Compact S-box Hardware Implementation with an Efficient MVP-CSE Algorithm

Xiaoqiang ZHANG, Ning WU, Chun ZENG

***Abstract*—In this paper, two compact architectures for hardware implementation of Advanced Encryption Standard S-box are presented based on a polynomial basis and a normal basis, respectively. Composite Field Arithmetic is employed in these architectures to reduce the area consumption and shorten the critical path. Furthermore, a novel Multi-Variable Patterns Common Subexpression Elimination algorithm is proposed to further optimize the S-box, which efficiently reduces the redundant resources of multiplicative inversion in $GF(2^4)$ module and isomorphism mapping functions. It is shown that both optimized S-boxes have good area-delay performances, and that the normal basis S-box uses 20.4% less XOR and a shorter critical path compared with the polynomial basis S-box. In 0.18μm COMS technology, the normal basis S-box has the smallest area-delay product, which is 10.32% and 19.64% smaller than that of the smallest area S-box and the shortest critical path S-box, respectively.**

***Index Terms*—AES, S-box, Composite Field Arithmetic (CFA), Multi-Variable Patterns Common Subexpression Elimination (MVP-CSE) algorithm, polynomial basis, normal basis.**

I. Introduction

Advanced Encryption Standard (AES) encryption algorithm is established by the National Institute of Standards and Technology (NIST) to replace the original DES encryption algorithm in 2001 [1]. It is one of the most important symmetric block ciphers. The block length of the AES algorithm is 128 bits with the key lengths of 128, 192 or 256 bits. Full computation of the AES encryption requires 10, 12 or 14 iterations, each containing four transformations: SubBytes, ShiftRows, MixColumns and AddRoundKey. The SubBytes transformation, commonly known as S-box, is a nonlinear substitution that guarantees a better security of the AES encryption. It takes the most resource and consumes the most power in the implementation. Hence, the hardware implementation efficiency of the AES encryption in terms of area, speed, security and power consumption mainly depends on the implementation of S-box [2].

In general, the AES encryption is applied in resource-limited systems, where the area optimization of the AES hardware implementation is highly desirable. Several small-area S-box designs using CFA constructions are proposed in [2]-[10], and they usually base on polynomial basis and normal basis. Summarizing from the previous works [2]-[10], Canright [3] presented the smallest S-box based on normal basis. However, the critical path was long in Canright's work. On the other hand, Zhang [8] proposed an AES S-box based on polynomial basis with the shortest critical path to date [5]. However, their work requires a large area.

In this paper, both area and critical path of S-box implementation are taken into account, with the focus on approaches to implement a very compact S-box efficiently. To reduce the hardware resource requirement of the S-box, we use pure combinational-logic through exploitation of Composite Field Arithmetic (CFA) in $GF(((2^2)^2)^2)$. A new MVP-CSE algorithm is proposed for subexpressions sharing. In Section II, two S-box circuit structures are derived, one based on the polynomial basis, and the other, the normal basis. In Section III, the squarer module and the constant multiplier module in $GF(2^4)$ are merged into a single one by using CFA, to reduce the gate counts significantly, while the critical path is shortened greatly. To optimize the multiplicative inversion circuit in $GF(2^4)$ and isomorphism mapping function circuit of the S-boxes, a new Multi-Variable Patterns Common Subexpression Elimination (MVP-CSE) algorithm is proposed in Section IV. In Section V, the evaluation of the resulting S-boxes and the comparison with previous works are discussed. Conclusions are given in Section VI.

II. Composite Field Implementation of S-box

The AES S-box is defined as the multiplicative inversion module in the finite field $GF(2^8)$ followed by an affine transformation [11]. The S-box is calculated by using (1).

$$F^T = M((X^T)^{-1}) + V^T \tag{1}$$

where X is the input state matrix. M is an 8×8 constant matrix, defined as M=[0x8*E*, 0x*C*7, 0x*E*3, 0x*F*1, 0x*F*8, 0x7*C*, 0x3*E*, 0x1*F*]. V is an 8-bit constant vector, defined as V=[0x63]. The row vectors of both M and V are represented in the hexadecimal format. The process of affine transformation is defined as follows: X^1 is multiplied by M first, then combined with a constant vector V in the affine transformation.

This study focuses on approaches to deduce and simplify the multiplicative inversion in $GF(2^8)$. Fig.1 shows the implementation block diagram of the S-box using the CFA technique (Fig.1(a)), the multiplicative inversion circuit structure based on the polynomial basis (Fig.1(b)), and

This work was supported by the National Natural Science Foundation of China (61376025), Industry-academic Joint Technological Innovations Fund Project of Jiangsu Province (BY2013003-11) he Funding of Jiangsu Innovation Program for Graduate Education (No. KYLX_0273), and the Fundamental Research Funds for the Central Universities.

