DESIGN OF POLYNOMIAL BASIS MULTIPLIERS OVER GF(2^{233})

*Vladimir Trujillo-Olaya, Jaime Velasco-Medina, Julio C. López-Hernández**

Grupo de Bionanoelectrónica, Escuela EIEE, Universidad del Valle, Cali, Colombia
* Instituto de computacao, UNICAMP, Campinas, Brasil
E-mail: vlatruo, jvelasco@univalle.edu.co, jlopez@ic.unicamp.br

ABSTRACT

This article addresses an efficient hardware implementations for multiplication over finite field GF(2^{233}). Multiplication in GF(2^n) is very commonly used in cryptography and error correcting codes. An efficient hardware could reduce the cost and development for these applications. This work presents the hardware implementation of polynomial basis. In this case, the multipliers were designed using bit-serial multiplication , bit-parallel multiplication, PCA based serial multiplication and PCA parallel based multiplication algorithms, the synthesis and simulation were carried out using Quartus II v.5.0 of Altera, and the designs were synthesized on the Stratix II EP2S60F1020C3. The simulation results show that the multipliers designed present a very good performance using small area.

1. INTRODUCTION

In order to protect or exchange confidential data, the cryptography and error correcting codes play an important role in the security of the information. Therefore, it is necessary to implement efficient cryptosystems, which can reduce the cost and development for these applications. In this context, public key cryptography based on elliptic curves is widely used because it presents higher security per key bit, and their two main applications are the private key exchange and the digital signature. Additionally, the Elliptic Curve Cryptosystems (ECC) can be used in applications where the computation resources are limited such as smart cards and cellular telephones. The ECC systems are included in the NIST and ANSI standards, and the principal advantage over other systems of public key like RSA is the size of the parameters, which are very small, however the ECC systems provide the same level of computational security.

The efficiency of an algorithm is often measured by the number of gates and the total gate delay, this work presents different algorithms for polynomial basis multiplication.

On the other hand, it is import to mention that the most expensive operation applied in elliptic curve based cryptosystems is the "scalar multiplication" of a large natural number with a point on an elliptic curve [1]. In this case, the performance of an elliptic curve cryptoprocessor depends on the multiplication over GF(2^m). Therefore, the multiplier is the most important functional block for elliptic curve cryptoprocessor design.

In the literature are presented a variety of algorithms and architectures for the polynomial basis multiplication over GF(2^m). In [2] G. Orlando and C. Paar present a super serial galois field multiplier over GF(2^{167}). In [3] M. Hütter, J Groβschädl and G. Kamendje present a versatile and scalable digit serial/parallel multiplier over GF(2^{256}). In [4] P. Kitsos, G Theorodiris and O. Koufopavlou present an efficient reconfigurable multiplier architecture over GF(2^{210}). In [5] C. Grabbe, M. Bednara, J. Teich, J. von zur Gathen and J. Shokrollahi present FPGA designs of parallel high performance GF(2^{233}).

This work addresses efficient hardware implementations for polynomial basis multiplication over GF(2^{233}). In this case, the multipliers designed present a good speed/area ratio, which is very suitable for elliptic curve cryptoprocessor design. Therefore, elliptic curve based cryptosystems can be used in applications that require small area, good speed and low consumption power, such as smart cards and cellular telephones.

This article is organized as follows. Initially, section 2 presents the arithmetic in finite field GF(2^m). Section 3 presents algorithms for polynomial basis multiplication over GF(2^m). Section 4 presents hardware architectures for polynomial basis multipliers. In section 5 the simulation results are presented. Finally, section 6 presents the conclusions and the future work.

2. ARITHMETIC IN THE FINITE FIELD GF(2^m)

A set of m linearly independent elements $\beta = \{\beta_0, \beta_1, ..., \beta_{m-1}\}$ of GF(2^m) is called a basis for GF(2^m).

A basis for GF(2^m) is important because any element $a \in$ GF(2^m) can be represented as a linear combination of the elements of β over GF(2). The two most common types of bases used in conventional hardware and software implementations are the polynomial basis and normal basis.

A polynomial basis for GF(2^m) is as follows: {1, α, α^2 , … , α^{m-1}} where α is a root of an irreducible polynomial $p(x) = x^m + \sum_{i=0}^{m-1} p_i x^i$ of degree *m* with coefficients $p_i \in$GF(2). When using polynomial basis, each element of the field is represented by a polynomial of the form $a(x) = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \ldots + a_2x^2 + a_1x + a_0$ all operations within the field are then performed modulo the polynomial *p(x)*.

Addition in GF(2^m) is implemented as component-wise XOR while a multiplication can be performed modulo an irreducible polynomial *p(x)*.

