New Bit-Level Serial $GF\left(2^m\right)$ Multiplication Using Polynomial Basis

Hayssam El-Razouk and Arash Reyhani-Masoleh
Department of Electrical and Computer Engineering
Western University
London, Canada
Email: helrazou@uwo.ca, areyhani@uwo.ca

***Abstract*—The Polynomial basis (PB) representation offers efficient hardware realizations of $GF\left(2^m\right)$ multipliers. Bit-level serial multiplication over $GF\left(2^m\right)$ trades-off the computational latency for lower silicon area, and hence, is favored in resource constrained applications. In such area critical applications, extra clock cycles might take place to read the inputs of the multiplication if the data-path has limited capacity. In this paper, we present a new bit-level serial PB multiplication scheme which generates its output bits in parallel after m clock cycles without requiring any preloading of the inputs, for the first time in the open literature. The proposed architecture, referred to as fully-serial-in-parallel-out (FSIPO), is useful for achieving higher throughput in resource constrained environments if the data-path for entering inputs has limited capacity, especially, for large dimensions of the field $GF\left(2^m\right)$.**

***Index Terms*—Bit-Level Multipliers, Finite Fields, Polynomial Basis, Serial Multiplication.**

I. Introduction

Many of today's state of the art digital systems utilize finite field arithmetic in applications such as error control coding, pseudo random number generation, digital signal processing, and cryptography [1], [2]. Therefore, improving the efficiency of finite field operations is very important for improving the performance of these applications.

In this perspective, binary extension fields, denoted as $GF\left(2^m\right)$, have received a great deal of attention due to their efficient hardware implementations. A $GF\left(2^m\right)$ have 2^m elements, where every element is represented by a unique set of m binary coordinates (0 or 1) with respect to (w.r.t) a basis. The basis is constructed from a set of m linearly independent field elements [1]. Polynomial basis (PB) and normal basis (NB) are widely used representations for field elements. In these representations, field addition of two elements is accomplished by simple bit-wise addition (Exclusive OR, i.e XOR) of the corresponding coordinates. On the other hand, field multiplication is more complicated than addition, and its complexity directly depends on the underlying field representation. In addition, field multiplication is the base for other more involved operations such as field exponentiation, division, and inversion. These operations are implemented as iterations of multiplications, and are used extensively in symmetric key cryptographic algorithms (for example [3], [4]) and asymmetric key cryptographic algorithms (for example the Diffie-Hellman key exchange algorithm [5] and Elliptic curve digital signature algorithm (ECDSA) [6]). Hence, the efficiency of the field multiplier affects the efficiency of upper system layers, and therefore, researchers are continuously striving to find efficient hardware implementations of $GF\left(2^m\right)$ multipliers. To this end, it is known that, polynomial basis representation offers low complexity field multiplications compared to other field representations [7]. This makes PB multipliers suitable for resource constrained applications, and hence, it is the focus of this work.

By finding a monic irreducible polynomial $p\left(x\right)=x^m+\sum_{i=1}^{\omega-2}x^{t_i}+1$ of degree m over $GF\left(2\right)$ with ω nonzero terms, a polynomial basis $\left\{\alpha^{m-1},\ldots,\alpha,1\right\}$ is constructed for the field $GF\left(2^m\right)$, where α is a root of $p\left(x\right)$ ($p\left(\alpha\right)=0$). An arbitrary $GF\left(2^m\right)$ element A is then represented as $A=\sum_{i=0}^{m-1}a_i\alpha^i=\left(a_{m-1},\ldots,a_0\right)$ w.r.t the PB, where $a_i\in GF\left(2\right)$, $0\leq i<m$, are the coordinates of A.

Two popular schemes for the multiplication of two $GF\left(2^m\right)$ elements in the PB representation are: the two-step classic multiplication scheme and the Matrix-vector scheme [2]. The first scheme starts by performing polynomial multiplication of the two input field elements, then, the result is reduced modulo the irreducible defining field polynomial [8]. In the second scheme, known as the Mastrovito multiplier [9], [10], [11], [12], one performs the field multiplication in terms of vector by matrix multiplication, in which both steps of the former scheme are combined into a single step.

In general, the different designs of PB multiplication fall under one of the two categories of parallel and serial computations. For achieving high throughput, parallel implementation is used where all output bits of the multiplication are generated in a single clock cycle [13], [9], [10], [11], [12], [14], [15]. For achieving low space complexity, bit-level serial computations are considered. In bit-level serial multiplication schemes, space complexity is reduced at the expense of increasing the number of clock cycles required for generating the m output bits (computational latency) to m clock cycles (in general) [16], [17], [18], [19], [20], [21], [22]. The proposed architecture in this work targets resource constrained applications, and hence, is for a bit-level PB multiplier.

In a bit-level serial implementation, the multiplication input/output bits are entered/generated either in parallel, or serially in the order of one bit per a clock cycle. For example, the authors of [17] present architectures for bit-level serial-in-

1063-6889/15 $31.00 © 2015 IEEE
DOI 10.1109/ARITH.2015.11

parallel-out (BL-SIPO) PB multipliers that generate the output bits in parallel after m clock cycles, where one input is loaded in parallel (in advance to computations) and the other input enters serially one bit per a clock cycle during computations, in either a most-significant-bit first (MSB) or least-significant-bit first (LSB) order. In 2008, the authors of [19] proposed a bit-level parallel-in-serial-out (BL-PISO) PB multiplier, where the two inputs are preloaded in parallel in advance to computations, after which the output bits are generated over m clock cycles, one bit per a clock cycle. In 2011, the authors of [21] proposed a parallel-in-parallel-out (PIPO) PB multiplication architecture, which requires preloading both inputs in advance to computations. The PIPO multiplier in [21] has a latency of $2t_{\omega-2}+1$ clock cycles to generate the m output bits in parallel, where $t_{\omega-2}$ denotes the second highest nonzero term of the field irreducible polynomial. The authors of [21] show that the proposed serial PIPO PB multiplier offers the lowest latency for the cases where $m \geq 2t_{\omega-2}-1$, however, the corresponding space complexity is quadratic in m.

As we mentioned above, the SIPO, PISO, and PIPO architectures require preloading of one or both inputs in advance to computations. For resource constrained applications, the preloading of inputs might take place serially due to limited capacity of the input data-paths. This introduces additional clock cycles to the total latency (loading + computation). In this work, we address this issue through the construction of a bit-level serial PB multiplier which reads both inputs serially, in an MSB order, while the computations are being conducted.

