International Journal of Electronics Communication and Computer Engineering
Volume 3, Issue 6, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

High Speed Bit-Parallel Systolic Multiplier over GF (2^m) for Cryptographic Application

S. Arul Mozhi
Asst. Prof., Dapartment of ECE,
Anna University

Beena Thomas
Asst. Prof., Dapartment of ECE,
Anna University

Abstract **— A bit parallel systolic multiplier in the finite field GF(2^m) over the polynomial basis where irreducible polynomial generate the field GF(2^m) is presented. The complexity of the proposed multiplier is compared in terms of area, latency and speed with other existing multipliers. The proposed multiplier has high throughput as compared with the traditional systolic multiplier. Moreover, this multiplier is highly regular, modular, and therefore, well-suited for VLSI implementation with fault tolerant design.**

Keywords **– AOP, Cryptography, Galois Field, Systolic Architecture.**

I. INTRODUCTION

Finite fields of the form GF(2^m) have found applications in the implementation of error-correcting codes such as Reed-Solomon (RS) codes and also certain cryptographic systems [1-3]. Addition and multiplication are two basic arithmetic operations in finite fields. Addition operation is easily realized using XOR gates but the multiplication operation is costly in terms of component counts and time delays [4]. Finite field multiplication is the most important arithmetic operation because it is nontrivial to implement in hardware and frequently required in both the encoding and decoding algorithm of cryptography. This paper focuses on the parallel implementation of fast and low-complexity multipliers over GF(2^m) since computing exponentiation, division, and computing multiplicative inverse can be performed by computing multiplication iteratively.

Normal basis representation offers the best performance in hardware [8-10] and polynomial basis representation is more efficient in software. Using the normal basis representation, the squaring of an element in GF(2^m) is readily shown to be a simple cyclic shift of its binary digits. The main difficulties for fast normal basis multiplication in software are due to the particular computation process. First, when multiplying two elements represented in normal basis according to the standard formula, the coefficients of their product need to be computed one bit at a time. Second, the "partial sums" computation of a given bit involves a series of which need to be computed sequentially in software. To avoid the above difficulties, the normal basis multiplier realized in hardware which performs the two computations in parallel [11]. Polynomial basis multipliers are more efficient and most widely used when compared with multipliers based on normal or dual basis because polynomial basis multiplication requires a polynomial multiplication followed by a modular reduction. In practice, these two steps can be combined.

Mastrovito [12] developed a new method for multiplication where a product matrix was introduced to combine the above two steps together. The dual basis multiplier uses the dual basis representation for the multiplicand and standard basis for the multiplier. The product is again in dual basis representation and this kind of multiplier can be used in RS encoding and decoding circuits and since dual basis multiplier have particularly low hardware requirements.

The multipliers over GF(2^m) may be either systolic or non systolic. In multiplier implementations, many architectures applied systolic array concept. In general, a non systolic architecture has global signals. Hence if 'm' becomes large, propagation delay also increases. But the systolic architecture does not suffer from this problem because the systolic architecture consists of replicated basic cells and each basic cell is connected with its neighbouring cells through pipelining, i.e., there are no global signals. Consequently, the systolic architecture is a better choice than the non systolic architecture for a high-speed VLSI implementation. Various architectures for non-systolic style have already been presented in [12]. Many bit parallel systolic multipliers have been proposed in [5-7] and the fields in these multipliers are defined by irreducible polynomial.

Bit-parallel systolic multipliers are very suitable for VLSI systems because they have more simple and regular architectures than other existing array multipliers. Another advantage of array multipliers is fault-tolerant design which can be easily realized. The fault-tolerant properties are very important for VLSI systems due to yield and maintenance. However, they are inefficient for cryptographic applications because of more time-consuming than existing array multipliers using AOP [13] and trinomial polynomials presented in [14]. To overcome this problem, a new fast array multipliers based on the LSB first bit multiplication algorithm is proposed. The multiplier architecture is altered to obtain minimum critical path delay and high speed.

