Novel Hardware Architecture for a Digit Serial Multiplier GF(2^m) for Cryptographic Applications

Mario A. García-Martínez[1] , Carlos Tejeda-Calderón[1] and Rubén Posada-Gómez[1]

[1] División de Estudios de Posgrado e Investigación, Instituto Tecnológico de Orizaba, Orizaba Veracruz, México.
{marioag1955, vctejeda}@yahoo.com.mx, pgruben@yahoo.com

Abstract. This paper presents the implementation in FPGA (*Field Programmable Gate Array*) of a digit-serial multiplier that operates efficiently over finite fields *GF(2^m)*. Arithmetic operations on *GF(2^m)*, specially multiplication, are fundamental for cryptography, error control coding and digital signal processing. Design and construction of efficient architectures to perform this arithmetic operation is of great practical concern. Bit-serial architectures for *GF(2^m)* multiplication are very efficient for space complexity, but they are not suitable for time complexity; in the other way, bit-parallel architectures compute this operation in a single clock cycle, but they require a great amount of physical chip-area. In digit-serial multiplication, groups of bits known like *digits,* are processed in a single clock cycle. This allows us to design the circuit in a rank of time and space efficiencies which can be defined by the selected digit size. We have constructed a digit serial multiplier that operates in the field *GF(2^{239})*. It is a field recommended by the NIST (*National Institute of Standard Technology*) for Elliptic Curve Cryptosystems (ECC). We have used the tools of computational package ISE Version 8.1i of Xilinx for our design: VHDL (*Hardware Description Language*) to describe the circuit, and ModelSim for the simulations of the multiplier, which has been implemented in a FPGA Spartan 3 in a card prototype of Digilent.

Keywords: *FPGA, digit size multiplication, finite fields, cryptography.*

1 Introduction

Arithmetic operations over finite fields *GF*(2^m) are widely used in cryptography, error control coding and signal processing. In particular, multiplication is specially relevant since other arithmetic operators, such as division or exponentiation, which usually utilize multipliers as building blocks. Hardware implementation of field multiplication may provide a great speedup in procedure's performance, which easily exceeds the one observed in software platforms.

We represent a finite field with q elements as *GF(q)*. For computational applications the fields of extension *GF(2)*, represented by *GF(2^m)* are very important due to his possible representation by digital logic. Representation of field elements has a fundamental importance to determine the efficiency of arithmetical architectures

A. Gelbukh, S. Suárez, H. Calvo (Eds.)
Advances in Computer Science and Engineering
Research in Copmuting Science 29, 2007, pp. 295-303

Received 16/06/07
Accepted 03/07/07
Final version 24/10/07

to compute the basic operations on the field. There are different basis to represent the elements of the field $GF(2^m)$, for example: polynomial or standard basis [1], normal basis [2] and dual basis [3]. Use of certain basis determines a particular type of algorithms and architectures for $GF(2^m)$ multiplication, associated with time and space complexity of the circuit. Considering $GF(2^m)$ like a vector space on $GF(2)$, the elements of the field can be seen like vectors of m-bits. In this situation, the arithmetic operation of sum is relatively little expensive, whereas the multiplication is the most important and one of most complex. One of the main application area of the digit-serial multipliers on finite fields $GF(2^m)$ is cryptography. Nowadays, these systems require a great efficient performance in speed, area requirements, power consumption and security.

Generally, software implementations of arithmetic operations on finite fields require great resources of calculation and great amounts of memory, which affect the performance of a calculation system. Due to this recently we found in the state-of-the-art several proposals of hardware implementations of such operators [4], [5], [6], [7], [10]. We presented here the FPGA implementation of a digit serial multiplier that operates over the field $GF(2^{239})$, that is a field recommended by the NIST (*National Institute of Standard Technology*) for Elliptic Curve Cryptosystems (ECC). We used reconfigurable devices like FPGA´s, due to its characteristic of reprogramming, which allows to a greater facility in verification and redesign process.

2 Algorithm Description

Finite field multiplication of two elements A and B in $GF(2^m)$ to obtain a result $C = A*B \bmod p(x)$ (where $p(x)$ it is the irreducible polynomial) can be made with different logical architectures: serial, parallel or digit serial. Digit serial multiplication algorithm introduced in [4] for binary fields $GF(2^m)$, is very efficient in area consumption, time and power, and we have use it in this work. Several coefficients of the multiplying B are processed at the same time. The number of coefficients that are parallel processing is named the *digit size*, and it is defined as D. Let $d = [m/D]$ be the total digit number.

