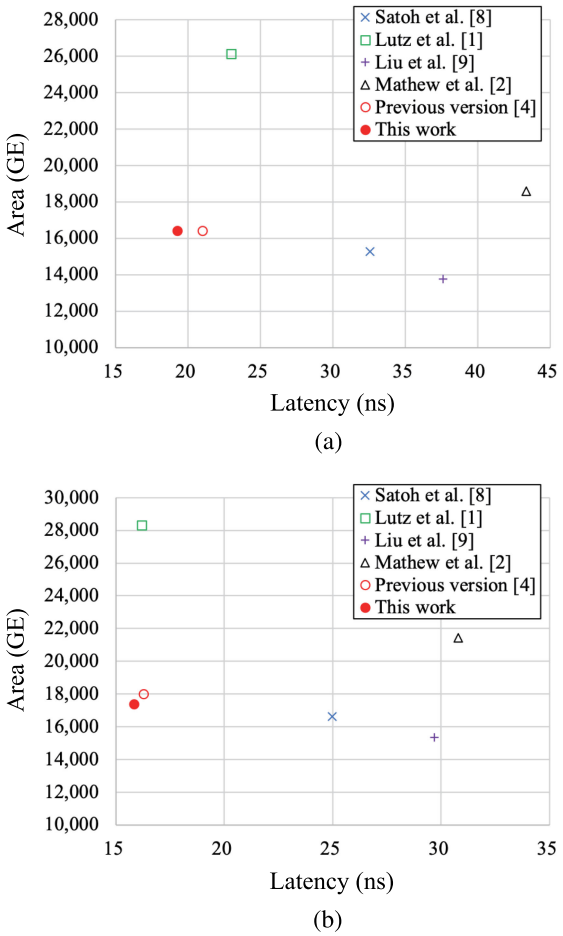

Fig. 6. Latency and area of AES encryption/decryption hardware architectures with (a) area and (b) area-speed optimization.

effect of pipelining, the datapaths of [1] and [2] were adjusted to the round-based nonpipelined architecture corresponding to the proposed datapath. Latency was calculated by assuming that the datapath of [1] requires 10 clock cycles to perform each encryption or decryption and the others require 11 clock cycles. This is because the initial key addition and first-round computation are performed with one clock cycle for [1]. On the other hand, Throughput of architectures except Lutz's [1]. is calculated assuming that they require 10 and 11 cycles to perform encryption/decryption per block. These architecture can perform consecutive encryption/decryption with 10 clock cycles per block if plaintext and ciphertext are input and output simultaneously in one clock cycle; otherwise, they require 11 cycles. Note that, in Lutz's [1] architecture, plaintext and ciphertext cannot be input and output simultaneously in one clock cycle.. Area includes the initial key, round key, data registers, and control logic in addition to combinational circuits for round datapaths. Note also that the key scheduling parts of [1] and [2] were implemented with the ones presented in this paper because there was no description for the key scheduling parts. (For [1], the isomorphic mapping was removed for application to the round function part.)

The results in Tables 1 and 2 show that our datapath achieves the lowest latency and highest efficiency compared with the conventional ones with tower-field inversion circuits. Although all operations are translated to the tower field in our architecture, the area and delay overheads of MixColumns and InvMixColumns are suppressed by the unification technique. More precisely, the critical path delay of the proposed datapath is given by only $T_{\mathrm{INV}}+T_{\mathrm{DeMUX}}+6T_{\mathrm{XOR}}+T_{4:1\mathrm{SEL}}$, where T_{INV}, T_{DeMUX}, and $T_{4:1\mathrm{SEL}}$ denote the delay for the inversion circuit, demultiplexer, and 4-to-1 selector, respectively. The delay of $6T_{\mathrm{XOR}}$ includes the unified linear operation ($3T_{\mathrm{XOR}}$), adder array ($2T_{\mathrm{XOR}}$), and AddRoundKey (T_{XOR}) in Round datapath. Since the critical path of conventional architectures includes at least three linear operations, adder array, one AddRoundKey, one inversion, and three selectors, the critical path delay of conventional ones should be given by larger than $T_{\mathrm{INV}}+12T_{\mathrm{XOR}}+3T_{2:1\mathrm{SEL}}$, where $T_{2:1\mathrm{SEL}}$ denotes the delay for a 2-in-1 selector. Thus, the lower logic depth of proposed datapath leads to the lower latency than the conventional ones, as indicated in Tables 1 and 2. Especially, even with a tower-field S-box, our architecture has an advantage with regard to the latency over Lutz's one with table-lookup-based inversion (i.e., very small T_{INV}). In addition, we can also confirm that the optimization of linear mappings based on multiplicative-offset clearly improves the area-time efficiency. As a result, our architecture is 51–63 percent more efficient in terms of the throughput per area than the conventional best. More precisely, the proposed datapath achieves approximately 63 percent higher efficiency than any conventional architecture and 9 percent more efficient than the previous version with area optimization. In addition, the proposed datapath has 51 and 7 percent higher throughput/gate efficiency than the conventional best and previous version when using area-speed optimization, respectively.

The results also suggest that the proposed architecture would perform an AES encryption or decryption with 38–46 and 44–51 percent smaller energy than the conventional best, respectively. More precisely, the proposed architecture reduces the power by 35–38 and 38–43 percent than the conventional best for encryption and decryption, respectively, because of the compressed datapath and the cutoff of an inactive path by a demultiplexer. Thanks to the lower latency and lower power consumption, the proposed architecture improves power-latency efficiency by 46 and 51 percent than conventional ones at encryption and decryption with area optimization, respectively. In the case of area-speed optimization, the improvement of power-latency efficiency is given by 38 and 44 percent for encryption and decryption, respectively. These results indicate that the proposed architecture can perform AES encryption and decryption with the lowest energy consumption. In addition, the proposed architecture improves the PL product by 23 percent (9 percent) and 20 percent (7 percent) at encryption (decryption) than the previous version with area and area-speed optimization, respectively, because of the multiplicative-offset. We can also confirm the advantage of the proposed architecture over the previous version in terms of power/energy consumption as well as the efficiency.