It is noted that, serial finite field multipliers with two serially-entered inputs have already been proposed. For example, the multiplier proposed by Feng [23] in 1989 is a bit-level serial structure for the NB representation which reads both of its inputs serially, one bit per a clock cycle, and generates its output bits in parallel after the m cycles. For convenience of reference, we denote Feng's multiplication scheme as a fully-serial-in-parallel-out (FSIPO). In 1992, the authors of [18] presented a bit-level MSB serial-serial PB multiplier which generates the m output bits serially over $2m$ clock cycles. In [18], the inputs to the multiplier enter serially bit-by-bit, starting with the MSB, over the first m clock cycles. After reading the serial inputs, the m output bits are then generated serially, one bit per a clock cycle, starting with the MSB. In 2009, the authors of [24] proposed a generic serial-serial multiplication/reduction architecture for $GF(q)$, where q can be a prime p, a power of a prime p^m, and where it is possible to have $p=2$. For the case of $GF(2^m)$, which is the focus of this paper, the serial-serial multiplication/reduction scheme proposed in [24] reads both of its multiplication inputs serially, one bit at a time, in either least or most-significant first order. The multiplication result is generated bit-by-bit, starting with the $(m+1)$-th clock cycle (an additional correction step is required in case of the least-significant first input order). Then, using the scheme in [24], all the m output bits are produced serially, after a total of $2m$ clock cycles, without using additional parallel-in-serial-out output register. It is noted that, the serial-serial multiplier in [24] is not a dedicated multiplication scheme in the sense that it works for any irreducible polynomial by reading it as one of its inputs. In order to allow for scalability, and to make the field multiplication generic, the serial-serial multiplier in [24] requires additional multiplexers and storage Flip-Flops (FF), in addition to a number of control signals.

On the other hand, using Feng's structure, one can generate the output bits serially over an additional m clock cycles, by using an independent parallel-in-serial-out output register. This is advantageous over the serial-serial schemes in [18], [24] for performing n consecutive multiplications. It is noted that, the serial-serial schemes presented in [18], [24] require $2mn$ clock cycles to complete n consecutive $GF(2^m)$ bit-level multiplications. The same number of n consecutive bit-level multiplications can be run using only $m(n+1)$ clock cycles based on the modified Feng's scheme (with an additional parallel-in-serial-out output register).

In this work, and to the best of our knowledge, we propose, for the first time in literature, a new architecture for $GF(2^m)$ bit-level FSIPO (BL-FSIPO) multiplier for dedicated PBs. Similar to Feng's bit-level serial NB multiplier, the new bit-level serial PB architecture generates the output bits in parallel after m iterations, while both inputs enter the multiplier bit-by-bit serially, one bit per a clock cycle starting from the most significant bit (MSB), as the computations are carried out. Therefore, the new MSB BL-FSIPO PB multiplier is expected to offer higher throughput in applications where parallel loading of inputs is not feasible due to limited inputs data-path capacity, specially, when the value of m is large.

The rest of the paper is organized as follows. In Section II, we present the proposed MSB BL-FSIPO PB multiplication scheme. Section III, presents comparisons between the proposed MSB BL-FSIPO PB multiplication scheme and the other existing counterparts. Section IV concludes the paper.

II. Proposed MSB BL-FSIPO PB Multiplier

This section presents the proposed MSB BL-FSIPO PB multiplier. The proposed architecture conducts the multiplication operation as the bits of the two inputs enter the multiplier in a bit-by-bit order, one bit per a clock cycle (for each input), starting from the most significant bit. Therefore, the proposed MSB BL-FSIPO PB multiplier generates the output bits in parallel after m clock cycles and does not require any preloading of the inputs. This is advantageous for achieving high output bit rates for applications where m is large and the parallel preloading of the inputs is not possible due to the limited sizes of the input data-paths. To the best of our knowledge, the presented PB multiplier architecture is proposed for the first time in literature. It is noted that, one can find similar formulations for an LSB construction of the FSIPO PB multiplier (we are currently working on it, in addition to digit-level extensions).

In the following, we first derive the required formulations. Then, we show the proposed architecture for the MSB BL-FSIPO PB multiplier. We conclude this section by studying the space and time complexities.

A. Formulations

In this section, we derive required formulations for the proposed MSB BL-FSIPO PB multiplication scheme. First, we define recursive bit-level construction of $GF(2^m)$ elements when represented in PB, by reading the field element bit-by-bit, starting from the most significant bit, as follows.

Lemma 1. *An arbitrary field element $A = (a_{m-1}, \ldots, a_0) \in GF(2^m)$ represented in the PB, can be constructed recursively, starting from the most significant bit a_{m-1}, as follows:*

$$A^{(i)} = a_{m-1-i} + A^{(i-1)}\alpha \tag{1}$$

where $i = 0, \ldots, m-1$ and $A = A^{(m-1)}$, given that $A^{(-1)} = 0$ with α being the root of the field's defining irreducible polynomial.

Proof. By using (1), for $i = 0, 1, \ldots, m-2$ we get

$$\begin{aligned}
A^{(0)} &= a_{m-1} + A^{(-1)}\alpha = a_{m-1},\\
A^{(1)} &= a_{m-2} + A^{(0)}\alpha = a_{m-2} + (a_{m-1})\,\alpha,\\
&\vdots\\
A^{(m-2)} &= a_1 + A^{(m-3)}\alpha\\
&= a_1 + (a_2 + \cdots (a_{m-2} + (a_{m-1})\,\alpha)\,\alpha \cdots)\,\alpha,
\end{aligned}$$

and hence, for $i = m-1$ we have

$$\begin{aligned}
A^{(m-1)} &= a_0 + A^{(m-2)}\alpha\\
&= a_0 + (a_1 + \cdots (a_{m-2} + (a_{m-1})\,\alpha)\,\alpha \cdots)\,\alpha,
\end{aligned}$$

that is $A^{(m-1)} = \sum_{i=0}^{m-1} a_i \alpha^i$, which completes the proof. □

Notice that the multiplication by α in (1) realizes a 1-bit left shift and does not require any reduction for $0 \le i < m$. Based on the recursive construction in (1), one obtains the multiplication of any two arbitrary $GF(2^m)$ elements A and B as follows.

Proposition 1. *Let A and B be two arbitrary $GF(2^m)$ elements represented in the PB generated by the degree m irreducible polynomial $p(x) = x^m + \sum_{i=1}^{\omega-2} x^{t_i} + 1$ with ω nonzero terms. Let us define $C_i = A^{(i)}B^{(i)}$ mod $p(\alpha)$, where $A^{(i)}$ and $B^{(i)}$ are given in* (1) *and α is the root of $p(x)$. Then, based on the following recurrence on C_i, one can compute the multiplication of A and B, as $AB = C_{m-1}$:*

$$\begin{aligned}
C_i = {} & a_{m-1-i}b_{m-1-i} + C_{i-1}\alpha^2 \bmod p(\alpha) +\\
& \left(a_{m-1-i}B^{(i-1)} + b_{m-1-i}A^{(i-1)}\right)\alpha,
\end{aligned} \tag{2}$$

$i = 0, \ldots, m-1$, where $C_{-1} = A^{(-1)}B^{(-1)}$ mod $p(\alpha) = 0$.

