International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 04 Issue: 05 | May -2017 www.irjet.net p-ISSN: 2395-0072

Serial-Out Bit-level Mastrovito Multipliers for High Speed Hybrid-Double Multiplication Architectures

Mrs Ramya K[1], Anupama H M[2], Anusha M K[3], Manjunath K N[4], Kishore Kumar[5]

[1]*AssociateProfessor, Dept. ofECE, BGSInstituteofTechnology, Karnataka, India*

[2,3,4,5] *Student, Dept. ofECE, BGSInstituteofTechnology, Karnataka, India*

ABSTRACT: The *Serial-out bit level multiplication scheme is characterized by an important latency feature. It has an ability to sequentially generate an output bit of the multiplication result in each clock cycle. However, the computational complexity of the existing serial-out bit-level multipliers in GF(2m) using normal basis representation, limits its usefulness in many applications; hence, an optimized serial-out bit-level multiplier using polynomial basis representation is needed. In this paper, we propose new serial-out bit-level Mastrovito multiplier schemes. We show that in terms of the time complexities, the proposed multiplier schemes outperform the existing serial-out bit-level schemes available in the literature. In addition, using the proposed multiplier schemes, we present new hybrid-double multiplication architectures. To the best of our knowledge, this is the first time such a hybrid multiplier structure using the polynomial basis is proposed. Prototypes of the presented serial-out bit-level schemes and the proposed hybrid-double multiplication architectures (10 schemes in total) are implemented over both GF (2163) and GF (2233), and experimental results are presented.*

Index Terms—serial-out, polynomial basis, bit-level multiplier, Mastrovito multiplier, hybrid-double multiplication

I. INTRODUCTION

FINITE field arithmetic has been widely applied in applications of different fields like error-control coding, cryptography, and digital signal processing [1], [2], [3], [4]. The arithmetic operations in the finite fields upon characteristic two GF(2m) have adopted widespread use in public-key cryptography such as point multiplication in elliptic curve cryptography [5], [6], and exponentiation-based cryptosystems [7], [8]. The finite field GF (2m) has 2m elements and each of its elements can be represented by its m binary coordinates based on the choice of field-generating polynomial. For such a representation, the addition is relatively straight-forward by bit-wise XORing of the corresponding coordinates of two field elements. On the other hand, the multiplication operation requires larger and slower hardware. Exponentiation, and division/inversion are other complex and time consuming operations and they are implemented by the iterative application of the multiplication operations. Much of the ongoing research in this area is focused on ending new architectures to implement the arithmetic multiplication operation more efficiently (see for example [9], [10], [11]).The implementation of finite field multipliers can be categorized, in terms of their structures, into three groups of parallel-level, digit-level and bit-level types. The bit-level multiplier scheme, which processes one bit of input per clock cycle, is area-efficient and suitable for resource-constrained and low-weighted devices. The bit level type multiplication algorithms, when the PB is used are classified as least significant bit first (LSB-first), and most significant bit first (MSB-first) schemes [16]. The bit-level multiplier can be further categorized into two types of either parallel or serial output. In the traditional parallel-out bit-level (POBL) multipliers [16], all of the output bits of the multiplication (from the first bit to the last bit) are generated at the end of the last clock cycle. Serial-out bit-level (SOBL) multipliers, on the other hand, generate an output bit of the product sequentially, after a certain number of clock cycles. Compared to the traditional parallel-out architecture multiplication scheme based on serial-out architecture i.e., SOBL has more advantages. For instance, combining a SOBL with a traditional LSB-first POBL one, would make fast exponentiation and inversion possible [17], [18]author of [19], has proposed a SOBL multiplication architecture that is constructed by the trinomials and the ω-nomials irreducible polynomials in GF (2m) using PB representation. In this paper, alternative schemes for the serial-out multiplication in the PB over GF (2m) for both trinomial and ω-nomial irreducible polynomial are developed. We summarize our contributions as follows:

- We have proposed a new scheme for the SOBL multiplication architecture in the PB over GF (2m) for the ω nomials, then we further optimized it for the irreducible trinomials. Both schemes have lower critical path delay compared to previously published results
- In order to investigate the applicability of the proposed SOBL schemes, we employed the proposed two SOBL schemes, and the SOBL scheme proposed in [19], to present, to our knowledge, the first approach for hybrid-double multiplication architecture in the PB over GF (2m).
- We extended the traditional POBL multiplier schemes presented in [16] to propose two new LSB first/MSB-first POBL double multiplication architectures, which perform two multiplications together after 2m clock cycles.
- To obtain the actual implementation results, all the proposed schemes, i.e., 2 SOBL multipliers, 3 hybrid-double multiplication architectures, 2 double multiplication architectures, and the counterpart ones, i.e., LSB-first POBL [16], MSB-first POBL [16], and SOBL scheme proposed in [19] are coded in VHDL (10 schemes in total), and implemented on ASIC technology over both GF(2163) and GF(2233).