The performance of the architecture in [2] was relatively lower for our experimental conditions because its critical path includes InvMixColumns for round key for operation-reordering and therefore becomes longer than those of other designs. In addition, InvMixColumns over a tower-field is

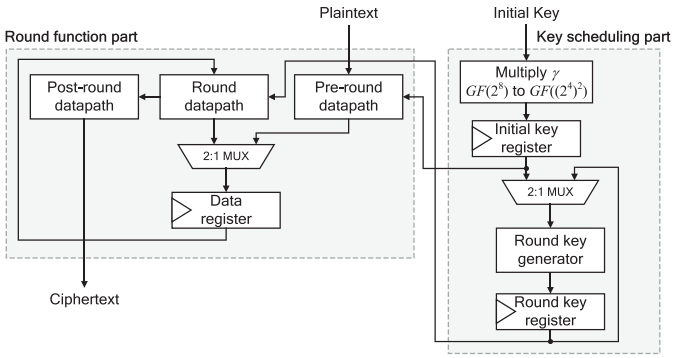

Fig. 7. Overall architecture of proposed AES encryption hardware.

more area-consuming than that over the AES field. This suggests that the architecture in [2] is not suitable for an on-the-fly key scheduling implementation. The architectures in [9], [21] have smaller areas than the proposed architecture; however, our architecture has a higher throughput. The increasing ratio of the throughput is larger than that of the circuit area because the architectures in [9], [21] use InvMixColumns to compute InvMixColumns for round key and require several additional selectors, respectively. Moreover, the proposed architecture has a higher efficiency than the previous version, because the multiplicative-offset can reduce the implementation cost of linear mappings as described in Section 3.3.

4 Proposed Encryption Architecture

4.1 Round Function Part

While unified AES encryption/decryption architecture is very important for many existing practical applications, AES hardware that supports only encryption is also highly in demand owing to the wider spread of the counter (CTR)-mode and inverse-free authenticated encryptions working with AES such as Galois/Counter-Mode (GCM) [26] and Offset Two Round (OTR) [27].

In this section, we propose an efficient AES encryption hardware architecture based on the same philosophy as Section 3. Fig. 7 shows the overview of the proposed AES encryption hardware architecture, and Fig. 8 shows the proposed round function part for encryption only, which are based on Eq. (14) and are basically derived by omitting the Decryption path of the proposed unified architecture shown

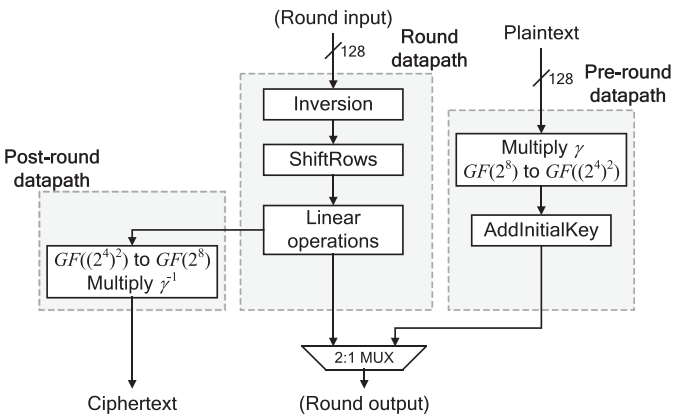

Fig. 8. Proposed round function part for encryption hardware.

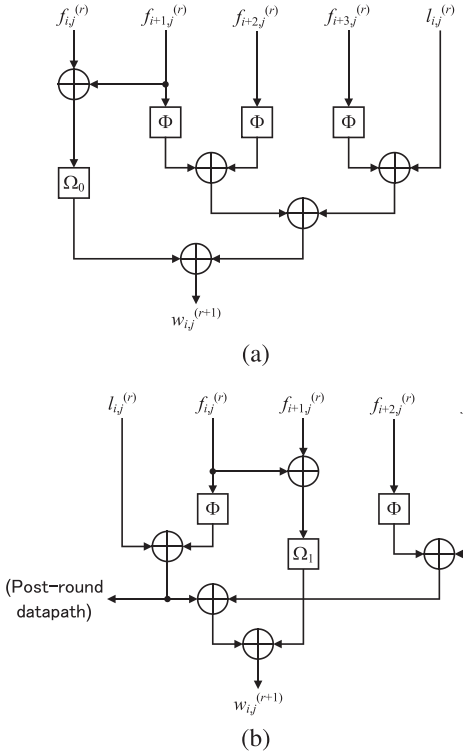

Fig. 9. Realization of Linear operations based on MixColumns factorization using (a) Ω_0 and (b) Ω_1.

in Figs. 1, 3, and 4. Note that the multiplicative-offset is applied to the architecture in Fig. 7.