II. PRELIMINARIES

Galois field (GF) is a popular name for a field with finite number of elements. The simplest example of a GF is the binary field which consist of elements {0, 1} and referred to as GF(2). We can create larger fields by extending GF(2) into vector space leading to finite fields of size 2^m. These are simple extensions of the base field GF(2) over 'm' dimensions. Field element can be derived by two alternate representations. In the first representation, all elements of GF(2^m) may be represented as powers of a primitive field element α[2], for example if m=8, then the

International Journal of Electronics Communication and Computer Engineering
Volume 3, Issue 6, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

field element is of the form α^n for n=0,1,....255. In the second representation, each element has an equivalent representation as a binary m-bit. A polynomial of the form p(x) = $(p_0 + p_1x + p_2x^2 + \cdots + p_mx^m)$ over GF (2) is called an all in one polynomial of degree 'm'.

Galois field includes three different basis namely polynomial (standard), normal and dual basis. The polynomial basis of GF(2^m) can be expressed in the form {1,α,α^2 ,.....α^{m-1}}, where 'α' □ GF(2^m) be the root of a primitive polynomial of degree 'm' over GF(2). The primitive polynomial p(x) can be written as $(1 + x^2+x^3)$ for the finite field over GF(2^3). When using a polynomial basis representation, any element of the field GF(2^m) can be expressed as a binary polynomial of degree at most (m-1). The standard alternative basis for polynomial basis is a normal basis which is represented in the form {α, α^2, α^4,..... $\alpha^{(2)\wedge m-1}$ } where 'α' is the root of an primitive polynomial of degree 'm' over GF(2). A dual basis is not a concrete basis like the polynomial basis and the normal basis rather it provides a way of using a second basis for computations. Consider two bases for elements in a finite field GF(2^m) such that

B_1 = { $\alpha_0,\alpha_1,\alpha_2,.....\alpha_{m-1}$ } and B_2={$\gamma_0, \gamma_1,\gamma_2,.....\gamma_{m-1}$} then B_2 can be considered a dual basis of B_1 provided that Tr (α_i . γ_j) = {0, if i ≠ j and 1, otherwise. 'Tr' denotes the trace function in dual basis. This basis provides a way to easily communicate between devices that use different bases, rather converting explicitly between the bases using the change of bases formula.

III. MULTIPLICATION ALGORITHM

Bit parallel systolic multiplication over GF(2^m) with irreducible polynomial is as follows. Let A (x) and B (x) be the two elements in GF(2^m), P(x) be the primitive polynomial used to generate the field GF(2^m) and C (x) be the result of multiplication where,

$$C(x) = A(x).B(x) \bmod P(x).$$

Then A (x), B (x), P (x) and C (x) can be expressed as follows:

$A(x) = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} +.....+ a_1x + a_0$
$B(x) = b_{m-1}x^{m-1} + b_{m-2}x^{m-2} +.....+ b_1x + b_0$
$P(x) = x^m + p_{m-1}x^{m-1} +.....+ p_1x + p_0$
$C(x) = c_{m-1}x^{m-1} + c_{m-2}x^{m-2} +.....+ c_1x + c_0$

The LSB-first multiplication can be performed as follows:

$$\begin{aligned} C(x) &= A(x) \cdot B(x) \bmod P(x) \\ &= b_0A(x) + b_1[A(x).x \bmod P(x)] + \\ &\quad b_2[A(x).x^2 \bmod P(x)] +.....+ \\ &\quad b_{m-1}[A(x).x^{m-1} \bmod P(x)]. \end{aligned}$$

In the LSB-first scheme, the multiplication starts with the LSB of the multiplier B (x) and each cell in the i^{th} step where (1 ≤ i ≤ m), performs the following computations. Multiplication over GF(2^m) is associative.

$$A(x)^{(i)} = [A(x)^{(i-1)}] \cdot x \bmod P(x)$$
$$C(x)^{(i)} = A(x)^{(i-1)}b_{i-1} + C(x)^{(i-1)}$$

Where $C(x)^{(0)} = 0$ and $A(x)^{(0)} = A(x)$

LSB-first multiplication algorithm
Input: P (x),A (x) and B (x)
Output: C (x) = A (x).B (x) mod P (x)
Initialize: $A(x)^{(0)} = A(x)$, $C(x)^{(0)} = 0$
For i = 1 to m do
For k = m-1 down to 0 do

$$a_k^{(i)} = a_{k-1}^{(i-1)} + a_{m-1}^{(i-1)}p_k$$
$$c_k^{(i)} = a_k^{(i-1)}b_{i-1} + c_k^{(i-1)}$$