Let A,B be:

$$A = \sum_{j=0}^{m-1} a_j \alpha^j , \quad B = \sum_{i=0}^{d-1} B_i \alpha^{D_i}$$

Where
$$B_i = \sum_{j=0}^{D-1} b_{D_{i+j}} \alpha^j , \quad 0 \le i \le d-1 \tag{1}$$

$$C \equiv AB \bmod p(x) = A \sum_{i=0}^{d-1} B_i \alpha^{D_i} \bmod p(x) \tag{2}$$

$$C \equiv [B_0 A + B_1 (A\alpha^D \bmod p(x)) + B_2 (A\alpha^D \alpha^D \bmod p(x)) + \ldots + B_{d-1} (A\alpha^{D(d-2)} \alpha^D \bmod p(x))] \bmod p(x) \quad (3)$$

Then, we present the next algorithm for multiplication.

Digit Size Multiplier Algorithm:

==========

Input: $A = \sum_{j=0}^{m-1} a_j \alpha^j$, where $a_i \in GF(2)$ and $B = \sum_{i=0}^{[\frac{m}{D}]-1} B_i \alpha^{D_i}$

where B_i is defined in equation 1.

Output: $C \equiv A * B = \sum_{i=0}^{m-1} c_i \alpha^i$ where $c_i \in GF(2)$

1. $C \leftarrow 0$
2. **for** $i = 0$ **to** $[\frac{m}{D}] - 1$ **do**
3. $C \leftarrow B_i A + C$
4. $A \leftarrow A\alpha^D \bmod p(x)$
5. **end for**
6. **Return** $(C \bmod p(x))$

==========

3 Digit Serial Multiplier Architecture

The digit size multiplication of *A(α)* and *B(α)* over finite fields is an operation more complex than addition and requires 3 steps for its calculation: [4] [9] [10].

- A polynomial multiplication
- A main reduction operation module the irreducible polynomial.
- A final reduction operation module the irreducible polynomial.

Figure 1 shows the architecture of digit serial/parallel multiplier traced from *LSD-First* algorithm in [4]. This architecture is also called single accumulator multiplier (*SAM*) since it uses a polynomial multiplication that is the multiplier core.

These architectures are widely used in hardware implementations for cryptographic applications. As you can see, the entrance polynomials *A* and *B* are 163 bits polynomials. They are introduced to multiplier core where the partial products and sums are computed. This operation is defined as $C = B_i * A + C$. Later, the main reduction $A = A * \alpha^D \bmod p(x)$ occurs, and finally the reduction operation $C \bmod p(x)$ is made. Polynomial multiplication circuit (*multiplier core*) computes the intermediate results (partial additions and products) and stores them in the accumulator *C*.

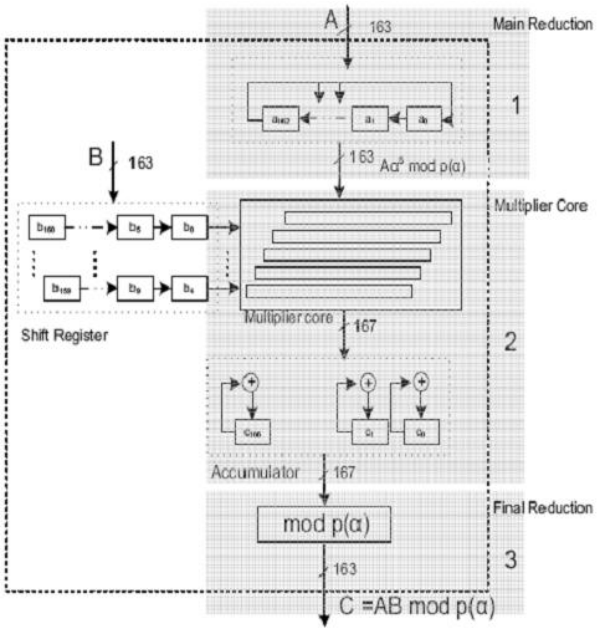

Figure 1. Digit multiplier architecture using a digit size (*D*=5) for *GF* (2^{163}).

In this operation are obtained *m* columns and *D* rows in each clock cycle. Figure 2 shows the structure of the multiplier core (step 3 of the algorithm).

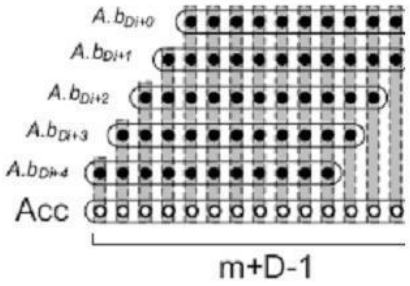

Figure 2. Multiplier core using a digit size (*D*=5) for *GF* (2^{163}).

Function of *main reduction circuit* is to shift *A* left by *D* positions and to reduce the result mod *p*(*x)* (step 4 of the algorithm). The figure 3 shows the structure of main reduction circuit.

