Cellular Automata Architecture for Elliptic Curve Cryptographic Hardware*

Jun-Cheol Jeon, Kee-Won Kim, Byung-Heon Kang, and Kee-Young Yoo**

Department of Computer Engineeing, Kyungpook National University,
Daegu, 702-701 Korea
{jcjeon33, nirvana, bhkang}@infosec.knu.ac.kr,
yook@knu.ac.kr

Abstract. Elliptic Curve Cryptosystems (ECC) are in the spotlight due to their significantly smaller parameters. The most costly arithmetic operation in ECC is division, which is performed by multiplying the inverse of a multiplicand. On the other hand, Cellular Automata (CA) have attracted a lot of attention regarding their potential for various applications. Thus, this paper presents an EC-based hardware architectural model for division based on CA over Galois Field GF(2^n). The proposed architectural model is highly regular, expandable, and it has reduced latency based on periodic boundary CA. The proposed architecture can be easily implemented into the hardware design of crypto-coprocessors.

1 Introduction

In cryptography, in order to achieve a high level of security, many public-key algorithms that rely on computations in GF(2^n) require a large fields, and some fields need to be as big as GF(2^{2000}). Hence, there is a need to develop an efficient algorithm for the multiplication in GF(2^n). Significantly smaller parameters, however, can be used in ECC than in other competitive systems such RSA and ElGamal, but with an equivalent level of security. The benefits of having smaller key sizes include faster computations and reductions in processing power, storage space, and bandwidth. This makes ECC ideal for constrained environments such as PDAs, cellular phones, and smart cards [1].

ECC was proposed as an alternative to the established public-key cryptosystems of RSA and ElGamal, and it has recently received a great deal of attention in various industries and academia [2, 3]. The main reason for this attention is that there is no sub-exponential algorithm that can solve discrete logarithm problem on a properly chosen elliptical curve. The main operation of ECC is an inverse/division operation, which can be regarded as a special case of exponentiation [4]. However, since a division operation is quite time consuming, efficient algorithms are required for practical applications, especially for a public key cryptosystem where operands can be as large as 512bits or even larger.

* This work was supported by the Brain Korea 21 Project in 2006.
** Corresponding author.

Finite field GF(2^n) arithmetic operations have recently been applied to a variety of fields, including cryptography and error-correcting codes [5]. A number of modern public key cryptography systems and schemes, for example, the Diffie-Hellman key pre-distribution, ElGamal cryptosystem, and ECC, require division and inversion operations [6]. Wang [7] proposed parallel-in parallel-out division architecture with a latency of ($2n^2$-1.5n) and a critical path of (T_{2AND}+ 3T_{2XOR}). Kim's serial-in serial-out architecture [8] has a latency of ($2n^2$-2n) and a critical path of (2T_{2AND}+3T_{2XOR}+ T_{MUX}). However, fast arithmetic architecture is still needed to design dedicated high-speed circuits.

Cellular automata have been used in evolutionary computation for over a decade. They have been used in various applications, such as parallel processing and number theory. CA architecture has been used to design arithmetic computations that Zhang [9] proposed an architecture with programmable cellular automata, Choudhury [10] designed an LSB multiplier based on CA, and Jeon [11] proposed a simple and efficient architecture based on periodic boundary CA.

This paper proposes CA architecture based on EC cryptography for division. We focused on the architecture in ECC, which uses restricted irreducible polynomials, especially, trinomials and pentanomials. The structure has a time complexity of (n^2-n)(T_{2AND}+T_{2XOR}+T_{MUX}) and a hardware complexity of (nAND+(n+2)/(n+6)XOR+ nMUX+4nREGISTER). In addition, our architecture can easily be expanded to be used in other public key cryptosystem with additional (n-2)/(n-6) XOR gates. Our architecture focuses on both area and time complexity.

The rest of this paper is organized as follows. The theoretical background, including finite fields, ECC, and CA, is described in Section 2. Section 3 presents the proposed division architecture based on CA, and we present our discussion, together with a comparison of the performances between the proposed architecture and previous works, in Section 4. Finally, Section 5 presents our conclusion.