Proof. By using the definition (1) for $A^{(i)}$ and $B^{(i)}$ in evaluating $C_i = A^{(i)}B^{(i)} \bmod p(\alpha)$, one obtains (2) as follows

$$\begin{aligned}
C_i &= \left(a_{m-1-i} + A^{(i-1)}\alpha\right)\left(b_{m-1-i} + B^{(i-1)}\alpha\right) \bmod p(\alpha)\\
&= a_{m-1-i}b_{m-1-i} + C_{i-1}\alpha^2 \bmod p(\alpha) +\\
&\quad \left(a_{m-1-i}B^{(i-1)} + b_{m-1-i}A^{(i-1)}\right)\alpha.
\end{aligned}$$

Notice that, the superscripts of the terms $A^{(i-1)}$ and $B^{(i-1)}$ in the right hand side are always less than $(m-1)$ for $0 \le i < m$. Hence, the intermediate variable elements $A^{(0)}$ to $A^{(m-2)}$ and $B^{(0)}$ to $B^{(m-2)}$ require a maximum of $m-1$ bits for their PB representations. Therefore, the multiplication by α which appears in the expression $\left(a_{m-1-i}B^{(i-1)} + b_{m-1-i}A^{(i-1)}\right)\alpha$ can be accomplished by a simple left shift without any reduction.

Having $AB = A^{(m-1)}B^{(m-1)} \bmod p(\alpha) = C_{m-1}$, then, by iterating for $i = 0, 1, \ldots, m-1$, one obtains the multiplication results after m iterations over (1) and (2). □

Based on (2), the multiplication of the two $GF(2^m)$ elements A and B, is reduced recursively to bit-wise AND operations, field additions, left shifts (for the multiplication by α), and a multiplication with the fixed field element α^2.

The following is an example for illustrating our proposed multiplication scheme.

Example 1. Table I lists the steps for multiplying the two $GF(2^3)$ field elements $A = \alpha = (0,1,0)$ and $B = \alpha^2 = (1,0,0)$, represented in the PB $\{\alpha^2, \alpha, 1\}$ which is defined by the irreducible trinomial $p(x) = x^3 + x + 1$.

For the realization of the multiplication of an arbitrary field element by the constant field element α^2, we have the following lemma.

Lemma 2. *The multiplication of an arbitrary $GF(2^m)$ element $A = \sum_{i=0}^{m-1} a_i\alpha^i$ by the fixed field element α^2, where α is the root of the field polynomial $p(x) = x^m + \sum_{i=1}^{\omega-2} x^{t_i} + 1$ with ω nonzero terms, is given by*
$A\alpha^2$ mod $p(\alpha) =$

$$\begin{cases}
a_{m-1}\sum_{i=1}^{\omega-2}\alpha^{t_i+1} + a_{m-2}\sum_{i=1}^{\omega-2}\alpha^{t_i} + & \\
\sum_{i=2}^{m-1} a_{i-2}\alpha^i + a_{m-1}\alpha + a_{m-2}, & t_{\omega-2} \neq m-1;\\
& \\
a'\alpha^{m-1} + a_{m-1}\sum_{i=1}^{\omega-3}\alpha^{t_i+1} + & \\
a''\sum_{i=1}^{\omega-3}\alpha^{t_i} + \sum_{i=2}^{m-2} a_{i-2}\alpha^i + & \\
a_{m-1}\alpha + a'', & t_{\omega-2} = m-1;
\end{cases} \tag{3}$$

where $a' = (a_{m-1} + a_{m-2} + a_{m-3})$ and $a'' = (a_{m-1} + a_{m-2})$.

Proof. Since $p(\alpha) = \alpha^m + \sum_{i=1}^{\omega-2}\alpha^{t_i} + 1 = 0$, then

$$\alpha^m \bmod p(\alpha) = \sum_{i=1}^{\omega-2}\alpha^{t_i} + 1, \tag{4}$$

and
α^{m+1} mod $p(\alpha) =$

$$\begin{cases}
\sum_{i=1}^{\omega-2}\alpha^{t_i+1} + \alpha, & t_{\omega-2} \neq m-1;\\
& \\
\alpha^{m-1} + \sum_{i=1}^{\omega-3}\alpha^{t_i+1} + & \\
\sum_{i=1}^{\omega-3}\alpha^{t_i} + \alpha + 1, & t_{\omega-2} = m-1;
\end{cases} \tag{5}$$

TABLE I: Example 1 for multiplying the two $GF\left(2^3\right)$ elements $A=\alpha=(0,1,0)$ and $B=\alpha^2=(1,0,0)$ using (1) and (2).

i	a_{2-i}	b_{2-i}	$A^{(i-1)}$	$B^{(i-1)}$
0	$a_2=0$	$b_2=1$	$A^{(-1)}=0$	$B^{(-1)}=0$
1	$a_1=1$	$b_1=0$	$A^{(0)}=a_2+A^{(-1)}\alpha=0$	$B^{(0)}=b_2+B^{(-1)}\alpha=1$
2	$a_0=0$	$b_0=0$	$A^{(1)}=a_1+A^{(0)}\alpha=1$	$B^{(1)}=b_1+B^{(0)}\alpha=\alpha$

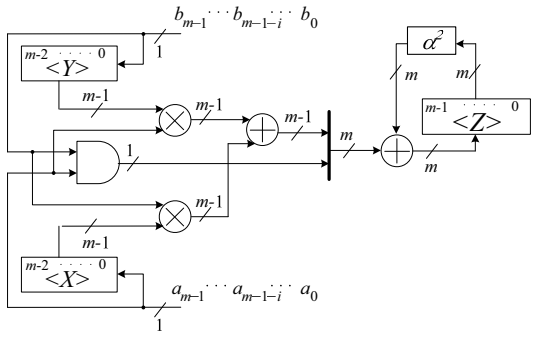
i	$a_{2-i}b_{2-i}$	$\left(a_{2-i}B^{(i-1)}+b_{2-i}A^{(i-1)}\right)\alpha$	$C_{i-1}\alpha^2 \bmod p(\alpha)$	C_i
0	$a_2b_2=0$	$\left(a_2B^{(-1)}+b_2A^{(-1)}\right)\alpha=0$	$C_{-1}\alpha^2 \bmod p(\alpha)=0$	$C_0=0$
1	$a_1b_1=0$	$\left(a_1B^{(0)}+b_1A^{(0)}\right)\alpha=\alpha$	$C_0\alpha^2 \bmod p(\alpha)=0$	$C_1=\alpha$
2	$a_0b_0=0$	$\left(a_0B^{(1)}+b_0A^{(1)}\right)\alpha=0$	$C_1\alpha^2 \bmod p(\alpha)=\alpha+1$	$C_2=\alpha+1=\alpha^3$

where last result (when $t_{\omega-2}=m-1$) is obtained by substituting for $\alpha^m \bmod p(\alpha)$ from (4) in $\alpha^{m+1}=\alpha^m\alpha \bmod p(\alpha)=\alpha^m \bmod p(\alpha)+\sum_{i=1}^{\omega-3}\alpha^{t_i+1}+\alpha$. Therefore, by substituting for $\alpha^m \bmod p(\alpha)$ and $\alpha^{m+1} \bmod p(\alpha)$ from (4) and (5), respectively, in $A\alpha^2 \bmod p(\alpha)$, which is equal to $\left(a_{m-1}\alpha^{m+1} \bmod p(\alpha)+a_{m-2}\alpha^m \bmod p(\alpha)+\sum_{i=2}^{m-1}a_{i-2}\alpha^i\right)$, one obtains (3). □