End for
End for
$C(x) = C^{(m)}(x)$

In the above algorithm $a_k^{(i)}$ and $c_k^{(i)}$ denote the k^{th} coefficient in $A^{(i)}$ (x) and $C^{(i)}$ (x) respectively and b_i denotes i^{th} coefficient of B (x) and p_k denotes k^{th} coefficient of P (x). Based on the algorithm, the signal flow graph (SFG) for systolic multiplier is drawn as shown in the figure 1 where 'm' denotes the size of the multiplier. From the SFG, it is shown that (m x m) cells are required to implement the multiplication over GF(2^m). The SFG is used for calculating the partial product and final output. The basic cell consists of two AND gates, two XOR gates. The internal architecture of the $(i,k)^{th}$ cell is given in figure 2. Note that in SFG, the right side column cells receive the input a_d from the left side column previous row cells. But there is no right side column for the right most column, hence the value of a_d for the entire right most column cells is zero.

The polynomial input and polynomial output in a cell is same since it is used only for computation. Each cell computes $a_k^{(i)}$ and $c_k^{(i)}$ which is the coefficient A $(x)^{(i)}$ and C $(x)^{(i)}$. The results of the basic cells in a row are given to the next row. The final result is obtained from the last row. Note that in LSB-first algorithm, the basic cell includes multiplying by x, current partial-product generation, and accumulation-to-previous-result.

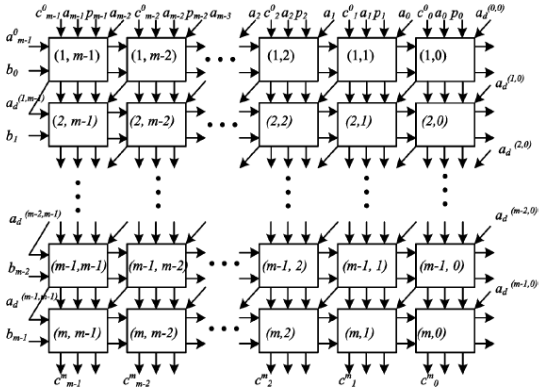

Fig.1. SFG for multiplication over GF(2^m)

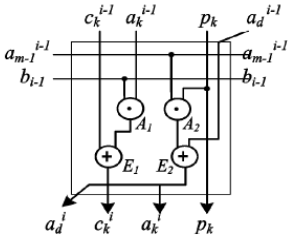

Fig.2. Circuit of $(i,k)^{th}$ cell

International Journal of Electronics Communication and Computer Engineering
Volume 3, Issue 6, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

These operations are performed concurrently in the LSB-first scheme, but sequentially in the MSB-first scheme. This multiplier requires m^2 identical cells and initial delay of '3m' clock cycles because of three latches in each cell.

In this architecture, in order to increase the speed of computation, latches were introduced in both the horizontal and vertical path using the concept of pipelining. Even though, there is an increase in speed of multiplication and reduction in critical path delay, the area consumed by single cell increases. Hence, the existing multiplier architecture is modified to obtain high speed and low critical path delay by eliminating the latches in each cell of entire multiplier architecture.

IV. Bit-Parallel Multiplier with Double Speeds

A modified version of existing multiplier is presented with the help of 8*8 bit-parallel systolic multiplier for GF(2^m). To increase the computing speed, the array multiplier in Figure 1 is modified. A modified array multiplier with double speed as comparing with that of existing multiplier is shown in Figure 3. The array multiplier in Figure 1 is divided into two parts. First, the operands A and B

