High-Performance Two-Dimensional Finite Field Multiplication and Exponentiation for Cryptographic Applications

Reza Azarderakhsh, *Member, IEEE*, and Mehran Mozaffari-Kermani, *Member, IEEE*

***Abstract*—Finite field arithmetic operations have been traditionally used in different applications ranging from error control coding to cryptographic computations. Among these computations are normal basis multiplications and exponentiations which are utilized in efficient applications due to their advantageous characteristics and the fact that squaring (and subsequent powering by two) of elements can be obtained with no hardware complexity. In this paper, we present 2-D decomposition systolic-oriented algorithms to develop systolic structures for digit-level Gaussian normal basis multiplication and exponentiation over $GF(2^m)$. The proposed high-performance architectures are suitable for a number of applications, e.g., architectures for elliptic curve Diffie–Hellman key agreement scheme in cryptography. The results of the benchmark of efficiency, performance, and implementation metrics of such architectures through a 65-nm application-specific integrated circuit platform confirm high-performance structures for the multiplication and exponentiation architectures presented in this paper are suitable for high-speed architectures, including cryptographic applications.**

***Index Terms*—Cryptography, finite field, Gaussian normal basis (GNB), security, systolic architecture.**

I. Introduction

THE PERFORMANCE and efficiency constraints of the hardware architectures of the systems embedded in diverse usage models utilizing finite field arithmetic including error control coding and cryptographic solutions necessitate having high-speed arithmetic units. Among these finite field arithmetic operations whose use is widespread and whose performance affects the aforementioned applications are exponentiations over $GF(2^m)$. In cryptography, much research work has been performed to achieve high-performance designs of the arithmetic units within [1]–[3]. Moreover, Gaussian normal basis (GNB) arithmetic operations have been utilized in the previous works for such applications (see [4], [5], [7], [8]).

Modular multiplication and exponentiation are key arithmetic operations for implementing error correcting codes (and Reed–Solomon codes [9]), and crypto-solutions Diffie–Hellman key exchange [10]. Elliptic curve discrete signature algorithm requires multiplications and single- and double-exponentiation. Double-exponentiation is widely applied in Schnorr- and ElGamal-like signature verifications [11], [12], and in digital signature standard [13]. For such applications, it is desirable to have high-performance exponentiation algorithms and hardware architectures. From the field element representation perspective, the major advantage of normal basis representation in $GF(2^m)$ is that squaring of field elements can be performed by cyclic shift of their coordinates. In this regard, these multipliers are effectively applied for deriving exponentiations. Exponentiations in $GF(2^m)$ are traditionally performed using the ordinary binary method. Reference [14] presents an exponentiation architecture using the normal basis. Moreover, Chiou and Lee [15] presented fast multiplexer-based double-exponentiation using the normal basis and the modified Booth's algorithm.

Digit-serial GNB multipliers can be employed for the applications where high-performance results are required. In the extended binary field $GF(2^m)$, various efficient systolic array multipliers have been suggested and can be classified into bit-parallel, bit-serial, and digit-serial architectures. Efficient bit-parallel systolic multipliers have the major advantage of high-throughout computations. However, these architectures for polynomial basis in $GF(2^m)$ usually require $O(m^2)$ XOR gates, AND gates, and 1-bit latches, and have $O(m)$ latency. Bit-serial systolic array multipliers require only $O(m)$ space complexity; nevertheless, they take longer computation time which in turn affects the performance metrics of their hardware implementations. For large finite fields in $GF(2^m)$, exponentiation can be performed through high-speed systolic array approach to achieve regular implementations [16]. Such arrays exhibit hardware structure units which are quite similar for varying choices of m, given $GF(2^m)$, leading to speed boost in the resulting structures. Low-complexity digit-level parallel-in parallel-out (DL-PIPO) GNB multipliers have been proposed in [17]–[19]. In [20] and [21], efficient semi-systolic arithmetic units for even-type GNB have been presented. Recently, Azarderakhsh *et al.* [5] presented a systolic array variant of GNB multiplier which outperforms previous architectures available in the literature. In order to achieve low-total computation times and thus high-performance architectures, in this paper, we propose two-dimensional (2-D) new systolic

Manuscript received October 1, 2014; revised February 18, 2015; accepted April 3, 2015. Date of publication April 21, 2015; date of current version September 17, 2015. This work was supported by the Texas Instruments Faculty Award. This paper was recommended by Associate Editor R. Karri.

R. Azarderakhsh is with the Department of Computer Engineering, Rochester Institute of Technology, Rochester, NY 14623-5603 USA (e-mail: rxaeec@rit.edu).

M. Mozaffari-Kermani is with the Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology, Rochester, NY 14623-5603 USA (e-mail: m.mozaffari@rit.edu).

Digital Object Identifier 10.1109/TCAD.2015.2424928

GNB exponentiation architectures over $GF(2^m)$ based on DL-PIPO structures (see also [6] for a recent related work). Utilizing our proposed systolic architectures, single- and double-exponentiation architectures are constructed leveraging systolic arrays. The main contributions of this paper are as follows (note that the 1-D variant for multiplications has been presented in [5]).

1) We propose 2-D systolic architectures for the GNB digit-level exponentiations. The proposed exponentiations with 2-D systolic architectures can achieve low-latency structures; specifically, the presented structures are based on 2-D systolic multipliers over $GF(2^m)$ which require the low latency of $\leq 3\lceil\sqrt[3]{m/d}\rceil$ clock cycles, where d is the selected digit-size.
2) Investing on the properties of normal basis representation, we utilize the proposed 2-D multiplication schemes to develop single- and double-exponentiation. Our analysis shows that the proposed exponentiations can achieve low-critical path delays and latency, suitable for high-performance applications.
3) Finally, using a 65-nm CMOS standard-cell library, for different field- and digit-size, we present and compare the results of our application-specific integrated circuit (ASIC) synthesis for DL-PIPO GNB multipliers to benchmark the efficiencies of the presented architectures.

We emphasize that the purpose of this paper is to push the speed limits of the multiplication and exponentiation computations which are suitable for high-performance applications in emerging, high-speed usage models. The rest of this paper is organized as follows. Section II presents the preliminaries regarding the utilized GNB multiplier and its optimized architecture. In Section III, we present the 2-D GNB digit-level systolic multiplications and single- and double-exponentiation architectures, along with the detailed analysis of their area and delay complexities. ASIC results are presented in Section IV. Finally, we conclude the proposed work in Section V.