Xiaoqiang ZHANG is with the College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing 210016, China (e-mail: zxq198111@qq.com).

Ning WU is with the College of Electronic and Information Engineering, NUAA, Nanjing 210016, China (e-mail: wunee@nuaa.edu.cn).

Chun ZENG is with the College of Electronic and Information Engineering, NUAA, Nanjing 210016, China (e-mail: daisyzeng0407201@126.com).

ISBN: 978-988-19253-9-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

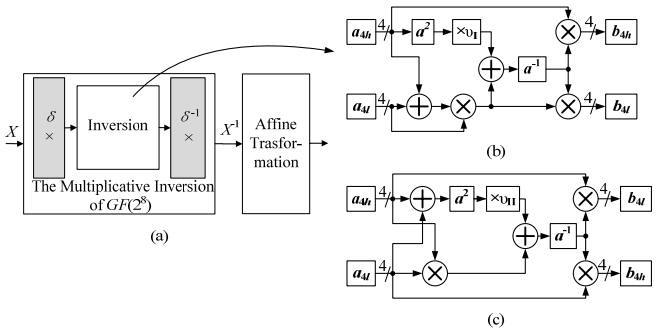

Fig.1 Implementations block diagram of S-box: (a) Implementation of S-box Based on CFA; (b) Inversion circuit structure based on polynomial basis; (c) Inversion circuit structures based on normal basis

normal basis (Fig.1(c)). Based on the CFA technique, the S-box can also be calculated by using (2). Two parts are included in deriving the multiplicative inversion in $GF(2^8)$: one is the derivation of an inversion module in the composite field $GF(((2^2)^2)^2)$, and the other is the calculation of mapping matrices δ and δ^{-1} in isomorphism mapping functions. The mapping matrices δ and δ^{-1} are the linear transformations between the finite field $GF(2^8)$ and the composite field $GF(((2^2)^2)^2)$.

$$F^T = M(\delta^{-1}(\delta X^T)^{-1}) + V^T \tag{2}$$

According to Fig.1(a), δ is the mapping matrix from the finite field to the composite field, and δ^{-1} is the inverse of δ. The relationship between δ and δ^{-1} is $\delta \times \delta^{-1}$=E, and E is a unit matrix. Usually, the mapping matrix δ^{-1} can be combined with the affine matrix M to simplify the circuit structure of the S-box. For easy presentation, the polynomial basis S-box is denoted as Case I, and the normal basis S-box, as Case II.

The structures of the inversion module and the value of the mapping matrix can be derived based on the irreducible polynomial coefficients of the composite field $GF((2^2)^2)$ and $GF((2^4)^2)$. The irreducible polynomial of the composite field $GF((2^4)^2)$, $GF((2^2)^2)$ and $GF(2^2)$ operation are denoted as follows:

$$\begin{cases} GF(((2^2)^2)^2): f(y) = y^2 + \tau y + \upsilon \\ GF((2^2)^2): f(z) = z^2 + \mathrm{T}z + \mathrm{N} \\ GF(2^2): f(w) = w^2 + w + 1 \end{cases} \tag{3}$$

where τ=$(0001)_4$, T=$(01)_2$, and υ_{I}=$(1100)_4$, N_{I}=$(10)_2$ in Case I [3], υ_{II}=$(0001)_4$, N_{II}=$(10)_2$ in Case II [4].

According to the selected coefficients τ, υ, T, N, the inversion module in Case I can be calculated by using (4), and that in Case II, by using (5). The inversion structure in Case I is shown in Fig. 1(b), and that in Case II, in Fig. 1(c).

$$A_I^{-1} = (a_{4h}^2 \upsilon_I + a_{4l}(a_{4l} + a_{4h}))^{-1}(a_{4h}X + (a_{4l} + a_{4h})) \tag{4}$$

$$A_{II}^{-1} = ((a_{4l}^2 + a_{4h}^2)\upsilon_{II} + a_{4l}a_{4h})^{-1}(a_{4l}Y^{16} + a_{4h}Y) \tag{5}$$

where $A_{\mathrm{I}} \in GF(2^8)$, $a_{4h}, a_{4l} \in GF(2^4)$; $A_{\mathrm{II}} \in GF(2^8)$, $a_{4h}, a_{4l} \in GF(2^4)$. In (4), the basis $(X,1)$ is chosen in the composite field $GF((2^4)^2)$. In (5), the basis (Y^{16}, Y) is chosen.

