IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 17, 2018, accepted June 26, 2018, date of publication June 29, 2018, date of current version July 30, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2851662

Efficient Implementation of Karatsuba Algorithm Based Three-Operand Multiplication Over Binary Extension Field

CHIOU-YNG LEE[1], (Senior Member, IEEE), CHIA-CHEN FAN[2], JIAFENG XIE[3], (Member, IEEE), AND SHYAN-MING YUAN[2]

[1]Department of Computer Information and Network Engineering, Lunghwa University of Science and Technology, Taoyuan 33306, Taiwan
[2]Department of Computer Science, National Chiao Tung University, Hsinchu 300, Taiwan
[3]Department of Electrical Engineering, Wright State University, Dayton, OH 45435, USA

Corresponding author: Jiafeng Xie (jiafeng.xie@wright.edu)

ABSTRACT Three-operation multiplication (TOM) over binary extension field is frequently encountered in cryptosystems such as elliptic curve cryptography. Though digit-serial polynomial basis multipliers are usually preferred for the realization of TOM due to their efficient tradeoff in implementation complexity, the Karatsuba algorithm (KA)-based strategy is rarely employed to reduce the complexity further. Based on this reason, in this paper, we derive a novel low-complexity implementation of TOM based on a new KA-based digit-serial multiplier. The proposed TOM is obtained through two novel coherent interdependent efforts: 1) mapping an efficient KA-based algorithm into a novel digit-serial multiplier and 2) obtaining a new TOM structure through the novel derivation of the TOM algorithm. From the estimated results, it is shown that the proposed structure has significant lower area-time-complexities when compared with the existing competing TOMs. The proposed TOM is highly regular with low-complexity, and hence can be employed in many cryptographic applications.

INDEX TERMS Digit-level serial-in parallel-out (DL-SIPO) multiplier, Karatsuba-algorithm (KA) decomposition, low-complexity, three-operand multiplication (TOM).

I. INTRODUCTION

Finite field arithmetic, especially multiplication, plays an important role in several applications such as elliptic curve cryptography (ECC), error correcting code, and signal processing [1]. For example, standards (NIST [1] and IEEE p1363 [2]) have recommended five binary extension field fields ($GF(2^m)$) for elliptic curve digital signature algorithm (ECDSA) implementation, e.g., $m = 163$, 233, 289, 409, and 571, many experts and scholars have devoted significant efforts on ECC designs for secure resource-constrained applications [3]–[5] like key exchange, authentication, digital signature, encrypt/decrypt, and so on. Basically, the main operation involved within ECC is the point multiplication (PM) kP, where k is an integer and P is given by a point on elliptic curves. We can use point addition and point doubling to perform the PM, i.e., left-to-right algorithm and right-to-left algorithm, where the point addition can be realized based on affine coordinates or projective coordinates. To achieve efficient implementation of PM on hardware platforms, optimized modular arithmetic operations are greatly needed. The finite field addition can be implemented by bitwise XORing, while multiplication is a complicated operation (to avoid inversion operation, point addition can employ the projective coordinates to have only finite field addition, squaring, and multiplication operations involved). Therefore, the finite field multiplication over $GF(2^m)$ is considered as the bottleneck of the PM, where the form of three-operand multiplication (TOM) is frequently encountered [1]–[3].

In binary field, hardware implementation of multiplication can be classified as bit-serial, bit-parallel, and digit-level architectures, respectively, based on their structuring styles. Bit-serial structure has the lowest circuit complexity but possesses a long calculation time; while the bit-parallel architecture involves a very high design area to obtain fast calculation. In order to achieve efficient time complexity, many scholars have proposed bit-parallel multipliers based on special polynomials, such as trinomials and pentanomials [6]–[9] (with relatively larger area occupation), for potential ECC implementation. The digit-level designs provide the trade-off between time and area complexities,

where they can be classified into three categories, namely, digit-level parallel-in serial-out (DL-PISO) [10], digit-level serial-in parallel-out (DL-SIPO) [11]–[13], and digit-level fully-serial-in parallel-out (DL-FSIPO) structures [14] (both systolic and non-systolic designs are included).

Karatsuba algorithm (KA) ([15], [16]) is a very efficient multiplication algorithm which can be used to obtain the subquadratic complexity multiplication. Based on the KA decomposition technique, the space complexity of the multiplier can be reduced from $O(m^2)$ to $O(m^{1.596})$. Recently, Lee *et al.* [17] and Lee and Meher [18] have presented a generalized (a, b)-way KA decomposition for digit-serial multiplication to achieve $O(m^{\log_a \frac{ab+a}{2}})$ space complexity, while the schoolbook digit-serial multiplier has $O(dm)$ space complexity (for example, (9,3)-way KA decomposition involves $O(m^{1.32})$ space complexity).

To obtain efficient structure for TOM, Lee *et al.* [19] have used KA decomposition to derive a bit-parallel TOM. Based on the polynomial basis of $GF(2^m)$, Lee *et al.* [20] have proposed a novel DL-SIPO non-KA-based TOM (NKATOM). Lee *et al.* [19] have proposed the bit-parallel TOM based on a KA approach. To further reduce the involved complexity, in this paper, we have defined a novel partial product formula to develop a novel DL-SIPO TOM structure based on KA decomposition, namely the DL-SIPO KA-based TOM (DL-SIPO KATOM). The proposed TOM is derived through two stages of two novel coherent interdependent efforts. At first, we present a novel KA based digit-serial multiplier with reduced space complexity. Secondly, based on a novel TOM algorithm, the proposed structure with reduced complexity is introduced. From the estimated results, we find that the proposed TOM has significant higher-throughput and lower area-complexity compared to the existing TOMs.

The rest of this paper is organized as follows. Section II briefly reviews the conventional KA decomposition technique and its complexity. In Section III, we introduce the proposed DL-SIPO multiplier based on the KA approach. In Section IV, we propose a novel partial product formula to derive our novel KATOM structure. Section V presents the complexity of the proposed structure and the comparison with the existing TOMs. Finally, we conclude the paper in Section VI.

II. REVIEW OF KARATSUBA ALGORITHM

Suppose that $A = a_0 + a_1x + \cdots + a_{n-1}x^{n-1}$ is a universal polynomial of degree $(n-1)$. KA [15] is one of the high-precision computations, which uses three subproducts of half-length operands to replace the original grade-school multiplication. For example, let n be a power of 2, two polynomials A and B can be splitted into $A = A_0 + x^{\frac{n}{2}}A_1$ and $B = B_0 + x^{\frac{n}{2}}B_1$, where A_0, A_1, B_0, and B_1 are four polynomials of degree $\frac{n}{2}$. Applying the divide-and-conquer algorithm, the product of A and B can be calculated as

$$AB = A_0B_0 + [(A_0 + A_1)(B_0 + B_1) + A_0B_0 + A_1B_1]x^{\frac{n}{2}} + A_1B_1x^n. \quad (1)$$

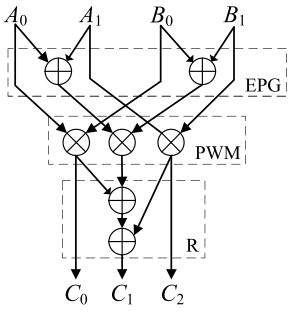


FIGURE 1. High-level description of the KA decomposition.