II. Review of the Traditional Digit-Level GNB Architectures

The set $N = \{\beta, \beta^2, \beta^{2^2}, \ldots, \beta^{2^{m-1}}\}$ is denoted as the normal basis of $GF(2^m)$ for $\beta \in GF(2^m)$ [β represents a normal element of $GF(2^m)$]. Let us assume that k is the multiplication order of 2 modulo p, and $m > 1$ and $T > 1$ be two integers such that $p = mT + 1$ be a prime number and $\gcd(mT/k,\ m) = 1$. Moreover, let us consider α as a primitive $mT + 1$th root of unity in $GF(2^{Tm})$. Then, for any primitive, the Tth root of unity k in $\mathbb{Z}_p$, $\beta = \sum_{i=0}^{T-1} \alpha^{k^i}$ generates a normal basis of $GF(2^m)$ over $GF(2)$ which is called the GNB of type T. GNB is a special class of normal basis over which the multiplication is based on a multiplication matrix $\mathbf{M}_{m\times m}$ [22]. For the sake of simplicity, it is a common practice to store the columns of 1s of the multiplication matrix $\mathbf{M}$ instead of the entire $\mathbf{M}$. Therefore, we only need to store those in rows 1 up to $m-1$ and build a new matrix $\mathbf{R}_{(m-1)\times T}$ [18]. The multiplication matrix has the following properties.

1) The matrix $\mathbf{M}_{m\times m}$ is symmetric.
2) All of its diagonal entries are zero except for the last entry.
3) The first row has just one nonzero entry.
4) row$(m-i)$ is the i-fold left cyclic shift of row(m).

For an element $A \in GF(2^m)$, squaring can be achieved by simple right cyclic shift of A, i.e., $A^2 = \sum_{i=0}^{m-1} a_i\beta^{2^{i+1}} = (a_{m-1}, a_0, a_1, \ldots, a_{m-2})$ [13]. The GNB exists for every $m > 1$ that is not divisible by eight [23]. The complexities of GNB multipliers (in terms of time and area) depend on their type $T > 1$. For the five binary fields recommended by the NIST, i.e., $m = 163, 233, 283, 409,$ and 571, the values of T are even, and are $4, 2, 6, 4,$ and 10, respectively.

Low-complexity DL-PIPO GNB multipliers have been proposed in [17] and [18] (optimized in [19]). The results of such schemes are available in parallel after $q = \lceil m/d\rceil$, $1 \leq d \leq m$, clock cycles [latency] (d is the digit-size in digit-level architectures [number of bits for each selected digit] and m is the field size). The time complexity of the digit-level GNB multiplier is $T_A + (\lceil\log_2 T\rceil + \lceil\log_2(d+1)\rceil)T_X$, and its area complexity is dm AND gates and $\leq d(m-1)/2(T-1) + dm$ XOR gates, where T_A and T_X are the delays of an AND and an XOR gate. The area complexity is further reduced by a common subexpression elimination algorithm proposed in [19] to $n_p + d(m-1)/2(T/2-1) + dm$ XOR gates, where $n_p \leq \min\{(d(m-1)/2T/2), \binom{m}{2}\}$ is a value used for the sake of brevity.

For very large-scale integration implementations, double-exponentiation using the normal basis and the modified Booth's algorithm has been originally developed in [15]. Here, we briefly review this double-exponentiation algorithm. Let A and B be two normal basis elements over $GF(2^m)$ and $P = p_0 2^0 + p_1 2^1 + \cdots + p_{m-1}2^{m-1}$ and $Q = q_0 2^0 + q_1 2^1 + \cdots + q_{m-1}2^{m-1}$ be two positive integers in m-bit binary representations. The double-exponentiation of the form $A^P B^Q$

Algorithm 1 Chiou–Lee's Double-Exponentiation [15]

Inputs: $A, B \in GF(2^m)$ and P, Q are two positive integers with m-bit binary representations.
Output: $C = A^P B^Q$.
1. Initial step:
   $C = 1$
   $U = AB$
   $V = 1$ (just used for clarity, in case $p_i q_i = 00$)
2. Multiplication step:
   for $i = 0$ **to** $m-1$ **do**
   if $(p_i q_i = 00)$ then $C = C \cdot V$
   if $(p_i q_i = 01)$ then $C = C \cdot B$
   if $(p_i q_i = 10)$ then $C = C \cdot A$
   if $(p_i q_i = 11)$ then $C = C \cdot U$
   $U = U^2$
   $A = A^2$
   $B = B^2$
   endfor

can be computed as

$$\begin{aligned} A^P B^Q &= A^{p_0 2^0 + p_1 2^1 + \cdots + p_{m-1} 2^{m-1}} B^{q_0 2^0 + q_1 2^1 + \cdots + q_{m-1} 2^{m-1}} \\ &= \left(A^{p_0} B^{q_0}\right)^{2^0} \left(A^{p_1} B^{q_1}\right)^{2^1} \cdots \left(A^{p_{m-1}} B^{q_{m-1}}\right)^{2^{m-1}} \\ &= U_0^{2^0} U_1^{2^1} \cdots U_{m-1}^{2^{m-1}} \end{aligned} \tag{1}$$

where $U_i = A^{p_i} B^{q_i}$ and p_i, $q_i \in GF(2)$ for $0 \le i \le m-1$. The function U_i, $0 \le i \le m-1$, utilizes the binary values p_i and q_i to determine one of the four values 1, A, B, and AB. Therefore, the multiplication of A and B needs to be precomputed before performing the double-exponentiation. Based on (1), the double-exponentiation is described in Algorithm 1.

III. Proposed 2-D GNB Exponentiation Architectures

In what follows, first, for reaching high-performance implementations, we present a 2-D systolic GNB multiplier which is capable of reaching efficient architectures. Then, the proposed 2-D systolic GNB exponentiation architectures are proposed.