II. METHODOLOGY OF THE SYSTEM

1. PRELIMINARIES

The binary extension field GF (2m) can be viewed as an m-dimensional vector space defined over GF (2) [1]. A set of m linearly independent vectors (elements of GF (2m)) is chosen to serve as the basis of representation. An explicit choice for a basis is the ordered set αm−1, ⋯, α2, α, 1, where α ∈ GF (2m) and is a root of an irreducible polynomial P(x). Each element is represented by a polynomial of degree m−1, whose coefficients are the binary digits 0 or 1. All arithmetic operations are performed modulo 2. A straightforward GF (2m) multiplication computations consists of two parts, the product of two field elements, followed by a modular reduction [20], [21]. Suppose A = (am−1, ⋯, a1, a0), B = (bm−1, ⋯, b1, b0) are two arbitrary field elements, i.e., A, B ∈ GF(2m), then to obtain the field multiplication of A and B, AB is computed first; it is then followed by the modular reduction, i.e., C , AB mod P(α). In [14], [15], Mastrovito has proposed an efficient dedicated parallel multiplication method that combines the two parts of the product and the modular reduction into a single step. He showed that the coordinates of C are obtained from the matrix-by-vector product of

$$\mathbf{c} = [cm-1, \cdots, c1, c0]^T = \mathbf{M} \cdot \mathbf{b}, \quad (1)$$

where T denotes the transposition; the column vector b = [bm−1, ⋯, b1, b0]T contains the coordinates of the multiplier B = (bm−1, ⋯, b1, b0) ∈ GF(2m), and M is an m × m binary matrix whose entries depend on the coordinates of A ∈ GF(2m). Sunar and Koc¸ [22] have studied the Mastrovito matrix M, and have presented a formulation for the Mastrovito algorithm using the irreducible trinomials. Halbuto˘gullari and Koc¸ in [23] have presented a new architecture for the Mastrovito multiplication and have also shown that the coefficient of the product AB can be obtained from the matrix-by-vector product of

$$\mathbf{d}=[d2m-2, \cdots, dm, dm-1, \cdots, d0]\ T = Z \cdot \mathbf{b},$$

where Z is a 2m−1×m binary matrix whose entries are

$$Z = \begin{pmatrix} a0 & 0 & \dots & 0 & 0 \\ a1 & a0 & \dots & 0 & 0 \\ \vdots & & & & \\ a_{m-1} & a_{m-2} & \dots & a1 & a0 \end{pmatrix}$$

2. NOTATIONS

Let us now introduce the following notations, which will be used in this paper: Column vectors are represented by small boldfaced characters. Matrices are represented by capital boldfaced characters, and to represent the entries of a matrix, we use the common notation used in the literature such as in [22], [23], [24], [25], and [19]. These notations are summarized in TABLE 1.

TABLE 1: List of notations.

Symbol	Description
b, b^T	Column and row vectors, respectively
M (i, :)	The i^{th}row of matrix **M**
M (:, j)	The j^{th}column of matrix **M**
M (i: j)	The entry with position (i,j) of the matrix **M**
$[v_j, \dots, v_i]$	The range of bits in the vector v from position i to position j, j >i.
$[r_j, \dots, r_i]$	The range of bits in the

	register [R] from position i to position j, j >i.
M[↓ n]	A down shift of the matrix **M** by n positions, emptied positions after the shifts are filled by zeros.
M(j,:)[→1]	A right shift of the j^{th} row of the matrix **M** by 1position, emptied positions after the shifts are filled by zeros.
v[f0, ↓ 1]	A down shift of the vector v by one-bit with cell f_0 fed in its upper-most bit, i.e., for the vector v of length l-bits
$e_i \|\| v^T$	The process of concatenating an element e_i and a vector **v**.