We can use three stages (evaluation polynomial generation (EPG) stage, point-wise multiplication (PWM) stage, and reconstruction (R) stage) to compute the product AB in (1). Observing three sub-products $\{A_0B_0,\ (A_0 + A_1)(B_0+B_1) = A_{01}B_{01},\ A_1B_1\}$ in (1), three stages can be defined as

- EPG stage: $EPG(B) = (B_0,\ B_0+B_1,\ B_1)$ and $EPG(A) = (A_0,\ A_0 + A_1,\ A_1)$.
- PWM stage: $D = PWM(EPG(A),\ EPG(B)) = (D_0,\ D_1,\ D_2)$, where $D_0 = A_0B_0$, $D_1 = (A_0+A_1)(B_0+B_1)$, and $D_2 = A_1B_1$.
- R stage: $C = (C_0,\ C_1,\ C_2) = \mathrm{R}(D) = (D_0,\ D_0 + D_1 + D_2,\ D_2)$.

Based on the recursive EPG step, each polynomial is splitted into three polynomials with their degrees reduced to about half of the original polynomial. The decomposition algorithm is completed after each polynomial degenerates into single-bit coefficient. The multiplication process based on the recursive KA scheme is shown in the functional block architecture of Fig. 1.

If n is a number of power of 3, two polynomials A and B can be represented by $A = A_0 + A_1x^{m/3} + A_2x^{2m/3}$ and $B = B_0+B_1x^{m/3}+B_2x^{2m/3}$, respectively, where A_i and B_i are $(\frac{m}{3})$-bit polynomials. Based on the 3-way KA decomposition, the product of A and B can be rewritten

$$C = AB = C_0 + C_1x^{m/3} + C_2x^{2m/3} + C_3x^m + C_4x^{4m/3}, \quad (2)$$

where

$$\begin{aligned} D_0 &= A_0B_0,\ D_1 = A_1B_1,\ D_2 = A_2B_2, \\ D_{01} &= (A_0 + A_1)(B_0 + B_1), \\ D_{12} &= (A_2 + A_1)(B_2 + B_1), \\ D_{02} &= (A_0 + A_2)(B_0 + B_2), \\ C_0 &= D_0,\ C_1 = D_{01} + D_0 + D_1, \\ C_2 &= D_{02} + D_0 + D_1 + D_2, \\ C_3 &= D_{12} + D_1 + D_2,\ C_4 = D_2. \end{aligned}$$

Let "S" and "D" to represent "space" and "delay", respectively. Table 1 lists the time and space complexities of each component for the 2-way and 3-way KA decompositions.

TABLE 1. Listing of the time and space complexities of the three components for KA decomposition with $n = b^i$.

b	Components	Recursive	Complexity	Delay
2	EPG	$S_X^{EPG}(n) = 3S_X^{EPG}(\frac{n}{2}) + \frac{n}{2}$ $D_X^{EPG}(n) = D_X^{EPG}(\frac{n}{2}) + 1$	$S_X^{EPG}(n) = n^{\log_2 3} - n$	$D^{EPG}(n) = \log_2 nT_X$
	PWM	$S_A^{PWM}(n) = 3S_A^{PWM}(\frac{n}{2})$ $D_A^{PWM}(n) = D_A^{PWM}(\frac{n}{2})$	$S_A^{PWM}(n) = n^{\log_2 3}$	$D^{PWM}(n) = T_A$
	R	$S_X^R(n) = 3S_X^R(\frac{n}{2}) + 2.5n-3$ $D_X^R(n) = D_X^R(\frac{n}{2}) + 2$	$S_X^R(n) = 3.5n^{\log_2 3} - 5n + 1.5$	$D^R(n) = 2\log_2 nT_X$
3	EPG	$S_X^{EPG}(n) = 6S_X^{EPG}(\frac{n}{3}) + n$ $D_X^{EPG}(n) = D_X^{EPG}(\frac{n}{3}) + 1$	$S_X^{EPG}(n) = n^{\log_3 6} - n$	$D^{EPG}(n) = \log_3 nT_X$
	PWM	$S_A^{PWM}(n) = 6S_A^{PWM}(\frac{n}{3})$ $D_A^{PWM}(n) = D_A^{PWM}(\frac{n}{3})$	$S_A^{PWM}(n) = n^{\log_3 6}$	$D^{PWM}(n) = T_A$
	R	$S_X^R(n) = 6S_X^R(\frac{n}{3}) + 6n - 11$ $D_X^R(n) = D_X^R(\frac{n}{3}) + 4$	$S_X^R(n) = \frac{41}{5}n^{\log_3 6} - 6n + \frac{11}{5}$	$D^R(n) = 4\log_3 nT_X$

X and A in S_X^* and D_A^* represent the corresponding complexity related to XOR and AND gates, respectively.

III. PROPOSED DIGIT-SERIAL KA-BASED MULTIPLICATION

The proposed subquadratic space complexity digit-serial multiplier based on KA decomposition is derived as follows. Let the field be constructed from an irreducible polynomial $F(x) = x^m + K(x)$, where $K(x) = \sum_{i=0}^{k} f_i x^i$ over $GF(2)$. We can find that if k is a very small value, $F(x)$ is abundant in $GF(2^m)$ (the low-weight polynomials $F(x)$, such as trinomials and pentanomials, exist in any field of $GF(2^m)$). Since $F(x) = 0$, we have

$$\begin{aligned} x^m &= K(x), \\ x^{m+1} &= xK(x), \\ &\vdots \\ x^{m+d} &= x^d K(x). \end{aligned}$$

Suppose that $n = \lceil \frac{m}{d} \rceil$, and m is divided by d, then based on $y = x^d$, we have

$$\overline{F}(y) = x^{nd-m}F(x) = y^n + \overline{K}, \tag{3}$$

where

$$\overline{K} = x^{nd-m}K(x).$$

Thus, polynomials $A = \sum_{i=0}^{m-1} a_i x^i$ and $B = \sum_{i=0}^{m-1} b_i x^i$ can be rewritten as $A = \sum_{i=0}^{n-1} A_i y^i$ and $B = \sum_{i=0}^{n-1} B_i y^i$, respectively, where $A_i = \sum_{j=0}^{d-1} a_{di+j}x^j$ and $B_i = \sum_{j=0}^{d-1} b_{di+j}x^j$. This polynomial formula is called the bivariate polynomial. The product of A and B in $GF(2^m)$ must follow the steps as:

1) Schoolbook multiplication: $T = AB$.
2) First reduction polynomial: $D = T \bmod \overline{F}(y)$.
3) Second reduction: $C = D \bmod F(x)$.