2 Preliminary

In this section, we discuss the mathematical background of the finite field and ECC, and the characteristics and properties of cellular automata.

2.1 Finite Fields

A finite field, which is a set of finite elements, can be defined by commutative law, associative law, distributive law and it contains for facilitates for addition, subtraction, multiplication, and division. A number of architectural models have already been developed to construct low complexity bit-serial and bit-parallel multiplications, by using various irreducible polynomials that can reduce the complexity of modular multiplication. Since a polynomial basis operation does not require a basis conversion, it can be readily matched to any input or output system. In addition, due to its regularity and simplicity, the ability to design and expand it into high-order finite fields, with a polynomial basis, is easier to realize than with other basis operations [12].

A finite field can be viewed as a vector space of dimension n over GF(2^n). That is, there exists a set of n elements {1, α, …, α^{n-2}, α^{n-1}} in GF(2^n), such that each

$A \in$ GF(2^n) can be written uniquely in the form $A = \sum A_i \alpha^i$, where $A_i \in \{0,1\}$. This section provides one of the most common bases of GF(2^n) over GF(2), which are polynomial bases [12, 13]. Let $f(x) = x^n + \sum_{i=0}^{n-1} f_i x^i$, where $f_i \in \{0,1\}$, for $i = 0, 1, \ldots,$ n-1, be an irreducible polynomial of degree n over GF(2). For each irreducible polynomial, there exists a polynomial basis representation. In such a representation, each element of GF(2^n) corresponds to a binary polynomial of less than n. That is, for $A \in$ GF(2^n) there exists n numbers $A_i \in \{0,1\}$ such that $A = A_{n-1}\alpha^{n-1} + \ldots + A_1\alpha + A_0$. In many applications, such as cryptography and digital communications, the polynomial basis is still the most widely employed criterion [7, 8, 14]. In the following, we confine our attention to computations that use the polynomial basis.

2.2 Elliptic Curve Cryptosystem

In ECC, computing kP is the most important arithmetic operation, where k is an integer and P is a point on the elliptic curve. This operation can be computed by the addition of two points k times. ECC can be done with at least two types of arithmetic, each of which gives different definitions of multiplication [15]. Two types of arithmetic are, namely, $\mathbf{Z}_p$ arithmetic (modular arithmetic with a large prime p as the modulus) and GF(2^n) arithmetic, which can be done with shifts and exclusive-ors.

We focused on GF(2^n) arithmetic operation. Let GF(2^n) be a finite field by definition. Then, the set of all solutions for equation E: $y^2 + xy = x^3 + a_2x^2 + a_6$, where a_2, $a_6 \in$ GF(2^n), $a_6 \neq 0$, together with the special point called the point at infinity $\boldsymbol{O}$, is a non-supersingular curve over GF(2^n). Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be points in E(GF(2^n)) given in affine coordinates [16]. Assume that P_1, $P_2 \neq \boldsymbol{O}$, and $P_1 \neq -P_2$. The sum $P_3 = (x_3, y_3) = P_1 + P_2$ is computed as follows; if $P_1 \neq P_2$ then $\lambda = (y_1 + y_2)/(x_1 + x_2)$, $x_3 = \lambda^2 + \lambda + x_1 + x_2 + a_2$, $y_3 = (x_1 + x_3)\lambda + x_3 + y_1$, and if $P_1 = P_2$ (called point doubling), then $\lambda = y_1 / x_1 + x_1$, $x_3 = \lambda^2 + \lambda + a_2$, $y_3 = (x_1 + x_3)\lambda + x_3 + y_1$ (see [16]).

In either case, the computation process requires one division, one squaring, and one multiplication. Squaring can be substituted by multiplication. From the point addition operation, it should be noted that no computation processes, except for addition, are performed at the same time due to the data dependency. Therefore, the sharing hardware between the division and multiplication processes is more desirable than the separated implementation of the division and multiplication processes [4, 13].

