On the Optimum Constructions of Composite Field for the AES Algorithm

Xinmiao Zhang, *Member, IEEE*, and Keshab K. Parhi, *Fellow, IEEE*

***Abstract*—In the hardware implementations of the Advanced Encryption Standard (AES) algorithm, employing composite field arithmetic not only reduces the complexity but also enables deep subpipelining such that higher speed can be achieved. In addition, it is more efficient to employ composite field arithmetic only in the SubBytes transformation of the AES algorithm. Composite fields can be constructed by using different irreducible polynomials. Nevertheless, how the different constructions affect the complexity of the composite implementation of the SubBytes has not been analyzed in prior works. This brief presents 16 ways to construct the composite field $GF(((2^2)^2)^2)$ for the AES algorithm. Analytical results are provided for the effects of the irreducible polynomial coefficients on the complexity of each involved subfield operation. In addition, for each construction, there exist eight isomorphic mappings that map the elements in $GF(2^8)$ to those in composite fields. The complexities of these mappings vary. An efficient algorithm is proposed in this brief to find all isomorphic mappings. Based on the complexities of both the subfield operations and the isomorphic mappings, the optimum constructions of the composite field for the AES algorithm are selected to minimize gate count and critical path.**

***Index Terms*—Advanced Encryption Standard (AES) algorithm, composite field, isomorphic mapping, multiplicative inversion.**

I. Introduction

CRYPTOGRAPHY plays an important role in the security of data transmission. The development of computing technology imposes stronger requirements on the cryptography schemes. The Data Encryption Standard (DES) has been the U.S. government standard since 1977. However, now, it can be cracked quickly and inexpensively. In 2000, the Advanced Encryption Standard (AES) [1] replaced the DES to meet the ever-increasing requirements for security.

The AES algorithm has broad applications, such as smart cards and cell phones, WWW servers and automated teller machines, and digital video recorders. Numerous architectures have been proposed for the hardware implementations of the AES algorithm [2]–[11]. Among these architectures, the design in [2] can achieve the highest speed while it is more efficient than the prior designs. The key idea in [2] is to employ composite field arithmetic in the computation of the multiplicative inversion in the SubBytes/InvSubBytes transformation of the AES algorithm. As a result, deep subpipelining is enabled, and hardware complexity is reduced.

Different construction schemes for composite fields are proposed for the AES algorithm in [7], [8], and [11]. In the design in [7], $GF(2^8)$ is decomposed into $GF((2^4)^2)$, and composite field arithmetic is applied to all the transformations in the AES algorithm. The optimum construction scheme for $GF((2^4)^2)$ is selected based on minimizing the total gate count in the implementation of all transformations. However, it is more efficient to apply composite field arithmetic only in the computation of the multiplicative inversion in the SubBytes and InvSubBytes transformations [2]. In this case, the construction scheme selected in [7] is no longer optimum. The schemes proposed in [8] and [11] apply composite field arithmetic only to the multiplicative inversion. In [8], $GF(2^8)$ is decomposed into $GF((2^4)^2)$, while in [11], $GF(2^8)$ is decomposed into $GF(((2^2)^2)^2)$. Nevertheless, each of them proposed only one possible way to construct the composite field. There exist other construction schemes with smaller gate counts and shorter critical paths.

Different irreducible polynomials can be used to construct the composite fields of the same order. This brief presents 16 ways to construct $GF(((2^2)^2)^2)$. Using composite field arithmetic, the complicated multiplicative inversion in $GF(2^8)$ is mapped to operations in subfields. This brief provides the analytical results of how the coefficients in the irreducible polynomials affect the complexities of the subfield operations. In addition, for each construction scheme, there exist eight isomorphic mappings with various complexities to map the elements between $GF(2^8)$ and $GF(((2^2)^2)^2)$. An efficient algorithm is proposed in this brief to find all the isomorphic mappings. Moreover, the lowest mapping complexity is provided for each proposed composite field construction scheme. Based on the complexities of both the subfield operations and the isomorphic mappings, the optimum constructions of the composite field $GF(((2^2)^2)^2)$ for the AES algorithm are proposed. Other composite field construction optimization approaches have been published recently [12], [13]. However, the approach in [12] is optimized based only on the complexity of isomorphic mappings. The approach in [13] optimizes for overall area requirement. Nevertheless, the critical path issue is ignored in the optimization process.