As mentioned above, the multiplication process involves sub-field multiplication steps, which is different from the traditional multiplication. Based on this multiplication scheme, suppose that Ay^i is denoted as $A^{(i)} = \sum_{j=0}^{n-1} A_j^{(i)} y^i$, then we can get $A^{(i)} = yA^{(i-1)} \bmod \overline{F}(y)$, where $\overline{F}(y) = y^n + \overline{K}$. Therefore, for the product $C = AB \bmod F(x)$, we can use two-step reduction polynomial to compute the product $C = AB$ as:

- Step-1 (first reduction): $T = A^{(0)}B_0 + A^{(1)}B_1 + \cdots + A^{(n-1)}B_{n-1}$, where $A^{(i)} = yA^{(i-1)} \bmod \overline{F}(y)$.
- Step-2 (second reduction): $C = T \bmod F(x)$.

For simplicity of discussion, let us define that $P_A = EPG(A)$, $P_A \odot P_B = PWM(P_A, P_B)$. Since $A^{(i)} = \sum_{j=0}^{n-1} A_j^{(i)} y^i$, where $A_j^{(i)}$ is a d-bit polynomial in variable x, $A^{(i)}B_j$ based on the KA approach can be expressed as

$$A^{(i)}B_j = \sum_{j=0}^{n-1} R(P_{A_j^{(i)}} \odot P_{B_i})y^j. \tag{4}$$

Consequently, Algorithm 1 shows the proposed digit-serial KA-based multiplication algorithm according to two-step reduction polynomials. Fig. 2 shows the corresponding digit-serial KA-based multiplier based on Algorithm 1. As shown in Fig. 2, the proposed multiplication architecture consists of $\times y$, EPG1, EPG2, Mult, recovery multiplication (RM), and final reduction polynomial (FRP) units.

Suppose that d is a power of b for $b = 2$ or 3, Table 2 lists the complexities of EPG, PWM, and R components for b-way KA decomposition. The complexity of each component in Fig. 2 is analyzed as follows:

- $\times y$ unit: This unit performs $A = Ay \bmod \overline{F}(y)$ in Step 2.5 of Algorithm 1. Define that $\times y$ unit involves $Q1$ XOR gates, where the value $Q1$ is based on the irreducible polynomial $F(x)$. Generally, we have $Q1 = d$ for trinomials or $Q1 = 4d$ for pentanomials (see the example of (15) later).
- EPG1 and EPG2 units: Since polynomial A is represented by a bivariate polynomial as $A = A_0 + A_1 y + \cdots + A_{n-1}y^{n-1}$, we use n EPG components in parallel to compute $P_{A_i} = EPG(A_i)$ for $0 \le i \le n-1$ (as seen in Step 2.3). In Step 2.2, we use one EPG component to compute $P_{B_i} = EPG(B_i)$. Thus, EPG1 and EPG2 have $(n+1)$ $S_X^{EPG}(d)$ space complexity with $D_X^{EPG}(d)$ delay.

TABLE 2. Listing of the complexities of the proposed structure and the existing digit-serial TOMs.

(a)

Structure	#XOR	#AND	#FF
[10]	$2d(m-1)T$	$2md$	$7m$
[22]*	$2md+4kd-2k+2d-2$	$2md+4kd-2k+4d-2$	$6m+2d+2k$
[23]	$2md+6m-6m/d+6$	$2md$	$4md+3m$
[24]	$2md$	$2md+2(d^2+d)/2$	$4m+2d$
NKATOM [20]	$(4d+1)m+4d^2+8d$	$4dm+4d^2$	$4m+2d$
KATOM (Fig. 4)	$(14.5n+9.5)d^{log_2 3}-14nd-6d+3.5n+1$	$(4n+4)d^{\log_2 3}$	$3nd+4d$

(b)

Structure	CPD	Latency
[10]	$T_A+(\lceil\log_2(T)\rceil+\lceil\log_2(m)\rceil)T_X$	$2\lceil m/d\rceil+2$
[22]	$T_A+(\lceil\log_2(d+1)\rceil)T_X$	$2\lceil m/d\rceil+2$
[23]	$2T_X$	$2\lceil m/d\rceil+2\lceil\log_2 d\rceil$
[24]	$T_A+(\lceil\log_2 d\rceil+2)T_X$	$2\lceil m/d\rceil$
NKATOM ([20])	$T_A+\lceil\log_2(d)\rceil T_X$	$\lceil m/d\rceil+2$
KATOM (Fig. 4)	$MAX(D_{t_0},D_{t_1},D_{t_2})\approx(2+2\log_2 d)T_X$	$n+2$

For the proposed one, $n=\lceil m/d\rceil$ (one can always check the corresponding references to get the detailed complexities, if there is a need).
*: The value of k is determined by the polynomial used, see [22].

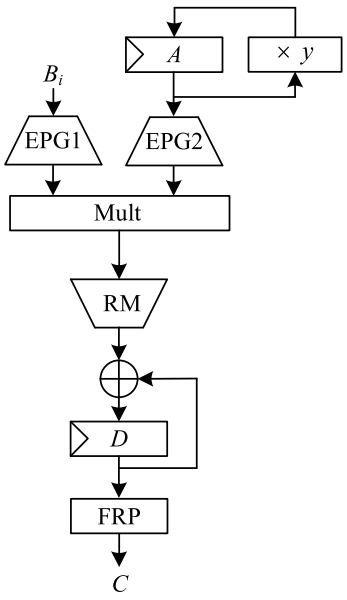


FIGURE 2. The proposed DL-SIPO KA-based multiplier.

- Mult unit: This step performs $T=\sum_{j=0}^{n-1}P_{B_i}\odot P_{A_j}y^j$ in Step 2.4 of Algorithm 1, which involves n PWM components. Therefore, the Mult unit requires $nS_A^{PWM}(d)$ space complexity.
- RM unit: The RM unit is based on the R component of KA decomposition to perform $D=D+\sum_{i=0}^{n-1}R(T_i)y^i$ in Step 2.4, and the result is stored in the register $<D>$. According to Step 2.4, we have obtained that RM unit is using n R components, namely, RM unit has $nS_X^R(d)$ space complexity with $D_X^R(d)$ XOR gate delay.
- FRP unit: This unit is operating $C=D \bmod F(x)=\sum_{i=0}^{n-1}D_iy^i \bmod F(x)$, where $y=x^d$. Based on the KA decomposition, each coefficient D_i has $(2d-1)$-bit digit-size. Thus, let us define $D_i=D_{0,i}+D_{1,i}y$, where $D_{0,i}$ and $D_{1,i}$ have d-bit and $(d-1)$-bit polynomials, respectively. We have

$$C=\overline{D}+D_{1,n-1}y^n \bmod F(x), \tag{5}$$

where

$$\overline{D}=D_{0,0}+(D_{1,0}+D_{0,1})y + \cdots+(D_{1,n-2}+D_{0,n-1})y^{n-1}.$$

The computation of $\overline{D}$ has $(n-1)(d-1)$ XOR gates. Since $\overline{D}+D_{1,n-1}y^n$ is $(nd+d-1)$-bit polynomial, the FRP unit involves $Q2=(n-1)(d-1)+2d$ XOR gates for trinomials, or $Q2=(n-1)(d-1)+4d$ XOR gates for pentanomials.