Here, we focus on the construction of "Linear operations," which performs the operations corresponding to Unified affine, Adder array, and AddRoundKey of the Encryption path in Fig. 3. Fig. 9 shows the block diagram of Linear operations for the jth column, where $f_{i,j}$ denotes the jth byte of input of Unified affine and Φ denotes the affine transformation with the multiplicative-offset over the tower field (i.e., $\Phi(f_{i,j}) = \Delta(\gamma(A(\gamma(\Delta'(f_{i,j}))))) + \Delta(\gamma c) = \Omega_2 = \Omega_3$), which is given by the same manner as the encryption/decryption architecture in Section 3. As shown in [20], according to $u_0 = \beta$ and $u_1 = \beta + 1$ in the MixColumn function, we implement only either Ω_0 or Ω_1 (corresponding to Fig. 9a or Fig. 9b, respectively) in our architecture because MixColumn function $\beta g_{i,j}^{(r)} + (\beta+1)g_{i+1,j}^{(r)} + g_{i+2,j}^{(r)} + g_{i+3,j}^{(r)}$ can be factorized as $\beta(g_{i,j}^{(r)} + g_{i,j+1}^{(r)}) + g_{i,j+1}^{(r)} + g_{i,j+2}^{(r)} + g_{i,j+3}^{(r)}$ or $(\beta+1)(g_{i,j}^{(r)} + g_{i,j+1}^{(r)}) + g_{i,j}^{(r)} + g_{i,j+2}^{(r)} + g_{i,j+3}^{(r)}$, where $g_{i,j}^{(r)}$ denotes the ith column and jth row input byte of MixColumns at the rth round over the AES field. This indicates that, because of the linearity of isomorphic mappings and multiplicative-offset, Linear operations can be realized by $w_{i,j}^{(r+1)} = \Omega_0(f_{i,j}^{(r)} + f_{i+1,j}^{(r)}) + \Phi(f_{i+1,j}^{(r)}) + \Phi(f_{i+2,j}^{(r)}) + \Phi(f_{i+3,j}^{(r)}) + l_{i,j}^{(r)}$ or $w_{i,j}^{(r+1)} = \Omega_1(f_{i,j}^{(r)} + f_{i+1,j}^{(r)}) + \Phi(f_{i,j}^{(r)}) + \Phi(f_{i+2,j}^{(r)}) + \Phi(f_{i+3,j}^{(r)}) + l_{i,j}^{(r)}$. In the proposed architecture, we employ the realization of (b) based on $\Omega_1(f_{i,j}^{(r)} + f_{i+1,j}^{(r)}) + \Phi(f_{i,j}^{(r)}) + \Phi(f_{i+2,j}^{(r)}) + \Phi(f_{i+3,j}^{(r)}) + l_{i,j}^{(r)}$, because the final round AddRoundKey can be efficiently implemented by the term $\Phi(f_{i,j}^{(r)})$, while $\Phi(f_{i,j}^{(r)})$ does not appear in another formula. In Fig. 9b, the output denoted by "Post-round datapath" is

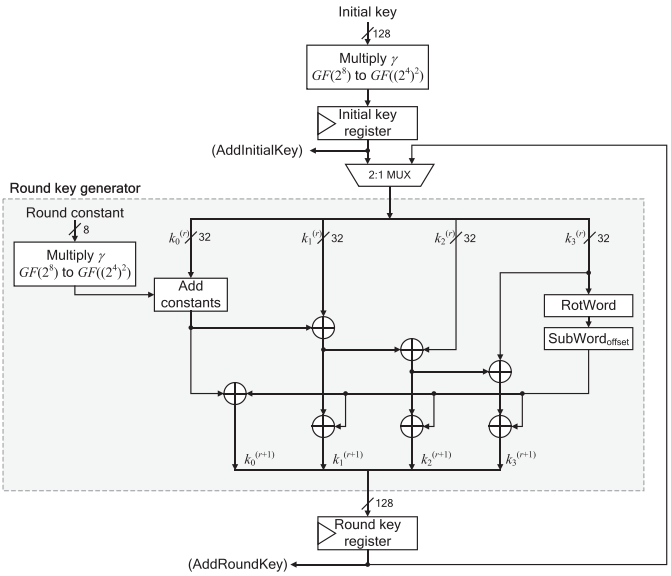

Fig. 10. Proposed key scheduling part for encryption hardware.

used for AddRoundKey at the final round with bypassing of MixColumns. Note that unified affine transformations and some XOR gates in Linear operations should be shared in an identical column (i.e., for the calculations of $w_{0,j}^{(r+1)}$, $w_{1,j}^{(r+1)}$, $w_{2,j}^{(r+1)}$, and $w_{3,j}^{(r+1)}$ for each j) for savings in the circuit area.

Linear mappings over $GF(2^8)$ including Ω_1 and Φ usually require a delay of $3T_{\text{XOR}}$, where T_{XOR} denotes the delay of a two-way XOR gates. This indicates that Linear operations of Fig. 9 requires a $6T_{\text{XOR}}$ delay because the delay of Φ is usually given by $3T_{\text{XOR}}$. On the other hand, if we find an optimal conversion matrix for Φ with a $2T_{\text{XOR}}$ delay, we can reduce the delay of Linear operations and can implement it with only a $5T_{\text{XOR}}$ delay. Actually, while we found no such a matrix with a $2T_{\text{XOR}}$ delay without multiplicative-offset, we successfully found a few optimal matrices in an exhaustive search of γ. Thus, the proposed encryption architecture has a low latency, and we can confirm again the effectiveness of multiplicative-offset.