2.1. Addition

The addition of two field elements of GF(2^m) is performed by adding the coefficients modulo 2, which is nothing else than bit-wise XOR-ing the coefficients of equals powers of *x*, that is if $a = (a_{m-1}a_{m-2}\ldots a_2a_1a_0)$ and $b = (b_{m-1}b_{m-2}\ldots b_2b_1b_0)$ are elements of GF(2^m), then $a + b = c = (c_0c_1c_2\ldots c_{m-1})$ where $c_i = (a_i + b_i)$ mod 2.

2.2. Multiplication

The multiplication of two field element C=AB, where $A(x) = \sum_{i=0}^{m-1} a_i x^i$, $B(x) = \sum_{i=0}^{m-1} b_i x^i$ and $C(x) = \sum_{i=0}^{m-1} c_i x^i$, finite field multiplication can be carried out by multiplying *A(x)* and *B(x)* and then performing reduction modulo *p(x)* or alternatively by interleaving multiplication and reduction, the multiplication is shown as follows:

$$(b(x)a_{m-1}x^{m-1} + \ldots + b(x)a_2x^2 + b(x)a_1x + b(x)a_0) \bmod p(x)$$

$$C(x) = \sum_{i=0}^{m-1} b(x)a_i x^i \bmod p(x)$$

3. ALGORITHMS FOR POLYNOMIAL BASIS MULTIPLICATION OVER GF(2^m)

The serial multiplier, sometimes referred to as "MSB first multiplier" is a polynomial basis multiplier and computes the GF(2^m) multiplication in *m* cycles.

The product is obtained by the addition of partial-products, and the reduction is interleaved with the addition steps and performed by additions of the irreducible polynomial. The algorithm is shown in Figure 1.

1. MSB first polynomial basis multiplication algorithm

Input: A,B ∈ GF(2^m) ***Output:*** C=AB mod *p(x)*

1. $C^{-1}(x)=0$
2. ***For*** k = 0 ***to*** m-1 ***do***
3. $C^i(x)=[C^{i-1}(x)x+b_{m-1-i}A(x)] \bmod p(x)$

Figure 1: *MSB first polynomial basis multiplication algorithm*

In [6], H. Li and C. N Zhang present a low complexity Programmable Cellular Automata (PCA) based versatile modular multiplier in GF(2^m). in this case, the PCA rules is shown in Table 1. Where Cm is configured as the coefficients of B(x) and Cr is configured as the coefficients of P(x), Xs is configured as coefficients of A(x), Xl and Xs are partial results of neighborhood PCA. The architecture of PCA cell is shown in Figure 2

Cm	Cr	S
0	0	Xl
0	1	Xl+Xr
1	0	Xl+Xm
1	1	Xl+Xr+Xm

Table 1: PCA rules

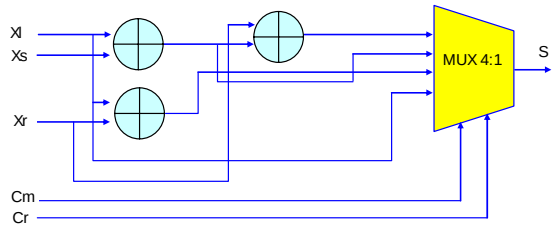

Figure 2: *PCA cell*

This work presents an architecture modular multiplier based on PCA (Programmable Cellular Automata) and the polynomial basis representation, the basic architecture of the multiplier is suitable for both parallel and serial multiplier. The algorithm is shown in Figure 3.

2. PCA based modular multiplication algorithm

Input: A(*x*),B(*x*), *p(x)* ***Output:*** C=AB mod *p(x)*

5. Reset PCA
6. Configure coefficients of *B*(*x*) as C_m, and coefficients of *P*(*x*) as C_r
7. Run PCA *m* clock cycles

Figure 3: *PCA based modular multiplication algorithm*

In [7] H. Wu presents a bit-parallel finite field multiplier which is implemented in two steps: polynomial multiplication and reduction modulo the irreducible polynomial.

1. Polynomial multiplication: S=AB

$S = \sum_{i=0}^{2m-2} s_i x^i$ and s_k is given by $s_k = \sum_{\substack{i+j=k \\ 0\le i,j\le m-1}}^{2m-2} a_i b_j$

2. Reduction modulo the irreducible polynomial:

$$\sum_{i=0}^{m-1} c_i x^i = \sum_{k=0}^{2m-2} s_k x^k \bmod p(x)$$

4. HARDWARE ARCHITECTURES FOR POLYNOMIAL BASIS MULTIPLIERS

In this section are presented the hardware architectures for polynomial basis multiplication over GF(2^{233}). In this case, MSB first multiplication, bit-parallel multiplication and modular multiplier based on PCA algorithms are implemented.