As shown in Fig. 2, the digit-serial multiplier is composed of three parts, so the designed multiplier requires $(n+2)$ clock cycles, and the critical-path delay (CPD) is $MAX(D_X^R(d)T_X, T_A+(D_X^{EPG}(d)+1)T_X)$. As analyzed above, the digit-serial multiplier has the following complexities:

$$\begin{aligned}
\#XOR &= (n+1)S_X^{EPG}(d)+nS_X^{PWM}(d)+nS_X^R(d)+Q1+Q2,\\
\#AND &= nS_A^{PWM}(d),\\
\#FF &= 3nd+nS_A^{PWM}(d)-n,\\
delay &= (n+2)MAX(D_X^R(d)T_X,\quad T_A+(D_X^{EPG}(d)+1)T_X).
\end{aligned} \tag{6}$$

IV. PROPOSED DIGIT-LEVEL SERIAL-IN PARALLEL-OUT THREE-OPERAND MULTIPLICATION

In this Section, we define a partial product formula to derive the proposed TOM algorithm to achieve an architecture with subquadratic space complexity.

Algorithm 1 The proposed DL-SIPO Multiplication Algorithm

Input: A and B are two polynomials in $GF(2^m)$
Output: $C = AB \bmod F(x)$
1. Initial step:
1.1. $A = A_0 + A_1 y + \cdots + A_{n-1} y^{n-1}$;
1.2. $B = B_0 + B_1 y + \cdots + B_{n-1} y^{n-1}$;
1.3. $D = 0$;
2. Multiplication step:
2.1. for $i = 0$ to $n-1$ do
2.2. $P_{B_i} = EPG(B_i)$;
2.3. $P_A = [P_{A_0}, P_{A_1}, \cdots, P_{A_{n-1}}]$, where $P_{A_i} = EPG(A_i)$;
2.4. $D = D + R(\sum_{j=0}^{n-1} P_{B_i} \odot P_{A_j} y^j)$;
2.5. $A = Ay \bmod \overline{F}(y)$;
2.6. end for
2.7. $C = D \bmod F(x)$;

A. DEFINITION OF THE PARTIAL PRODUCT FORMULA

Let the bivariate polynomial A in $GF(2^m)$ be written as $A = A_0 + A_1 y + \cdots + A_{n-1} y^{n-1}$ over $GF(2)$ with $y = x^d$. We can define the following polynomial formula as

$$A^{(i)} = A_0 + A_1 y + \cdots + A_i y^i. \tag{7}$$

When $i = 0$, we have $A^{(0)} = A_0$. In general, the polynomial $A^{(i)}$ can be re-expressed as

$$A^{(i)} = A_i y^i + A^{(i-1)}. \tag{8}$$

In order to derive the proposed TOM, let us define first the novel partial formula in the following theorem.

Theorem 1: Let A and C be two polynomials in $GF(2^m)$ constructed by the irreducible polynomial $F(y)$. We can define the partial product $D^{(i)} = (A_0 + A_1 y + \cdots + A_i y^i) C y^i \bmod \overline{F}(y)$, where $\overline{F}(y) = y^n + \overline{K}$. The partial product $D^{(i)}$ can then be re-expressed as $D^{(i)} = D^{(i-1)} y + A_i C y^{2i} \bmod \overline{F}(y)$.

Proof: Assume that the partial product is defined by $D^{(i)} = (A_0 + A_1 y + \cdots + A_i y^i) C y^i \bmod \overline{F}(y)$. The product $D^{(i)}$ can be rewritten as $D^{(i)} = (A_0 + A_1 y + \cdots + A_i y^i) C y^i \bmod \overline{F}(y) = [(A_0 + A_1 y + \cdots + A_{i-1} y^{i-1}) C y^{i-1}] y + A_i C y^{2i} \bmod \overline{F}(y) = D^{(i-1)} y + A_i C y^{2i} \bmod \overline{F}(y)$. □

Let us denote $C^{(i)} = C y^{2i} \bmod \overline{F}(y)$. The partial product $D^{(i)}$ in Theorem 1 can be re-expressed as

$$D^{(i)} = D^{(i-1)} y + A_i C^{(i)} \bmod \overline{F}(y). \tag{9}$$

Besides that, we can list each partial product $D^{(i)}$ as follows:

$$\begin{aligned} D^{(0)} &= A_0 C \bmod \overline{F}(y), \\ D^{(1)} &= (A_0 + A_1 y) C y \bmod \overline{F}(y) = D^{(0)} y + A_1 C^{(1)} \bmod \overline{F}(y), \\ &\cdots \\ D^{(i)} &= D^{(i-1)} y + A_i C^{(i)} \bmod \overline{F}(y). \end{aligned}$$

As stated previously, we can use the iterative relation of (2) to compute each partial product $D^{(i)}$. Following this, we employ (2) to derive a new digit-serial TOM in Section IV-B.

B. PROPOSED KA-BASED THREE-OPERAND MULTIPLIER

Using the polynomial presentation of (8), the product of $A^{(i)}$ and $B^{(i)}$ is rewritten as

$$\begin{aligned} A^{(i)} B^{(i)} &= (A_i y^i + A^{(i-1)})(B_i y^i + B^{(i-1)}) \\ &= A_i B_i y^{2i} + (A_i B^{(i-1)} + B_i A^{(i-1)}) y^i + A^{(i-1)} B^{(i-1)} \\ &= (A_i B^{(i)} + B_i A^{(i-1)}) y^i + A^{(i-1)} B^{(i-1)}. \end{aligned} \tag{10}$$

Given the recursive formula in (10), the partial product $A^{(i)} B^{(i)}$ can be obtained as

$$\begin{aligned} A^{(i)} B^{(i)} = A_0 B_0 + (A_1 B^{(1)} + B_1 A^{(0)}) y + \cdots + (A_i B^{(i)} \\ + B_i A^{(i-1)}) y^i = P^{(i)} + Q^{(i)}, \end{aligned} \tag{11}$$

where

$$\begin{aligned} P^{(i)} &= A_0 B_0 + A_1 B^{(1)} y + \cdots + A_i B^{(i)} y^i \bmod \overline{F}(y), \\ Q^{(i)} &= B_1 A^{(0)} y + B_2 A^{(1)} y + \cdots + B_i A^{(i-1)} y^i \bmod \overline{F}(y). \end{aligned}$$

We find that, since $i = n-1$, $C^{(n-1)} = P^{(n-1)} + Q^{(n-1)}$ is exactly the product of A and B. Based on the recurrence $A^{(i)} B^{(i)}$ in (11), the TOM is derived as

$$E = ABC \bmod F(x) = P^{(n-1)} C + Q^{(n-1)} C \bmod F(x), \tag{12}$$

where C is another polynomial in $GF(2^m)$. In the followings, we give the process to derive two partial products $P^{(n-1)} C$ and $Q^{(n-1)} C$.