3. REDUCTION PROCESS

Let us first define an irreducible polynomial with ω nonzero terms, i.e., [19]

$$P(x) = x^m + \sum_{i=1}^{w-1} x^{t_i} \quad (3)$$

Where $m/2 > t1 > t2 > \cdots > t\omega-2 > t\omega-1 = 0$. Then from (3), we define two new sets: T is a set of degrees of nonzero terms in (3), and N consists of ω−1 elements, which are the differences between m and the others contains the non-zero terms in (3), i.e.,T ={0, t1, ⋯, tω−2}, and N ={0, Δ1, ⋯, Δω−2}, where Δ1 = m−tω−2, Δ2 = m−tω−3,⋯, Δω−2 = m−t1. In the Mastrovito matrix M, which is shown in fig (1) can be deduced by reducing the matrix Z in (2) (3). that the entries of the matrix M can be obtained as

M=(L+Q.U) (4)

where L is an m×m lower triangular Toeplitz matrix is an (m−1)×m upper triangular Toeplitz matrix

$$\mathbf{L}= \begin{pmatrix} a_0 & 0 & 0 & 0 \dots & 0 \\ a_1 & a_0 & 0 & 0 \dots & 0 \\ \vdots & & & & \\ a_{m-2} & a_{m-3} & \dots & a_1 & a_0 & 0 \\ a_{m-1} & a_{m-2} & \dots & a_2 & a_1 & a_0 \end{pmatrix} \quad (5)$$

$$\mathbf{U}= \begin{pmatrix} 0 & a_{m-1} & a_{m-2} & \dots & a_1 \\ 0 & 0 & a_{m-1} & \dots & a_2 \\ \vdots & & & & \\ 0 & 0 & \dots & a_{m-1} & a_{m-2} \\ 0 & 0 & \dots & 0 & a_{m-1} \end{pmatrix}$$

And Q is a reduction matrix, which is formalized in [24], [26], and [25] as

$$Q=\sum_{n\in N} \hat{Q}[\rightarrow n] \quad (6)$$

Where

$$\hat{Q} = \sum_{t\in T} I_{m*(m-1)} [\downarrow t] \quad (7)$$

Where $I_{m*(m-1)}$ represents an m× (m−1) identity matrix. Then, using (6) and (7) the matrix **M** in (4) can be written as [24]

$$\mathbf{M = L+S+ \sum_{t\in T-\{0\}} S [\downarrow t],} \quad (8)$$

Where the matrix S is an m×m upper triangular Toeplitz matrix with the follows

$$S= \begin{pmatrix} 0 & s_{m-1} & s_{m-2} & \dots & s_1 \\ 0 & 0 & s_{m-1} & \dots & s_2 \\ \vdots & & & & \\ 0 & 0 & \dots & 0 & s_{m-1} \\ 0 & 0 & \dots & 0 & 0 \end{pmatrix} \quad (9)$$

Where the row 0 of S, i.e., S (0, :) can be computed as [24]

$$S(0,:) = [0, s_{m-1} \dots s_1] = \sum_{n\in N} U(0,:)[\rightarrow n] \quad (10)$$

III.PROPOSED SERIAL-OUT BIT-LEVEL MASTROVITO MULTIPLICATION ALGORITHM

From (4) and (8), one can define a matrix P as

$$\mathbf{P=Q.U=S+\sum_{t\in T-\{0\}} S[\downarrow t]} \quad (11)$$

In (11), the rows produced due to the reductions corresponding to the x^{t_i} terms in (3) are identical to the rows produced at the first reduction iteration. Thus, we can store the elements of row S (0, :), so that they can be added later to obtain the rows t_i, $1 \le i \le \omega - 2$, of the matrix **P,** i.e., **P** (t_i, :), for $ti \in T - \{0\}$. Then, the rows P (j, :), for $0 \le j \le m-1$ can be obtained as

$$\mathbf{P}(j,:)= \begin{cases} S(0,:), & \text{for } j = 0, \\ P(j-1,:)[\rightarrow 1], & \text{for } 0<j \;\&\; j\neq t_i \\ P(j-1,:)[\rightarrow 1]+S(0,:), & \text{for } j = t_i, \end{cases} \quad (12)$$

for $1 \le i \le \omega-2$.