For the DL-PIPO architecture due to the symmetry of the multiplication matrix $\mathbf{R}_{(m-1)\times T}$, and the fact the two input operands are both available we can define reduced matrix $u_{m-1/2\times T}$ which has half of the rows of $\mathbf{R}_{(m-1)\times T}$ as previously proposed in [18]. Assume that for the elements of the input A (preloaded in a register denoted by $\langle X\rangle$) we have $\langle X\rangle = (x_0, x_{m-1}, x_{m-2}, \ldots, x_2, x_1) = \bar{A} \gg 1$, where $\bar{A} = \sum_{i=0}^{m-1} a_{m-1-i}\beta^{2^i}$. Since the reduced multiplication matrix $u_{m-1/2\times T}$ is multiplied only by the input operand B which is preloaded to the register $\langle Y\rangle$, then the product of A and B using the reduced matrix $u_{m-1/2\times T} = [u_k]_{k=1}^{m-1/2}$, where u_k is the row k, can be obtained using the following formulation:

$$C = AB = \sum_{i=0}^{m-1} J^{2^i}(X \gg i, B \gg i) \tag{2}$$

where

$$J(X, Y) = X \odot P(Y)$$

and $P(Y) = (y_1, s'(1, Y), s'(2, Y), \ldots, s'(2, Y), s'(1, Y))$, $s'(k, B) = (B \ll R(2k, 1)) \oplus (B \ll R(2k, 2)) \oplus \cdots \oplus (B \ll R(2k, T))$, $1 \le k \le m - 1/2$. For each coordinate, a $J(X, Y)$ function with appropriately shifted inputs must be computed. For detailed information, one needs to refer to [18]. Let $q = \lceil m/d \rceil$, $1 \le d \le m$, then one can write the product C in (2) as

$$C = \sum_{i=0}^{q-1} L^{2^{id}}(X \gg id, B \gg id) \tag{3}$$

where

$$L(X, B) = \sum_{j=0}^{d-1} J^{2^j}(X \gg j, B \gg j). \tag{4}$$

Let k and n be two integers to satisfy $q = k^2 n$. Note that if q is not divisible by k^2, we can zero-pad X and B to satisfy $q = k^2 n$. To derive the 2-D digit-level systolic multiplier, let us decompose (3) into the sum of n partial results as

$$C = C_0 + C_1^{2^{k^2 d}} + \cdots + C_{n-1}^{2^{(n-1)k^2 d}} \tag{5}$$

where

$$C_i = \sum_{j=0}^{k-1} C_{ij}^{2^{kjd}} \tag{6}$$

$$C_{ij} = \sum_{z=0}^{k-1} L^{2^{dz}}(X_{ij} \ll dz, B_{ij} \ll dz),$$

$$X_{ij} = X \gg k^2 id + kjd \tag{7}$$

and

$$B_{ij} = B \gg k^2 id + kjd.$$

Each of the partial results C_{ij} is the sum of k partial products in (4) and the partial result C_i in (6) is the sum of k-term partial result C_{ij}. In this regard, the proposed systolic multiplier with 2-D systolic array structure for computing partial result C_i is shown in Fig. 1. In Fig. 1, the proposed 2-D systolic multiplier is composed of k systolic arrays, $(k-1)$ cyclic shifting (CS) circuits, $(k-1)$ accumulation circuit (AC1) architectures, and one AC2 architecture. Each CS block provides kd cyclic shifts to the right and is only rewiring in hardware.

Proposition 1: Let the field be constructed from even type-T GNB. Then, the proposed 2-D systolic GNB multiplier needs the maximum latency of $3\lceil\sqrt[3]{m/d}\rceil$ clock cycles with the number of processing element (PEs) selected by $\lceil\sqrt[3]{(m/d)}\rceil^2$.

Proof: Let k and n be two positive integers to satisfy $\lceil m/d\rceil = k^2 n$, where d is selected digit-size. Since GNB multiplication uses k^2 PEs to construct the 2-D systolic array architecture, if this circuit computes C_i, then, we have $2k$ clock cycles. Therefore, the GNB multiplication requires $2k + n$ clock cycles. The first derivative of $2k + n(= 2k + q/k^2)$ needs to be zero and we have: $2 - 2q/k^3 = 0$ which leads to $k = \sqrt[3]{q} = \sqrt[3]{m/d}$. So, we select $k = n = \lceil\sqrt[3]{m/d}\rceil$ (GNB multiplication uses k^2 PEs, i.e.,$\lceil\sqrt[3]{(m/d)}\rceil^2$), and the 2-D systolic multiplier requires the latency of $2k + n = 3\lceil\sqrt[3]{m/d}\rceil$ clock cycles. Whenever, m/d is not perfect cube, the computation might be performed in fewer cycles and hence the proof is complete. ■

Remark 1: According to Proposition 1, the latency of our proposed 2-D systolic GNB multiplier is at most $3\lceil\sqrt[3]{m}\rceil$ clock cycles.

By using 2-D systolic array implementation, the proposed 2-D systolic multiplier is composed of $\lceil\sqrt[3]{(m/d)^2}\rceil$ PEs, $(\lceil\sqrt[3]{m/d}\rceil - 1)$ CSs, $(\lceil\sqrt[3]{m/d}\rceil - 1)$ AC1s, and one AC2. Using this structure, the latency of the GNB multiplier can reach the minimum latency of $\le 3\lceil\sqrt[3]{m/d}\rceil$ clock cycles. For instance, using digit-size of one, the 2-D systolic multiplier over $GF(2^{409})$ has the latency of 24 clock cycles.

Let A be a field element in $GF(2^m)$ over an even type-T GNB and let P be a positive m-bit integer with binary representation as $P = (p_0, p_1, \ldots, p_{m-1})_2$. Assume that B is represented by $B = A^{2^{\lceil m/2\rceil}}$ and P is represented by

$$P = p_0 2^0 + p_1 2^1 + \cdots + p_{m-1} 2^{m-1} = K + 2^{\lceil \frac{m}{2} \rceil} Q \tag{8}$$