The additive inverse and multiplicative inverses in GF(2^n) can be calculated efficiently using the extended Euclidean algorithm. Division and subtraction are defined in terms of additive and multiplicative inverses: A-B is A+(-B) in GF(2^n) and A/B is $A\cdot(B^{-1})$ in GF(2^n). Here, the characteristic 2 finite fields GF(2^n) used should have $n \in \{113, 131, 163, 193, 233, 239, 283, 409, 571\}$ [4]. Addition and multiplication in GF(2^n) should be performed by using one of the irreducible binary polynomials of degree n. This restriction is designed to facilitate interoperability while enabling implementers to deploy efficient implementations that are capable of meeting common security requirements [16].

The rule that is used to pick acceptable reduction polynomials is the followings: If a degree n binary irreducible trinomial, $f(x) = x^n + x^k + 1$, for $n > k \geq 1$ exists, then, the irreducible trinomial with the smallest possible k should be used, otherwise the degree n binary irreducible pentanomial can be used, $f(x) = x^n + x^{k3} + x^{k2} + x^{k1} + 1$, for $n > k3 > k2 > k1 \geq 1$, with $k3$ being as small as possible, $k2$ being as small as possible for the given $k3$, and $k1$ as small as possible for the given $k3$ and $k2$, . These polynomials enable the efficient calculations of field operations.

2.3 Cellular Automata

A CA is a collection of simple cells arranged in a regular fashion. CAs can be characterized by four properties: Cellular geometry, neighborhood specifications, the number of states per cell, and the rules that are used to compute to a new state. The next state of a CA depends on the current state and rules [17].

A one-dimensional cellular automaton consists of a linearly connected array of n cells, each of which takes the value of 0 or 1, and a transition function $f(s)$ on the state configuration, s, with q variables. The value of the cell state s_i is updated in parallel, by using this function in discrete time steps as $s_i^{t+1}=f(s^t_{i+j})$ where $-r \leq j \leq r$ [10]. The parameter q is usually an odd integer, i.e. $q = 2r+1$, where r is often named the radius of the function $f(s)$; the possible configurations and the total number of rules for the two state CAs, with the radius r of the neighborhood are 2^q and 2^n where $n = 2^q$. A new value of the ith cell is calculated by using the value of the ith cell itself and the values of the r neighboring cells to the right and left of the ith cell.

Furthermore, if the same rule applies to all the cells in a CA, the CA is called a uniform or regular CA, whereas if different rules apply to different cells, it is called a hybrid CA. In addition, in the structure of CAs, the boundary conditions should be taken into consideration since there exists no left neighbor of the leftmost cell and right neighbor of the rightmost cell, among the cells that compose the CA. According to the conditions, they are divided into three types: Null Boundary CA (NBCA), Periodic Boundary CA (PBCA), and Intermediate Boundary CA (IBCA). In this paper, we only consider PBCA, which is mainly used in this area because of their efficient cyclic properties.

We employ the characteristics of PBCA, which is that the left neighbor of the leftmost cell becomes the rightmost cell and they are adjacent to each other. If the next state is determined by 2-bit shift to the left, it can be expressed as $s_i^{t+1}= s^t_{i-2}$ ($0 \leq i \leq n-1$). This means that the next state of the ith cell is changed by the second right neighbor of the current ith cell. The proposed architecture carries out shift operations and modular reductions using the introduced property.

3 CA Architecture for Division

This section presents an A/B architecture based on cellular automata. Finite field division in GF(2^n) can be performed by using multiplication and inverse processes; that is, $A/B = AB^{-1}$, where A and B are the elements of GF(2^n). Here, the multiplicative inverse of the field element B can be obtained by recursive squaring and multiplication, since the field element B can be expressed as

$$B^{-1} = B^{2^n-2} = (B(B(B \cdots B(B(B)^2)^2 \cdots)^2)^2)^2 \qquad (1)$$

Division also can be easily induced by equation (1).