The irreducible polynomial $\mathrm{P}(x) = x^8+x^4+x^3+1$ in $GF(2^8)$ is selected in the AES encryption, whereas the δ and δ^{-1} can be calculated by using the CFA technique according to the following equations:

$$\begin{cases} \delta_I = [0xC2, 0x4A, 0x78, 0x63, 0x75, 0x35, 0x7B, 0x05] \\ \delta_I^{-1} = [0xAE, 0x0C, 0x79, 0x7C, 0x6E, 0x46, 0x22, 0x47] \end{cases} \tag{6}$$

$$\begin{cases} \delta_{II} = [0xE7, 0x71, 0x63, 0xE1, 0x8B, 0x01, 0x61, 0x054F] \\ \delta_{II}^{-1} = [0x12, 0xEB, 0xED, 0x42, 0x7E, 0xB2, 0x22, 0x04] \end{cases} \tag{7}$$

To reduce the area and shorten the delay of the S-box, the concrete method can be summarized as follows: Modules $\times\upsilon$ and a^2 are merged into one module $\upsilon \times a^2$, and a pure combinational-logic expression is derived for a^{-1} utilizing the CFA technique; then, the proposed MVP-CSE algorithm is used to reduce the gate counts and shorten the critical path in a^{-1} module, the isomorphism mapping functions, and the affine transformation.

III. CFA Operation for S-box Optimization

In this section, the detailed architectures are presented to optimize the modules of the $\times\upsilon$, a^2, and the a^{-1} that are shown in Fig.1(b) and Fig.1(c) of the S-box. Each module's implementation is derived by using the CFA technique to reduce area and to increase speed. The detailed calculation process of architectures for $\times\upsilon_{\mathrm{II}}$ and a_{II}^2 in Case II is introduced afterwards, and the modules of $\times\upsilon_{\mathrm{I}}$ and a_{I}^2 are optimized in the same fashion.

The multiplications in Fig.1(c) are derived based on the normal basis according to the following equation:

$$\begin{aligned} A_4(Z)B_4(Z) = {} & (a_{2h}b_{2h} + (a_{2l} + a_{2h})(b_{2l} + b_{2h})\mathrm{N})Z^4 \\ & + (a_{2l}b_{2l} + (a_{2l} + a_{2h})(b_{2l} + b_{2h})\mathrm{N})Z \end{aligned} \tag{8}$$

where A_4, $B_4 \in GF(2^4)$, and they are represented as $A_4(Z)=a_{2h}Z^4+a_{2l}Z$, and $B_4(Z)=b_{2h}Z^4+b_{2l}Z$, respectively, using the normal basis (Z^4,Z). a_{2h}, a_{2l}, b_{2h}, $b_{2l} \in GF(2^2)$.

The multiplication in (8) can be decomposed into $GF(2^2)$ and then further into $GF(2)$. In $GF(2)$, a multiplication is simply an AND gate. According to υ_{II}=$(0001)_4$, N_{II}=$(10)_2$, the modules $\times\upsilon_{\mathrm{II}}$ and a_2 are computed based on the CFA technique and merged into one single module $\upsilon_{\mathrm{II}} \times a_2$. The $\upsilon_{\mathrm{II}}a^2$ can be calculated by using (9).

$$\begin{aligned} \upsilon_{II}a^2 = {} & (a_{h0} + a_{l0})W^2Z^4 + (a_{h1} + a_{l1})WZ^4 \\ & + (a_{l1} + a_{l0})W^2Z + a_{l0}WZ \end{aligned} \tag{9}$$

where the normal basis (W^2, W) are chosen in $GF(2^2)$, and $a_{2h}=a_{h1}W^2+a_{h0}W$, $a_{2l}=a_{l1}W^2+a^{l0}W$, $b_{2h}=b_{h1}W^2+b_{h0}W$, $b_{2l}=b_{l1}W^2+b_{l0}W$. a_{h1}, a_{l1}, b_{h0}, $b_{l0} \in GF(2)$.

Fig. 2 illustrates the detailed opeartions of $\times\upsilon_{\mathrm{II}}$ and a_2, together with the merged structure of $\upsilon_{\mathrm{II}} \times a_2$. As observed from Fig. 2, $\times\upsilon_{\mathrm{II}}$ and a_{II}^2 are implemented by 4 XOR gates and 6 XOR gates, respectively, while the merged module $\upsilon_{\mathrm{II}}a^2$ only needs 3 XOR gates. Table I summarizes the resource consumption and critical path optimization of $\times\upsilon$ and a^2 by using the CFA technique for both Case I and Case II. After

