Low Complexity Systolic Architecture for Modular Multiplication over $GF(2^m)$

Hyun-Sung Kim¹ and Sung-Woon Lee²

¹ Kyungil University, School of Computer Engineering,
712-701, Kyungsansi, Kyungpook Province, Korea

² Tongmyong University, Dept. of Information Security,
Busan, Korea

Abstract. The modular multiplication is known as an efficient basic operation for public key cryptosystems over $GF(2^m)$. Various systolic architectures for performing the modular multiplication have already been proposed based on a standard basis representation. However, they have high hardware complexity and long latency. Thereby, this paper presents a new algorithm and architecture for the modular multiplication in $GF(2^m)$. First, a new algorithm is proposed based on the LSB-first scheme using a standard basis representation. Then, bit serial systolic multiplier is derived with a low hardware complexity and small latency. Since the proposed architecture incorporates simplicity, regularity, and modularity, it is well suited to VLSI implementation and can be easily applied to modular exponentiation architecture. Furthermore, the architecture will be utilized for the basic architecture of crypto-processor.

1 Introduction

The arithmetic operations in the finite field have several applications in error-correcting codes[1], cryptography[2, 3], digital signal processing[4], and so on. Information processing in such areas usually requires performing multiplication, inverse/division, and exponentiation. Among these operations, the modular multiplication is known as the basic operation for public key cryptosystems over $GF(2^m)$ [2- 4]. Exponentiation is computed efficiently by the sequences of modular multiplications. And division and inverse can be regarded as a special case of exponentiation because $B^{-1} = B^{2^m - 2}$ [5, 6].

Recently, three types of multipliers over $GF(2^m)$ have been proposed that are easily realized using VLSI techniques. These are normal, dual, and standard basis multipliers, which have their own distinct features. The normal and dual basis multipliers need basis conversion, while the standard does not. In the following, we restrict our attention to the standard basis multiplier.

Numerous architectures for modular multiplication in $GF(2^m)$ have been proposed in [7-10] over the standard basis. In 1984, Yeh et al. proposed two systolic array architectures with the LSB-first modular multiplication [7]. Wang et al. in [8] proposed two systolic architectures with the MSB-first fashion with less control problems as compared to [6]. Jain et al. proposed another multiplier architecture [9]. Its latency is smaller than those of other standard-basis multipliers, but there are broadcast lines in

the circuit. Wu et al. in [10] proposed bit-level systolic arrays with a simple hardware complexity with the MSB-first modular multiplication.

This paper proposes a new algorithm and its architecture for the modular multiplication with a bit-serial systolic architecture. The proposed algorithm supports the LSB-first scheme and the proposed architecture has a good time and area complexity compared to the previous multipliers.

2 Modular Multiplication

A finite field GF(2^m) has 2^m elements and it is assumed that all the (2^m-1) non-zero elements of GF(2^m) are represented using the standard basis. Let A(x) and B(x) be two elements in GF(2^m) and F(x) be the primitive polynomial, where A(x) = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + ... + a₁x + a₀, and B(x) = b_{m-1}x^{m-1} + b_{m-2}x^{m-2} + ... + b₁x + b₀, where a_i and b_i ∈ GF(2) (0 ≤ i ≤ m-1). A finite field of GF(2^m) elements is generated by a primitive polynomial of degree m over GF(2). Let F(x) be an irreducible polynomial that generates the field and is expressed as F(x) = x^m + f_{m-1}x^{m-1} + ... + f₁x + f₀. If α is the root of F(x), then F(α) = 0, and F(α) ≡ α^m = f_{m-1}α^{m-1} + ... + f₁α + f₀, where f_i ∈ GF(2) (0 ≤ i ≤ m-1). To compute the AB operation, the following equation is the common LSB-first algorithm

$$\begin{aligned} P &= AB \bmod F(x) \\ &= b_0[A\alpha^0 \bmod F(x)] + b_1[A\alpha \bmod F(x)] + b_2[A\alpha^2 \bmod F(x)] + \dots \\ &\quad + b_{m-1}[A\alpha^{m-1} \bmod F(x)] \end{aligned} \quad (1)$$

A new algorithm for the modular multiplication is derived from the equation 1 that is suitable for systolic array implementation. The modular reduction is necessary because of the operation [Aαⁱ mod F(x)] on the i-th step. We just try to concentrate on the modular reduction from the first term of the above equation, b₀A mod F(x). The subsequent terms in the above equation are accumulated until reaching the end. The procedure of the new algorithm is as follows:

First,

$$\begin{aligned} P_1 &= b_0[A\alpha^0 \bmod F(x)] \\ &= \left[\sum_{k=0}^{m-1} a_k b_{m-1} \alpha^k \right] \alpha^0 \bmod F(x) \end{aligned} \quad (2)$$