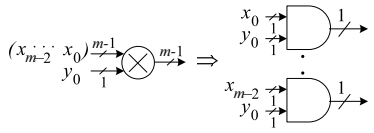
Although this work accounts for $t_{\omega-2}=m-1$, however, and except for $p(x)=x^2+x+1$, this value of $t_{\omega-2}$ is not common in practice for $m>2$ (for example, the five $GF(2^m)$ recommended by NIST for ECDSA [6]). Next, we present the architecture of proposed MSB BL-FSIPO PB multiplier.

B. Architecture

Here, we present the proposed architecture of our MSB BL-FSIPO PB multiplier, as shown in Figure 1a. The architecture



(a)



(b)

Fig. 1: (a) Architecture of the proposed MSB BL-FSIPO PB multiplier. (b) Architecture of $\otimes$ module.

in Figure 1a is designed based on the formulations (1) and (2). In this design, $\langle X\rangle$ and $\langle Y\rangle$ are left shift registers, which respectively store the bits of $A^{(i-1)}$ and $B^{(i-1)}$ (see (1)) during the i-th iteration, for $0\leq i<m$, where $A,B\in GF(2^m)$ represent the inputs to the multiplier. It is noted that, the coordinate of α^{m-1} is always zero in the PB representation of all the intermediate elements $A^{(0)}$ to $A^{(m-2)}$ and $B^{(0)}$ to $B^{(m-2)}$, according to (1). Then, it is sufficient to have only $(m-1)$-bits, in each of $\langle X\rangle$ and $\langle Y\rangle$. Moreover, during the i-th iteration, the vertical thick line in Figure 1a represents the concatenation of $a_{m-1-i}b_{m-1-i}$ with the 1-bit left shift of the $(m-1)$-bits of $(a_{m-1-i}\langle Y\rangle+b_{m-1-i}\langle X\rangle)$. This is done in order to compute the expression $a_{m-1-i}b_{m-1-i}+\left(a_{m-1-i}B^{(i-1)}+b_{m-1-i}A^{(i-1)}\right)\alpha$ in (2) (the multiplication by α is accomplished through the 1-bit left shift). In the same figure, the block which is denoted by α^2 represents the multiplication of the current state of register $\langle Z\rangle$ by the fixed field element α^2. Hence, by adding the result of this constant multiplication to the concatenated signal $a_{m-1-i}b_{m-1-i}+\left(a_{m-1-i}B^{(i-1)}+b_{m-1-i}A^{(i-1)}\right)\alpha$ (see Figure 1), one obtains $C_i=A^{(i)}B^{(i)} \bmod p(\alpha)$ in accumulator $\langle Z\rangle$, after the i-th clock signal, according to (2). Therefore, by initializing the three registers $\langle X\rangle$, $\langle Y\rangle$, and $\langle Z\rangle$ of Figure 1a, with zeros, we generate the result $C_{m-1}=AB=A^{(m-1)}B^{(m-1)} \bmod p(\alpha)$ in accumulator $\langle Z\rangle$ after m iterations.

As a graphical illustration of Example 1, Figure 2 presents the state of the corresponding $GF\left(2^3\right)$ MSB BL-FSIPO PB multiplier during the different iterations of computations (for multiplying the two field elements $A=\alpha=(0,1,0)$ and $B=\alpha^2=(1,0,0)$), based on the architecture which has been introduced in this section. It is noted that, in this figure, the underlined leftmost bit of $A^{(i-1)}$ and $B^{(i-1)}$, respectively, are always zero, which represent the missing (not required) leftmost FF in registers $\langle X\rangle$ and $\langle Y\rangle$.

In the following, we study the space and time complexities of the proposed MSB BL-FSIPO PB multiplier.

C. Space and Time Complexities

In this section, we start by deriving the space and time complexities for the multiplication of an arbitrary field element represented in the PB by the constant field element α^2, where $\alpha\in GF(2^m)$ is the root of the field's irreducible polynomial. After this, we give the space and time complexities of the proposed MSB BL-FSIPO PB multiplier.

Lemma 3. *Denote by T_X the propagation delay of a two-inputs XOR gate. Define the function $\delta(k)$, where $k\in\mathbb{Z}$ (the set of integers), as: $\delta(k)=1$ if $k=1$, otherwise $\delta(k)=0$. Let N_{α^2} and T_{α^2} represent the number of two-inputs XOR gates and the maximum propagation delay, respectively, which are*

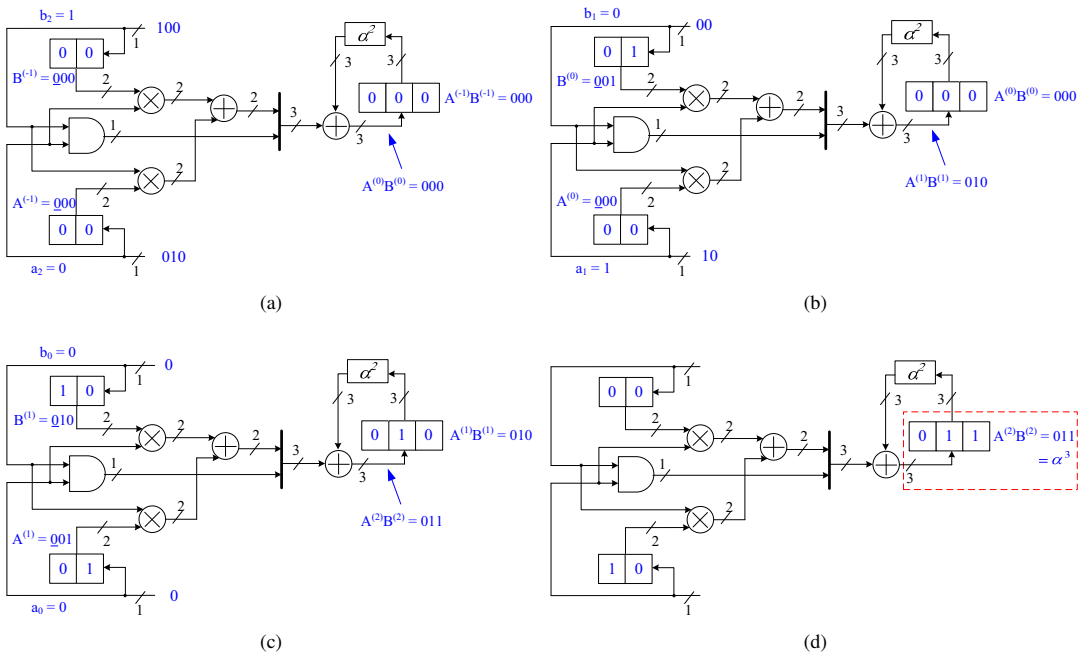


Fig. 2: State of the corresponding $GF\left(2^3\right)$ MSB BL-FSIPO PB multiplier for Example 1, throughout the different iterations of computation. (a) initial state. $i=0$. (b) state after first clock cycle. $i=1$. (c) state after second clock cycle. $i=2$. (d) state after third clock cycle, where the result $\alpha^3=\alpha+1$ is stored in the output register surrounded by the dotted rectangle.