ISBN: 978-988-19253-9-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

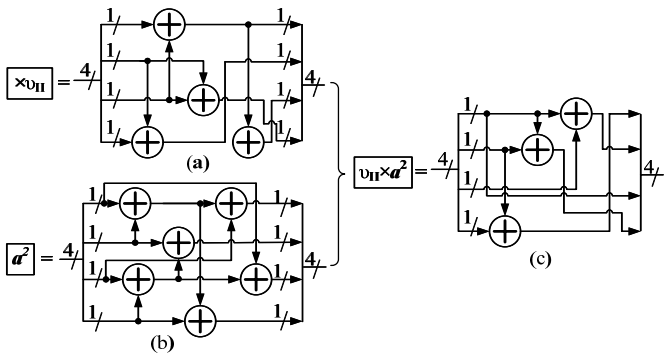

Fig.2 The process for merging modules: (*a*)×*υ*; (*b*) a^2 ; (*c*)$\upsilon_{II}\times a^2$

being merged into one module, the $\upsilon_I a^2$ operation saves 42.9% in the resource consumption and shortens the critical path by 50% in Case I, while the $\upsilon_{II} a^2$ operation saves 70.0% in the resource consumption and shortens the critical path by

TABLE I
THE OPTIMIZATION OF ×υ AND a^2 OPERATION

Approach	×υ AND a^2	Resource Consumption	Critical Path
Case I	Separate	7 XOR	4 XOR
	Merged	4 XOR	2 XOR
Case II	Separate	10 XOR	4 XOR
	Merged	3 XOR	1 XOR

75% in Case II.

The expressions structures of the inversion circuits a_I^{-1} and a_{II}^{-1} in $GF(2^4)$ shown in Fig. 2(a) and Fig. 2(b) are similar to those of (4) and (5). Similar to Eqs. (8) and (9), a_I^{-1} and a_{II}^{-1} can be represented as follows:

$$a_I^{-1} = (a_{2h}^2 \mathrm{N}_I + a_{2l}(a_{2l} + a_{2h}))^{-1}(a_{2h}X + (a_{2l} + a_{2h})) \quad (10)$$

$$a_{II}^{-1} = ((a_{2l}^2 + a_{2h}^2)\mathrm{N}_{II} + a_{2l}a_{2h})^{-1}(a_{2l}Z^4 + a_{2h}Z) \quad (11)$$

To further simplify calculations, we substitute $N_I=(10)_2$ into (10), and $N_{II}=(10)_2$ into (11). The direct implementations of the resulting equations for a_I^{-1} and a_{II}^{-1} are given by (12) and (13).

IV. MVP-CSE ALGORITHM FOR SUBEXPRESSIONS SHARING OPTIMIZATION IN S-BOX

The subexpressions can be shared to reduce the complexity of the a^{-1} module in $GF(2^4)$, the mapping function $\delta\times$ module, and the combined inverse mapping and affine transformation $M\delta^{-1}\times$ module in S-box. The approach to select a subexpression for sharing can be described as identifying patterns. A single variable is used to replace the identified pattern. In [12], it has been proven that selecting a pattern to eliminate is an NP-complete problem. To find an optimal solution, the proposed MVP-CSE algorithm for S-box optimization focuses on the number of variables in each candidate pattern. It employs an exhaustive search algorithm for the situation that several patterns can be selected for elimination. As a result, the MVP-CSE algorithm is efficient and yields a better solution.

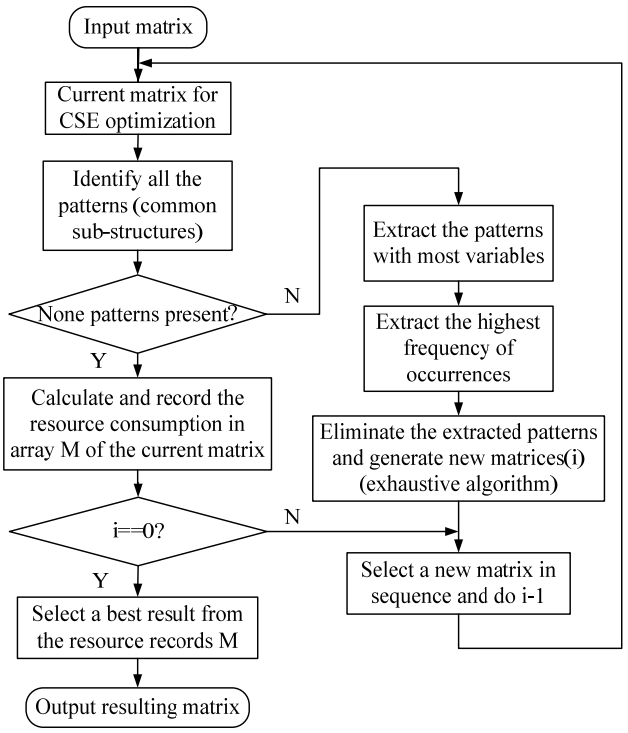

Fig.3 Flowchart of MVP-CSE algorithm

A. MVP-CSE Algorithm

In the MVP-CSE algorithm, the patterns with most variables are extracted at each iteration. If there are more than one pattern, the patterns with the highest frequency of occurrences will be selected. From the highest occurrence patterns, an exhaustive search algorithm is employed to extract each of them and replace the pattern with a new variable. The process continues until no more common subexpressions are found. The detail of the MVP-CSE algorithm is summarized in Fig. 3.