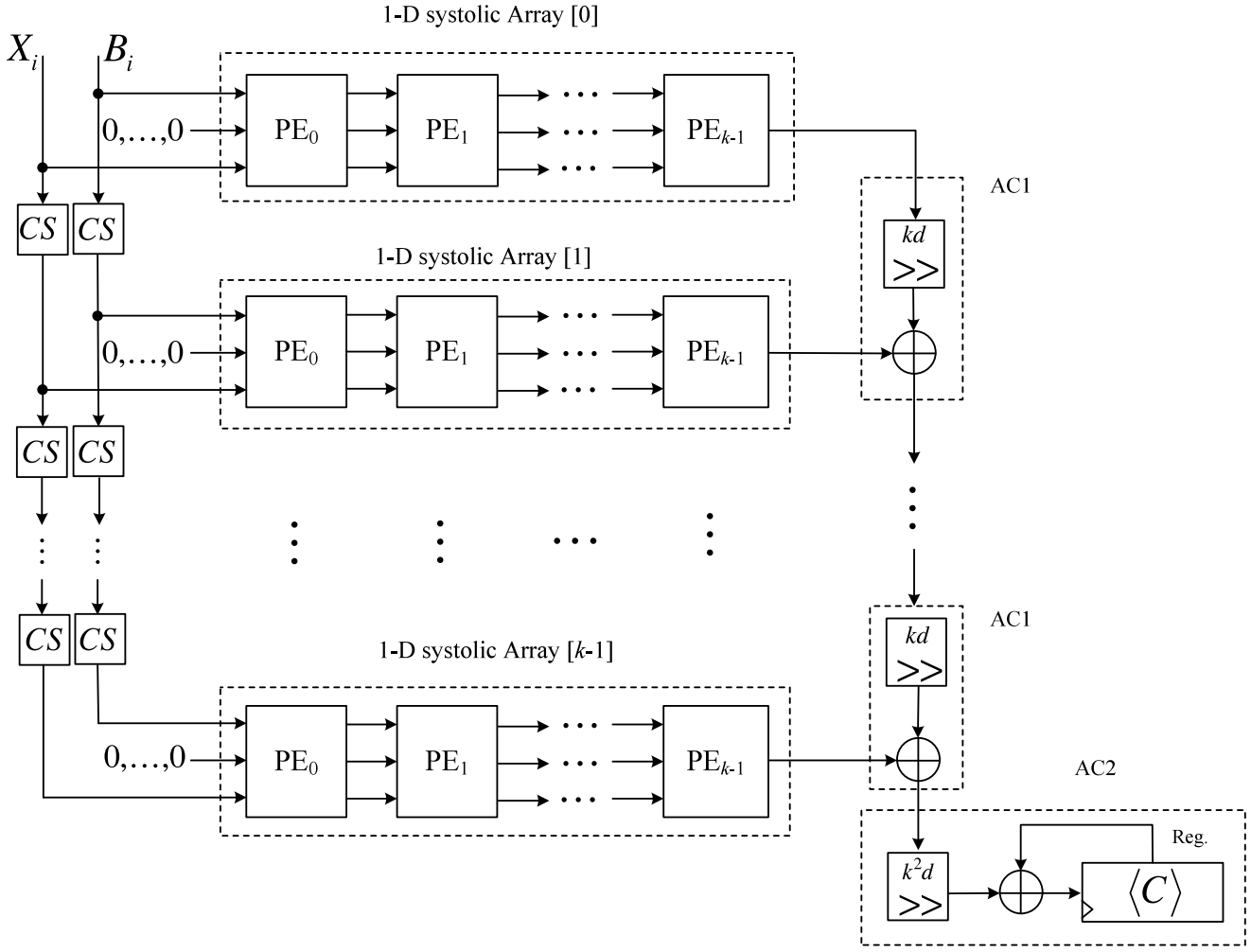

Fig. 1. Proposed architecture for 2-D digit-level systolic GNB multiplier.

where

$$K = k_0 + k_1 2 + \cdots + k_{\lceil \frac{m}{2} \rceil - 1} 2^{\lceil \frac{m}{2} \rceil - 1}$$
$$Q = q_0 + q_1 2 + \cdots + q_{\lceil \frac{m}{2} \rceil - 1} 2^{\lceil \frac{m}{2} \rceil - 1}$$
$$k_i = p_i$$
$$q_i = p_{\lceil \frac{m}{2} \rceil + i}.$$

Note that if m is an odd integer, the coefficient $q_{\lceil m/2 \rceil - 1}$ is set to zero. Thus, computing the exponentiation in the form of A^P yields to

$$\begin{aligned} A^P &= A^{K+2^{\lceil \frac{m}{2} \rceil} Q} = A^K B^Q \\ &= \left(A^{k_0} B^{q_0}\right)^{2^0} \left(A^{k_1} B^{q_1}\right)^{2^1} \cdots \left(A^{k_{\lceil \frac{m}{2} \rceil - 1}} B^{q_{\lceil \frac{m}{2} \rceil - 1}}\right)^{2^{\lceil \frac{m}{2} \rceil - 1}} \\ &= (U_0)^{2^0} (U_1)^{2^1} \cdots \left(U_{\lceil \frac{m}{2} \rceil - 1}\right)^{2^{\lceil \frac{m}{2} \rceil - 1}} \end{aligned} \tag{9}$$

where $U_i = A^{k_i} B^{q_i}$ for $0 \le i \le \lceil m/2 \rceil - 1$ and k_i, $q_i \in GF(2)$. We can use 4-to-1 multiplexers (denoted hereafter by $\text{MUX}_{4\times1}$) to realize the function U_i. The inputs of $\text{MUX}_{4\times1}$ are A, B, AB, and 1. Note that the multiplication of A and B is precomputed before performing the exponentiation. For computing (9), we require $\lceil m/2 \rceil$ multiplications repeated using the normal basis multiplier and the squarer circuit. It is noted that squaring is cost-free in hardware in the normal basis representation.

Let us first utilize the regular systolic array for computing the GNB multiplication. Applying this method, let us define $r = \lceil \sqrt{m/2} \rceil$, then, the exponentiation in (9) is decomposed by r-term partial results as follows:

$$A^P = C_0 C_1 \cdots C_{r-1} \tag{10}$$

where

$$C_i = (U_{ri})^{2^{ri}} (U_{ri+1})^{2^{ri+1}} \cdots (U_{ri+r-1})^{2^{ri+r-1}}. \tag{11}$$

From the relation of $U_i = A^{k_i} B^{q_i}$, let us define

$$U_i^{(h)} = (U_i)^{2^h} = \left(A^{k_i} B^{q_i}\right)^{2^h} = \left(A^{2^h}\right)^{k_i} \left(B^{2^h}\right)^{q_i} \tag{12}$$

where h is a positive number. Therefore, the partial result in (11) can be rewritten as

$$\begin{aligned} C_i &= \left(U_{ri}^{(ri)}\right)^{2^0} \left(U_{ri+1}^{(ri)}\right)^{2^1} \cdots \left(U_{ri+r-1}^{(ri)}\right)^{2^{r-1}} \\ &= \left(\left(\left(U_{ri+r-1}^{(ri)}\right)^2 U_{ri+r-2}^{(ri)}\right)^2 \cdots\right)^2 U_{ri}^{(ri)}. \end{aligned} \tag{13}$$