However, in this step we can overlap the modular multiplication of [Aα mod F(x)] for the next step operations. For the better understanding, we will use the symbol A⁽ⁱ⁾ for the i-th snapshot for A, which has the value of Aαⁱ mod F(x). Thereby, our algorithm computes two operations as follows

$$\begin{cases} P^{(1)} = b_0 A^{(0)} = \left[\sum_{k=0}^{m-1} a^{(0)}_k b_{m-1} \alpha^k \right]; \\ A^{(1)} = A^{(0)} \alpha \bmod F(x) = \left[\sum_{k=0}^{m-1} a^{(0)}_k \alpha^k \right] \alpha \bmod F(x) = \left[\sum_{k=0}^{m-1} a^{(1)}_k \alpha^k \right]; \end{cases} \quad (3)$$

where

$$\begin{cases} p_k^{(1)} = a_k^{(0)} b_{m-1} \quad (k = 0, \dots, m-1); \\ a_k^{(1)} = a_{m-1}^{(0)} f_k + a_{k-1}^{(0)} \quad (k = 1, \dots, m-1); \\ a_0^{(1)} = a_{m-1}^{(0)} f_0; \end{cases} \quad (4)$$

In the general case,

$$\begin{cases} P^{(i)} = P^{(i-1)} + b_{i-1} A^{(i-1)} = [\sum_{k=0}^{m-1} (p_k^{(i-1)} + a_k^{(i-1)} b_{m-1}) \alpha^k]; \\ A^{(i)} = A^{(i-1)} \alpha \bmod F(x) = [\sum_{k=0}^{m-1} a_k^{(i-1)} \alpha^k] \alpha \bmod F(x) = [\sum_{k=0}^{m-1} a_k^{(i)} \alpha^k]; \end{cases} \quad (5)$$

where

$$\begin{cases} p_k^{(i)} = p_k^{(i-1)} + a_k^{(i-1)} b_{m-1} \quad (k = 0, \dots, m-1); \\ a_k^{(i)} = a_{m-1}^{(i-1)} f_k + a_{k-1}^{(i-1)} \quad (k = 1, \dots, m-1); \\ a_0^{(i)} = a_{m-1}^{(i-1)} f_0; \end{cases} \quad (6)$$

Finally,

$$P^{(m)} = P^{(m-1)} + b_{m-1} A^{(m-1)} = [\sum_{k=0}^{m-1} a_k^{(m-1)} b_{m-1} \alpha^k]; \quad (7)$$

where $p_k^{(m)} = p_k^{(m-1)} + a_k^{(m-1)} b_{m-1} \quad (k = 0, \dots, m-1)$.

Thus the multiplication of two elements A and B in $\text{GF}(2^m)$ can be computed using the above new recursive algorithm. From the algorithm, we can derive an efficient systolic multiplier by following the procedures in [8-9].

3 Systolic Modular Multiplier

This section proposes a serial systolic multiplier. Fig.1 shows the dependence graph for our new multiplication over $\text{GF}(2^4)$. The inputs A and F enter the array in parallel from the top row, while B is from the leftmost column. The output P is transmitted from the bottom row of the array in parallel.

Fig 2 (a) shows a basic cell in Fig. 1 for the general case where the circuit function is primarily governed by the following recurrence equation:

$$\begin{cases} p_k^{(i)} = p_k^{(i-1)} + a_k^{(i-1)} b_{m-1} \quad (k = 0, \dots, m-1); \\ a_k^{(i)} = a_{m-1}^{(i-1)} f_k + a_{k-1}^{(i-1)} \quad (k = 1, \dots, m-1); \\ a_0^{(i)} = a_{m-1}^{(i-1)} f_0; \end{cases} \quad (8)$$

where the k -th bit ($p_k^{(i)}$) of P_i is the intermediate result of the product. Fig. 2 (b) shows a processing element for the case that the cell is located in the last row.

Fig. 1. Dependence graph in $GF(2^4)$

(a) PE1

(b) PE2

Fig. 2. Processing elements

$$p_k^{(m)} = p_k^{(m-1)} + a_k^{(m-1)}b_{m-1} \ (k = 0, \dots, m-1) ; \tag{9}$$

Note that the processing element in the bottom row, Fig. 2 (b), is very simple and reduces the total cell complexity compared to previous architectures.

Fig. 3. Proposed systolic multiplier in GF(2^m)

(a) PE3

(b) PE4

Fig. 4. Processing elements

An optimal systolic array among the several systolic arrays is derived by the projection vector $[1,0]^T$ and schedule vector $[2,1]^T$ from the dependency graph in Fig. 1. Fig. 3 shows the proposed bit-serial systolic multiplier. The dot (“•”) on the data flow means the buffer for one time step delay. Let $I = (I_{ctl}, I_{f(x)}, I_B, I_A, I_P)$ denote the layout of input values for the systolic array. I_{ctl} , $I_{f(x)}$, I_B , I_A , and I_P are data sequences of control signals, $f(x)$, B , A , and P , respectively. Each sequence is as follows:

$$\begin{aligned} I_{ctl} &= \{1, 0, 0, \dots, 0\} \\ I_{f(x)} &= \{\bullet, f_{m-1}, f_{m-2}, \dots, f_0\} \\ I_B &= \{b_{m-1}, b_{m-2}, \dots, b_0\} \\ I_A &= \{a_{m-1}, a_{m-1}, a_{m-2}, \dots, a_0\} \\ I_P &= \{0, 0, 0, \dots, 0\} \end{aligned}$$

The “•” of the digit sequences means a time delay of input data. Only the first bit among m bits of the control signal has the value 1 as the values of a and b should be held for m clock times.