- Computing $P^{(n-1)} C$: Based on the novel partial product formula in Theorem 1, $P^{(i)} C$ in (12) can be rewritten as

$$\begin{aligned} &P^{(i)} C \\ &= A_0 B^{(0)} C + A_1 B^{(1)} C y + \cdots + A_i B^{(i)} C y^i \bmod \overline{F}(y) \\ &= A_0 D_p^{(0)} + A_1 D_p^{(1)} + \cdots + A_i D_p^{(i)} \bmod \overline{F}(y), \end{aligned} \tag{13}$$

where

$$\begin{aligned} D_p^{(i)} &= (B_0 + B_1 y + \cdots + B_i y^i) C y^i \bmod \overline{F}(y) \\ &= D_p^{(i-1)} y + B_i C y^{2i} \bmod \overline{F}(x). \end{aligned}$$

Algorithm 2 illustrates the computation of $P^{(i)} C$ according to (13). Based on Algorithm 2, Fig. 3 shows the novel digit-serial KA-based multiplier for computing $P^{(i)} C$. In order to reduce the CPD, Fig. 3 is decomposed into three units (t_0, t_1, and t_2 units). We then use the KA decomposition to analyze the time and space complexities of these three units (based on Section III). The obtained complexities of EPG1, EPG2, Mult, RM, $\times y$, and FRP components are already listed in Section III. The t_0 unit performs Steps 6.1, 6.2, 6.3, and 6.6 of Algorithm 2, and it involves EPG1, EPG2, Mult-1, RM1, $\times y$, and $\times y^2$ components. The t_1 unit performs Steps 6.4 and 6.5 of Algorithm 2, and it involves EPG1, EPG2, Mult-2, RM2, and Add2 components. The t_2 unit performs Steps 8 of Algorithm 2, and it involves FRP component. At the initial step, register C is set as zero. After $(n+1)$ clock cycles, the product result is stored in register $< E >$, one extra clock cycle is required in the t_2 unit to produce the final result $P^{(n-1)} C$. Therefore, the computation of

Algorithm 2 Computing the Product $P^{(n-1)}C$ Based on KA Approach

Input: A, B, and C in $GF(2^m)$
Output: $P^{(n-1)}C = \sum_{i=0}^{n-1} A_i D_p^{(i)} \bmod \overline{F}(y)$, where $D_p^{(i)} = (B_0 + B_1 y + \cdots + B_i y^i)Cy^i \bmod \overline{F}(y)$ and $y = x^d$
Initial step:
1. $A = A_0 + A_1 y + \cdots + A_{n-1} y^{n-1}$
2. $B = B_0 + B_1 y + \cdots + B_{n-1} y^{n-1}$
3. $C = C_0 + C_1 y + \cdots + C_{n-1} y^{n-1}$
4. $D = 0$
5. $E = 0$
Multiplication step:
6. for $i = 0$ to n-1
   6.1. $P_{B_i} = EPG(B_i)$
   6.2. $P_C = [P_{C_0}, P_{C_1}, \cdots, P_{C_{n-1}}]$, where $P_{C_i} = EPG(C_i)$
   6.3. $D = (D \times y + R(P_{B_i} \odot P_C) \bmod \overline{F}(y)$, where $P_{B_i} \odot P_C = [P_{B_i} \odot P_{C_0}, P_{B_i} \odot P_{C_1}, \cdots, P_{B_i} \odot P_{C_{n-1}}]$
   6.4. $P_D = [P_{D_0}, P_{D_1}, \cdots, P_{D_n}]$ and $P_{A_i} = EPG(A_i)$, where $P_{D_i} = EPG(D_i)$
   6.5. $E = E + R(P_{A_i} \odot P_D)$
   6.6. $C = Cy^2 \bmod \overline{F}(y)$
7. end for
8. $E = FRP(E)$

$P^{(n-1)}C$ needs $(n+2)$ clock cycles, and the CPD is $MAX(D_{t_0}, D_{t_1}, D_{t_2})$.

- Computing $Q^{(n-1)}C$: Since $Q^{(n-1)}C = B_1 A^{(0)} Cy + \cdots + B_i A^{(n-2)} Cy^{n-1} \bmod \overline{F}(y)$, we can find that the term $D_q^{(i)} = A^{(i-1)} Cy^i$ in $Q^{(n-1)}C$ is unsuitable for the partial product formula in Theorem 1. To solve this problem, $D_q^{(i)}$ multiplied by y can be rewritten as

$$\begin{aligned} \overline{D}_q^{(i)} &= D_q^{(i)} y = \overline{A} Cy^i \bmod \overline{F}(x) \\ &= (\overline{A}_0 + \overline{A}_1 y + \cdots + \overline{A}_i y^i) Cy^i \bmod \overline{F}(x), \end{aligned} \quad (14)$$

where

$$\begin{aligned} \overline{A}_0 &= 0, \\ &\cdots, \\ \overline{A}_j &= A_{j-1}, \quad \text{for } 1 \le j \le i. \end{aligned}$$

As show in (14), we can then have

$$\overline{A} = \sum_{i=0}^{n-1} \overline{A}_i y^i = A \gg 1,$$

where the symbol "$\gg 1$" denotes the right shifting of polynomial A by sub-polynomial with d-bits. We can find that the result $\overline{D}_q^{(i)}$ is suitable for the partial product formula in Theorem 1. From (14), we have obtained $D_q^{(i)} = \overline{D}_q^{(i)} y^{-1}$. Thus, $Q^{(n-1)}C$ can be expressed as

$$\begin{aligned} Q^{(n-1)}C &= y^{-1} \overline{Q}^{(n-1)} C \bmod F(x) \\ &= y^{-1} (B_0 \overline{D}_q^{(0)} + B_1 \overline{D}_q^{(1)} \\ &\quad + \cdots + B_{n-1} \overline{D}_q^{(n-1)}) \bmod F(x). \end{aligned} \quad (15)$$

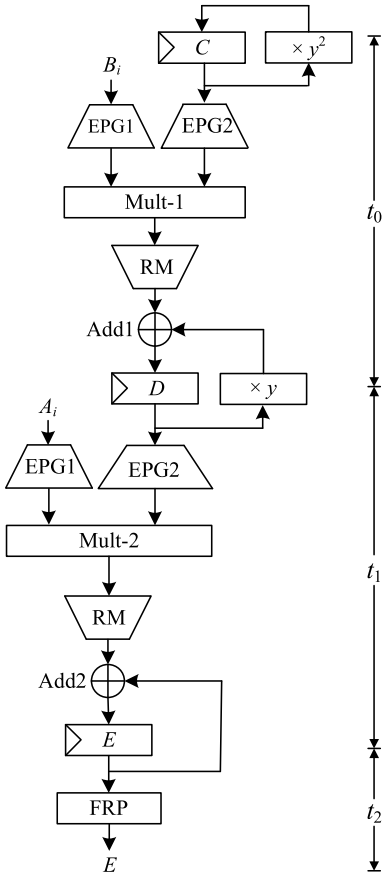


FIGURE 3. The proposed digit-serial KA-based multiplier for computing $P^{(i)}C$.