Algorithm 2 Proposed Single-Exponentiation Algorithm

Inputs: $A \in GF(2^m)$ and P is a positive integer with m-bit binary representation.
Output: $C = A^P$.
1. Initial step:
1.1 $B = A^{2^{\lceil \frac{m}{2} \rceil}}$
1.2 $C = 1$
1.3 $D = AB$
1.4 $U = 1$
1.5 $K = k_0 + k_1 2 + \cdots + k_{\lceil \frac{m}{2} \rceil - 1} 2^{\lceil \frac{m}{2} \rceil - 1}$, where $k_i = p_i$
1.6 $Q = q_0 + q_1 2 + \cdots + q_{\lceil \frac{m}{2} \rceil - 1} 2^{\lceil \frac{m}{2} \rceil - 1}$, where $q_i = p_{\lceil \frac{m}{2} \rceil + i}$
2. Multiplication step:
2.1 **for** $i = 0$ **to** $r - 1$ **do**
2.2 $C_i = 1$
2.3 $B = B \gg ir$
2.4 $A = A \gg ir$
2.5 $D = D \gg ir$
2.6 **for** $j = r - 1$ **to** 0 **do**
2.7 $C_i = C_i^2$
2.8 if $(k_{ir+j} q_{ir+j} = 00)$ $C_i = C_i \cdot U$
2.9 if $(k_{ir+j} q_{ir+j} = 01)$ $C_i = C_i \cdot B$
2.10 if $(k_{ir+j} q_{ir+j} = 10)$ $C_i = C_i \cdot A$
2.11 if $(k_{ir+j} q_{ir+j} = 11)$ $C_i = C_i \cdot D$
2.12 **endfor**
2.13 $C = C \cdot C_i$
2.14 **endfor**
3. **Return** $C = A^P$.

From the structure of (13), computing C_i requires precomputing the value $A^{2^{ri}} B^{2^{ri}}$. Based on the above, the proposed exponentiation is based on Algorithm 2. In the initial step, the register $\langle C \rangle$ is initialized with $1 \in GF(2^m)$, where "1" $= (1, 1, \ldots, 1)$ in GNB. Moreover, the values of $B = A^{2^{\lceil m/2 \rceil}}$ and $AB = A^{2^{\lceil m/2 \rceil}} \times A = A^{2^{\lceil m/2 \rceil}+1}$ are precomputed as the inputs of the $\text{MUX}_{4\times1}$. Therefore, in this step, we need one multiplication for computing AB. In addition to this multiplication, from the structure of (10), computing exponentiation is divided by r-term partial products C_i. In the multiplication step, computing each partial product C_i requires r-time GNB multiplications, to be stored in the register $\langle C \rangle$. Thus, computing exponentiation in the multiplication step requires $(2r - 1)$ GNB multiplications. As a result, we require the total number of $2r$ GNB multiplications which determines the latency of the entire operation.

For clarity, we give the following illustrative example to go over the time and space complexities of the proposed scheme.

Example 1: Let the element $A \in GF(2^{17})$ and $P = p_0 2^0 + p_1 2^1 + \cdots + p_{16} 2^{16}$ be a positive 17-bit integer. Since $\lceil 17/2 \rceil = 9$, let us define that $B = A^{2^9}$, $K = k_0 2^0 + k_1 2^1 + \cdots + k_8 2^8$, and $Q = q_0 2^0 + q_1 2^1 + \cdots + q_8 2^8$, where $k_i = p_i$ for $0 \le i \le 8$, $q_i = p_{i+9}$ for $0 \le i \le 7$, and $q_8 = 0$. Let us define $r = \sqrt{9} = 3$. Given (10), the exponentiation $C = A^P$ can be represented as $C = A^P = A^{K + 2^9 Q} = A^K B^Q = C_0 C_1 C_2$, where $C_i = ((U_{3i+2}^{(3i)})^2 U_{3i+1}^{(3i)})^2 U_{3i}^{(3i)}$, and $U_{3i+j}^{(3i)} = (A^{2^{3i}})^{k_{3i+j}} (B^{2^{3i}})^{q_{3i+j}}$ for $0 \le i, j \le 2$. We select $P = K + 2^9 Q = 104\,853$, where $K = 405 = (110010101)_2$ and $Q = 204 = (011001100)_2$.

Let us now propose the constructed scheme for the 2-D systolic array architecture for computing the GNB multiplication. Let us define $r = \lceil \sqrt[3]{m/2} \rceil$ (note to differentiate this r and the one for the 2-D multiplier array approach), then, the exponentiation in (9) can be represented by

$$A^P = C_0 C_1 \cdots C_{r-1}$$

where

$$C_i = \prod_{j=0}^{r-1} C_{i,j}, \tag{14}$$

$$C_{ij} = \prod_{l=0}^{r-1} \left(U_{ir^2+jr+l}^{(ir^2+jr)} \right)^{2^l}. \tag{15}$$

From (15), C_{ij} requires $r - 1$ GNB multipliers to perform the product of r-term $U_{ir^2+jr+l}^{(ir^2+jr)}$ for $0 \le l \le r - 1$. Moreover, C_i in (14) is represented by the product of r-term C_{ij}. Thus, based on (14) and (15), Fig. 2 depicts the proposed 2-D multiplier array to calculate the C_i multiplication. In this figure, the multiplier M_S in the multiplier array $[i]$ computes the multiplication result of the multiplier array $[i]$ multiplied by the previous product results. After finalizing the 2-D array multiplication for computing C_i, the multiplication using M_f is performed, see, $C = C \cdot C_i$, to be stored in the register $\langle C \rangle$. According to the structure of Fig. 2, computing each C_i needs $2r$ GNB multiplications (latency). Since the exponentiation A^P is represented by the product of r-term C_i for $0 \le i \le r-1$, the proposed 2-D multiplier array structure for computing exponentiation requires the total number of $3r$ multiplication delays as its latency.