$$C = AB^{-1} = A(B(B(B\cdots B(B(B)^2)^2\cdots)^2)^2)^2$$

The above equation can be generalized as follows:

$$C_0 = B$$
$$C_i = B(C_{i-1})^2 = B^{2^{i+1}-1}, (1 \le i \le n-2) \tag{2}$$
$$C_{n-1} = A(C_{n-2})^2 = AB^{2^n-2} = AB^{-1} \tag{3}$$

In equation (2), we compute the C_i value for the n-2 clock cycles. Then, we can compute the division results by equation (3). Meanwhile, we assume that $XY = D = (D_{n-1} \ldots D_1\ D_0)$, where X and Y are the field elements; equation (4) is held for a certain k in a reduction trinomial, i.e. $f(x) = x^n + x^k + 1$.

$$D_{n-2}\cdot x^{n-1} + D_{n-3}\cdot x^{n-2} + \ldots + (D_{n-1}\oplus D_{k-1})x^k + \ldots + D_1\cdot x^2 + D_0\cdot x^1 + D_{n-1} \tag{4}$$

In equation (4), shift operations and modular reduction are performed by a given transition rule and reduction trinomial. The operation shown in equation (4) should be performed twice in order to accomplish C_i for n-2 times. Therefore, trinomials and pentanomials based multiplications are computed as the next equations respectively.

$$D_{n-3}\cdot x^{n-1} + D_{n-4}\cdot x^{n-2} + \ldots + \gamma x^{k+1} + \delta x^k + \ldots + D_0\cdot x^2 + D_{n-1}\cdot x^1 + D_{n-2}, \tag{5}$$

where $\gamma = D_{n-1}\oplus D_{k-1}$ and $\delta = D_{n-2}\oplus D_{k-2}$

$$D_{n-3}\cdot x^{n-1} + D_{n-4}\cdot x^{n-2} + \ldots + \zeta x^{k3+1} + \eta x^{k3} + \ldots + \zeta' x^{k2+1} + \eta' x^{k2} + \ldots + \zeta'' x^{k1+1} + \eta'' x^{k1} + \ldots + D_0\cdot x^2 + D_{n-1}\cdot x^1 + D_{n-2}, \tag{6}$$

where $\zeta = D_{n-1}\oplus D_{k3-1}$, $\zeta' = D_{n-1}\oplus D_{k2-1}$, $\zeta'' = D_{n-1}\oplus D_{k1-1}$, $\eta = D_{n-2}\oplus D_{k3-2}$, $\eta' = D_{n-2}\oplus D_{k2-2}$, and $\eta'' = D_{n-2}\oplus D_{k1-2}$.

The architectural model based on the above equations (5) and (6) are shown in Fig. 1 and Fig. 2. In terms of the CA concept, both architectural models are based on 5-neighborhood and $s_i^{t+1} = s_{i-2}^t$ ($0 \le i \le n-1$), which is also expressed as rule 2863311530.

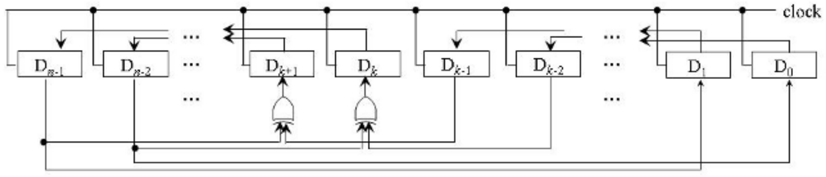

Fig. 1. Multiplication architecture based on irreducible trinomials

In order to satisfy equation (3), A should be multiplied by the square of C_{n-2}. The result is $C = A\cdot B^{-1}$ and when $A = 1$, the algorithm realizes the inverse operation B^{-1}. Fig. 3 shows the division architecture. Each initial value is such that cellular automata have all zeros ($C_i^r = 0$, $0 \le i \le n-1$); the B register and shift register have B_i values ($B_i^r = B_{n-1} \ldots B_2 B_1 B_0$).