$$a_I^{-1} = \begin{pmatrix} a_{h1} \\ a_{h0} \\ a_{l1} \\ a_{l0} \end{pmatrix}^{-1} = \begin{pmatrix} a_{h1}a_{h0}a_{l1} + a_{h1}a_{l0} + a_{h1} + a_{h0} \\ a_{h1}a_{h0}a_{l1} + a_{h1}a_{h0}a_{l0} + a_{h1}a_{l0} + a_{h0}a_{l1} + a_{h0} \\ a_{h1}a_{h0}a_{l1} + a_{h1}a_{l1}a_{l0} + a_{h1}a_{l1} + a_{h0}a_{l0} + a_{h1} + a_{h0} + a_{l1} \\ a_{h1}a_{h0}a_{l1} + a_{h1}a_{h0}a_{l0} + a_{h1}a_{l1}a_{l0} + a_{h0}a_{l1}a_{l0} + a_{h1}a_{l0} + a_{h0}a_{l1} + a_{h0} + a_{l1} + a_{l0} \end{pmatrix} \quad (12)$$

$$a_{II}^{-1} = \begin{pmatrix} a_{h1} \\ a_{h0} \\ a_{l1} \\ a_{l0} \end{pmatrix}^{-1} = \begin{pmatrix} a_{h0}a_{l1}a_{l0} + a_{h1}a_{l1} + a_{h0}a_{l1} + a_{l1} + a_{l0} \\ a_{h1}a_{l1}a_{l0} + a_{h1}a_{l1} + a_{h0}a_{l1} + a_{h0}a_{l0} + a_{l0} \\ a_{h1}a_{h0}a_{l0} + a_{h1}a_{l1} + a_{h1}a_{l0} + a_{h1} + a_{h0} \\ a_{h1}a_{h0}a_{l1} + a_{h1}a_{l1} + a_{h1}a_{l0} + a_{h0}a_{l0} + a_{h0} \end{pmatrix} \quad (13)$$

ISBN: 978-988-19253-9-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

For example, considering the constant multiplication matrix [0x9, 0xf, 0xf, 0x5], the MVP-CSE algorithm is applied to the matrix in (14). According to the flowchart in Fig. 3, the algorithm identifies the patterns "x_4+x_3+x_2" to be eliminated at the first iteration. A new variable "b_1" is generated to replace the pattern "x_4+x_3+x_2". In the second iteration, "x_3+x_2" has the highest frequency of occurrences, and it is identified and replaced by a new variable "b_2". The computation process can be expressed as follows:

$$\begin{bmatrix} y_4 \\ y_3 \\ y_2 \\ y_1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} x_4 \\ x_3 \\ x_2 \\ x_1 \end{bmatrix} = \begin{bmatrix} x_4 + x_3 + x_2 \\ x_3 + x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_3 + x_2 + x_1 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 + x_1 \\ b_2 \\ b_1 + x_1 \end{bmatrix} \tag{14}$$

$$b_1 = x_4 + x_3 + x_2; b_2 = x_3 + x_2$$

B. The MVP-CSE for inverse a^{-1} module in GF(2^4)

According to (12) in section III, the resource consumption of the a_I^{-1} module is 21XORs+25ANDs. The a_{II}^{-1} module needs 18XORs+16ANDs in (13). The MVP-CSE method can be employed to optimize both the XOR gates and AND gates by using the same selection rules. The computation process of applying the MVP-CSE to (12) and (13) can be expressed as:

$$a_I^{-1} = \begin{pmatrix} a_{h1} + (a_{h1}a_{l0} + (a_{h1}a_{h0}a_{l1} + a_{h0})_{b1})_{b2} \\ (b_2 + a_{h1}a_{h0}a_{l0} + a_{h0}a_{l1})_{b3} \\ b_1 + (a_{h1}a_{l1}a_{l0} + a_{l1})_{b4} + a_{h1}a_{l1} + a_{h0}a_{l0} + a_{h1} \\ b_3 + b_4 + a_{h0}a_{l1}a_{l0} + a_{l0} \end{pmatrix} = \begin{pmatrix} a_{h1} + ((a_{h1}a_{l0})_{c1} + (a_{h1}(a_{h0}a_{l1})_{c2} + a_{h0})_{b1})_{b2} \\ (b_2 + c_1a_{h0} + c_2)_{b3} \\ b_1 + (c_1a_{l1} + a_{l1})_{b4} + a_{h1}a_{l1} + a_{h0}a_{l0} + a_{h1} \\ b_3 + b_4 + c_2a_{l0} + a_{l0} \end{pmatrix} \tag{15}$$