Thanks to the register-retiming, operation-reordering, and multiplicative-offset, our encryption architecture has the lowest logic depth among conventional architectures. Let T_{AND} be the delay of a two-way AND gate. The architecture presented by Nekado *et al*. [20] has the delay of $4T_{\text{AND}} + 13T_{\text{XOR}}$,[4] which was lower than any other conventional tower-field architecture until now, to the best of our knowledge. The proposed architecture can achieve the delay of $3T_{\text{AND}} + 11T_{\text{XOR}} + T_{2:1\text{SEL}}$, where $T_{2:1\text{SEL}}$ denotes the delay of a two-in-one selector. Note that, as in the encryption/decryption hardware in Section 3, the critical path should be on the Round datapath part but not Post-round datapath in Fig. 7b, because the adder array and multiplexer in Round datapath would be longer than $GF((2^4)^2)$ to $GF(2^8)$ in Post-round datapath. Thus, we can confirm the higher efficiency of the proposed architecture, given that a 2-in-1 multiplexer has a delay similar to a two-way logic gate such as AND and XOR gates [10].

4.2 Key Scheduling Part

Fig. 10 shows the proposed key scheduling part datapath for the encryption hardware. The encryption key scheduling part is basically derived by removing the decryption datapath from the encryption/decryption key scheduling part shown in Fig. 4 as well as the round function part, and it can be implemented according to Eq. (16). However, in contrast to the encryption/decryption key scheduling part, some XOR gates excluding the output of $\text{KeyEx}_{\text{offset}}$ should be factorized for achieving a smaller area, because the factorization can be performed without degrading the critical delay nor changing the functionality as shown in Fig. 10.

4.3 Performance Evaluation

Tables 3 and 4 show the performance of the proposed AES encryption hardware, and these data were derived with the same conditions as those in Tables 1 and 2 in Section 3.4. For comparison, we also show performance evaluation results for the conventional hardware.[5] As typical architectures, we evaluated two AES encryption hardware architectures denoted by SASEBO IPs [23], which are published as open-source intellectual property cores (IPs) for Side-channel Attack Standard Evaluation BOards (SASEBOs). In the columns of SASEBO IPs [23], "Table" and "Tower field" indicate that the hardware architectures are based on a table-based and tower-field S-boxes, respectively. The source codes for these architectures were derived from [23]. In addition, we also evaluated state-of-the-art architectures presented by Gueron and Mathew in [28]. In the column of Gueron and Mathew [28], "Native" and "Mapped" indicate that the all round operations and only SubBytes were performed over a tower field, respectively. Since we found no public source codes for the architectures of [28], the source codes for these syntheses were described by the authors referring to [28], as well as Section 3.4. SASEBO IPs require 12 clock cycles for one block encryption, while the proposed architecture and the architecture by Gueron and Mathew can be implemented such that one block encryption is performed with 11 clock cycles, according to their description. In Tables 3 and 4, the rows denoted by "10 (11) cycles" and "11 (12) cycles" denote the throughput or efficiency in the case and not the case that the plaintext and ciphertext blocks are input and output simultaneously in one clock cycle, respectively. Fig. 11 also shows the latency and area obtained by the synthesis results.

In Tables 3 and 4, while the SASEBO IP with the table-based S-box has a shorter latency and higher throughput than our architecture, the table-based S-box requires an approximately two times larger circuit area than ours. The SASEBO IP with a tower-field S-box has a smaller circuit area, while our architecture has a shorter latency, which results in higher efficiency of our architectures. Here, it is interesting that the SASEBO IP with a table-based S-box has a lower power consumption than that with a tower-field

4. The delay does not include that of multiplexers because neither concrete implementation, datapath, nor architecture was shown in [20]; and therefore, a quantitative comparisons and evaluations with [20] are difficult in this paper.

5. Since the previous version [7] did not present encryption architecture, we do not compare the proposed encryption architecture with the previous version.

TABLE 3
Synthesis Results for Proposed and Conventional AES Encryption Hardware Architectures With Area Optimization

		Area (GE)	Latency (ns)	Max. freq. (MHz)	Throughput (Gbps) 10 (11) cycles	Throughput (Gbps) 11 (12) cycles	Power (μW) @100MHz	Efficiency (Kbps/GE) 10 (11) cycles	Efficiency (Kbps/GE) 11 (12) cycles	PL product
SASEBO	Table	21,291.67	16.68	719.42	8.10	7.67	373	393.18	360.42	6,222
IPs [23]	Tower field	10,529.33	30.72	390.63	4.55	4.17	536	431.69	395.72	16,466
Gueron and	Mapped	12,467.67	26.51	414.94	5.31	4.83	588	426.00	387.27	15,588
Mathew [28]	Native	11,311.00	28.38	387.60	4.96	4.51	619	428.62	398.75	17,567
This work		11,257.33	17.38	632.91	8.10	7.36	295	719.64	654.22	5,127

TABLE 4
Synthesis Results for Proposed and Conventional AES Encryption Hardware Architectures With Area-Speed Optimization

		Area (GE)	Latency (ns)	Max. freq. (MHz)	Throughput (Gbps) 10 (11) cycles	Throughput (Gbps) 11 (12) cycles	Power (μW) @100MHz	Efficiency (Kbps/GE) 10 (11) cycles	Efficiency (Kbps/GE) (12) cycles	PL product
SASEBO	Table	23,085.00	11.64	1,030.93	12.00	11.00	352	519.66	476.35	4,097
IPs [23]	Tower field	11,431.67	23.04	520.83	6.06	5.56	513	530.16	485.98	11,820
Gueron and	Mapped	13,249.33	21.78	505.05	6.46	5.88	655	487.92	443.57	14,266
Mathew [28]	Native	12,108.33	23.87	460.83	5.90	5.36	755	487.16	442.87	18,022
This work		12,127.00	13.97	787.40	10.08	9.16	279	831.10	755.54	3,898

S-box in spite of their circuit area, rather than the tower-field S-boxes by hand at gate level. The tower-field S-box is known as a power-consuming circuit due to dynamic hazards [11], and Design Compiler would be good at synthesizing the table-based S-box with consideration of not only the low latency but also the low power consumption. Nevertheless, our architectures have a lower power consumption and lower (or at least comparable) power-latency product