required for the hardware realization of (3)*, then* $N_{\alpha^2}=$

$$\begin{cases} 2\left(\omega-2\right)-\sum_{j=1}^{\omega-2}\delta\left(t_j-t_{j-1}\right)+r, & t_{\omega-2}\neq m-1; \\ 2\left(\omega-2\right)-\sum_{j=1}^{\omega-3}\delta\left(t_j-t_{j-1}\right), & t_{\omega-2}=m-1; \end{cases} \tag{6}$$

$$T_{\alpha^2}=\begin{cases} \left(1+d\right)T_X, & t_{\omega-2}\neq m-1; \\ \left(2-\delta\left(t_{w-2}-t_{w-3}\right)\right)T_X, & t_{\omega-2}=m-1; \end{cases} \tag{7}$$

where $d=\left\lceil\frac{\sum_{j=2}^{\omega-2}\delta(t_j-t_{j-1})}{\omega}\right\rceil\in\{0,1\}$, $r=\left\lceil\frac{\sum_{j=1}^{\omega-2}\delta(t_j-t_{j-1})}{\omega}\right\rceil\in\{0,1\}$, $t_0=0$, *and* $p\left(x\right)=x^m+\sum_{i=1}^{\omega-2}x^{t_i}+1$ *is the field's irreducible polynomial.*

Proof. We prove (7) and (6), based on (3) as follows:

1- $t_{\omega-2}\neq m-1$:

The field additions of the two terms $a_{m-1}\sum_{i=1}^{\omega-2}\alpha^{t_i+1}$ and $a_{m-2}\sum_{i=1}^{\omega-2}\alpha^{t_i}$ to $\left(\sum_{i=2}^{m-1}a_{i-2}\alpha^i+a_{m-1}\alpha+a_{m-2}\right)$ result in one level of XORing (i.e. propagation delay of T_X) consisting of a total of $2\left(\omega-2\right)$ two-inputs XOR gates, if there is no j, $2\leq j\leq\omega-2$, where $t_j=t_{j-1}+1$. However, if there are some j, $2\leq j\leq\omega-2$, such that $t_j=t_{j-1}+1$, therefore, each such occurrence in $a_{m-1}\sum_{i=1}^{\omega-2}\alpha^{t_i+1}$ and $a_{m-2}\sum_{i=1}^{\omega-2}\alpha^{t_i}$ results in adding $\left(a_{m-1}+a_{m-2}\right)\alpha^{t_j}$ to $\left(\sum_{i=2}^{m-1}a_{i-2}\alpha^i+a_{m-1}\alpha+a_{m-2}\right)$, which results in a delay of $2T_X$. This has been compensated for by adding dT_X in (7). On the other hand, the expression $\left(a_{m-1}+a_{m-2}\right)$ is common for all coefficients of α^{t_j} where $t_j=t_{j-1}+1$, $1\leq j\leq\omega-2$. Hence, we account for this case in (6) by: a) adding r to count for the XOR gate which generates $\left(a_{m-1}+a_{m-2}\right)$. b) for each j, $1\leq j\leq\omega-2$, where $t_j=t_{j-1}+1$, subtract $\delta\left(t_j-t_{j-1}\right)=1$, as depicted by (6).

2- $t_{\omega-2}=m-1$:

It is noted that the generation of $a^{'}=\left(a_{m-1}+a_{m-2}+a_{m-3}\right)$ and $a^{"}=\left(a_{m-1}+a_{m-2}\right)$ in (6) requires 2 two-inputs XOR gates with a propagation delay of $2T_X$. In addition, the field additions of the two terms $a_{m-1}\sum_{i=1}^{\omega-3}\alpha^{t_i+1}$ and $a^{"}\sum_{i=1}^{\omega-3}\alpha^{t_i}$ to $\left(a^{'}\alpha^{m-1}+\sum_{i=2}^{m-2}a_{i-2}\alpha^i+a_{m-1}\alpha+a^{"}\right)$, result in one level of XORing (propagation delay of T_X) consisting of a total of $2\left(\omega-3\right)$ two-inputs XOR gates if $t_{\omega-3}<m-2$ and $t_1>1$, and there is no j, $2\leq j\leq\omega-3$, such that $t_j=t_{j-1}+1$. Therefore, in this case there is a total of $2\left(\omega-3\right)+2=2\left(\omega-2\right)$ XORs with a propagation delay of $2T_X=\max\left\{2T_X,T_X\right\}$.

However: a) if $t_{\omega-3}=m-2$, then the coefficient of α^{m-1} becomes $a^{'}+a_{m-1}=a_{m-2}+a_{m-3}$, which requires only a delay of T_X (hence, subtracting $\delta\left(t_{w-2}-t_{w-3}\right)$ for this case in (7)) and one less XOR gate compared to $a^{'}$. Notice that, in this case we still require another XOR gate to generate $a^{"}$. b) if there are some j, $1\leq j\leq\omega-3$, such that $t_j=t_{j-1}+1$, then, each such occurrence in $a_{m-1}\sum_{i=1}^{\omega-3}\alpha^{t_i+1}$, $a^{"}\sum_{i=1}^{\omega-3}\alpha^{t_i}$, and

$a_{m-1}\alpha$ results in the coefficient a_{m-2} due to $a_{m-1}\alpha^{t_{j-1}+1} + a^{"}\alpha^{t_j} = a_{m-2}\alpha^{t_j}$, where adding $a_{m-2}\alpha^{t_j}$ to $\sum_{i=2}^{m-2} a_{i-2}\alpha^i$ requires only one XOR. This has been accounted for in (6) by subtracting $\delta(t_j - t_{j-1}) = 1$ if $t_j = t_{j-1} + 1$. □

The space complexity of proposed MSB BL-FSIPO PB multiplier (Figure 1a) is as follows.