The structure of this brief is as follows. In Section II, the architecture for the implementation of SubBytes using composite field arithmetic is introduced. Section III provides different construction schemes for the composite field $GF(((2^2)^2)^2)$. Section IV discusses how the coefficients of the field polynomials affect the complexity of each block in the SubBytes implementation. An efficient scheme is presented in Section V to find all the isomorphic mappings for each possible construction of the composite field. The lowest achievable mapping complexity for

Manuscript received July 12, 2005; revised April 14, 2006. This work was supported by the Army Research Office under Grant W911NF-04-1-0272. This paper was recommended by Associate Editor C.-T. Lin.

X. Zhang is with the Department of Electrical Engineering and Computer Science, Case Western Reserve University, Cleveland, OH 44106-7071 USA (e-mail: xinmiao.zhang@case.edu).

K. K. Parhi is with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: parhi@ece.umn.edu).

Digital Object Identifier 10.1109/TCSII.2006.882217

each construction scheme is listed, and some comparison results with prior works are provided. Section VI concludes this brief.

II. Composite Field Implementations of the SubBytes in AES

In the AES algorithm, the message is divided into blocks of 128 bits. Each block is divided into 16 bytes, and each byte is considered as an element of $GF(2^8)$. Although different irreducible polynomials can be used to construct $GF(2^8)$, the one specified for the AES algorithm is $P(x) = x^8 + x^4 + x^3 + x + 1$. The AES algorithm is carried out in a number of rounds. Each round in the encryption consists of four transformations, namely: 1) SubBytes; 2) ShiftRows; 3) MixColumns; and 4) AddRoundKey. The decryption consists of the inverse transformations. Among the four transformations involved in the encryption, the SubBytes is the most complicated. In this transformation, we need to compute the multiplicative inverse of each byte in $GF(2^8)$, followed by an affine transformation. Denoting each byte by S, the SubBytes can be described by

$$S' = MS^{-1} + C$$

where M is an 8 × 8 binary matrix, and C is an 8-bit binary vector.

Although two finite fields of the same order are isomorphic, the complexities of the field operations may heavily depend on the representations of the field elements. Composite field arithmetic can be employed to reduce the hardware complexity of the multiplicative inversion in $GF(2^8)$. We call two pairs $\{GF(2^n), Q(y) = y^n + \Sigma_{i=0}^{n-1} q_i y^i, q_i \in GF(2)\}$ and $\{GF((2^n)^m), P(x) = x^m + \Sigma_{i=0}^{m-1} p_i x^i, p_i \in GF(2^n)\}$ a composite field [14] if

- $GF(2^n)$ is constructed from $GF(2)$ by $Q(y)$;
- $GF((2^n)^m)$ is constructed from $GF(2^n)$ by $P(x)$.

Composite fields will be denoted by $GF((2^n)^m)$, and a composite field $GF((2^n)^m)$ is isomorphic to the field $GF(2^k)$ for $k = nm$. Additionally, composite fields can be built iteratively from lower order fields. For example, the composite field of $GF(2^8)$ can be built iteratively from $GF(2)$ using the irreducible polynomials

$$\begin{cases} GF(2) \Rightarrow GF(2^2), & P_0(x) = x^2 + x + 1 \\ GF(2^2) \Rightarrow GF\left((2^2)^2\right), & P_1(x) = x^2 + x + \phi \\ GF\left((2^2)^2\right) \Rightarrow GF\left(\left((2^2)^2\right)^2\right), & P_2(x) = x^2 + x + \lambda \end{cases} \tag{1}$$

where $\phi \in GF(2^2)$, $\lambda \in GF((2^2)^2)$, and the values of ϕ, λ satisfy that $P_1(x)$ is irreducible over $GF(2^2)$ and $P_2(x)$ is irreducible over $GF((2^2)^2)$. Moreover, an isomorphic mapping function $f(x) = \delta \times x$ and its inverse need to be applied to map the representation of an element in $GF(2^8)$ to its composite field and vice versa. The 8 × 8 binary matrix δ is decided by the field polynomials of $GF(2^8)$ and its composite field.