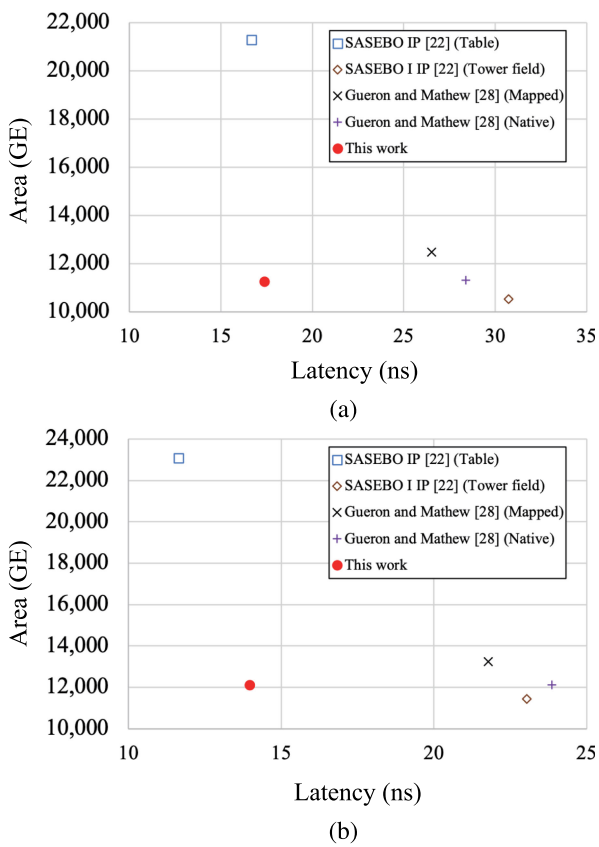

Fig. 11. Latency and area of AES encryption hardware architectures with (a) area and (b) area-speed optimization.

than SASEBO IP with a table-based S-box, which clearly shows the usefulness of the proposed unification techniques of linear mappings and multiplicative-offset. In addition, we can also confirm that our architectures have higher efficiency and lower power/energy consumption than state-of-the-art architectures by Gueron and Mathew, thanks to the unification technique and multiplicative-offset. Thus, we can confirm that the proposed architecture achieves 64 and 58 percent higher efficiency in terms of bps/GE than conventional representative and state-of-the-art architectures with area and area-speed optimization, respectively. We can also confirm the advantages of the proposed architecture in terms of efficiency and power/energy consumption.

The critical path delay of the proposed datapath is given by $T_{\mathrm{INV}} + 5T_{\mathrm{XOR}} + T_{2:1\mathrm{SEL}}$, where $5T_{\mathrm{XOR}}$ corresponds to the delay of Linear operation. The conventional architectures except for SASEBO IP with table-based S-box requires greater than $T_{\mathrm{INV}} + 8T_{\mathrm{XOR}} + T_{2:1\mathrm{SEL}}$ or $T_{\mathrm{INV}} + 11T_{\mathrm{XOR}} + T_{2:1\mathrm{SEL}}$, because their critical paths include two or three linear operations. In contrast, SASEBO IP with table-based S-box has a delay of $T_{\mathrm{INV}} + 7T_{\mathrm{XOR}} + T_{2:1\mathrm{SEL}}$ with a very small T_{INV} by the table-based S-box. As aforementioned, SASEBO IP with table-based S-box has a lower latency than the proposed architecture. However, the latency of proposed architecture is still comparable to the SASEBO IP with table-based S-box as shown in Tables 3 and 4, whereas the proposed architecture has approximately 47 percent smaller circuit area with both area and area-speed optimization. This also indicates the area-latency efficiency of the proposed architecture.

5 Discussion

5.1 Comparison With Other Conventional Architectures

The above comparative evaluation was done with the proposed and some conventional but representative datapaths. There are other previous works focusing on area-time efficiency by round-based architectures. However, such previous works do not provide for concrete implementation

and/or exhibit better performance than the abovementioned conventional datapaths. For example, a hardware AES description with a short critical path was presented in [20], which employed an redundant representation and unification techniques to reduce the critical path delay. However, we could not evaluate the efficiency by ourselves because of the lack of a detailed description about the implementation. The AES processor in [29] is also a typical high throughput encryption architecture, but we cannot evaluate it in this paper because of the lack of descriptions of its building blocks. Another AES encryption/decryption architecture with a high throughput was presented in [21]. However, the architecture had a lower throughput/area efficiency compared to the architecture in [9] according to that paper. Moreover, low latency AES encryption and decryption architectures based on twisted-BDD and hardware T-box in [10] were not evaluated in this paper because this would require a sophisticated back-end design (i.e., place and route).

5.2 Possibility of Pipelining

The proposed design employs a round-based architecture without block-wise parallelism such as pipelining. However, the block-wise parallelism are exploitable in the parallelizable modes and authenticated encryptions (e.g., CTR mode, GCM [26], OTR [27], and Offset CodeBook (OCB) [30]), by the trade-off between the area and the throughput by pipelining [28], [31]. A simple way to obtain a pipelined version of the proposed architecture is to unroll the rounds and insert pipeline registers between them. The datapath can be further pipelined by inserting registers into the round datapath. The proposed datapath can be efficiently pipelined by placing the pipeline register at the output of the inversion with a good delay balance between the inversion and the following circuit. For example, the synthesis results for the proposed datapath using the area-speed optimization with the NanGate 45-nm standard-cell library indicated that the inversion circuit had a delay of 0.60 ns, and the remainder had a delay of 0.66 ns. Accordingly, pipelining would achieve a throughput of 17.63 Gbps, which is nearly twice that without pipelining. Thus, the proposed datapath is also suitable for pipelined implementation.