Proposition 2. *The total number of gates in the proposed MSB BL-FSIPO PB multiplier of Figure 1 is as follows:*

$$\begin{cases} \#ANDs = 2m-1, & \#FFs = 3m-2, \\ \#XORs = 2m + N_{\alpha^2} - 1, \end{cases} \tag{8}$$

where N_{α^2} is given in (6)*.*

Proof. The total number of two-inputs AND gates which is required for the hardware realization of the proposed architecture in Figure 1a equals to $(m-1)+(m-1)+1 = 2m-1$. Similarly, and from the same figure, one finds the total number of FF to be $(m-1)+(m-1)+m = 3m-2$. For the total number of two-inputs XOR gates, it consists of the XOR gates of the $(m-1)$-bit, and the m-bit, bitwise modulo-2 additions, in addition to the XOR gates which form the multiplication by the constant α^2, i.e., N_{α^2}. Therefore, the total number of two-inputs XOR gates is $(m-1)+m+N_{\alpha^2} = 2m+N_{\alpha^2}-1$. □

For the time complexity of the proposed MSB BL-FSIPO PB multiplier, we derive it in terms of the propagation delay through the corresponding levels of two-inputs AND and two-inputs XOR gates along the multiplier longest path, as follows.

Proposition 3. *Maximum propagation delay (PD) through the proposed MSB BL-FSIPO PB multiplier of Figure 1a is:*

$$PD = \max\{T_{\alpha^2} + T_X, T_A + 2T_X\}, \tag{9}$$

where T_A denotes the propagation delay of a single two-inputs AND gate and T_{α^2} is given in Lemma 3.

Proof. As one can see from Figure 1a, there are two main paths in the proposed design of the MSB BL-FSIPO PB multiplier. The first path is between the shift registers $\langle X\rangle$ and $\langle Y\rangle$, from one side, and the accumulator $\langle Z\rangle$, from the other side, which has a corresponding propagation delay of $T_A + 2T_X$. The second path lies between the output and input of the accumulator $\langle Z\rangle$, which passes through the m-bits field adder and the module α^2 (multiplication by the constant α^2). This path has a propagation delay equals to $T_{\alpha^2} + T_X$, where T_{α^2} is the propagation delay contributed by the α^2 module. Therefore, the propagation delay of the proposed MSB BL-FSIPO PB multiplier takes the value of the maximum propagation delay between these two paths. □

In practice, $\omega \in \{3, 5\}$. For example, $\omega = 3$ for ECDSA's $GF\left(2^{233}\right)$ and $GF\left(2^{409}\right)$. The following is a corollary about space and time complexities in case of trinomials.

Corollary 1. *If the $GF(2^m)$ is generated by an irreducible trinomial ($\omega = 3$), then, the total number of XOR gates and the propagation delay in the proposed MSB BL-FSIPO PB multiplier in Figure 1a evaluate to*

$$\#XORs = 2m+1, \tag{10}$$

$$PD = \begin{cases} T_A + 2T_X, & t_1 \neq m-1; \\ \max\{3T_X, T_A + 2T_X\}, & t_1 = m-1. \end{cases} \tag{11}$$

Proof. From (6), for $\omega = 3$, we have $N_{\alpha^2} = 2(3-2) = 2$ for either $t_1 \neq m-1$ or $t_1 = m-1$. Then, we obtain (10) by substituting for $N_{\alpha^2} = 2$ in (8).

On the other hand, in (7), $d = 0$ when $\omega = 3$ and $t_{\omega-2} = t_1 \neq m-1$, which gives $T_{\alpha^2} = T_X$. In the same formulation, for $t_1 = m-1$, $\delta(t_{w-2} - t_{w-3}) = 0$ (for $m > 2$)[1], hence $T_{\alpha^2} = 2T_X$. Substituting T_{α^2} in (9) completes the proof. □

The other three ECDSA fields are defined by irreducible pentanomials ($\omega = 5$ for $GF\left(2^{163}\right)$, $GF\left(2^{283}\right)$, and $GF\left(2^{571}\right)$) [6]. As mentioned earlier, $t_{\omega-2} < m-1$ in these fields. Specifically, for the two fields $GF\left(2^{283}\right)$ and $GF\left(2^{571}\right)$, there is no j such that $t_j = t_{j-1} + 1$ for $1 \leq j \leq \omega - 2$ in their corresponding pentanomials ($t_0 = 0$). The following corollary gives the number of two-inputs XOR gates and propagation delay when the field polynomial has $t_{\omega-2} < m-1$ and $t_j \neq t_{j-1}+1$ for $1 \leq j \leq \omega-2$ (including pentanomials).

Corollary 2. *Let the $GF(2^m)$ be defined by an irreducible polynomial in which $t_{\omega-2} < m-1$ and there is no $t_j = t_{j-1}+1$ for $1 \leq j \leq \omega-2$. Then, the total number of XOR gates and propagation delay in the proposed MSB BL-FSIPO PB multiplier in Figure 1a, become*

$$\#XORs = 2(m+\omega) - 5,\ PD = T_A + 2T_X. \tag{12}$$

Proof. From (6) and (7), if $t_{\omega-2} < m-1$ and $t_j \neq t_{j-1}+1$ for $1 \leq j \leq \omega-2$, then $d = 0$ and $\sum_{j=1}^{\omega-2} \delta(t_j - t_{j-1}) = r = 0$. This gives $N_{\alpha^2} = 2(\omega-2) = 2\omega - 4$ and $T_{\alpha^2} = T_X$ in (6) and (7), respectively. Substituting for N_{α^2} and T_{α^2}, respectively, in (8) and (9), we obtain (12). □

Remark 1. The field $GF\left(2^{163}\right)$ recommended by NIST for ECDSA is defined by the irreducible pentanomial $x^{163} + x^7 + x^6 + x^3 + 1$ in which $t_3 = t_2 + 1$ [6]. For this field, one can have similar derivation as the one done in Corollary 2 to find that the total number of two-inputs XOR gates required satisfies (12), while the propagation delay becomes $\max\{3T_X, T_A + 2T_X\}$.

Remark 2. In (11) and Remark 1, $\max\{3T_X, T_A + 2T_X\}$ is determined by the greater between T_A and T_X, which depends on the specific implementation technology.

In the following, we conduct a comparison between the proposed MSB BL-FSIPO PB multiplier and other existing bit-level serial PB multiplication schemes.

[1]For special case of $m = 2$, although $x^2 + x + 1$ has $t_1 = m-1$, however since $a_{m-3} = 0$ in (3), there is no XOR gate needed to generate $a_{m-1} + a^{'} = a_{m-2}$, therefore $\#XORs = 2m$ while $PD = T_A + 2T_X$.