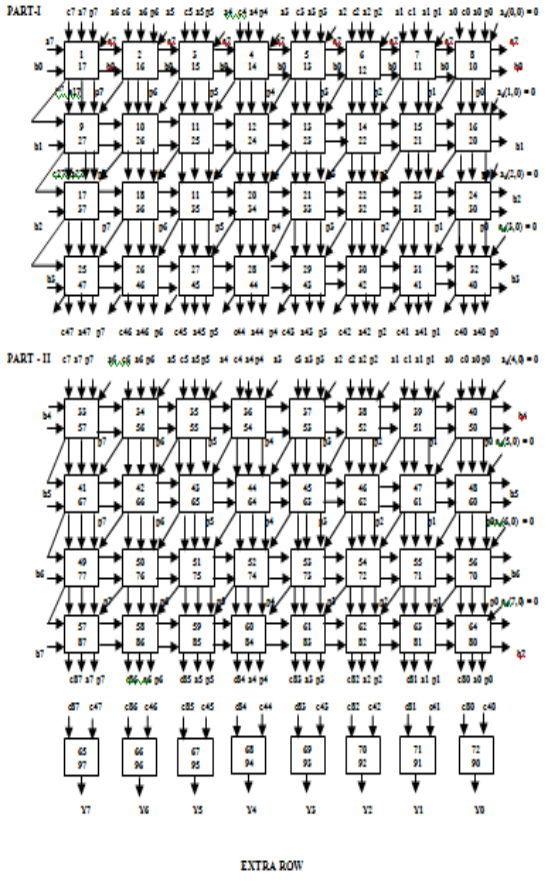

Fig.3. SFG for proposed bit-parallel multiplier over GF (2m)

are loaded into multiplier array. One extra row is added at the last to add the results in Part-I and Part-II. The cells in this extra row are same as the normal cells shown in Figure 2. But these cells perform only XOR operation to add the results of two parts. After 4 clock cycles, partial results are obtained in Part-I and Part-II and then they are added in cells in the extra row at the fifth clock cycle. But the existing multiplier requires eight clock cycle to produce the final output.

The latency of the array multiplier in Fig 3 is 5 clock cycles. In general, we can divide m*m array multiplier into two parts and then the latency becomes (m/2 + 1) clock cycles. In practice, 'm' is a large number if such multipliers are applied in cryptosystems. Therefore, the extra clock cycle for cells at the extra row is negligible. Thus the proposed bit-parallel systolic multiplier operates in high speed, low critical path and it consumes very less extra area than the existing bit-parallel multiplier.

V. Analysis and Discussion

The different versions (classical and proposed) of bit parallel systolic multiplier over GF(2^m) for irreducible polynomial, general polynomial and AOP have been designed and coded in Verilog HDL. This was done to validate our claims and demonstrate that our technique is efficient in terms of speed and delay. The designs were simulated using ModelSim and were tested for functionality by giving various inputs. The architecture of the existing multiplier was adopted to implement parallel computation of multiplications, by reducing the complexity of basic cell and ultimately the overall system complexity can be reduced.

Table I reveals that the proposed multiplier requires small extra circuit than the general polynomial multiplier but it requires less hardware when comparing with the AOP and irreducible polynomial multipliers. The latency of the proposed multiplier is also shorter than the other systolic multipliers of GF(2^m). The speed is increased by partition of arrays and parallelism. Figure 4 gives the graphical comparison in terms of area, speed and delay for existing and proposed multipliers.

Table I : Comparison of Various Systolic Multipliers over GF (2^m)

Multipliers	General polynomial multiplier	AOP based multiplier	Irreducible trinomial multiplier	Proposed Multiplier
Items				
Generating polynomial	General form	AOP form	Trinomial form	General form
Array type	Systolic	Systolic	Systolic	Systolic
Number of cells	m^2	$(m+1)^2$	U cell - m^2, V cell – (m-1	m^2 + (1 extra row)
2-input AND gate	2	1	U cell - 1, V cell - 0	2
2-input XOR gate	2	1	U cell - 1, V cell - 1	2
1-bit latches	7	3	5	3
Computation time per cell	$T_A + T_X$	$T_A + T_X$	$T_A + T_X$	$T_A + T_X$
Latency	3m	m+1	2m-1	m/2+1