A. Application in Double-Exponentiation

Let A and B be two field elements and P and Q be two positive integers. Then, the computation of the form $A^P B^Q$ is called "double-exponentiation." Double-exponentiation is widely applied in Schnorr- and ElGamal-like signature verifications [11], [12]. From Algorithm 1, the multiplexer-based double-exponentiation using normal basis representation has been firstly proposed in [15]. In [24], a new hybrid GNB multiplier to improve the performance of double-exponentiation is proposed. According to (1), two proposed exponentiation architectures can be extended to implement double-exponentiation. The space complexity of double-exponentiation architectures is presented through the following proposition.

Proposition 2: The 2-D multiplier array architecture for computing the double-exponentiation can be constructed by $\lceil \sqrt[3]{m^2} \rceil + 1$ multipliers, $m \lceil \sqrt[3]{m^2} \rceil$ $\text{MUX}_{4\times1}$ components. In addition, computing double-exponentiation requires $3 \lceil \sqrt[3]{m} \rceil$ multiplication delays (latency) using 2-D multiplier array architectures.

Table I lists the comparison of various double-exponentiation architectures. In this table, it is shown that our proposed 2-D multiplier array architectures for computing the double-exponentiation have lower multiplication delays (latencies) and areas in comparison with the counterparts presented. Based on this table and the discussions in

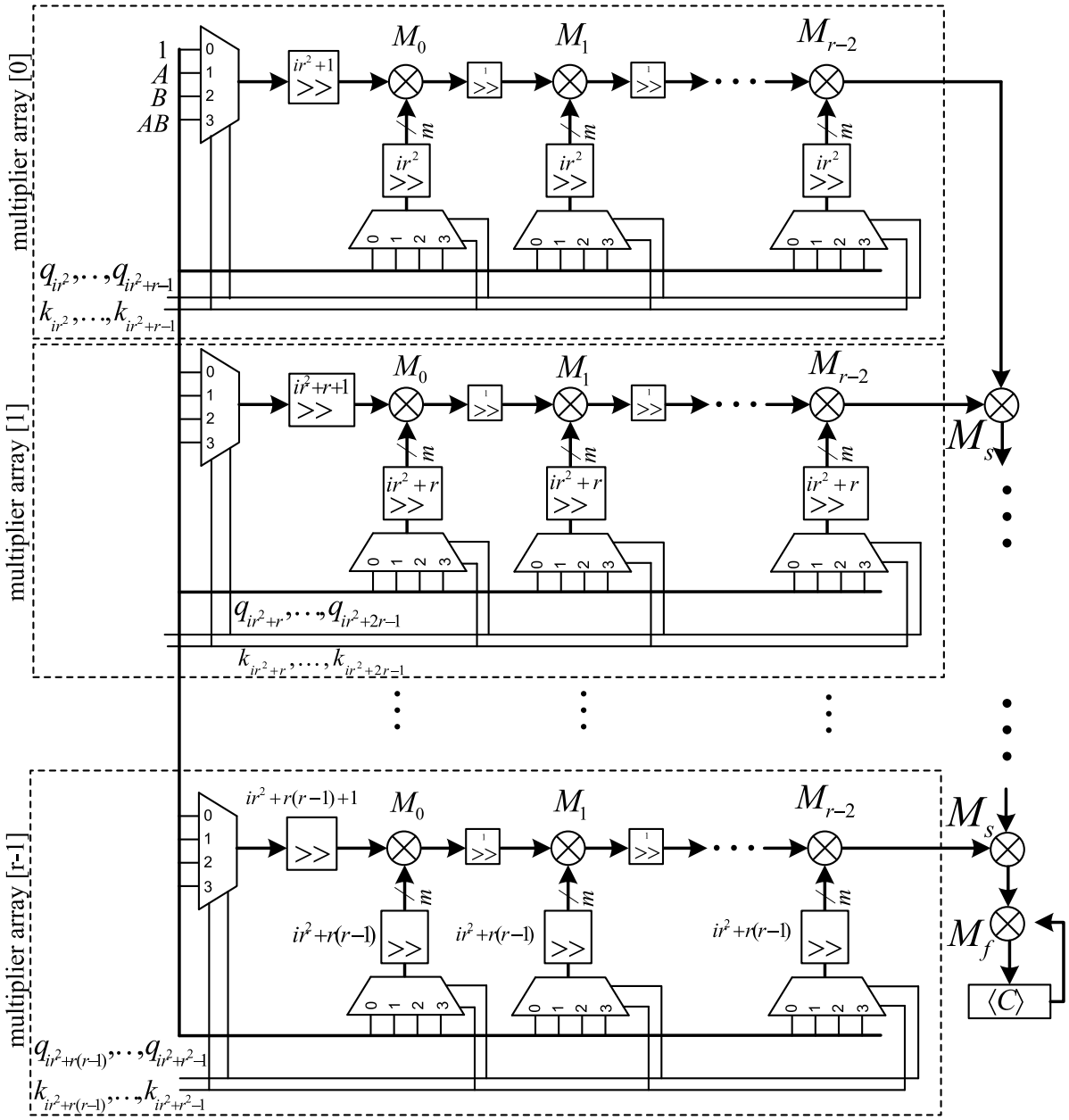

Fig. 2. 2-D multiplier array architecture for computing the exponentiation over $GF(2^m)$.

TABLE I
COMPARISON OF VARIOUS SINGLE- AND DOUBLE-EXPONENTIATION ARCHITECTURES OVER $GF(2^m)$

Architecture	Basis	Latency (# of multipliers delays)	Multiplier type	Area (# of multipliers)
Single-exponentiation				
Wang-Pei [14]	NB	m	MO	1
A-RM [24]	GNB	$\lceil \frac{m}{4} \rceil$	Hybrid	1
Chiou-Lee [15]	NB	$\lceil \frac{m}{2} \rceil$	Systolic	$\lceil \frac{m}{2} \rceil$
Fig. 1	GNB	$3\left\lceil \sqrt[3]{\frac{m}{2}} \right\rceil$	2-D systolic	$\left\lceil \sqrt[3]{\left(\frac{m}{2}\right)^2} \right\rceil + 1$
Double-exponentiation				
Chiou-Lee [15]	NB	m	Systolic	m
A-RM [24]	GNB	$\lceil \frac{m}{2} \rceil$	Hybrid	1
Fig. 1	GNB	$3\left\lceil \sqrt[3]{m} \right\rceil$	2-D systolic	$\left\lceil \sqrt[3]{m^2} \right\rceil + 1$

this section, we note that our proposed double-exponentiation architectures are based on the developed GNB multipliers and they outperform the traditional exponentiation schemes.