TABLE II: Space / time complexities of different bit-level PB multipliers. Space complexity is reported in terms of # of FF, two-inputs AND and XOR gates, and 2-to-1 1-bit multiplexers (for either logic implementation or inputs preloading). Time complexity appears in terms of levels of two-inputs AND (T_A) and XOR (T_X) gates, and 2-to-1 1-bit multiplexers (T_M). $p(x) = x^m + \sum_{i=1}^{\omega-2} x^{t_i} + 1$ is the field irreducible polynomial. N_{α^2} and T_{α^2} are given in (6) and (7), respectively. $T^{'} = (1 + \lceil \log_2(\omega - 1) \rceil + \lceil \log_2(m) \rceil) T_X$ and $T^{"} = (1 + \lceil \log_2(m-1) \rceil + \lceil \log_2(n) \rceil) T_X$, where n and l_i ($0 \leq i < n$), respectively, are the number of nonzero entries in column 0 of the reduction matrix, and their row locations [19].

Multiplier	FF	AND	XOR	2-to-1 1-bit MUX*	Propagation Delay	Parallel Loading		Serial Loading
						2-to-11-bit MUX	Latency	Latency
LSB BL-SIPO [17]	$2m$	m	$m + \omega - 2$	0	$T_A + T_X$	m	m	$2m$
MSB BL-SIPO [17]	$2m$	m	$m + \omega - 2$	0	$T_A + T_X$	m	m	$2m$
BL-PISO [19]	$3m + t_{\omega-2} - 1$	$2m - 1$	$(n+1)(m-1) + \omega - 2 + \sum_{i=1}^{n-1} l_i$	0	$T_A + \max\left\{T^{'}, T^{"}\right\}$	$2m$	m	$2m$
PIPO [21]	$5m - 1$	$\frac{m^2+m}{2}$	$\frac{m^2+m}{2}$	$4m$	$T_A + \lceil \log_2 m \rceil T_X + 2T_M$	$2m$	$2t_{\omega-2} + 1$	$m + 2t_{\omega-2} + 1$
MSB BL-FSIPO (Figure 1)	$3m - 2$	$2m - 1$	$2m + N_{\alpha^2} - 1$	0	$\max\{T_{\alpha^2} + T_X, T_A + 2T_X\}$	0	m	m

* These multiplexers are used in the multiplication logic which are different from the ones used for parallel preloading of inputs.

TABLE III: Space and time complexities for NIST recommended field $GF\left(2^{233}\right)$ defined by the irreducible $x^{233} + x^{74} + 1$.

Multiplier	FF	AND	XOR	2-to-1 1-bit MUX*	Propagation Delay	Parallel Loading		Serial Loading
						2-to-11-bit MUX	Latency	Latency
LSB BL-SIPO [17]	466	233	234	0	$T_A + T_X$	233	233	466
MSB BL-SIPO [17]	466	233	234	0	$T_A + T_X$	233	233	466
BL-PISO [19]	772	465	538	0	$T_A + 10T_X$	466	233	466
PIPO [21]	1164	27261	27261	932	$T_A + 8T_X + 2T_M$	466	149	382
MSB BL-FSIPO (Figure 1)	697	465	467	0	$T_A + 2T_X$	0	233	233

* These multiplexers are used in the multiplication logic which are different from the ones used for parallel preloading of inputs.

TABLE IV: Space and time complexity estimates for the multipliers which are listed in Table III based on the standard 65nm CMOS library measures. Total gate counts are estimated in terms of total NAND gate equivalence (GE) while MPD denotes the maximum propagation delay. Latency denotes the total number of clock cycles required to generate the 233-bits of output. TP is throughput (@ 1 GHz) and TP/G denotes throughput per total GE measured in Kbps/Gate. SIL and PIL denote "Serial Input Loading" and "Parallel Input Loading", respectively.

Multiplier	MPD	GE		Latency		TP/G @ 1 GHz	
	ns	PIL	SIL	PIL	SIL	PIL	SIL
LSB BL-SIPO [17]	0.07	2973	2507	233	466	336	199
MSB BL-SIPO [17]	0.07	2973	2507	233	466	336	199
BL-PISO [19]	0.43	5484	4552	233	466	182	110
PIPO [21]	0.41	95759	94827	149	382	16	6
MSB BL-FSIPO (Figure 1)	0.11	4129	4129	233	233	242	242

III. Comparisons

In this section, we compare the proposed MSB BL-FSIPO PB multiplier to other existing serial PB multipliers. In this work, theoretical propagation delay and space complexity, respectively, are given in terms of the levels of logic gates and the count of logic gates, Flip-Flops (FF), and multiplexers (which are used for logic implementation and / or parallel preloading of inputs), for the different serial PB multiplication schemes which are listed in Table II. Results based on CAD tools-based realizations will be considered in future projects.

From Table II, it is noted that, the proposed MSB BL-FSIPO PB multiplier is advantageous for the case of serial inputs preloading since it offers a lower latency for generating the m output bits, compared to the other listed multipliers (when running at the same clock speed). This feature results in low-latency fast multiplication in resource constrained applications where the input data-path might have limited capacity for reading elements from large finite fields. For the time complexity, and similar to the BL-SIPO PB multipliers which are listed in Table II, the proposed MSB BL-FSIPO PB multiplier offers a constant propagation delay that is independent of the dimension of the $GF(2^m)$. While the proposed MSB BL-FSIPO PB multiplier has larger space complexity and propagation delay compared to the BL-SIPO PB multipliers listed in Table II, on the other hand, it offers lower space complexity and propagation delay compared to the BL-PISO and PIPO serial PB multipliers listed in the same table.

For further comparisons, we investigate the case of $GF\left(2^{233}\right)$ recommended by NIST defined by an irreducible trinomial $p(x) = x^{233} + x^{74} + 1$. Then, for this case, the resulting space and time complexities of the multipliers which are listed in Table II are reported in Table III. Also, Table IV estimates the corresponding space and time complexity readings based on the 65nm CMOS standard library's statistics. In this technology library, the NAND gate equivalences (GEs) for a two-inputs AND, two-inputs XOR, D-type FF, and a 2-to-1 1-bit Multiplexer, when reported based on synthesis results using the Synopsys Design Vision tool [25], are 1.25, 2, 3.75, and 2, respectively. In addition, based on same tool using same technology library, the maximum propagation delays

(MPD) for a two-inputs AND, two-inputs XOR, and 2-to-1 1-bit multiplexer are 0.03ns, 0.04ns, and 0.03ns respectively.

From Table IV, one can see that the listed PIPO serial PB multiplier offers the best latency in case of parallel preloading of its inputs. However, it has lowest efficiency (in terms of throughput per NAND gate equivalence measured at 1 GHz) in both parallel and serial preloading scenarios, compared to all the other listed multiplication schemes. This is mainly due to the relatively large space complexity of the PIPO multiplier.

It is noted that the BL-SIPO PB multipliers in Table IV offer the best space complexity and highest operating frequency. In addition, the BL-SIPO PB multipliers in this table show the best efficiency, in case of parallel preloading of inputs.