In the composite field $GF((2^4)^2)$, an element can be expressed as $s_h x + s_l$, where $s_h, s_l \in GF(2^4)$, and x is the root of $P_2(x)$. Using the extended Euclidean algorithm, the multiplicative inverse of $s_h x + s_l$ modulo $P_2(x)$ can be computed as

$$(s_h x + s_l)^{-1} = s_h \Theta x + (s_h + s_l)\Theta \tag{2}$$

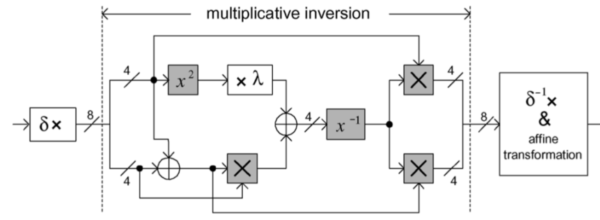

Fig. 1. Implementation of the SubBytes transformation.

where $\Theta = (s_h^2\lambda + s_h s_l + s_l^2)^{-1}$. According to (2), the multiplicative inversion in $GF(2^8)$ can be carried out in $GF((2^4)^2))$ by the architecture illustrated in Fig. 1 [2]. The inverse isomorphic mapping is combined with the affine transformation to reduce the hardware complexity.

The multiplication in $GF(2^4)$ can be further decomposed into multiplications in $GF((2^2)^2)$ to reduce complexity as shown in Fig. 2(a). In addition, based on further decomposition into $GF((2^2)^2)$, the bit expressions can be derived for other blocks in Fig. 1. Then, substructure sharing can be employed accordingly to reduce the complexities of these blocks. For example, it was proposed in [8] to use $\phi = \{10\}_2$ and $\lambda = \{1100\}_2$, where $\{\cdot\}_2$ denotes the number in the braces in binary form. In this case, the squarer in $GF(2^4)$, $\times\lambda$ block, and $\times\phi$ block can be implemented as illustrated in Fig. 2(c)–(e), respectively.

Composite fields can be constructed using different irreducible polynomials. In this brief, we only consider those in the form of (1). The values of ϕ and λ decide the complexities of subfield operations. In addition, for each fixed set of irreducible polynomials used in composite field construction, there exist multiple isomorphic mappings whose complexities vary.

III. Choosing the Coefficients in Irreducible Polynomials

$x^2 + x + 1 = 0$ is the only irreducible polynomial of degree two over $GF(2)$. Hence, it is the only possible choice for $P_0(x)$ in (1). The choices of $\phi \in GF(2^2)$ and $\lambda \in GF((2^2)^2)$ need to satisfy that $P_1(x) = x^2 + x + \phi$ is irreducible over $GF(2^2)$ and $P_2(x) = x^2 + x + \lambda$ is irreducible over $GF((2^2)^2)$.

A polynomial over $GF(2^q)$ is irreducible if it cannot be factored into nontrivial polynomials other than itself over $GF(2^q)$. Since the degree of $P_1(x)$ is two, if $P_1(x)$ can be factored into nontrivial polynomials, it will be factored into the form of $P_1(x) = (x + \mu)(x + \nu)$, where $\mu, \nu \in GF(2^2)$. Therefore, the test of irreducibility can be carried out by examining if any elements of $GF(2^2)$ are roots of $P_1(x)$. Using this scheme, it can be derived that the only values of ϕ that make $x^2 + x + \phi$ irreducible over $GF(2^2)$ are $\phi = \{10\}_2$ and $\phi = \{11\}_2$.

The values of λ, which make $P_2(x)$ irreducible over $GF((2^2)^2)$, can be derived in a similar way. Alternatively, we can evaluate $x^2 + x$ on each element of $GF((2^2)^2)$. The evaluation results are all the values of λ that make $P_2(x)$ nonirreducible. For example, assume $\beta^2 + \beta = \gamma(\beta, \gamma \in GF((2^2)^2))$. Then, $\beta^2 + \beta + \gamma = 0$. Hence, $P_2(x) = x^2 + x + \gamma = 0$ has a root β. Therefore, $P_2(x)$ can be factored into $(x + \beta)(x + \theta)(\theta \in GF((2^2)), \beta\theta = \gamma)$, which is nonirreducible. The values of λ, which make $P_2(x)$ irreducible, consist of the field elements not equaling to any of the evaluation results. Different representations for field elements of $GF((2^2)^2)$ can be developed when $\phi = \{10\}_2$ and $\phi = \{11\}_2$,