The *final reduction circuit* reduce the contents in the accumulator to get the final result *C* (step 6 of the algorithm). Figure 4 shows the structure of final reduction. The figures 2, 3 and 4 denotes an AND gate with a black dot and a XOR gate as a vertical line between two black dots.

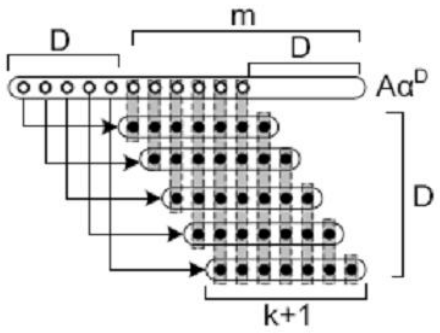

Figure 3. Main reduction circuit using a digit size (*D*=5) for *GF* (2^{163}).

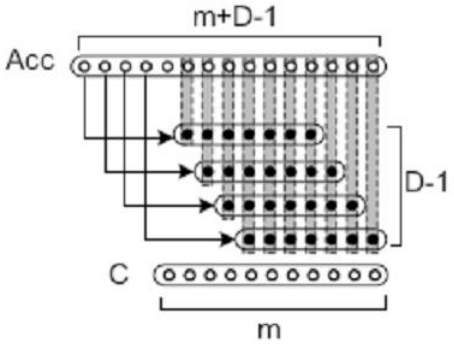

Figure 4. Final reduction circuit using a digit size (*D*=5) for *GF* (2^{163}).

4 Implementation of Digit Serial Multiplier in FPGA

All tests and measurements were made in a prototype card of Digilent that contains a FPGA SPARTAN3: XC3S200-FT256. The FPGA contains 200,000 gates, 960 cell logic blocks (CLBs) and 1,920 slices.

4.1 Previous calculations of space and time complexities

The space complexity based on the number of logic gates is shown in Table 1.

Table 1. Space complexity of the digit multiplier.

Irreducible Polynomial	Area Complexity (Gates)
General	*(m+k)D* XORs + *(m+k+1)D* AND*s*

In table 2 is shown the space requirements for a finite field $GF(2^{239})$ that use the irreducible polynomial $P(x)= x^{239} + x^5 +1$.

Table 2. Space complexity of the digit multiplier that operates in the finite field $GF(2^{239})$ for different values of *D*.

m	***D***	***Space complexity***	
		Gates	**Slices**
239	5	2445	1222
239	10	4890	2445
239	30	14670	7335
239	60	29340	14670

We have determined the time complexity previously making the following considerations:

Frequency of FPGA that is handled by prototype card SPARTAN 3 is 50 *Mhz*.

Then:

f = 50 *Mhz* ; T= 1 / 50 MHz = 0.02 *µseg in one clock cycle*
T_{LSDE} = 0.02 *µseg* * *clock cycles (d= Total digits or clock cycles)*
Where: *d = m/D clock cycles*

Example of time using the digit multiplier over a finite field $GF(2^{239})$.

Digit Multiplier with *m*=239 bits and *D*=20
d=m/D = 12 *digits or clock cycles*
T= 0.02 µseg * 12 = **0.24 *µseg***

Figure 5 shows the time complexity in microseconds using the finite field $GF(2^{239})$ for different values of *D*. The frequency used is 50 *MHz*, with a *T* = 0.02 *µseg.*

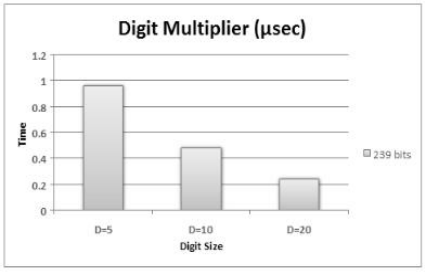

Figure 5. Time complexity in *µseg* for digit serial multiplier for different values of *D*.

4.1 Implementation Results

Figure 6 shows the area complexity (*slices*) reported by the synthesis tool for different values of *D*. We can observed that for *m*=239 bits the space of FPGA is used almost in its totality with a value of *D*=5, that it requires 1,918 slices of a total of 1,920 slices from the FPGA.

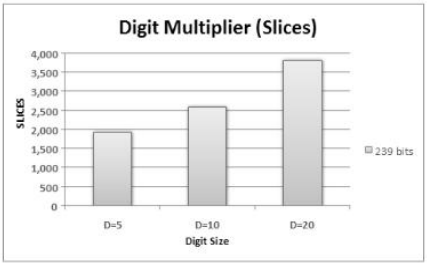

Figure 6. Area complexities (*slices*) for digit serial multiplier in FPGA.

In the figure 7, we presented the time complexity according to the clock cycles.