Fig 4 (a) shows a basic cell in Fig. 3 for the general case where the circuit function is primarily governed by the equation 8. Fig. 4 (b) shows a processing element for the case that the cell is located in the last row using the equation 9.

The structure of a systolic array is similar to Yeh *et al.* in [4] except the input scheduling and the structure of the last processing element. Since the horizontal path of each cell only requires two delay elements, except for the last cell, the latency is $3m-2$.

Table 1. Comparison of the bit-serial systolic architectures in GF(2^m)

Circuit Item	Yeh et al. [7]	Wang et al. [8]	Proposed	
No. of cells	m	m	m	
Function	$AB+C$	$AB+C$	$AB+C$	
Throughput	$1/m$	$1/m$	$1/m$	
Latency	$3m$	$3m$	$3m-2$	
Critical path	$T_{AND} + T_{XOR-2}$	$T_{AND} + T_{XOR-3}$	$T_{AND} + T_{XOR-2}$	
Cell complexity	2 AND 2 XOR 7 latches 2 MUX	2 AND 1 XOR ₃ 7 latches 2 MUX	PE3	2 AND 2 XOR 7 latches 2 MUX
			PE4	1 AND 1 XOR 2 latches 1 MUX
Algorithm fashion	LSB	MSB	LSB	

4 Analysis

Our multiplier was simulated and verified using the ALTERA MAX+PLUS II simulator. Table 1 shows comparisons between the proposed architecture and the related circuits. We will give a comparison of systolic architectures with Yeh et al.'s in [7] and Wang et al.'s in [8].

Before the comparison, it was assumed that AND and XOR_i denote a 2-input AND gate and i -input XOR, respectively. T_{AND} and T_{XOR_i} are the propagation delay of a 2-input AND gate and i -input XOR gate, respectively. Latches are 1-bit latch and MUX is a 2-input 1-output switch. Table 1 shows that our multiplier has a good area and time complexity compared with the previous architectures.

5 Conclusion

This paper has explored a new algorithm for computing the modular multiplication into a low-complexity systolic architecture in $\text{GF}(2^m)$. A comparison between related systolic architectures reveals that the new systolic architecture has lower property than the conventional architectures for the hardware complexity and the latency. Furthermore, it can be used as the basic architecture for computing an inverse/division operation. Moreover, the architecture has a simplicity, regularity, and modularity. Thereby, it is well suited to VLSI implementation and it can be easily utilized for the crypto-processor chip design.

References

1. W.W.Peterson and E.J.Weldon, *Error-correcting codes*, MIT Press, MA, 1972.
2. D.E.R.Denning, *Cryptography and data security*, Addison-Wesley, MA, 1983.
3. A.Menezes, *Elliptic Curve Public Key Cryptosystems*, Kluwer Academic Publishers, Boston, 1993.
4. I.S.Reed and T.K.Truong, "The use of finite fields to compute convolutions," *IEEE Trans. Inform. Theory*, 21, 1975, pp.208-213.
5. H.S.Kim, *Bit-Serial AOP Arithmetic Architecture for Modular Exponentiation*, PhD. Thesis, Kyungpook National University, 2002.
6. S.W.Weï, "VLSI architectures for computing exponentiations, multiplicative inverses, and divisions in $\text{GF}(2^m)$," *IEEE Trans. Circuits and Systems*, 44, 1997, pp.847-855.
7. C.S.Yeh, S.Reed, and T.K.Truong, "Systolic multipliers for finite fields $\text{GF}(2^m)$," *IEEE Trans. Comput.*, vol.C-33, Apr. 1984, pp.357-360.
8. C.L.Wang and J.L.Lin, "Systolic Array Implementation of Multipliers for Finite Fields $\text{GF}(2^m)$," *IEEE Trans. Circuits and Systems*, vol.38, July 1991, pp796-800.
9. S. K. Jain and L. Song, "Efficient Semisystolic Architectures for finite field Arithmetic," *IEEE Trans. on VLSI Systems*, vol. 6, no. 1, Mar. 1998.
10. C.W.Wu and M.K.Chang, "Bit-Level Systolic Arrays for Finite-Field Multiplications," *Journal of VLSI Signal Processing*, vol. 10, pp.85-92, 1995.
11. S.Y.Kung : 'VLSI Array Processors,' Prentice-Hall, 1987.
12. K.Y.Yoo : 'A Systolic Array Design Methodology for Sequential Loop Algorithms,' PhD. thesis, Rensselaer Polytechnic Institute, New York, 1992.