IV. ASIC SYNTHESIS AND BENCHMARK

To derive the performance and implementation metrics of the proposed architecture for digit-serial systolic multipliers,

TABLE II
ASIC (65-nm CMOS LIBRARY) SYNTHESIS RESULTS FOR THE PREVIOUS AND THE PRESENTED MULTIPLIERS OVER $GF(2^{409})$

Multiplier	Digit-size	Latency	Area [μm^2]	CPD [ns]	Total time (CPD$\times$ latency)
Proposed 2-D systolic architecture	13	12	241,209	1.53	18.4
	16	*9*	*259,006*	*1.57*	*14.1*
DL-PIPO [18], [17] (optimized in [19])	13	32	91,312	1.50	48.0
Digit-serial systolic [25]	13	64	50,917	1.34	85.8

Note 1. The numbers in Italic typeface for digit-size 16 are presented to show that the total time decreases ($\sim$23%) at the expense of slight increase in area ($\sim$7%) compared to digit-size 13.
Note 2. Although the merit of the proposed architectures is to achieve high-performance structures, the time$\times$area (efficiency) of the proposed work is also better than [18], [17] (optimized in [19]) and [25].

we have implemented them on ASIC platform. We have used Taiwan Semiconductor Manufacturing Company 65-nm standard-cell library for the ASIC results. We have implemented our proposed 2-D architectures for two representative digit sizes ($m = 409$ and $T = 4$) and the results are reported in Table II. This field size is chosen to support the security level requirements for the cryptographic computations.

We note that the proposed 2-D systolic GNB multiplier needs the maximum latency of $3\lceil\sqrt[3]{m/d}\rceil$ clock cycles with the number of PEs selected by $\lceil\sqrt[3]{(m/d)}\rceil^2$. Benchmark through hardware platforms is essential in determining the effectiveness of the devised approaches. As seen in this table, at the expense of higher area, the proposed architectures achieve better total time of computation (18.4 and 14.1 ns) compared to the other structures. These result in higher performance for the proposed systolic structures using the GNB.

When the concern is more toward performance and not area, for instance, in server-side security, using the presented approaches, we can fulfill the requirement. We target high-performance applications that require fastest computation results at the cost of employing more silicon area. This high performance is not achieved by the previous works. As seen in Table II, using the proposed 2-D architecture, we achieve higher performance compared to the previously presented counterparts. For instance, in comparison to the work presented in [19], our timing results are about 2.6 times faster for the digit-size 13. We should note that choosing larger digit sizes for the architecture presented in [19] is not efficient and makes the routing more difficult and degrades the maximum operating clock frequency. We would also like to note that systolic structures intend to boost the performance of the architectures, as seen in the table. One main advantage of the proposed architectures is that they give the architects the ability of selecting different digit sizes based on the requirements of the designs and the performance objectives to achieve. Finally, compared to the research work presented in [26], for the field size 409 and digit-size 13, the proposed multipliers achieve the maximum latency of 12 while the aforementioned digit-level systolic design has the maximum latency of 32 (we emphasize that this is the maximum latency obtained and depending on the digit-size, lower latencies can be achieved for both architectures). For comparing the total times of [26] and the presented work, we derive the corresponding values through multiplying the latencies and the critical path delays; based on which, the ratio is 4 which confirms the performance of the proposed work. In addition, compared to the super-systolic architecture of [26], the presented work achieves lower latency. It should be noted that the digit-serial systolic multiplier of [25] is a fully systolic architecture and comparison in terms of the digit sizes might not be fair. However, we compare the area-time results which gives better comparison of our presented 2-D systolic architecture.

V. CONCLUSION

In this paper, a high-performance GNB 2-D digit-level systolic multiplication and exponentiation have been presented. The derived systolic architectures are new and their major advantages are their low-latency and high-performance implementations. Specifically, based on our proposed multiplication schemes, we have derived the 2-D multiplier array architectures for computing exponentiations which can achieve low-multiplication delays (latencies) as compared to the existing exponentiation architectures. The proposed architectures are also suitable for implementing double-exponentiation. Finally, although the merit of the proposed architectures is to achieve high-performance structures, the time$\times$area (efficiency) of the proposed architectures are higher than some of the previous works as well.

ACKNOWLEDGMENT

The authors would like to thank the reviewers and the Associate Editor for their useful feedback and comments. The first author would like to thank B. Bahloul-Azarderakhsh for help and support.

REFERENCES

[1] J. Imana, R. Hermida, and F. Tirado, "Low complexity bit-parallel multipliers based on a class of irreducible pentanomials," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 12, pp. 1388–1393, Dec. 2006.

[2] M. Cenk, C. Nègre, and M. A. Hasan, "Improved three-way split formulas for binary polynomial and Toeplitz matrix vector products," *IEEE Trans. Comput.*, vol. 62, no. 7, pp. 1345–1361, Jul. 2013.

[3] M. Cenk, C. Nègre, and M. A. Hasan, "Improved three-way split formulas for binary polynomial multiplication," in *Proc. Conf. Sel. Areas Cryptography*, Toronto, ON, Canada, 2011, pp. 384–398.

[4] K. Järvinen and J. Skyttä, "On parallelization of high-speed processors for elliptic curve cryptography," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 16, no. 9, pp. 1162–1175, Sep. 2008.

[5] R. Azarderakhsh, M. Mozaffari-Kermani, S. Bayat-Sarmadi, and C.-Y. Lee, "Systolic Gaussian normal basis multiplier architectures suitable for high-performance applications," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, to be published.