5.3 Choice of S-box Implementation

We employed a tower-field S-box presented by Ueno *et al.* [8], [15]. Actually, Ueno's S-box is the most efficient S-box architecture applicable to our AES hardware architecture, which employ tower-field arithmetic, decomposition of (inv)S-box into inversion and (inv)affine transformation, and merge of linear operations including isomorphic mapping.

There are several tower-field S-box implementation that have smaller area, lower latency, and/or higher efficiency than the above one. For example, Boyar *et al.* presented a very small S-box description based on a logic minimization [16]. In 2018, Reyhani-Masoleh *et al.* presented smaller and more efficient S-boxes than Boyar's and Ueno's S-boxes on the basis of architectural and gate-level optimizations. In 2019, Maximov and Ekdahl further improved the performance of AES S-box [18] by means of new logic minimization techniques. They presented three S-box designs, each of which are the fastest, smallest, or most efficient until now. However, these S-boxes cannot be applied to our architecture. These S-boxes unifies isomorphic mapping, affine transformation, and linear operations in inversion as a big XOR matrix operation at the input and/or output of S-box to achieve smaller area and lower latency. Therefore, these (inv)S-boxes cannot be decomposed into inversion and (inv)affine transformation, and cannot be applied to our AES hardware architecture that utilizes the S-box decomposition. An extension of our architecture for efficient application of such S-boxes remains in future work.

In contrast to tower-field S-boxes, direct-mapping-based (or table-based) S-box architecture including BDD-based ones can be implemented with very lower latency and relatively small power consumption at costs of circuit area and efficiency. However, we should implement table-based inversion and (inv)affine transformation separately in our architecture whereas the table can directly implement the functionality of S-box, which indicates that the decomposition of S-box can be latency and area overheads for the table-based implementation. In addition, our optimization techniques for unifying linear operations may not work well with the table-based implementation, because the table-based implementation does not require isomorphic mappings which are efficiently unified in our architecture.

Thus, the tower-field S-box which can be efficiently decomposed into inversion and (inv)affine transformation like Ueno's S-box [8], [15] is suitable to our architecture that is intended to achieve a higher throughput/gate efficiency (and S-boxes in, for example, [9], [12], [13] are possible candidates).

5.4 Application of Countermeasure Against Side-Channel Attacks

Another discussion point is how the proposed architecture can be resistant to side-channel attacks, especially against differential power analyses (DPAs) [32]. A masking countermeasure would be based on a masked tower-field inversion circuit [33]. The major features of the countermeasure are to replace the inversion with a masked inversion and to duplicate other linear operations. Such a countermeasure can also be applied to the proposed datapath. In addition, hiding countermeasures, such as wave dynamic differential logic (WDDL) [34], which replaces the logic gates with a complementary logic style, would also be applicable, and the hardware efficiency would be proportionally lower with respect to the results in Tables 1 and 2.

More sophisticated masking-based countermeasures such as threshold implementation (TI) and a consolidated masking scheme (CMS) [35], [36] would also be applicable to the proposed datapath in principle in the same manner as other conventional ones. On the other hand, such countermeasures, especially against higher-order DPAs, require a considerable area overhead and more random bits compared with the aforementioned countermeasures. When applying such countermeasures, the area overhead would be critical for some applications. In addition, TI- and CMS-based inversion circuits should be pipelined to reduce the resulting circuit area (i.e., the number of shares). To divide the circuit delay equally, it would be better to insert a pipeline register at the middle of the Encryption and Decryption path in Fig. 3.

6 Conclusion

This paper presented a new efficient round-based AES architecture that supports encryption only and both encryption and decryption. The proposed datapath utilizes new operation-reordering and register-retiming techniques to unify critical components with fewer additional selectors. Consequently, Our datapath has the lowest critical path delay compared to conventional ones with tower-field S-boxes. We also presented a new technique for optimizing matrices for linear operations named multiplicative-offset. The multiplicative-offset can improve the efficiency of AES hardware architecture by approximately 9 percent without any overhead. The proposed and conventional AES datapaths were implemented with compatible round-based architectures and evaluated by logic synthesis with the NanGate 45-nm open-cell library. The synthesis results suggested that the proposed architecture was approximately 51–64 percent more efficient than the best conventional architecture in terms of the throughput per area. In addition, as a result of gate-level timing simulations with back-annotation, we also confirmed that the proposed architecture can perform encryption/decryption with the lowest power/energy consumption.

Acknowledgments

This research has been supported by JSPS KAKENHI Grant No. 17H00729 and No. 19K21526, and JST PRESTO Grant No. JPMJPR18M3.

References

[1] A. Lutz *et al.*, "2 Gbit/s hardware realizations of RIJNDAEL and SERPENT: A comparative analysis," in *Proc. Int. Workshop Cryptographic Hardware Embedded Syst.*, 2002, pp. 144–158.

[2] S. K. Mathew *et al.*, "53 Gbps native $GF(2^4)^2$ composite-field AES-encrypt/decrypt accelerator for content-protection in 45 nm high-performance microprocessors," *IEEE J. Solid-State Circuits*, vol. 46, no. 4, pp. 767–776, Apr. 2011.

[3] S.-Y. Lin and C.-T. Huang, "A high-throughput low-power AES cipher for network applications," in *Proc. IEEE Asia South Pacific Design Autom. Conf.*, 2007, pp. 595–600.

[4] M. Liskov, R. L. Rivest, and D. Wagner, "Tweakable block ciphers," *J. Cryptology*, vol. 24, no. 3, pp. 588–613, 2011.