Therefore, we can use similar structure of Fig. 3 to compute $\overline{Q}^{(n-1)}C \bmod F(x)$ in (15).

According to the preceding analysis, the derived $P^{(n-1)}C$ and $Q^{(n-1)}C$ formulas have the same structures. Note that the KA block recombination (KABR) approach [21] so far leads the best KA decomposition. Based on the KABR decomposition, Algorithm 3 shows the proposed KATOM based on (13) and (15). Fig. 4 shows the proposed DL-SIPO KATOM based on Algorithm 3. As shown in Fig. 4, the proposed structure is divided into thee units (t_0, t_1, and t_2). In the followings, we analyze the complexities of thee units:

- t_0 unit: This unit performs Steps 6.1, 6.2, 6.3, and 6.6 of Algorithm 3. It involves two EPG1 components, one EPG2 component, two Mult-1 components, two Add1 components, two $\times y$ components, and one $\times y^2$ component. The value $Q1$ is the space complexity of $\times y$ component. As shown in Fig. 4, we have obtained that, based on the KA decomposition approach, EPG1 has one EPG component; EPG2 has n EPGs; Mult-1 has n PWM components. Add1 component involves nd XOR gates. Thus, t_0 unit has $(n+2)S_X^{EPG}(d) + 2nS_X^R(d) + 2S_X^{Add1} + 2S_X^{\times y} + S_X^{\times y^2}$ XOR gates and $2nS_A^{PWM}(d)$ AND gates, and its CPD is $D_{t_0} = (1 + D_X^{EPG}(d) + D_X^R(d))T_X + T_A$.

Algorithm 3 The Proposed TOM Based on KA Approach

Input: A, B, and C in $GF(2^m)$
Output: $P^{(n-1)}C = \sum_{i=0}^{n-1} A_i D_p^{(i)} \bmod \overline{F}(y)$, where $D_p^{(i)} = (B_0 + B_1 y + \cdots + B_i y^i) C y^i \bmod \overline{F}(y)$ and $y = x^d$
Initial step:
1. $A = A_0 + A_1 y + \cdots + A_{n-1} y^{n-1}$
2. $B = B_0 + B_1 y + \cdots + B_{n-1} y^{n-1}$
3. $C = C_0 + C_1 y + \cdots + C_{n-1} y^{n-1}$
4. $D_1 = D_2 = 0$
5. $E = 0$
Multiplication step:
6. for $i = 0$ to $n - 1$
6.1. $P_{B_i} = EPG(B_i)$, $P_{\overline{A}_i} = EPG(\overline{A}_i)$, and $P_C = [P_{C_0}, P_{C_1}, \cdots, P_{C_{n-1}}]$
6.2. $T_1 = P_{B_i} \odot P_C$ and $T_2 = P_{\overline{A}_i} \odot P_C$
6.3. $D_1 = (D_1 \times y + R(T_1)) \bmod \overline{F}(y)$ and $D_2 = (D_2 \times y + R(T_2)) \bmod \overline{F}(y)$
6.4. $P_{D_1} = [P_{D_{1,0}}, P_{D_{1,1}}, \cdots, P_{D_{1,n}}]$, $P_{D_2} = [P_{D_{2,0}}, P_{D_{2,1}}, \cdots, P_{D_{2,n}}]$, $P_{B_i} = EPG(B_i)$, and $P_{A_i} = EPG(A_i)$
6.5. $E = E + R(P_{A_i} \odot P_{D_1} + P_{A_i} \odot P_{D_2} \times y^{-1})$
6.6. $C = Cy^2 \bmod \overline{F}(y)$
7. end for
8. $E = FRP(E)$

- t_1 unit: This unit performs $P_{D_1} = [P_{D_{1,0}}, P_{D_{1,1}}, \cdots, P_{D_{1,n}}]$, $P_{D_2} = [P_{D_{2,0}}, P_{D_{2,1}}, \cdots, P_{D_{2,n}}]$, $P_{B_i} = EPG(B_i)$, $P_{A_i} = EPG(A_i)$, and $E = E + R(P_{A_i} \odot P_{D_1} + P_{A_i} \odot P_{D_2} \times y^{-1})$ in Steps 6.4 and 6.5 of Algorithm 3. As shown in Fig. 4, the symbol "<<1" is performed by $\times y^{-1}$ without doing modulo reduction, namely, it is done by right-to-left shifting. The computation of $E = E + R(P_{A_i} \odot P_{D_1} + P_{A_i} \odot P_{D_2} \times y^{-1})$ involves one Add2 component, "<<1", two EPG1 components, two EPG2 components, Add3 component, and two Mult-2 components. The Mult-2 involves $(n + 1)$ PWM components with T_A delay. Add2 performs the sum of $P_{A_i} \odot P_{D_1} + P_{A_i} \odot P_{D_2} \times y^{-1}$, where P_{D_1} involves $(n+1)P_{D_{1,i}}$ sub-product results, and each $P_{D_{1,i}}$ has $S_X^{PWM}(d)$ bits. Thus, Add2 has $nS_X^{PWM}(d)$ XOR gates with T_X delay. Add3 component has $nd + d$ XOR gates. RM2 has $(n + 1)$ R components. Thus, t_1 unit has $2nS_A^{PWM}$ AND gates and $(n + 1)S_X^R(d) + (2n + 4)S_X^{EPG}(d) + S_X^{Add2} + S_X^{Add3}$ XOR gates, and the CPD is $D_{t_1} = T_A + (D_X^{EPG}(d) + D_X^R(d) + 2)T_X$.
- t_2 unit: This unit performs $E = FRP(E)$ in Steps 8 of Algorithm 3. t_2 unit involves one FRP component, which has Q2 XOR gates, and the CPD is $D_{t_3} = 2T_X$.