4.1. MSB first based multipliers

The hardware multiplier based on the MSB first multiplication, uses *m* cells and computes the multiplication in m cycles. The hardware architecture for the polynomial basis multiplier is shown in Figure 4.

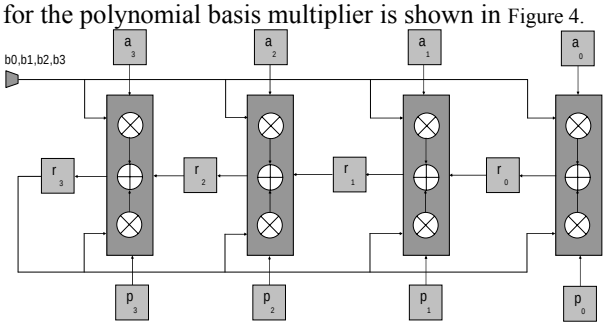

Figure 4: *MSB first based multiplier in GF(2^4)*

4.2. Serial and parallel PCA multiplier

An array of PCA cells determine the architecture of the polynomial multiplier GF(2^n), in this case in Figure 5 and Figure 6 is shown an serial and parallel multiplier over GF(2^4) respectively.

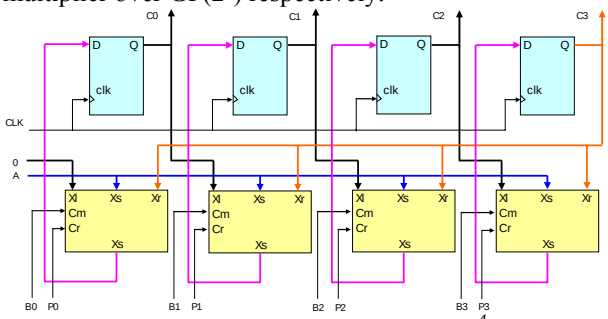

Figure 5: *Serial multiplier in GF(2^4)*

4.3. Parallel multiplier

The hardware architecture for the parallel multiplier algorithm for GF(2^{233}) is presented in Figure 7. In this case, the two modules correspond to the polynomial multiplication and modulo reduction respectively, polynomial multiplication module uses an array which uses XOR and AND functions, where m^2 AND gates and $(m-1)^2$ XOR gates are used. The equation presents the modular reduction as follows:

$$C_{i=0,1,2\ldots k-2} = S_i + S_{m+i} + S_{2m-k+i}$$

$$C_{k-1} = S_{k-1} + S_{m+k-1}$$

$$C_{i=k\ldots 2k-2} = S_i + S_{m+i} + S_{m-k+i} + S_{2m-2k+i}$$

$$C_{i=2k-1\ldots m-2} = S_i + S_{m+i} + S_{m-k+i}$$

$$C_{m-1} = S_{m-1} + S_{2m-k-1}$$

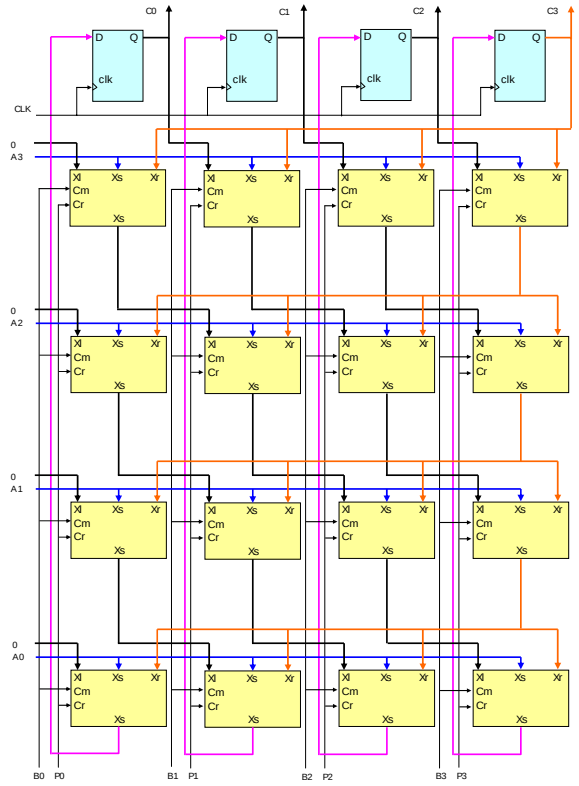

Figure 6: *Parallel multiplier in GF(2^4)*

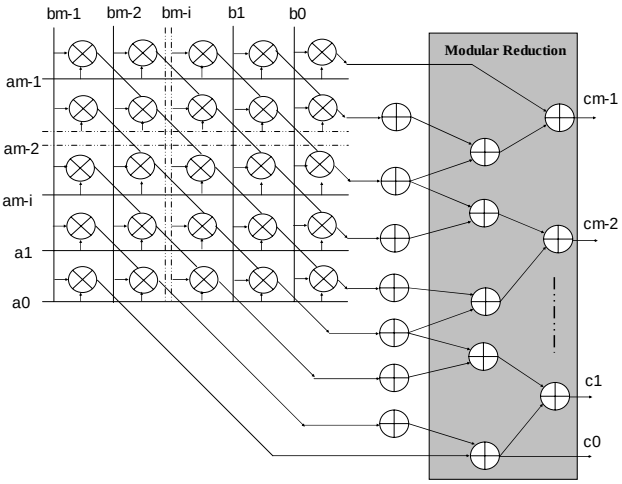

Figure 7*: Hardware architecture for GF(2^4) multiplier based on parallel multiplier algorithm*

5. SIMULATION RESULTS

In order to verify the performance of the multipliers, several simulations were carried out. The simulation results for hardware implementations are shown in Tables 1 and 2. The multipliers are implemented on the FPGA EP2S60F1020C3, and the simulation and synthesis were carried out using Quartus II version 5.0.

Serial	LC combinationals	LC registers	F_{MAX}(MHz)
MSB	163	163	215.8
PCA	163	163	215.8

Table 1: *Simulation results for serial multiplier*

parallel	Logic elements	LC registers	F_{MAX}(MHz)
Parallel	30909	0	34.44
PCA	26569	163	4.64

Table 2: *Simulation results parallel multiplier*

As could be observed from Tables 1, the MSB first and PCA algorithm based multipliers present a good performance using small area, which is very suitable for elliptic curve cryptoprocessor design. In Table 2, the Parallel multiplier present a good performance using smaller area than the PCA algorithm based multiplier.

6. CONCLUSIONS AND FUTURE WORK

This article presents the design of efficient hardware implementations for the polynomial basis multiplication over GF(2^{233}). In this case, the multipliers were designed using bit-serial multiplication, bit-parallel multiplication, PCA based serial multiplication and PCA parallel based multiplication algorithms for the multiplication over GF(2^m).

The MSB first and PCA algorithm based multipliers present a good performance this allows that elliptic curve based cryptosystems can support applications economically feasible such as smart cards and cellular telephones. The multipliers were simulated using Quartus II of Altera and synthesized on the FPGA EP2S60F1020C3.

The future work, will be oriented to design hardware for squarin and inversion using polynomial basis over GF(2^{233}), design a fast parallel multiplier over GF(2^{233}) and to implement new multiplication algorithms.

7. ACKNOWLEDGMENT

This work was sponsored by Altera Corporation through the University Program. The authors give a special thanks to Mrs Ralene Marcoccia of Altera Corporation.

8. BIBLIOGRAPHY

[1] M. Jung, "FPGA Based Implementation Of An Elliptic Curve Coprocessor Utilizing Synthesizable VHDL code", Darmstadt University of Technology. Available at http:// www.vlsi.informatik.tu-darmstadt.de/staff/ mjung/ publications/comprehensive.pdf

[2] G. Orlando, C. Paar, "a super serial galois fields multiplier for FPGAs and its application to public key algorithms", ieeexplore.ieee.org/iel5/6529/17422/00803685.pdf?arnumber=803685

[3] M. Hütter, J Großschädl and G. Kamendje "A versatile and scalable digit serial/parallel multiplier architecture for finite field GF(2^m)", www.iaik.tu-graz.ac.at/research/ publications / 2003 / ITCC2003_VSD.pdf

[4] P. Kitsos, G Theorodiris and O. Koufopavlou, "an efficient reconfigurable multiplier architecture for galois field GF(2^m) ", microelectronic journal 34 (2003) 975-980.

[5] C. Grabbe, M. Bednara, J. Teich, J. von zur Gathen and J. Shokrollahi, "FPGA design of parallel high performance GF(2^{233}) multipliers", ieeexplore.ieee.org/iel5/8570/27136/01205958.pdf? isnumber=&arnumber=1205958

[6] H. Li and C. N Zhang, "Efficient cellular automata versatile multiplier for GF(2^m)", http://www.iis.sinica.edu.tw /JISE/2002/ 2002 07_01.pdf.

[7] H. Wu, "bit-parallel finite field multiplier and squarer using polynomial basis", http//: www. ieeexplore.ieee.org/iel5/12/21897/01017695.pdf?arnumber=1017695.