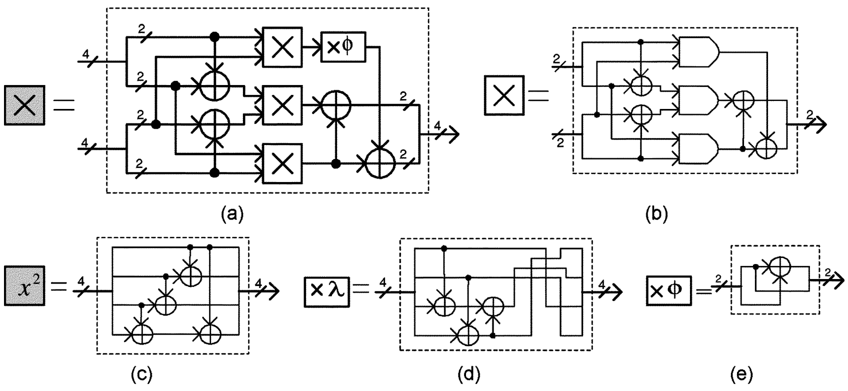

Fig. 2. Implementations of individual blocks. (a) Multiplier in $GF((2^2)^2)$. (b) Multiplier in $GF(2^2)$. (c) Squarer in $GF(2^4)$. (d) Constant multiplier $(\times\lambda)$. (e) Constant multiplier $(\times\phi)$.

respectively. It can be derived that there are eight possible values of λ that make $x^2+x+\lambda$ irreducible over $GF((2^2)^2)$ constructed by using each of $\phi=\{10\}_2$ and $\phi=\{11\}_2$. These values of λ are

$$\begin{aligned}\lambda&=\{1000\}_2, & \lambda&=\{1100\}_2\\ \lambda&=\{1001\}_2, & \lambda&=\{1101\}_2\\ \lambda&=\{1010\}_2, & \lambda&=\{1110\}_2\\ \lambda&=\{1011\}_2, & \lambda&=\{1111\}_2.\end{aligned}$$

All together, there are sixteen ways to construct the composite field $GF(((2^2)^2)^2)$ using irreducible polynomials in the form of (1).

In the composite field implementation of the SubBytes illustrated in Figs. 1 and 2, the coefficients ϕ and λ affect the complexities of the following blocks.

- The value of ϕ affects the complexities of
  1) constant multiplier $\times\phi$;
  2) squarer in $GF(2^4)$;
  3) multiplier in $GF((2^2)^2)$;
  4) inversion in $GF(2^4)$;
  5) constant multiplier $\times\lambda$;
  6) isomorphic mapping and inverse.
- The value of λ affects the complexities of
  1) constant multiplier $\times\lambda$;
  2) isomorphic mapping and inverse.

In the next section, analytical results are provided on how the coefficients ϕ and λ affect the complexity of each block.

IV. Effects of Irreducible Polynomial Coefficients

Taking x as the root of $P_0(x)$, an element $a\in GF(2^2)$ can be expressed as a_1x+a_0, where a_1, $a_0\in GF(2)$. It can be derived that

$$\begin{aligned}(a_1x+a_0)(x+1)&=a_1x^2+(a_0\oplus a_1)x+a_0\\ &=a_0x+(a_0\oplus a_1).\end{aligned}$$

Hence, in the case of $\phi=\{11\}_2$, the constant multiplier $\times\phi$ can be implemented by one XOR gate. The constant multiplication by $\phi=\{10\}_2$ also takes one XOR gate [2]. Therefore, the two choices of ϕ lead to the same complexity for the $\times\phi$ block. As it can be observed from Fig. 2(a), the multiplier in $GF((2^2)^2)$ consists of multipliers in $GF(2^2)$, a constant multiplier $\times\phi$, and modulo 2 adders. The complexity of the $GF(2^2)$ multiplier does not change with ϕ or λ. Accordingly, the complexity of the $GF((2^2)^2)$ multiplier is the same for the two possible values of ϕ.