At the initial step, register C is set as zero. After $(n + 1)$ clock cycles, the product result is stored in register $< E1 >$ and one more clock cycle is required for the t_3 unit to obtain the final result $E = ABC$. Therefore, the proposed KATOM structure needs $(n + 2)$ clock cycles, and the CPD is $MAX(D_{t_0}, D_{t_1}, D_{t_2})$. From the analysis above, the proposed

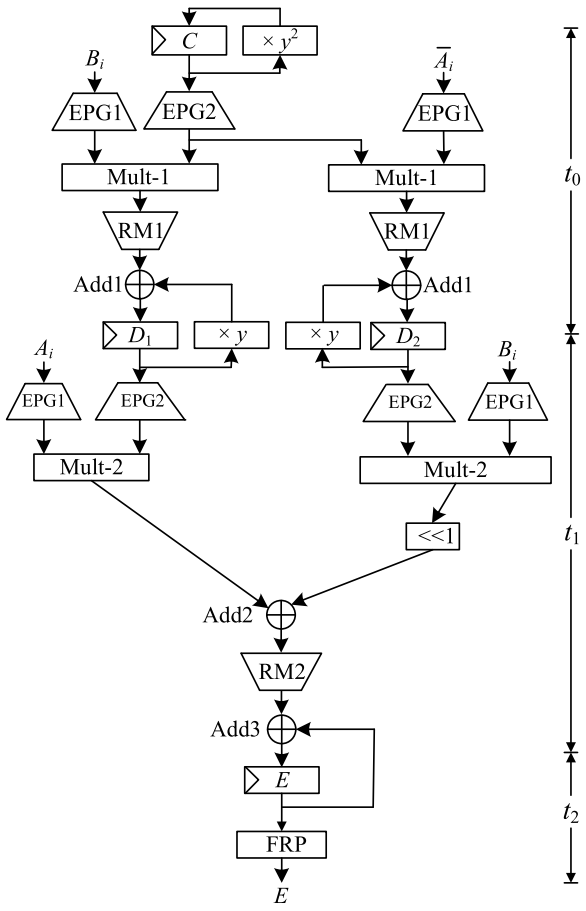


FIGURE 4. The proposed structure for computing TOM.

DL-SIPO KATOM is estimated as

$$\begin{aligned} \#XOR &= (3n+6)S_X^{EPG}(d) + S_X^{Add2} + 2S_X^{Add1} + S_X^{Add3} \\ &\quad + (3n+1)S_X^R(d) + 2S_X^{\times y} + S_X^{\times y^2} + S_X^{FRP}, \\ \#AND &= (4n+2)S_A^{PWM}(d), \\ \#FF &= 4nd + d, \\ delay &= MAX(D_{t_0}, D_{t_1}, D_{t_2}), \end{aligned} \tag{16}$$

where

$$\begin{aligned} S_X^{Add1} &= nd, \\ S_X^{Add2} &= nS_X^{PWM}(d), \\ S_X^{Add3} &= nd + d, \\ S_X^{\times y} &= Q1 = d, \\ S_X^{\times y^2} &= 2Q1, \\ S_X^{FRP} &= Q2 = (n-1)(d-1) + 2d. \end{aligned}$$

V. COMPLEXITY AND COMPARISON

The complexities of the proposed KATOM are evaluated on the situation when the field is generated by trinomials. Generally, the implementation of TOM can be realized through strategies such as digit-serial [20] structure,

TABLE 3. The synthesized results for our proposed structure and the best existing TOM over $GF(2^{409})$.

	Digit	Latency	#XOR	#AND	#FF	CPD	ACT	Area	ADP
NKATOM [20]	16	28	44,268	28,940	7,542	0.3	8.4	80,751	678,308
	32	16	91,152	60,060	7,687	0.34	5.1	158,900	810,390
KATOM (Fig. 4)	16	28	40,663	9,135	7,886	0.4	11.2	57,685	646,072
	32	16	67,262	13,961	8,248	0.48	7.68	89,472	687,145

ADP: Area-delay product = area × ACT.
Note that in [20], the authors have shown their design outperforms the ones of [10], [22], [23], and [24], we hence only list the design of [20] as comparison.

KA-based bit-parallel [19] design, and 2 two-operand multiplier ones [10], [22], [23], [24]. While the implementation of the proposed KATOM structure is based on the digit-serial approach (as seen in Fig. 4) combined with KA decomposition. Table 2 shows the comparison of the proposed structure and the existing TOM structures [20], [10], [22], [23], [24]. As shown in this table, for the same digit-size d, different structures have different area and time complexities determined by their structuring styles. But it is worth mentioning that the proposed KATOM structure can obtain subquadratric space complexity, which leads to lower area complexity than the existing ones.

To further estimate the area-time complexities of all these designs, we have used the FreePDK base kit [25] and the 45-*nm* NanGate's library to synthesize the proposed and the existing TOMs. Note that in [20], Lee *et al.* have shown their design outperforms the ones of [10], [22], [23], and [24], we hence only list the design of [20] as comparison. Both designs are synthesized at 1 GHz clock frequency. We have chosen digit-size of $d = 16$ and 32 to synthesize our proposed structure and the corresponding TOM ([20]) over $GF(2^{409})$. After that, we estimate the CPD (*ns*), latency (clock cycles), average computation time (ACT) (*ns*), and area complexity (um^2) of the two designs, respectively. Table 3 shows the synthesized results for our proposed structure and the existing TOM. As shown in this table, we find that the proposed KATOM structure, for the digit-size 16 and 32, has about 28.6% and 43.7% savings in area-complexity, respectively, when compared to the existing TOM, namely the proposed TOM has significant lower area-complexity than the existing one. Moreover, one can find that the proposed design has smaller area-delay product (ADP) than the competing one (at most 15.21% smaller), the overall area-time-complexities of the proposed TOM is better than the existing one though the proposed structure has slightly higher delay-complexity than the existing TOM.

VI. CONCLUSION

In this paper, through two interdependent stages' efforts, we have presented a novel KA-based TOM for low-complexity implementation. A novel digit-serial KA multiplier is introduced first. Then, we have defined a new partial product formula to obtain an efficient derivation of TOM algorithm. Based on the proposed algorithm, we have proposed an efficient KATOM structure to further reduce the space complexity (based on the proposed KA multiplier). As shown from the estimated results, the proposed KATOM structure has significant lower area complexity and smaller ADP when compared to the competing TOM. The proposed KATOM is quite regular and therefore can be extended and employed in many cryptographic applications.