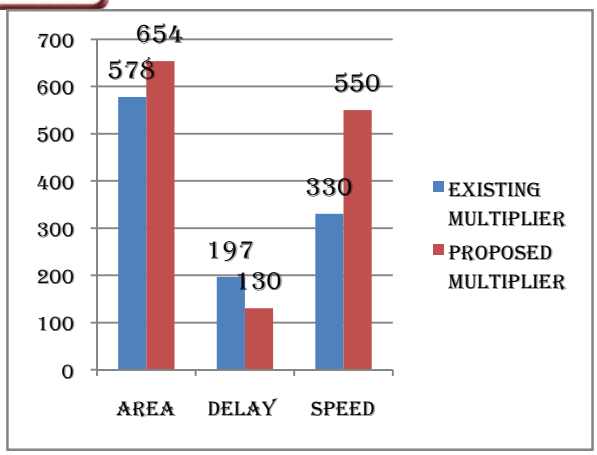

Fig.4. Comparison results of area delay and speed

VI. CONCLUSION

We have presented a bit-parallel systolic multiplier over GF(2^m) fields with double speed. We also have showed that the proposed array multiplier is efficient in terms of delay and speed comparing with the other existing multipliers. Further, the proposed multiplier can be designed with k-times speeds only dividing the array multiplier into k parts. Only few extra rows are required for the m*m array multiplier with k-times speeds.

REFERENCES

[1] Macwilliams F.J. and Sloane N.J.A *"The theory of error correcting codes,"* (North – Holland, New York), 1997.

[2] H.C.A Van Tilborg *"An Introduction to Cryptography,"* kluwer Academic Publishers, 1998.

[3] Peter Sweeney, *"Error Control Coding from theory to practice,"* John Wiley & Sons, 2002.

[4] G.B. Agnew, T. Beth, E.C. Mullin and S.A. Vanstone, *"Arithmetic operations in GF (2^m),"* Journal cryptography, 6, 3-13,.1993.

[5] J. L. Massey and J. K Omura, *"computation method and apparatus for finite field arithmetic,"* U.S.patent, 4, 587, 627, May 1986.

[6] R. C. Mullin, *"Multiple Bit multiplier,"* U.S. Patent 5, 787, 028, July 1998.

[7] R.C.Mullin,I. M.Onyszchuk,S. A.Vanstone *"Computational methods and apparatus for Finite field arithmetic,"* U.S.patent, 4,745,568, May 1988.

[8] C. L. Wang and J. L. Lin, *"Systolic array implementation of multipliers for GF (2^m),"* IEEE Trans. Circuits . Syst., vol. 38, no. 7, pp. 796–800, Jul 1991.

[9] M. A. Hassan and V. K. Bhargava *"Bit-serial systolic divider and multiplier for finite fields GF(2^m),"* IEEE Transactions on Computers, 41(8):972{980, Aug 1992.

[10] S.K.Jain,L.Song, and K.K.Parthi, *"Efficient Semi - systolic architectures for finite field arithmetic,"* IEEE Trans Very Large Scale (VLSI) Syst., vol. 6, no. 1, pp. 101 – 113, Mar 1998.

[11] Peng Ning and Yiqun Lisa Yin, *"Efficient Software implementation for finite field multiplication in normal basis,"* 2001.

[12] E. D. Mastrovito *"VLSI Architectures for Computations In Galois Fields,"* PhD thesis, Linkoping University, Dept. Electr. Eng., Linkoping, Sweden, 1991.

[13] C. Y. Lee, E. H. Lu, and J. Y. Lee, *"Bit parallel systolic Multipliers for GF (2^m) fields defined by all-one and equally – spaced polynomials,"* IEEE Trans. Comput., vol. 50, no. 5, pp. 385–393, May 2001.

[14] C. Y. Lee, *"Low complexity bit – parallel systolic multiplier over GF (2^m) using irreducible trinomial,"* IEEE Comput. Digit. Tech., vol. 150, no. 1, 2003.

[15] C.Y. Lee, *"Low - complexity parallel systolic Montgomery multipliers over GF(2^m) using Toeplitz Matrix – vector representation,"* IEICE Trans. Fundam., 2008.

AUTHOR'S PROFILE

S. Arul Mozhi
is a staff member of SNS College of Engineering, Anna University, Chennai, had received B.E. Degree from Anna University in 2008 and completed M.E. VLSI Degree from Avinashilingam University in 2012. She is currently working on VLSI testing tools for implementing the research work.

Beena Thomas
is a staff member of Rathinam Institute of Engineering and Technology, Anna University, Chennai, had received B.E. Degree from Anna University in 2009 and completed M.E. VLSI Degree from Avinashilingam University in 2012. She is currently working on VLSI back end tools for implementing the research work.