[5] K. Minematsu, "Beyond-birthday-bound security based on tweakable block cipher," in *Proc. Int. Workshop Fast Softw. Encryption*, 2009, pp. 308–326.

[6] Y. Naito, "Tweakable Blockciphers for efficient authenticated encryptions with beyond the birthday-bound security," *IACR Trans. Symmetric Cryptology*, no. 2, pp. 1–26, 2017.

[7] R. Ueno, S. Morioka, N. Homma, and T. Aoki, "A high throughput/gate AES hardware architecture by compressing encryption and decryption datapaths—toward efficient CBC-mode implementation," in *Proc. Int. Conf. Cryptographic Hardware Embedded Syst.*, 2016, pp. 538–558.

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

[9] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A compact Rijndael hardware architecture with S-box optimization," in *Proc. Int. Conf. Theory Appl. Cryptology Inf. Security*, 2001, pp. 239–254.

[10] S. Morioka and A. Satoh, "A 10 Gbps full-AES crypto design with a twisted-BDD S-Box architecture," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 12, no. 7, pp. 686–691, Jul. 2004.

[11] S. Morioka and A. Satoh, "An optimized S-Box circuit architecture for low power AES design," in *Proc. Int. Workshop Cryptographic Hardware Embedded Syst.*, 2002, pp. 172–186.

[12] D. Canright, "A very compact S-box for AES," in *Proc. Int. Workshop Cryptographic Hardware Embedded Syst.*, 2005, pp. 441–455.

[13] Y. Nogami, K. Nekado, T. Toyota, N. Hongo, and Y. Morikawa, "Mixed bases for efficient inversion in $\mathbb{F}_{((2^2)^2)^2}$ and conversion matrices of SubBytes of AES," in *Proc. Int. Workshop Cryptographic Hardware Embedded Syst.*, 2010, pp. 234–247.

[14] A. Rudra, P. K. Dubey, C. S. Jutla, V. Kumar, J. Rao, and P. Rohatgi, "Efficient Rijndael encryption implementation with composite field arithmetic," in *Proc. Int. Workshop Cryptographic Hardware Embedded Syst.*, 2001, pp. 171–184.

[15] R. Ueno, N. Homma, Y. Sugawara, Y. Nogami, and T. Aoki, "Highly efficient $GF(2^8)$ inversion circuit based on redundant GF arithmetic and its application to AES design," in *Proc. Int. Workshop Cryptographic Hardware Embedded Syst.*, 2015, pp. 63–80.

[16] J. Boyar, P. Matthews, andP. Peralta, "Logic minimization techniques with applications to cryptology," *J. Cryptology*, vol. 47, no. 2, pp. 280–312, 2013.

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

[18] A. Maximov and P. Ekdahl, "New circuit minimization techniques for smaller and faster AES SBoxes," *IACR Trans. Crypt. Hardware Embedded Syst.*, pp. 91–125, 2019.

[19] I. Hammad, K. El-Sankary, and E. El-Masry, "High-speed AES encryptor with efficient merging techniques," *IEEE Embedded Syst. Lett.*, vol. 2, no. 3, pp. 67–71, Sep. 2010.

[20] K. Nekado, Y. Nogami, and K. Iokibe, "Very short critical path implementation of AES with direct logic gates," in *Proc. Int. Workshop Security*, 2012, pp. 51–68.

[21] P.-C. Liu, H.-C. Chang, and C.-Y. Lee, "A 1.69 Gb/s area-efficient AES crypto core with compact on-the-fly key expansion unit," in *Proceedings of ESSCIRC*, 2009, pp. 404–407.

[22] "NanGate FreePDK45 open cell library," Jan. 2016, [Online]. Available: http://www.nangate.com/?page_id=2325

[23] Tohoku University, "Cryptographic hardware project," 2015. [Online]. Available: http://www.aoki.ecei.tohoku.ac.jp/crypto/

[24] D. Canright, "Canright web page," 2015. [Online]. Available: http://faculty.nps.edu/drcanrig/

[25] T. Sasao, "And-Exor expressions and their optimization," in *Logic Synthesis and Optimization*, vol. 212. New York, NY, USA: Kluwer Academic Publishers, 1993, pp. 287–312.

[26] D. A. McGrew and J. Viega, "The Galois/Counter Mode of operation (GCM)," 2005. [Online]. Available: http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/gcm-revised-spec.pdf

[27] K. Minematsu, "Parallelizable rate-1 authenticated encryption from pseudorandom functions," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, 2014, pp. 275–292.

[28] S. Gueron and S. Mathew, "Hardware implementation of AES using area-optimal polynomials for composite-field representation $GF(2^4)^2$ of $GF(2^8)$," in *Proc. IEEE 23nd Symp. Comput. Arithmetic*, 2016, pp. 112–117.

[29] I. Verbauwhede, P. Schaumont, and H. Kuo, "Design and performance testing of a 2.29-GB/s Rijndael processor," *IEEE J. Solid-State Circuits*, vol. 38, no. 3, pp. 569–572, Mar. 2003.

[30] P. Rogaway, N. Bellare, J. Black, and T. Krovetz, "OCB: A block-cipher mode of operation for efficient authenticated encryption," *ACM Trans. Inf. Syst. Security*, vol. 6, no. 3, pp. 365–403, 2003.

[31] A. Hodjat and I. Verbauwhede, "Area-throughput trade-offs for fully pipelined 30 to 70 Gbits/s AES processors," *IEEE Trans. Comput.*, vol. 50, no. 4, pp. 366–372, Apr. 2006.