However, the proposed MSB BL-FSIPO PB multiplier offers lower latency, compared to the BL-SIPO, BL-PISO, and PIPO, in case of serial inputs loading. In this case, as a result of its low latency, the proposed MSB BL-FSIPO PB multiplier offers the best efficiency (at same frequency). In comparison to the BL-PISO and BL-PIPO PB multipliers, which are listed in Table IV, the proposed MSB BL-FSIPO PB multiplier is advantageous in terms of space complexity, operating frequency, as well as latency and efficiency in case of serial loading of inputs.

It is also worth noting that, in case of parallel loading of inputs, the BL-PISO PB multiplier generates its first output bit with a latency of 1 clock cycle, while proposed BL-FSIPO, as well as the BL-SIPO PB multipliers, require 233 cycles after which all output bits are generated in parallel. For the same case, the PIPO PB multiplier [21] requires 149 cycles to generate all 233 output bits, in parallel.

IV. Conclusion

In this paper, we have introduced a new bit-level serial multiplication scheme for the elements of $GF\,(2^m)$, based on the PB representation. The proposed formulation for the bit-level PB multiplication is based on a recursive definition of the field elements. The recursive definition constructs an element bit-by-bit, one bit per a clock cycle, starting from the most significant bit. Based on this recursive formulation, we have proposed, and to the best of our knowledge, the first architecture for dedicated MSB-first bit-level fully-serial-in-parallel-out (BL-FSIPO) PB multiplier in the literature. The proposed MSB-first BL-FSIPO PB multiplier does not require any preloading of its inputs. Therefore, it is advantageous for achieving high throughput in applications where the parallel preloading of the inputs is not possible (if the input data-path size is limited). For this specific case of serial preloading of the inputs, we have shown, based on the provided theoretical analysis, that our proposed MSB BL-FSIPO PB multiplier offers the highest throughput and efficiency, when compared to other bit-level serial PB multiplication schemes.

Acknowledgment

Authors would like to thank the reviewers for their constructive comments. This work has been supported in part by Natural Sciences and Engineering Council (NSERC) Scholarships and Grants to both authors.

References

[1] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and their Applications*. New York, NY, USA: Cambridge University Press, 1986.

[2] J. Deschamps, J. Imaña, and G. Sutter, *Hardware Implementation of Finite-Field Arithmetic*. McGraw-Hill Education, 2009.

[3] 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.

[4] H. El-Razouk, A. Reyhani-Masoleh, and G. Gong, "New Implementations of the WG Stream Cipher," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 9, pp. 1865–1878, Sep. 2014.

[5] W. Diffie and M. Hellman, "New Directions in Cryptography," *IEEE Trans. Inf. Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976.

[6] *Digital Signature Standard (DSS)*, Federal Information Processing Standards (FIPS), U.S. National Institute of Standards and Technology (NIST) Std. FIPS 186-4, July 2013.

[7] "IEEE Standard Specifications for Public-Key Cryptography," *IEEE Std 1363-2000*, p. i, 2000.

[8] H. Wu, "Bit-Parallel Finite Field Multiplier and Squarer Using Polynomial Basis," *IEEE Trans. Comput.*, vol. 51, no. 7, pp. 750 –758, July 2002.

[9] E. Mastrovito, "VLSI Designs for Multiplication Over Finite Fields GF(2m)," in *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, ser. Lecture Notes in Computer Science, T. Mora, Ed. Springer Berlin Heidelberg, 1989, vol. 357, pp. 297–309.

[10] B. Sunar and C. Koc, "Mastrovito Multiplier for All Trinomials," *IEEE Trans. Comput.*, vol. 48, no. 5, pp. 522 –527, May 1999.

[11] A. Halbutogullari and C. Koc, "Mastrovito Multiplier for General Irreducible Polynomials," *IEEE Trans. Comput.*, vol. 49, no. 5, pp. 503 –518, May 2000.

[12] A. Reyhani-Masoleh and M. Hasan, "Low Complexity Bit Parallel Architectures for Polynomial Basis Multiplication Over GF(2^m)," *IEEE Trans. Comput.*, vol. 53, no. 8, pp. 945 – 959, Aug. 2004.

[13] T. C. Bartee and D. I. Schneider, "Computation With Finite Fields," *Information and Control*, vol. 6, no. 2, pp. 79 – 98, 1963.

[14] A. Cilardo, "Fast Parallel GF(2^m) Polynomial Multiplication for All Degrees," *IEEE Trans. Comput.*, vol. 62, no. 5, pp. 929–943, May 2013.

[15] M. Cenk, M. Hasan, and C. Negre, "Efficient Subquadratic Space Complexity Binary Polynomial Multipliers Based on Block recombination," *IEEE Trans. Comput.*, vol. 63, no. 9, pp. 2273–2287, Sep. 2014.

[16] P. Scott, S. Tavares, and L. Peppard, "A Fast VLSI Multiplier for GF(2^m)," *IEEE J. Sel. Areas Commun.*, vol. 4, no. 1, pp. 62–66, Jan. 1986.

[17] T. Beth and D. Gollman, "Algorithm Engineering for Public Key Algorithms," *IEEE J. Sel. Areas Commun.*, vol. 7, no. 4, pp. 458–466, 1989.

[18] M. Hasan and V. Bhargava, "Division and Bit-Serial Multiplication over GF(q^m)," *Computers and Digital Techniques, IEE Proceedings E*, vol. 139, no. 3, pp. 230–236, May 1992.

[19] A. Reyhani-Masoleh, "A New Bit-Serial Architecture for Field Multiplication Using Polynomial Bases," in *Cryptographic Hardware and Embedded Systems - CHES 2008*, ser. Lecture Notes in Computer Science, E. Oswald and P. Rohatgi, Eds. Springer Berlin Heidelberg, Jan 2008, no. 5154, pp. 300–314.

[20] G. N. Selimis, A. P. Fournaris, H. E. Michail, and O. Koufopavlou, "Improved Throughput Bit-Serial Multiplier for GF(2^m) Fields," *Integration, the VLSI Journal*, vol. 42, no. 2, pp. 217 – 226, 2009.

[21] J. Imaña, "Low Latency GF(2^m) Polynomial Basis Multiplier," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 5, pp. 935–946, May 2011.

[22] S. Namin, H. Wu, and M. Ahmadi, "Power Efficiency of Digit Level Polynomial Basis Finite Field Multipliers in GF(2^{283})," in *2012 19th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, Dec. 2012, pp. 897–900.

[23] G.-L. Feng, "A VLSI Architecture for Fast Inversion in GF(2^m)," *IEEE Trans. Comput.*, vol. 38, no. 10, pp. 1383–1386, 1989.

[24] A. Al-Khoraidly and M. K. Ibrahim, "Finite field serial-serial multiplication/reduction structure and method," U.S. Patent US7 519 644 B2, Apr., 2009. [Online]. Available: http://www.google.com/patents/US7519644

[25] Synopsys, http://www.synopsys.com/.