Compared to a general multiplier, the implementation of a squarer can be simplified. In the case $\phi=\{11\}_2$, taking the 4 bits of $a\in GF((2^2)^2)$ as $\{a_3,a_2,a_1,a_0\}$, it can be derived that each bit in $a^2=\{a_3',a_2',a_1',a_0'\}$ can be computed as

$$\begin{aligned}a_3'&=a_3\\ a_2'&=a_3\oplus a_2\\ a_1'&=(a_3\oplus a_2)\oplus a_1\\ a_0'&=(a_1\oplus a_0)\oplus a_2.\end{aligned}$$

Hence, the implementation of the squarer in $GF(2^4)$ takes 4 XOR gates with 2 XOR gates in the critical path by employing substructure sharing. In the case $\phi=\{10\}_2$, the architecture for the squarer in $GF(2^4)$ is provided in [2]. It can be observed that the squarer has the same gate count and critical path when $\phi=\{11\}_2$ and $\phi=\{10\}_2$.

The complexity of the inversion in $GF(2^4)$ is only dependent on ϕ. In addition, the complexity of the direct computation approach as in [2] can achieve lower complexity than those employing further decompositions as in [8]. In the case $\phi=\{10\}_2$, the equations to compute each bit in the inverse can be found in [2]. Using substructure sharing, the inversion can be implemented by 14 XOR gates and 8 AND gates with 3 XOR gates and 2 AND gates in the critical path. It can be derived that in the case of $\phi=\{11\}_2$, each bit in $a^{-1}=\{a_3^{-1},a_2^{-1},a_1^{-1},a_0^{-1}\}$ can be computed by

$$\begin{cases}a_3^{-1}=a_2\oplus a_0a_3\oplus a_1a_2a_3\\ a_2^{-1}=a_3\oplus a_0a_3\oplus a_1a_2\oplus a_0a_2a_3\oplus a_1a_2a_3\\ a_1^{-1}=a_1\oplus a_2\oplus a_0a_2\oplus a_0a_3\oplus a_1a_2\oplus a_1a_3\\ \qquad\oplus a_1a_2a_3\oplus a_0a_1a_3\\ a_0^{-1}=a_0\oplus a_1\oplus a_3\oplus a_0a_2\oplus a_0a_3\oplus a_1a_2\\ \qquad\oplus a_0a_1a_2\oplus a_0a_1a_3\oplus a_0a_2a_3\oplus a_1a_2a_3.\end{cases}\tag{3}$$

Using substructure sharing, the equations in (3) can be computed by 14 XOR gates and 8 AND gates with 3 XOR gates and 2 AND gates in the critical path. Compared to the case when $\phi=\{10\}_2$, the computation of the inverse in $GF(2^4)$ in the case of $\phi=\{11\}_2$ requires the same number of gates and has the same critical path.

TABLE I
GATE COUNT FOR EACH $\times\lambda$ IMPLEMENTATION

ϕ	λ	gate #	ϕ	λ	gate #
$\{10\}_2$	$\{1000\}_2$	4 XOR	$\{11\}_2$	$\{1000\}_2$	3 XOR
	$\{1001\}_2$	5 XOR		$\{1001\}_2$	4 XOR
	$\{1010\}_2$	4 XOR		$\{1010\}_2$	3 XOR
	$\{1011\}_2$	5 XOR		$\{1011\}_2$	4 XOR
	$\{1100\}_2$	3 XOR		$\{1100\}_2$	4 XOR
	$\{1101\}_2$	4 XOR		$\{1101\}_2$	5 XOR
	$\{1110\}_2$	4 XOR		$\{1110\}_2$	5 XOR
	$\{1111\}_2$	3 XOR		$\{1111\}_2$	4 XOR

TABLE II
OPTIMUM VALUES OF ϕ AND λ

$\phi = \{10\}_2$	$\lambda = \{1100\}_2, \{1111\}_2$
$\phi = \{11\}_2$	$\lambda = \{1000\}_2, \{1010\}_2$

The complexity for the constant multiplier $\times\lambda$ is affected by both values of ϕ and λ. Taking x as a root of $P_1(x)$, an element $b \in GF((2^2)^2)$ can be expressed as $b_1x+b_0(b_1, b_0 \in GF(2^2))$. Similarly, λ can be expressed as $\lambda_1x+\lambda_0(\lambda_1, \lambda_0 \in GF(2^2))$. Accordingly, the product $c = b \times \lambda$ can be computed as

$$\begin{aligned} c = c_1x + c_0 &= b \times \lambda \\ &= (b_1x + b_0)(\lambda_1x + \lambda_0) \\ &= b_1\lambda_1x^2 + (b_0\lambda_1 + b_1\lambda_0)x + b_0\lambda_0 \\ &= (b_1\lambda_1 + b_0\lambda_1 + b_1\lambda_0)x + (b_1\lambda_1\phi + b_0\lambda_0). \end{aligned}$$