From the Toeplitz matrix L, which is shown in (5), one can see that the rows L(j, :), for $0 \le j \le m-1$ can be obtained as

$$\mathbf{L}(j,:)=\begin{cases}[a0, 0 \ldots 0], & \text{for } j=0,\\ L(j-1,:)\,[a_j,\rightarrow 1], & \text{for } 0<j\le m-1.\end{cases} \quad (13)$$

From (12) and (13), the row j of the matrix M in (4), i.e., M(j, :), for $0 \le j \le m-1$, is obtained as

$$\mathbf{M}(j,:)=\begin{cases}\mathbf{L}(0,:)+S(0,:), & j=0,\\ \mathbf{M}(j-1,:)[a_j,\rightarrow 1], & 0<j\&j\neq t_i,\\ \mathbf{M}(j-1,:)[a_j,\rightarrow 1]+S(0,:), & j=t_i,\end{cases} \quad (14)$$

For $1 \le i \le \omega-2$.
From (10) and (13), one can see that the row 0 of the matrix **M** in (14) can be obtained as

$$\mathbf{M}(0,:)=L(0,:)+S(0,:)=[a0,s_{m-1},s_{m-2},\cdots,s_1]. \quad (15)$$

After calculating **M** (j, :) and based on (1), one can serially obtain cj, for $0 \le j \le m-1$ as

$$c_j=\mathbf{M}(j,:)\cdot\mathbf{b}. \quad (16)$$

IV.PROPOSED SOBL MULTIPLICATION ALGORITHM FOR ω-NOMIALS

From (10), (14), (15), and (16) ,we write a algorithm which outlines the process of serially generating the coordinates C starting from c0 to ending cm−1 for the multiplication of the two field elements A and B.

Algorithm1:ProposedSerial-OutBit-Level Mastrovito Multiplier for ω-nomials $x^m+x^{t1}+\cdots+x^t\omega-2+1$

Input: The parameters of the ω-nomial irreducible polynomial: m, t1, ⋯, tω−2,
A =(am−1, ⋯, a0), B=(bm−1, ⋯, b0)∈GF (2^m).
Output: cj, where C = (cm−1, ⋯, c0) = AB mod P (α).
/* Set signal vectors sT, yT, and zT of length m−1, m−1, and m bits, respectively */
Initialize: yT= [ym−2, ⋯, y0] = (am−1, ⋯, a1);
zT = [zm−1, ⋯, z0] = (bm−1, ⋯, b0) ;
sT = [sm−1, ⋯, s1] = (am−1, ⋯, a1).
/* Compute sT = S (0, :) */
Step 1: For i = 1 to ω−2 **do**
Step1.1:Δi=m−tω−1−i;
Step1.2:s^T=[sm−1,⋯,s1]+[0,⋅,0,am−1,⋯, aΔi+1];
Step 2: End For
/* Set a signal vector wT of length m−1 bits, and initialize it with S (0, :), and set a signal vector xT of length m bits, and initialize it with M (0, :) */
Step 3: $w^T \leftarrow s^T$;$x^T \leftarrow$ a0||s^T;
Step 4: For j = 0 to m−1 **do**
/* Compute the inner product: cj = M (j, :) ⋅b */
Step 4.1: Output cj = xT •z;
/* Update xT with M (j+1, :) */
Step 4.2: If j 6= ti−1 **Then**
/* M (j+1, :) = M (j, :) [aj+1, → 1] */
Step 4.2.1:xT← [y0, xm−1, ⋯, x1];
Step 4 .3: **Else** /* j = ti−1 */
/* M (j+1, :) = M (j, :) [aj+1, → 1] + S (0, :) */
Step 4.3.1: xT← [y0, xm−1 + wm−2, ⋯, x1 + w0];
Step 4 .4: **End If**
Step 4.5: yT← [y0, ym−2, ⋯, y1];
Step 5: **End For**

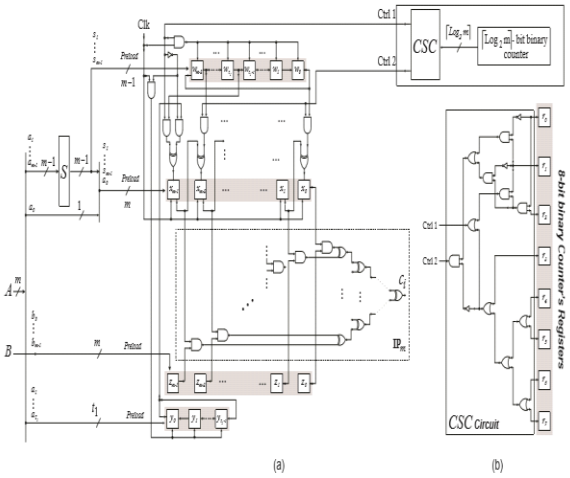


Fig. 1: The proposed serial-out bit-level (SOBL)