$$a_{II}^{-1} = \begin{pmatrix} a_{h0}a_{l1}a_{l0} + (a_{h1}a_{l1} + a_{h0}a_{l1} + a_{l0})_{b1} + a_{l1} \\ b_1 + a_{h1}a_{l1}a_{l0} + a_{h0}a_{l0} \\ a_{h1}a_{h0}a_{l0} + (a_{h1}a_{l1} + a_{h1}a_{l0} + a_{h0})_{b2} + a_{h1} \\ b_2 + a_{h1}a_{h0}a_{l1} + a_{h0}a_{l0} \end{pmatrix} = \begin{pmatrix} (a_{h0}a_{l1})_{c3}a_{l0} + ((a_{h1}a_{l1})_{c1} + c_3 + a_{l0})_{b1} + a_{l1} \\ b_1 + c_1a_{l0} + (a_{h0}a_{l0})_{c4} \\ (a_{h1}a_{l0})_{c2}a_{h0} + (c_1 + c_2 + a_{h0})_{b2} + a_{h1} \\ b_2 + c_1a_{h0} + c_4 \end{pmatrix} \tag{16}$$

The XOR gates are optimized firstly, then the AND gates, by utilizing the MVP-CSE. As a result, the optimized multiplicative inversions in $GF(2^4)$ have a good area performance. Case I only needs 13XORs + 8ANDs, eliminated 38.1% XOR gates and 68.0% AND gates compared with (12). Case II needs only 12XORs +8ANDs, eliminating 33.3% XOR gates and 50.0% AND gates compared to (13).

TABLE II
THE RESOURCES CONSUMPTION AND CRITICAL PATH IN GF (2^4) MULTIPLICATIVE INVERSION

Approach	Resources Consumption		Critical Path		Area-delay
	XOR	AND	XOR	AND	
[3]	9	2NOR+ 2NAND+ 6AND	5	2	1.53
[5]case III	13	9	5	2	1.91
[6]case III	13	9	3	1	1.09
[8]	14	8	3	2	1.41
case I	13	8	3	2	1.32
case II	12	8	3	1	1

Compared with the works of [3], [5]-[6], [8], the resource consumption and critical path of multiplicative inversions in $GF(2^4)$ are shown in Table II. The area-delay in Case II is used as the basic unit for reference.

Table II shows that Case II has the best performance in area-delay and the shortest critical path. The reference [3] presented the smallest multiplicative inversion in $GF(2^4)$, but a long critical path.

C. The MVP-CSE for $\delta^{-1}\times$ and $M\delta^{-1}\times$ functions

According to the section II, the implementation of S-box includes two parts, namely the multiplicative inversion in $GF(2^8)$ and the affine transformation. The multiplicative inversion in $GF(2^8)$ can be decomposed into the ones in $GF((2^4)^2)$, which includes the isomorphism mapping

$$\delta_I X = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_7 \\ x_6 \\ x_5 \\ x_4 \\ x_3 \\ x_2 \\ x_1 \\ x_0 \end{bmatrix} = \begin{bmatrix} x_7 + x_6 + x_1 \\ x_6 + x_3 + x_1 \\ x_6 + x_5 + x_4 + x_3 + x_0 \\ x_6 + x_5 + x_1 + x_0 \\ x_6 + x_5 + x_4 + x_2 + x_0 \\ x_5 + x_4 + x_2 + x_0 \\ x_6 + x_5 + x_4 + x_3 + x_1 + x_0 \\ x_2 + x_0 \end{bmatrix} = \begin{bmatrix} x_7 + (x_6 + x_1)_{y4} \\ y_4 + x_3 \\ (x_6 + x_3 + y_3)_{y1} \\ y_4 + (x_5 + x_0)_{y5} \\ x_6 + y_2 \\ ((y_5 + x_4)_{y3} + x_2)_{y2} \\ y_1 + x_1 \\ x_2 + x_0 \end{bmatrix} \tag{18}$$

ISBN: 978-988-19253-9-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

functions and the inversion. The affine matrix M and the mapping matrix δ^{-1} can be merged into a single matrix $M\delta^{-1}$. The matrices $M\delta_I^{-1}$ in Case I and $M\delta_{II}^{-1}$ in Case II are shown separately in (17).

$$\begin{cases} M\delta_I^{-1} = [0xE3, 0x81, 0xBE, 0xE0, 0xC8, 0x21, 0x0F, 0x31] \\ M\delta_{II}^{-1} = [0x28, 0x88, 0x41, 0xA8, 0xF8, 0x6D, 0x32, 0x52] \end{cases} \quad (17)$$