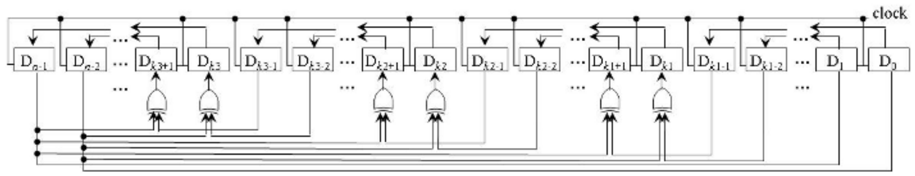

Fig. 2. Multiplication architecture based on irreducible pentanomials

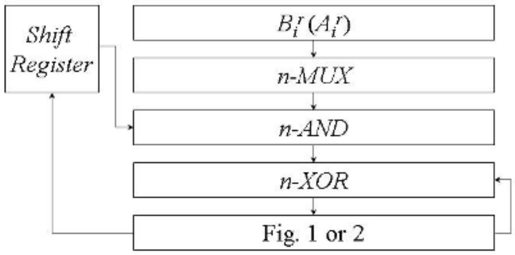

Fig. 3. Division architecture based on Fig. 1 and Fig. 2

For initial n clock cycles, the values in the shift register are sequentially broadcast to the connected line in order to multiply the values in B_i^r, and C_i^r is evolved based on PBCA. After the evolution of C_i^r, the values are transferred to the shift register, just as the initialized form in the shift register. $(C_{n-2})^2$ is computed according to the process described by n-2 times.

At the end of the transition, the system chooses A_i values in the B register for the final resultant values. It is possible to perform A/B division in $n(n-1)$ clock cycles by using n AND gates, $n+2/n+6$ XOR gates, n Muxes, and $4n$ bits registers. In addition, extra equipment, such as the control signals for transferring results in cellular automata to the shift register and the changing of the values in the B register, right after deriving the values of B^{-1}.

Moreover, our architecture can be easily applied to other public key cryptosystems by using general irreducible polynomials. In Fig. 1 and 2, by using additional $n-2/n-6$ XOR gates, the proposed architecture can perform a general division operation. Although the architecture is used as a general divider, it has the same latency as in Fig. 3 because of the parallelization.

4 Comparison and Analysis

A comparison of the proposed CA architecture for EC-based cryptography, with existing structures, was performed focusing on time and hardware complexity issues. As such, Wang's [7] and Kim's [8] division architectural methods were chosen.

Wang proposed a parallel-in parallel-out A/B architectural model, which has a latency of $2n^2$-$1.5n$ and a critical path of $(T_{2AND}+ 3T_{2XOR})$ over GF(2^n). Kim proposed a serial-in serial-out A/B architectural model, which has a latency of $2n^2$-$2n$ and critical path of $(2T_{2AND}+ 3T_{2XOR}+ T_{MUX})$ over GF(2^n).

Table 1. A comparison of the performance of the *A*/*B* Circuits

Item	Wang et al. [7]	Kim et al. [8]	Fig. 3	
Irreducible polynomial	general	general	Trinomial/ pentanomial	general
Critical path	$T_{2AND}+3T_{2XOR}$	$2T_{2AND}+3T_{2XOR}+T_{MUX}$	$T_{2AND}+T_{2XOR}+T_{MUX}$	$T_{2AND}+T_{2XOR}+T_{MUX}$
Latency	$2n^2$-1.5n	$2n^2$-2n	$2n^2$-n	$2n^2$-n
Hardware Complexity -Registers(R) -Latch(L) -Inverter(I)	$3n^3$ -$3n^2$ AND $3n^3$ -$3n^2$ XOR 8.5n^3 -8.5n^2 (L)	$4n^2$-7n+3 AND $3n^2$-5n+2 XOR $14n^2$-22n+8 (L) n^2-2n+1Mux $3n^2$-6n+2 (I)	n AND n+2/n+6 XOR 4n (R) n Mux	n AND 2n XOR 4n (R) n Mux
I/O format	Parallel-in parallel-out	Serial-in serial-out	Serial-in parallel-out	

In general, parallel architectural models need much more hardware equipment than serial fashion architectural models, and the issue of latency is reversed. The proposed architectural model, however, has less complexity than the serial or parallel fashion architectural methods, with respect to both space and time. Our architectural model has been constructed based on ECC. However, our architecture can be easily applied to other public cryptosystems with additional n-2/n-6 XOR gates, while the existing systolic architectural models including those of Wang's and Kim's, hardly reduce the level of complexity, although they apply the restrict polynomials.