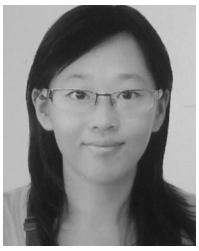
REFERENCES

[1] *Digital Signature Standard (DSS), Federal Information Processing Standards Publication 186-2*, Standard, National Institute Standards Technology, 2000.
[2] *Standard Specifications for Public Key Cryptography*, IEEE Standards 1363-2000, 2004. [Online]. Available: http://grouper.ieee.org/groups/1363/P1363/index.html
[3] A. J. Menezes, *Elliptic Curve Public Key Cryptosystems* (Springer International Series in Engineering and Computer Science). New York, NY, USA: Springer, 1993.
[4] Z. Liu, J. Groszschaedl, Z. Hu, K. Jarvinen, H. Wang, and I. Verbauwhede, "Elliptic curve cryptography with efficiently computable endomorphisms and its hardware implementations for the Internet of Things," *IEEE Trans. Comput.*, vol. 66, no. 5, pp. 773–785, May 2017.
[5] P. Saravanan and P. Kalpana, "Performance analysis of reversible finite field arithmetic architectures over GF(p) and $GF(2^m)$ in elliptic curve cryptography," *J. Circuits, Syst., Comput.*, vol. 24, no. 8, pp. 1550–1562, 2015.
[6] Y. Li and Y. Chen, "New bit-parallel Montgomery multiplier for trinomials using squaring operation," *Integr., VLSI J.*, vol. 52, pp. 142–155, Jan. 2016.
[7] Y. Li, G.-L. Chen, and J.-H. Li, "Speedup of bit-parallel Karatsuba multiplier in $GF(2^m)$ generated by trinomials," *Inf. Process. Lett.*, vol. 111, no. 8, pp. 390–394, 2011.
[8] F. Rodríguez-Henríquez and Ç. K. Koç, "Parallel multipliers based on special irreducible pentanomials," *IEEE Trans. Comput.*, vol. 52, no. 12, pp. 1535–1542, Dec. 2003.
[9] J. Xie, J. He, and P. K. Meher, "Low latency systolic Montgomery multiplier for finite field $GF(2^m)$ based on pentanomials," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 2, pp. 385–389, Feb. 2013.
[10] A. H. Namin, H. Wu, and M. Ahmadi, "A word-level finite field multiplier using normal basis," *IEEE Trans. Comput.*, vol. 60, no. 6, pp. 890–895, Jun. 2011.
[11] S. Talapatra, H. Rahaman, and J. Mathew, "Low complexity digit serial systolic Montgomery multipliers for special class of $GF(2^m)$," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 5, pp. 847–852, May 2010.
[12] A. Rezai and P. Keshavarzi, "High-performance scalable architecture for modular multiplication using a new digit-serial computation," *Microelectron. J.*, vol. 55, pp. 169–178, Sep. 2016.
[13] J.-S. Pan, C.-Y. Lee, and P. K. Meher, "Low-latency digit-serial and digit-parallel systolic multipliers for large binary extension fields," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 12, pp. 3195–3204, Dec. 2013.
[14] P. H. Namin, R. Muscedere, and M. Ahmadi, "A fully serial-in parallel-out digit-level finite field multiplier in $\mathbb{F}_{2^m}$ using redundant representation," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, vol. 64, no. 11, pp. 1337–1341, Nov. 2017.
[15] A. Weimerskirch and C. Paar, "Generalizations of the Karatsuba algorithm for efficient implementations," Ruhr Univ. Bochum, Bochum, Germany, Tech. Rep., 2003.
[16] P. L. Montgomery, "Five, six, and seven-term Karatsuba-like formulae," *IEEE Trans. Comput.*, vol. 54, no. 3, pp. 362–369, Mar. 2005.

[17] C.-Y. Lee, C.-S. Yang, B. K. Meher, P. K. Meher, and J.-S. Pan, "Low-complexity digit-serial and scalable SPB/GPB multipliers over large binary extension fields using (b,2)-way Karatsuba decomposition," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 11, pp. 3115–3124, Nov. 2014.
[18] C.-Y. Lee and P. K. Meher, "Subquadratic space-complexity digit-serial multipliers over $GF(2^m)$ using generalized (a,b)-way Karatsuba algorithm," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 4, pp. 1091–1098, Apr. 2015.
[19] C.-Y. Lee, P. K. Meher, and C.-P. Chang, "Efficient M-ary exponentiation over $GF(2^m)$ using subquadratic KA-based three-operand Montgomery multiplier," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 11, pp. 3125–3134, Nov. 2014.
[20] C.-Y. Lee, C.-C. Fan, and S.-M. Yuan, "New digit-serial three-operand multiplier over binary extension fields for high-performance applications," in *Proc. 2nd IEEE Int. Conf. Comput. Intell. Appl. (ICCIA)*, Sep. 2017, pp. 498–502.
[21] C.-H. Liu, C.-Y. Lee, and P. K. Meher, "Efficient digit-serial KA-based multiplier over binary extension fields using block recombination approach," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 8, pp. 2044–2051, Aug. 2015.
[22] S. Kumar, T. Wollinger, and C. Paar, "Optimum digit serial $GF(2^m)$ multipliers for curve-based cryptography," *IEEE Trans. Comput.*, vol. 55, no. 10, pp. 1306–1311, Oct. 2006.
[23] J. Xie, P. K. Meher, and Z.-H. Mao, "Low-latency high-throughput systolic multipliers over $GF(2^m)$ for NIST recommended pentanomials," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 3, pp. 881–890, Mar. 2015.
[24] R. Azarderakhsh and A. Reyhani-Masoleh, "Parallel and high-speed computations of elliptic curve cryptography using hybrid-double multipliers," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 6, pp. 1668–1677, Jun. 2015.
[25] *NanGate Standard Cell Library*. [Online]. Available: http://www.si2.org/openeda.si2.org/projects/nangatelib/

CHIOU-YNG LEE (SM'07) received the bachelor's degree in medical engineering and the M.S. degree in electronic engineering from Chung Yuan Christian University, Taiwan, in 1986 and 1992, respectively, and the Ph.D. degree in electrical engineering from Chang Gung University, Taiwan, in 2001.

From 1988 to 2005, he was a Research Associate with the Chunghwa Telecommunication Laboratory, Taiwan. From 2001 to 2005, he taught courses related to finite fields at Ching Yun University. He is currently a Professor with the Department of Computer Information and Network Engineering, Lunghwa University of Science and Technology. His research interests include computations in finite fields, error-control coding, signal processing, and digital transmission systems. He is a senior member of the IEEE Computer Society.



CHIA-CHEN FAN received the bachelor's degree in network engineering from the Lunghwa University of Technology in 2011 and the M.S. degree in electronic engineering from the Taipei University of Technology, Taiwan, in 2013.

She is currently pursuing the Ph.D. degree with the Department of Computer Science, National Chiao Tung University. Her research interests include computations in finite fields, error-control coding, and signal processing.

JIAFENG XIE (M'15) received the B.E. degree in measurement and control technology and instrumentation from Yanshan University, China, in 2006, the M.E. degree in control science and engineering from Central South University, China, in 2010, and the Ph.D. degree in electrical engineering from the University of Pittsburgh in 2014.

He is currently an Assistant Professor with the Department of Electrical Engineering, Wright State University. His research interests include VLSI cryptographic circuits design, intelligent system fault detection, hardware security, and VLSI signal image processing systems.

He is currently serving on the Editorial Board of *Microelectronics Journal* (Elsevier).

SHYAN-MING YUAN received the M.S. degree in computer science from the University of Maryland at Baltimore County, Catonsville, in 1985, and the Ph.D. degree in computer science from the University of Maryland at College Park, College Park, in 1989.

From 1989 to 1990, he was a Member of Technical Staff with ATC, CCL, ITRI, Taiwan. He is currently a Professor with the Department of Computer Science, National Chiao Tung University. His current research interests include distributed system design, fault-tolerant computing, network management, computer-supported cooperative work, multimedia application environments, and intelligent computer-assisted learning in distinct cooperative learning environments.