TABLE III
THE MVP-CSE OPTIMIZATION RESULTS FOR $\delta\times$ AND $M\delta^{-1}\times$ FUNCTIONS

MCM matrix	Without MVP-CSE (XOR)		With MVP-CSE (XOR)		Optimization Rate
	Area	Delay	Area	Delay	
δ_I	24	3	12	4	50.0%
$M\delta_I^{-1}$	21	3	16	3	23.8%
δ_{II}	24	3	14	3	41.7%
$M\delta_{II}^{-1}$	17	3	11	3	35.3%

The constant matrices δ_I, $M\delta_I^{-1}$, δ_{II} and $M\delta_{II}^{-1}$ in S-box are optimized utilizing the MVP-CSE. The process and result of δ_I optimization are given in (18). The MVP-CSE is also applied to the remaining constant matrices mentioned above in the same fashion as (18).

Table III shows the MVP-CSE optimization results of the matrices δ_I, $M\delta_I^{-1}$, δ_{II} and $M\delta_{II}^{-1}$. The area reductions range from 23% to 50%, while the critical path delay increases slightly (one extra XOR gate delay in δ_I).

The reference [5] takes a double-variable pattern CSE algorithm to optimize the mapping matrices, yielding the resource consumption of 29 XORs and the critical path of 6 XORs. For comparison, Case I requires 26 XORs in resource consumption with the critical path of 7 XORs, and Case II needs only 25 XORs in resource consumption with the critical path of 6 XORs. The results show that the case II has a better optimization result than those of Case I and [5]. Compared with [5], Case I saves 10.3% XOR gates with a slight critical path length increase, and Case II reduces 13.8% XOR gates while having the same critical path.

V. IMPLEMENT RESULTS AND ANALYSIS

The performance comparison between Case I and Case II is illustrated in Table IV. Combining with Fig.1, the CFA algorithm is used for the $\upsilon\times a^2$ module optimization and the combinational logic expression derivation of the a^{-1} module. Moreover, the MVP-CSE algorithm is efficient in reducing the redundant resource of the a^{-1} module and the mapping matrix modules. The results show that, Case II reduces 20.4% XORs in resource consumption with a shorter critical path compared with Case I.

From the performance analysis of Cases I and II, and compared with prior works [2], [3], [5]-[8], the results of resource consumption and critical path are shown in Table V. In the table, a polynomial basis was employed for [2], [7]-[8] as well as Case I. Comparing Case I with [2] and [7], Case I uses less resource and has a shorter critical path. In addition, Case I saves 7.5% XOR gates with a small critical path sacrifice compared with [8]. The normal basis was used in [3], [5] and Case II, whereas [6] combined both polynomial basis and normal basis for S-box implementation. Case II uses less resource and has a shorter critical path than [5] and [6]. Compared with [3], it shortens the critical path length by 9.1%XORs + 25.0%ANDs with slight larger area consumption. The results show that Case I and Case II optimized by using the CFA and MVP-CSE algorithms achieve a good area-delay performance, since these algorithms consider both area and delay factors comprehensively.

TABLE V
THE RESOURCES CONSUMPTION AND CRITICAL PATH OF AES S-BOX IN DIFFERENT CASES

Approach	Resources Consumption		Critical Path	
	XOR	AND	XOR	AND
[2]	126	36	25	4
[3]	91	36	22	4
[5]caseIII	96	36	20	4
[6]caseIII	117	35	20	3
[7]	123	36	23	4
[8]	120	35	19	4
Case I	111	35	20	4
Case II	93	35	20	3

In the 0.18μm CMOS technology, the area consumption is

TABLE IV
THE PERFORMANCE COMPARISON OPTIMIZED AES S-BOXES IN COMPOSITE FIELD COMPUTATION

Approach	Modules	Inversion of Composite Field				$M\delta^{-1}$		δ		S-box			
		RC		CP		RC	CP	RC	CP	RC		CP	
		XOR	AND	XOR	AND	XOR	XOR	XOR	XOR	XOR	AND	XOR	AND
Case I	$GF(2^4)$ Inversion	13	8	3	2	14	3	12	4	111	35	20	4
	$GF(2^4)$Multiplication	20	9	4	1								
	$\upsilon\times a^2$	4	--	2	--								
	$GF(2^8)$ Inversion	85	35	13	4								
Case II	$GF(2^4)$ Inversion	12	8	3	1	14	3	11	3	93	35	20	3
	$GF(2^4)$ Multiplication	10	9	4	1								
	$\upsilon\times a^2$	3	--	1	--								
	$GF(2^8)$ Inversion	68	35	14	3								

RC: Resources Consumption, CP: Critical Path

ISBN: 978-988-19253-9-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE VI
THE AREA-DELAY PRODUCTS OF S-BOXES IN 0.18μM CMOS TECHNOLOGY

Approach	Delay (ns)	Area(μm^2)	Area-delay product(μm^2•ns)
[3]	26	2900.6208	75416.1408
[8]	23	3659.0400	84157.9200
Case I	24	3419.5392	82068.9408
Case II	23	2940.5376	67632.3648

$26.6112\mu m^2$ in a XOR gate and $13.3056\mu m^2$ in an AND gate, while the standard delay is 1ns in a XOR gate as well as in an AND gate. Using these parameters, Table VI lists the costs of S-box implementation in [3], [8], Case I, and Case II.