Therefore, two values need to be computed for the constant multiplication by λ, i.e.,

$$c_1 = b_1\lambda_1 + b_0\lambda_1 + b_1\lambda_0 \tag{4}$$
$$c_0 = b_1\lambda_1\phi + b_0\lambda_0. \tag{5}$$

One way to simplify the computations in the above two equations is to make λ_0 or λ_1 zero such that the terms consisting of the multiplications with λ_0 or λ_1 can be eliminated from the computation. Alternatively, if $\lambda_0 = \lambda_1$, the terms $b_1\lambda_0$ and $b_1\lambda_1$ can be cancelled out from (4). From the discussions in Section III, λ_1 can only be $\{11\}_2$ or $\{10\}_2$. Therefore, the values of λ, which minimize the complexity of $\times\lambda$, can be $\{1100\}_2$, $\{1111\}_2$, $\{1000\}_2$, or $\{1010\}_2$. Table I lists the gate counts for the implementations of $\times\lambda$ for all possible combinations of ϕ and λ. The critical path for each implementation has two XOR gates. It can be observed from Table I that the combinations of ϕ and λ, which minimize the cost of the constant multiplication by λ, are those listed in Table II. The results in Table II agree with the analytical results.

V. OPTIMUM ISOMORPHIC MAPPINGS

For a fixed set of irreducible polynomials in (1), there exist multiple isomorphic mappings between $GF(((2^2)^2)^2)$ and $GF(2^8)$. The complexities of these mappings vary. In this section, the lowest achievable complexity of the isomorphic mappings for each combination of ϕ and λ is provided.

An algorithm is proposed in [14, Ch. 2.2] to find the isomorphic mapping matrices when the involved field polynomials are primitive. However, the irreducible polynomial $P(x) = x^8 + x^4 + x^3 + x + 1$ specified for the AES algorithm is not primitive. Therefore, the algorithm in [14] cannot be applied directly. Another algorithm is proposed in [7] to find isomorphic mappings for nonprimitive field polynomials. However, this algorithm has very high complexity. To find a mapping for $GF(2^k)$, an average of $(\Phi(2^k-1)\cdot(2^k-1))/2k$ checkings is needed, where $\Phi(\cdot)$ denotes Euler's totient function.

Assume α is a root of $P(x) = x^8 + x^4 + x^3 + x + 1$. Then, the set $\{1, \alpha, \alpha^2, \ldots, \alpha^7\}$ forms a standard basis for $GF(2^8)$. To construct isomorphic mappings for $GF(2^8)$, we need to find eight base elements $1, \beta, \beta^2, \ldots, \beta^7$ of $GF(((2^2)^2)^2)$ to which the base elements $1, \alpha, \alpha^2, \ldots, \alpha^7$ are mapped. The isomorphic mapping matrix δ is formed by taking the binary representation of β^j as the entries of its $(8-j)$th column. Since $P(x)$ is not primitive, α is not a primitive element. Accordingly, the element β to which α is mapped is not primitive. Assume δ is the isomorphic mapping matrix and $a, b, c \in GF(2^8)$. If $a = b + c$, then $\delta\underline{a} = \delta(\underline{b} + \underline{c}) = \delta\underline{b} + \delta\underline{c}$, where $\underline{a}$, $\underline{b}$, and $\underline{c}$ are the column vectors formed by the bits in the standard basis representation of a, b, and c, respectively. Hence, additive homomorphism is always held for such isomorphic mappings. In addition, if the β to which α is mapped satisfies $P(\beta) = 0$, then multiplicative homomorphism will also hold. Based on these discussions, the isomorphic mappings between $GF(2^q)$ constructed using irreducible polynomial $P(x)$ and its composite field can be found by the algorithm described below.