The proposed serial-out bit-level (SOBL) Mastrovito multiplier architecture for the !-nomial. (a) The highlevel architecture. (b) The implementation of the control signal circuit (CSC) that generates the signals Ctrl1 and Ctrl2 from the 8-bit binary counter's registers for the GF(2^{163}) field constructed by

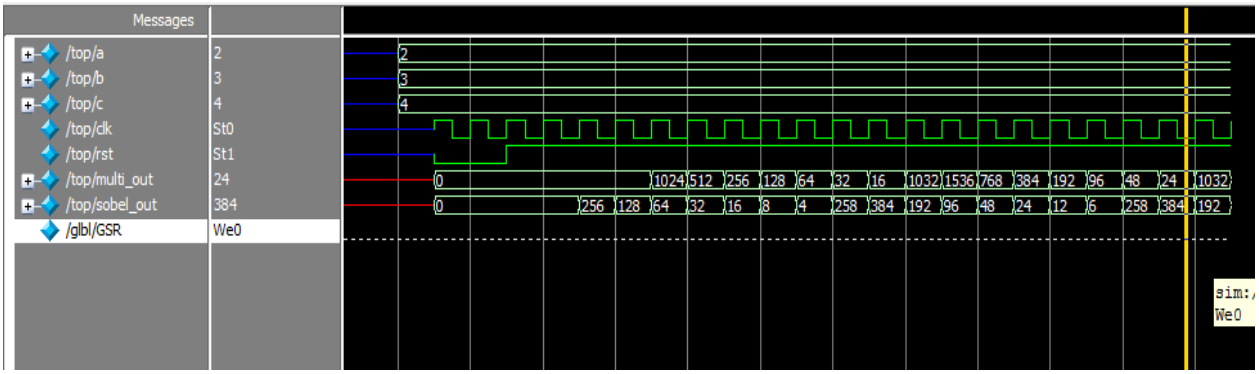
$$P(x) = x^{163} + x^7 + x^6 + x^3 + 1.$$

V.CONCLUSION

We have presented new hardware schemes for the serial-out bit-level (SOBL) multiplier in PB representation over GF (2^m) for both the ω-nomial and the irreducible trinomial. Compared to previously published results in terms of time complexities, the work presented here outperform the existing SOBL multiplier schemes. which perform two multiplications after 2m clock cycles. Then, we proposed three hybrid-double multiplication architectures in PB over GF (2^m). These hybrid multiplier structures perform two multiplications with latency comparable to the latency of a single multiplication, i.e.,

after m + 1 clock cycles. For the practical purposes, all the 10 schemes presented in this work have been implemented in ASIC technology over both GF (2^{163}) and GF (2^{233}), and the area, timing, power consumption, and energy results have been presented.

VI. SIMULATION RESULTS



VII. REFERENCES

[1] R. Lidl, and H. Niederreiter, Introduction to Finite Fields and Their Applications. 2nd Ed., Cambridge Univ. Press, Cambridge, UK, Aug. 1994.
[2] R. E. Blahut, Theory and Practice of Error Control Codes.AddisonWesley, Reading, MA, May 1983.
[3] A. J Menezes, I. F. Blake, X. Gao, R. C. Mullin, S. A. Vanstone, and T. Yaghoobian, Applications of Finite Fields. Kluwer Academic Publishers, Boston, MA, 1993.
[4] R. E. Blahut, Fast Algorithms for Digital Signal Processing. 1st Ed., Addison-Wesley, Reading, MA, Sept. 1985.
[5] V. S. Miller, "Use of Elliptic Curves in Cryptography,"
[6] N. Koblitz, "Elliptic Curve Cryptosystems," Mathematics of Computation, vol. 48, no. 177, pp. 203-209, Jan. 1987.
[7] T. Elgamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," IEEE Trans.
[8] W. Diffie, and M. Hellman, "New Directions in Cryptography," IEEE Trans. Inf. Theory, vol. 22, no. 6, pp. 644-654, Nov. 1976.
[9] M. A. Hasan, A. H. Namin, and C. Negre, "Toeplitz Matrix Approach for Binary Field Multiplication ," IEEE Trans. VLSI Systems, vol. 20, no. 3, pp. 449-458, Mar. 2012.
[10] H. Wu, "Bit-Parallel Polynomial Basis Multiplier for New Classes of Finite Fields," IEEE Trans. Computers, vol. 57, no. 8, pp. 10231031, Aug. 2008.