[32] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proc. Annu. Int. Cryptology Conf.*, 1999, pp. 388–397.

[33] E. Oswald, S. Mangard, N. Pramstaller, and V. Rijmen, "A side-channel analysis resistant description of the AES S-box," in *Proc. Int. Workshop Fast Softw. Encryption*, 2005, pp. 413–423.

[34] K. Tiri and I. Verbauwhede, "A logic level design methodology for a secure DPA resistant ASIC or FPGA implementation," in *Proc. Conf. Design Autom. Test Europe*, 2004, vol. 1, pp. 246–251.

[35] S. Nikova, V. Rijmen, and M. Schläffer, "Secure hardware implementation of nonlinear functions in the presence of glitches," *J. Cryptology*, vol. 24, no. 2, pp. 292–321, 2011.

[36] O. Reparaz, B. Bilgin, S. Nikova, B. Gierlichs, and I. Verbauwhede, "Consolidating masking schemes," in *Proc. Annu. Cryptology Conf.*, 2015, pp. 764–783.

Rei Ueno is an assistant professor in Research Institute of Electrical Communication, Tohoku University, and is currently working with the JST as a researcher for a PRESTO project. His research interests include arithmetic circuits, cryptographic implementations, formal verification, and hardware security. He received the Kenneth C. Smith Early Career Award in Microelectronics at ISMVL 2017. He is a member of the IEEE.

Sumio Morioka received the BE, ME, and PhD degrees in computer science from Osaka University, Japan, in 1992, 1994, and 1997, respectively. For 1997-2016, he was a senior researcher in central research laboratories of NTT, IBM, Sony and NEC, and a visiting researcher of Imperial College London. In 2016, he joined Interstellar Technologies Inc., Japan, as a chief designer of avionics system for commercial space launch vehicles. His research interests include LSI architecture, EDA, formal methods and security systems. He received the Sony MVP 2004 Award for the development of a hardware security processor for PlayStation Portable and PLAYSTATION3. He is a member of the IEEE, IEICE, and a senior member of IPSJ.

Noriyuki Miura received the BS, MS, and PhD degrees in electrical engineering from the Keio University, Yokohama, Japan. He is currently an associate professor with Kobe University, Kobe, Japan, and concurrently a JST PRESTO researcher, working on hardware security and next-generation heterogeneous computing system. He is currently serving as a TPC Member for A-SSCC and Symposium on VLSI Circuits. He received the Top ISSCC Paper Contributors 2004-2013 and the IACR CHES Best Paper Award in 2014. He is a member of the IEEE.

Kohei Matsuda received the BS and MS degrees in computer science from Kobe University, Kobe, Japan, in 2015 and 2017, respectively, where he is currently working toward the PhD degree with the Graduate School of System Informatics. His current research interests include circuit-level countermeasure against physical attacks and design methodology for cryptographic processors. He is a member of the IEEE.

Makoto Nagata received the BS and MS degrees in physics from Gakushuin University, Tokyo, in 1991 and 1993, respectively, and the PhD degree in electronics engineering from Hiroshima University, Hiroshima, in 2001. He is a professor of the Graduate School of Science, Technology and Innovation, Kobe University, Kobe, Japan. He served as a technical program chair (2010-2011) and symposium chair (2012-2013) for Symposium on VLSI circuits. He is currently chairing Technology Directions subcommittee for International Solid-State Circuits Conference (ISSCC) and an associate editor for the IEEE Transactions on VLSI Systems. He is a senior member of IEEE and IEICE.

Shivam Bhasin received the bachelor's degree from UP Tech, India, in 2007, the master's degree from Mines Saint-Etienne, France, in 2008, and the PhD degree from Telecom Paristech, in 2011. He is a senior research scientist and principal investigator at Physical Analysis and Cryptographic Engineering Laboratory, Temasek labs, Nanyang Technical University Singapore, since 2015. His research interests include embedded security, trusted computing and secure designs. Before NTU, Shivam held position of Research Engineer in Institut Mines-Telecom, France. He was also a visiting researcher at UCL, Belgium (2011) and Kobe University, Japan (2013). He regularly publishes at top peer reviewed journals and conferences. Some of his research now also forms a part of ISO/IEC 17825 Standard. He is a member of the IEEE.

Yves Mathieu is currently a full professor at Institut Mines-telecom/TELECOM ParisTech. He is the vice-chair for education of the Communication and Electronics Department. He undertakes research activities inside the "Safe and Secure Hardware" team with a focus on ASIC design.

Tarik Graba received the master degree (DEA: Diplme d'tude Approfondies) and the PhD degree in electrical engineering from Pierre et Marie Curie University (UPMC), Paris, France, in 2003 and 2006. He is currently associate professor at Institut Mines-telecom/Telecom ParisTech in the Communication and Electronics Department. His research activities include digital ASIC and system on chip design, and hardware security.

Jean-Luc Danger is currently a full professor at Institut Mines-telecom/TELECOM ParisTech. He is the head of the digital electronic system research team whose the main research topics are about security/safety of embedded systems and implementation of complex algorithms with physical constraints. He authored more than 200 scientific publications, 20 patents, and cofounded the company Secure-IC in 2010.

Naofumi Homma received the BE degree in information engineering, and the MS and PhD degrees in information sciences from Tohoku University, Sendai, Japan, in 1997, 1999, and 2001, respectively. He is currently a professor with the Research Institute of Electrical Communication at Tohoku University. For 2002–2006, he also joined the Japan Science and Technology Agency (JST) as a researcher for the PRESTO project. His research interests include computer arithmetic, EDA methodology, high performance/secure VLSI computing, and hardware security. He is a senior member of the IEEE.