Algorithm 1

```
initialization: $t = 1$, stop $= 0$
                flag$(i) = 0$ for $i = 1, 2, \ldots, 2^q - 1$
while stop $== 0$
    {$\omega = dectobin(t, q)$
    $P(\omega)$
    if $P(\omega) == 0$
        mapping found, output $\beta = \omega$
        $stop = 1$
    else
        index$(j) = bintodec(\omega^{2^j})$, for $j = 0, 1, 2, \ldots, q-1$
        flag(index$(j)$) $= 1$, for $j = 0, 1, 2, \ldots, q-1$
    find the minimum integer $l > t$, such that flag$(l) = 0$
    $t = l$
}
```

In Algorithm 1, $\omega = dectobin(t, q)$ means first convert the integer t to a q-bit binary number, and then take these q bits as the standard basis representation for ω. Similarly, $t = bintodec(\omega)$ stands for taking the standard basis representation of ω as an q-bit binary number, and then converting this number into an integer as the value of t. It may be noted that the computation of $P(\omega)$ is carried out according to the field operations specified in the composite field. The basic idea of this algorithm is to test if an element ω of the composite field is a root of $P(x)$. If not, then none of the other elements in the same conjugacy class as ω, namely ω^{2^j} $(j = 1, 2, \ldots, q-1)$, is a root of $P(x)$. The next element to be tested is selected excluding the elements in the conjugacy classes of all tested elements. It can be derived that, on average, $(2^q - 1)/2q$ checkings are needed to find isomorphic mappings for a field of order 2^q.

TABLE III
COMPLEXITY OF OPTIMUM ISOMORPHIC MAPPING AND INVERSE

ϕ	λ	isomorphic mapping		inverse mapping+affine	
		gate count	critical path	gate count	critical path
$\{10\}_2$	$\{1000\}_2$	11 XOR	3 XOR	16 XOR	3 XOR
	$\{1001\}_2$	10 XOR	3 XOR	19 XOR	3 XOR
	$\{1010\}_2$	13 XOR	3 XOR	16 XOR	5 XOR
	$\{1011\}_2$	15 XOR	3 XOR	16 XOR	3 XOR
	$\{1100\}_2$	11 XOR	3 XOR	18 XOR	5 XOR
	$\{1101\}_2$	13 XOR	4 XOR	16 XOR	3 XOR
	$\{1110\}_2$	12 XOR	4 XOR	17 XOR	3 XOR
	$\{1111\}_2$	11 XOR	5 XOR	19 XOR	3 XOR
$\{11\}_2$	$\{1000\}_2$	11 XOR	5 XOR	17 XOR	4 XOR
	$\{1001\}_2$	12 XOR	3 XOR	17 XOR	3 XOR
	$\{1010\}_2$	11 XOR	3 XOR	17 XOR	3 XOR
	$\{1011\}_2$	11 XOR	3 XOR	17 XOR	3 XOR
	$\{1100\}_2$	11 XOR	3 XOR	18 XOR	3 XOR
	$\{1101\}_2$	13 XOR	3 XOR	16 XOR	3 XOR
	$\{1110\}_2$	11 XOR	3 XOR	18 XOR	4 XOR
	$\{1111\}_2$	12 XOR	3 XOR	17 XOR	3 XOR

The β computed by Algorithm 1 is not the only element α can be mapped to. α can also be mapped to the other elements in the same conjugacy class as β. It can be computed that for each of the combinations of ϕ and λ, there are eight isomorphic mappings. The optimum isomorphic mapping can be selected based on minimizing gate count. Table III shows the complexity of the optimum isomorphic mapping and its inverse for each combination of ϕ and λ.

The optimum constructions of the composite field for the AES algorithm can be selected by taking the complexities of the involved subfield operations and isomorphic mappings into account. From the discussions in Section IV and Section V, it can be concluded that the implementation of the multiplicative inversion and the affine transformation in SubBytes has the least gate count and the shortest critical path when $GF(((2^2)^2)^2)$ is constructed by using either $\phi = \{11\}_2$, $\lambda = \{1010\}_2$ or $\phi = \{10\}_2$, $\lambda = \{1000\}_2$. The lowest complexity for isomorphic mapping and inverse can be achieved for these two constructions when the root α of $P(x)$ is mapped to $\beta = \{01010011\}_2$ and $\beta = \{01111010\}_2$, respectively.