It is observed from Table VI that the area-delay product of Case I is 2.48% less than that of [8], while that of Case II is 10.32% less than that of [3]. The results illustrate that Cases I and II achieves a better performance in the area-delay product, with Case II achieving the best one. For example, Case II saves 17.59%, 10.32%, and 19.64% in the area-delay product, compared to Case I, [3], and [8], respectively.

VI. CONCLUSIONS

This paper discussed the optimization of the compact AES S-box implementation. Two structures of S-box have been proposed based on a polynomial basis and a normal basis, respectively. A CFA algorithm is used to combine $\times v$ and a^2 modules to reduce the area-delay product. In addition, a new MVP-CSE algorithm is proposed for subexpressions sharing in mapping functions and a^{-1} module. As a result, the proposed techniques are efficient in saving the resources and shortening the critical path. Implementing the S-box combinational-logic circuits in 0.18μm COMS technology, the results show that the case based on the normal basis achieves the best area-delay product, saving 17.59%, 10.32%, and 19.64% in the area-delay product, respectively, over the case based on the polynomial basis, [3] and [8].

The combinational-logic circuits of S-box based on either polynomial basis or normal basis are adapted to the pipelined structure, which could further improve the operating frequency of S-box as well as the data throughput. However, high-speed S-box design generally results in a high power consumption. Therefore, future work will be carried out to jointly optimize the speed and power consumption of the S-box hardware implementation.

REFERENCES

[1] National Institute of Standards and Technology (NIST), "Advanced Encryption Standard (AES)", FIPS Publication 197, Nov. 2001.
[2] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A compact Rijndael hardware architecture with S-box optimization", In Advances in Cryptology-ASIACRYPT 2001, LNCS 2248, 2001, pp. 239–245.
[3] D. Canright, "A very compact Rijndael S-box," Technical report NPS-MA-04-001, Naval Postgraduate School, 2005
[4] Z. Xinmiao, "High-speed VLSI Architectures for Error-correcting Codes and Cryptosystems," Ph.D. Minnesota, University of Minnesota, 2005.
[5] M.M.Wong, M.L.D.Wong, A.K.Nandi, I. Hijazin, "Construction of Optimum Composite Field Architecture for Compact High-Throughput AES S-Boxes," IEEE Transactions on Very Large Scale Integration (VLSI) Systems. vol. 20, no. 6, 2012, pp.1151-1155.
[6] M.M.Wong, M.L.D.Wong, A.K.Nandi, I. Hijazin, "Composite field GF(((22)2)2) Advanced Encryption Standard (AES) S-box with algebraic normal form representation in the subfield inversion," Circuits, Devices & Systems IET. Vol. 5, Nov. 2011, pp. 471-476.
[7] N. Mentens, L. Batinan, B. Preneeland, and I. Verbauwhede, "A systematic evaluation of compact hardware implementations for the Rijndael S-box," In Topics in Cryptology-CT-RSA 2005, vol. 3376, 2005, pp. 323-333.
[8] X. Zhang, K. K. Parhi, "On the Optimum Constructions of Composite Field for the AES Algorithm," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 53, Oct. 2006, pp.1153-1157.
[9] Atri Rudra, Pradeep K. Dubey, Charanjit S. Jutla, Vijay Kumar, Josyula R. Rao, and Pankaj Rohatgi, "Efficient Rijndael encryption implementation with composite field arithmetic," In Cryptographic Hardware and Embedded System-CHES2001, LNCS 2162, 2001, pp.171-184.
[10] M. Mozaffari-Kermani, R. Reyhani-Masoleh, "A low-cost S-box for the advanced encryption standard using normal basis," IEEE Int. Conf. Electro/Information Technology 2009, 2009, pp. 52-55.
[11] A.M. Youssef, S.E. Tavares. "Affine equivalence in the AES round function," In Discrete Applied Mathematics, vol. 148, no. 2, 2005, pp.161-170.
[12] R. Pasko, P. Schaumont, V. Derudder, S. Vernalde, and D. Durackova, "A new algorithm for elimination of common subexpressions". IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 18, no. 1, 1999, pp.58-68.

ISBN: 978-988-19253-9-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)