5 Conclusion

This paper has presented CA architecture in order to compute *A*/*B* modulo irreducible trinomials and pentanomials, which are restricted in the Certicom Standard for ECC. We have proposed a simple CA hardware architectural model that is the most costly arithmetic operation scheme in ECC over GF(2^n). The proposed architectural model includes the characteristics of both an evolutionary PBCA and the restricted polynomials, and it has minimized both time and hardware complexity concerns. Moreover, we have shown that our architectural model can be easily applied to general division architectural method with no additional latency needed. Our CA architecture has a regularity and modularity. Accordingly, it can be used as a basic architecture not only for ECC, but also for other public key cryptosystems.

References

1. I. Lopez and R. Dahab, An overview of Elliptic Curve Cryptography, University of Campinas Press, Brazil (2000)
2. N. Koblitz, Elliptic curve cryptosystems, Mathematics of Computation, Vol. 48. (1987) 203-209
3. V. Miller, Use of Elliptic Curves in Cryptography, Advances in Cryptology-CRYPTO'85, Springer-Verlog Lecture Notes in Computer Science, Vol. 218. (1986)

4. A. J. Menezes, Elliptic Curve Public Key Cryptosystems, Boston, MA: Kluwer Academic Publishers (1993)
5. T. R. N. Rao and E. Fujiwara, Error-Control Coding for Computer Systems, Englewood Cliffs, NJ: Prentice-Hall (1989)
6. W. Drescher, K. Bachmann, and G. Fettweis, "VLSI Architecture for Non Sequential Inversion over GF(2^m) using the Euclidean Algorithm," The International Conference on Signal Processing Applications and Technology, Vol. 2. (1997) 1815-1819
7. C. L. Wang and J. H. Guo, "New Systolic Arrays for C+AB^2, inversion, and division in GF(2^m)", IEEE Trans. on Computer, Vol. 49, No. 10. (2000) 1120-1125
8. N. Y. Kim and K. Y. Yoo, "Systolic architecture for inversion/division using AB^2 circuits in GF(2^m)", Integration, the VLSI journal, Vol. 35. (2003) 11-24
9. C. N. Zhang, M. Y. Deng, and R. Mason, "A VLSI Programmable Cellular Automata Array for Multiplication in GF(2^n)," PDPTA '99 International Conference (1999)
10. P. Pal. Choudhury and R. Barua, "Cellular Automata Based VLSI Architecture for Computing Multiplication and Inverses in GF(2^m)," IEEE 7th International Conference on VLSI Design (1994) 279-282
11. Jun-Cheol Jeon and Kee-Young Yoo, "An Evolutionary Approach to the Design of Cellular Automata Architecture for Multiplication in Elliptic Curve Cryptography over Finite Fields," Lecture Notes in Artificial Intelligence PRICAI 2004: Trends in Artificial Intelligence (LNAI 3157), Springer-Verlag, Vol. 3157. (2004) 241-250
12. A. J. Menezs, Applications of Finite Fields, Boston, MA: Kluwer Academic Publishers (1993)
13. IEEE P1363, Standard Specifications for Public Key Cryptography (2000)
14. S. W. Wei, "VLSI architecture of divider for finite field GF(2^m)", IEEE International Symposium on Circuit and Systems, Vol. 2. (1998) 482-485
15. C. Kaufman, R. Perlman, and M. Speciner, Network Security private communication in a public world, New Jersey: Prentice Hall (2002)
16. SEC 1: Elliptic Curve Cryptography version 1.0, Certicom Reserch (2000)
17. J. Von Neumann, The theory of self-reproducing automata, University of Illinois Press, Urbana and London (1966)