Table IV shows some comparison results with prior works. Using the proposed optimum construction of composite field, the SubBytes can be implemented by 120 XOR gates and 35 AND gates with 19 XOR gates and 4 AND gates in the critical path. In [12], the complexity of the isomorphic mapping is measured by the number of 1s in the matrix. This is an incorrect approach when substructure sharing is employed. The matrix with the largest number of 1s may have more terms that can be shared, which may leads to a smaller total gate number. Considering substructure sharing, the optimum approach proposed in [12] is only three gates less than that in [8] with two gates less in the critical path. The approach in [13] achieves minimum gate count by using normal basis for field element representation and sharing terms between multipliers with a common input. However, the critical path of this approach is much longer than that of the construction proposed in this brief. In addition, further area reduction can be achieved by introducing NOR gates and NOT gates in the design [13]. The numbers in Table IV are based on a theoretical analysis. These numbers are not the result of synthesizing by using any cell library.

TABLE IV
GATE COUNT AND CRITICAL PATH FOR COMPOSITE IMPLEMENTATIONS OF SUBBYTES

	total gate		critical path	
	XOR	AND	XOR	AND
[8]	126	36	25	4
[12]	123	36	23	4
[11]	110	58	19	4
[13]	91	36	23	4
This approach	120	35	19	4

VI. SUMMARY

In this brief, the optimum constructions of the composite field for the AES algorithm are presented. How the coefficients of the field polynomials affect each block in the composite field implementation of the SubBytes transformation is analyzed. In addition, an efficient algorithm is proposed to find isomorphic mappings when the involved field polynomials are not primitive. The optimum constructions are selected by considering the complexities of both the involved subfield operations and the isomorphic mappings. Future work will address composite field constructions using irreducible polynomials in other forms.

REFERENCES

[1] *Advanced Encryption Standard (AES)*, FIPS PUB 197, Nov. 26, 2001, Federal Information Processing Standards publication 197.

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

[3] K. U. Jarvinen, M. T. Tommiska, and J. O. Skytta, "A fully pipelined memoryless 17.8 Gbps AES-128 encryptor," in *Proc. Int. Symp. FPGA*, Monterey, CA, Feb. 2003, pp. 207–215.

[4] G. P. Saggese, A. Mazzeo, N. Mazocca, and A. G. M. Strollo, "An FPGA based performance analysis of the unrolling, tiling and pipelining of the AES algorithm," in *Proc. FPL*, Portugal, Sep. 2003, pp. 292–302.

[5] F. Standaert, G. Rouvroy, J. Quisquater, and J. Legat, "Efficient implementation of Rijndael encryption in reconfigurable hardware: Improvements and design tradeoffs," in *Proc. CHES*, Cologne, Germany, Sep. 2003, pp. 334–350.

[6] X. Zhang and K. K. Parhi, "Implementation approaches for the advanced encryption standard algorithm," *IEEE Circuits Syst. Mag.*, vol. 2, no. 4, pp. 24–46, Fourth Quarter 2002.

[7] A. Rudra, P. K. Dubey, C. S. Jutla, V. Kumar, J. R. Rao, and P. Rohatgi, "Efficient implementation of Rijndael encryption with composite field arithmetic," in *Proc. CHES*, Paris, France, May 2001, pp. 171–184.

[8] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A compact Rijndael hardware architecture with S-box optimization," in *Proc. ASIACRYPT*, Gold Coast, Australia, Dec. 2000, pp. 239–254.

[9] M. McLoone and J. V. McCanny, "Rijndael FPGA implementation utilizing look-up tables," in *Proc. IEEE Workshop Signal Process. Syst.*, Sep. 2001, pp. 349–360.

[10] H. Kuo and I. Verbauwhede, "Architectural optimization for a 1.82 Gbits/sec VLSI implementation of the AES Rijndael algorithm," in *Proc. CHES*, Paris, France, May 2001, pp. 51–64.

[11] J. Wolkerstorfer, E. Oswald, and M. Lamberger, "An ASIC implementation of the AES S-boxes," in *Proc. RSA Conf.*, San Jose, CA, Feb. 2002, pp. 67–78.

[12] N. Mentens, L. Batina, B. Preneel, and I. Verbauwhede, "A systematic evaluation of compact hardware implementations for the Rijndael S-box," in *Proc. Topics Cryptology—CT-RSA*, San Francisco, CA, Feb. 2005, pp. 323–333.

[13] D. Canright, "A very compact S-box for AES," in *Proc. Cryptographic Hardware and Embedded Syst.*, Edinburgh, U.K., Sep. 2005, pp. 441–455.

[14] C. Paar, "Efficient VLSI architecture for bit-parallel computations in Galois field," Ph.D. dissertation, Inst. Exp. Math.,, Univ. Essen, Essen, Germany, 1994.