[6] R. Azarderakhsh, M. Mozaffari-Kermani, and K. Järvinen, "Secure and efficient architectures for single exponentiation in finite field suitable for high-performance cryptographic applications," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 3, pp. 332–340, Mar. 2015.
[7] K. Järvinen and J. Skyttä, "Fast point multiplication on Koblitz curves: Parallelization method and implementations," *Microprocessors Microsyst.*, vol. 33, no. 2, pp. 106–116, Mar. 2009.
[8] R. Azarderakhsh and A. Reyhani-Masoleh, "Efficient FPGA implementation of point multiplication on binary Edwards and generalized Hessian curves using Gaussian normal basis," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 8, pp. 1453–1466, Aug. 2012.
[9] I. S. Reed and G. Solmon, "Polynomial codes over certain finite fields," *SIAM J. Appl. Math.*, vol. 8, pp. 300–304, Jun. 1960.
[10] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976.
[11] C.-P. Schnorr, "Efficient signature generation by smart cards," *J. Cryptol.*, vol. 4, no. 3, pp. 161–174, 1991.
[12] T. E. Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. 31, no. 4, pp. 469–472, Jul. 1985.
[13] *National Institute of Standards and Technology*, DSS Standard FIPS 186-2, Jan. 2000.
[14] C. Wang and D. Pei, "A VLSI design for computing exponentiations in $GF(2^m)$ and its application to generate pseudorandom number sequences," *IEEE Trans. Comput.*, vol. 39, no. 2, pp. 258–262, Feb. 1990.
[15] C. W. Chiou and C.-Y. Lee, "Multiplexer-based double-exponentiation for normal basis of $GF(2^m)$," *Comput. Secur.*, vol. 24, pp. 83–86, Feb. 2005.
[16] P. K. Meher, "Systolic and non-systolic scalable modular designs of finite field multipliers for Reed–Solomon codec," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 17, no. 6, pp. 747–757, Jun. 2009.
[17] S. Kwon, K. Gaj, C. H. Kim, and C. P. Hong, "Efficient linear array for multiplication in $GF(2^m)$ using a normal basis for elliptic curve cryptography," in *Proc. Workshop Cryptograph. Hardw. Embedded Syst.*, Cambridge, MA, USA, 2004, pp. 76–91.
[18] A. Reyhani-Masoleh, "Efficient algorithms and architectures for field multiplication using Gaussian normal bases," *IEEE Trans. Comput.*, vol. 55, no. 1, pp. 34–47, Jan. 2006.
[19] R. Azarderakhsh and A. Reyhani-Masoleh, "A modified low complexity digit-level Gaussian normal basis multiplier," in *Proc. Int. Workshop Arithmet. Finite Fields*, Istanbul, Turkey, 2010, pp. 25–40.
[20] C. W. Chiou, C.-C. Chang, C.-Y. Lee, T.-W. Hou, and J.-M. Lin, "Concurrent error detection and correction in Gaussian normal basis multiplier over $GF(2^m)$," *IEEE Trans. Comput.*, vol. 58, no. 6, pp. 851–857, Jun. 2009.
[21] Z. Wang and S. Fan, "Efficient Montgomery-based semi-systolic multiplier for even-type GNB of $GF(2^m)$," *IEEE Trans. Comput.*, vol. 61, no. 3, pp. 415–419, Mar. 2012.
[22] D. W. Ash, I. F. Blake, and S. A. Vanstone, "Low complexity normal bases," *Discrete Appl. Math.*, vol. 25, no. 3, pp. 191–210, 1989.
[23] A. Menezes *et al.*, *Applications of Finite Fields*. Boston, MA, USA: Kluwer Academic, 1993.
[24] R. Azarderakhsh and A. Reyhani-Masoleh, "A low complexity hybrid architecture for double-multiplication using Gaussian normal basis," *IEEE Trans. Comput.*, vol. 62, no. 4, pp. 744–757, Apr. 2013.
[25] 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.
[26] P. K. Meher, "Systolic and super-systolic multipliers for finite field $GF(2^m)$ based on irreducible trinomials," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 5, pp. 1031–1040, May 2008.

Reza Azarderakhsh (M'12) received the B.Sc. degree in electrical and electronic engineering and the M.Sc. degree in computer engineering from the Sharif University of Technology, Tehran, Iran, in 2002 and 2005, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Western Ontario, London, ON, Canada, in 2011.

He joined the Department of Electrical and Computer Engineering, University of Western Ontario, as a Limited Duties Instructor, in 2011. He was an Natural Sciences and Engineering Research Council of Canada (NSERC) Post-Doctoral Research Fellow with the Center for Applied Cryptographic Research and the Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada. He is currently a Faculty Member with the Department of Computer Engineering, Rochester Institute of Technology, Rochester, NY, USA. His current research interests include finite field and its application, elliptic curve cryptography, and pairing-based cryptography.

Dr. Azarderakhsh was a recipient of the NSERC Post-Doctoral Research Fellowship in 2011. He serves as the Guest Editor of the IEEE TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS for the Special Issue of Emerging Security Trends for Biomedical Computations, Devices, and Infrastructures in 2015 and 2016.

Mehran Mozaffari-Kermani (M'11) received the B.Sc. degree in electrical and computer engineering from the University of Tehran, Tehran, Iran, in 2005, and the M.E.Sc. and Ph.D. degrees from the Department of Electrical and Computer Engineering, University of Western Ontario, London, ON, Canada, in 2007 and 2011, respectively.

He joined the Advanced Micro Devices, Markham, Ontario, Canada, as a Senior ASIC/Layout Designer, integrating sophisticated security/cryptographic capabilities into a single accelerated processing unit. In 2012, he joined the Department of Electrical Engineering, Princeton University, Princeton, NJ, USA, as a Natural Sciences and Engineering Research Council of Canada (NSERC) Post-Doctoral Research Fellow. He is currently with the Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology (RIT), Rochester, NY, USA. He has been recognized as the Featured Faculty Member in research with the School of Engineering, RIT in 2014. His current research interests include emerging security/privacy measures for deeply embedded systems, cryptographic hardware systems, fault diagnosis and tolerance in cryptographic hardware, very large-scale integration reliability, and low-power secure and efficient FPGA and ASIC designs.

Dr. Mozaffari-Kermani was a recipient of the NSERC Post-Doctoral Research Fellowship in 2011 and the Texas Instruments Faculty Award (Douglas Harvey) in 2014. He currently serves as an Associate Editor for *ACM Transactions on Embedded Computing Systems* and the Lead Guest Editor for the IEEE TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS for the Special Issue of Emerging Security Trends for Biomedical Computations, Devices, and Infrastructures in 2015 and 2016. He has served as the Lead Guest Editor for the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING for the Special Issue of Emerging Security Trends for Deeply-Embedded Computing Systems in 2014 and 2015. He is currently a Technical Committee Member for a number of related conferences including International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, Workshop on Fault Diagnosis and Tolerance in Cryptography, Workshop on Security and Privacy in Radio Frequency Identification, Workshop on Lightweight Security, and International Workshop on the Arithmetic of Finite Fields.