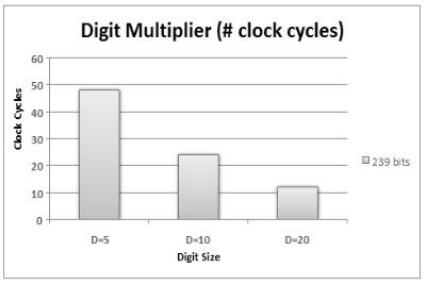

Figure 7. Time complexity of the digit serial multiplier according to the clock cycles for different values in *D*.

4 Comparison with Others Architectures

We compared our multiplier with some multipliers proposed in [7], [9], [10].

Table 3. Time complexity for Software/Hardware implementations reported in [7].

Implementation	sw/hw	*m*	Mult.	Platforms
López (1999)	SW	162	10.5mS	UltraSparc 300 Mhz
Savas (2000)	SW	160	18.3μS	Micro ARM 80 Mhz
Rodríguez (2000)	SW	163	5.4μS	Pentium II 450 Mhz
Rosner (1998)	HW	168	4.5mS	FPGA-XC4052 16 Mhz
Orlando (1999)	HW	167	0.21mS	FPGA-XCV400 76 Mhz
Lee (2000)	HW	192	2.88μS	*Not implemented*
García (2004)	HW	239	3.1 μS	Virtex-300 (75 Mhz)

Table 4. Time complexity for digit serial architectures reported in [9].

m=167	Digit Size	Clock (Mhz)	Montgomery (msec)
	4	85.7	0.55
	8	75.5	0.35
	16	76.7	0.21

Table 5 Time complexities of digit multipliers reported in [4].

Digit Size D=16	Field *m*	Platform	Time (msec)
	155	VLSI 40 Mhz	3.9
	155	Xilinx FPGA XC4020XL, 15 Mhz	18.4
	113	Xilinx FPGA XCV300, 45 Mhz	3.7
	155	VLSI, 66 Mhz	5.7
	167	Xilinx FPGA XCV400E, 76.7 Mhz	0.21

In the presented tables, we can observe a greater efficiency in operation time of our multiplier compared with the results reported in the state of the art. The digit

multiplier that we have presented computes a multiplication in a time of 0.24 μseg using a digit size D=20 for a finite field $GF(2^{239})$.

4 Conclusions

We have presented the implementation in a FPGA Spartan3 of Xilinx, of a digit serial multiplier that operates in the field $GF(2^{239})$ and that uses an irreducible polynomial $P(x) = x^{239} + x^5 + 1$, which are values suggested by the NIST for cryptographic applications of elliptical curves ECC. Has been shown that with the selection of the digit D, can be obtained an efficient implementation in the FPGA considering the time and space complexities that are required for specific applications. A direct application of our multiplier will be the construction of a cryptoprocessor for Elliptic Curve Cryptography, considering that is an important component for several systems that they require of a great performance in speed, area, power consumption and security.

Acknowledgments. We thank the suggestions of anonymous referees that helped us to improve the final presentation of this paper.

4 References

1. Mastrovito, E.D.: VLSI Architectures for Multiplication Over Finite Fields GF(2^m). Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes. Proc. Sixth Int Conf., AAECC-6, New York: Springer-Verlag, Roma, pp. 297-309, July 1988.
2. Omura,J and Massey, J.: Computational Method and Apparatus for Finite Field Arithmetic. U.S. Patent Number 4,587,627, May 1986.
3. Fenn, S.T.J., Benaissa, M., Taylor, D.: GF(2^m) Multiplication and Division Over the Dual Basis. IEEE Trans. Computers, vol.45, no.3, pp. 319-327, March 1996.
4. Song, L., Parhi, K.: Low Energy Digit Serial Parallel Finite Field Multipliers. Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis. 1997.
5. Kumar, K., Wollinger, T.: Optimized Digit Multipliers for Elliptic Curve Cryptography. Communication Security Group (COSY). Ruhr-Universitaet Bochum, Germany, 2005.
6. Paar, C.: A New Architecture for a Parallel Finite Field Multiplier with Low Complexity Based on Composite Fields. IEEE Trans. Computers, vol. 45, No. 7, pp. 856-861, 1996.
7. García-Martínez, M.A.: Construcción de operadores básicos sobre Campos Finitos $GF(2^m)$. PhD Tesis . Cinvestav, IPN. México D.F. December, 2004.
8. Orlando, G.: Efficient Elliptic Curve Processor Architectures for Field Programmable Logic. ECE Dept. Worcester Polytechnic Institute, Germany. 2002
9. Paar, C.: Reconfigurable Hardware in Modern Cryptography. ECE Dept. Worcester Polytechnic Institute, Germany. 2006.
10. Baz, E.: Implementación en Hardware Reconfigurable de Multiplicadores sobre Campos Finitos GF(2^m). Master Tesis. División de Estudios de Postgrado. Instituto Tecnológico de Orizaba. Orizaba Ver. December, 2006.