GF(2^m)상에서 셀룰러 오토마타를 이용한 새로운 AB^2 연산기 설계[1)]

하경주*, 구교민**, 김현성***, 이형목***, 전준철***, 유기영***

*경산대학교, **대구교육대학교, ***경북대학교

Design of a New AB^2 Multiplier over GF(2^m) using Cellular Automata

Kyeoung-Ju Ha*, Kyo-Min Ku**, Hyun-Sung Kim***, Hyoung-Mok Lee***, Jun-Cheol Jeon***, Kee-Young Yoo***

*Dept. of Information processing, Kyungsan Univ. **Daegu Nationl Univ. of Education ***Dept. of Computer Engineering, Kyungpook Nationl Univ.

요 약

본 논문에서는 셀룰러 오토마타를 이용하여, GF(2^m)상에서 AB^2 연산을 m 클럭 사이클만에 처리할 수 있는 새로운 연산기를 설계하였다. 이는 대부분의 공개키 암호화 시스템에서의 기본 연산인 유한 필드 상의 모듈러 지수(modular exponetiation) 연산기 설계에 효율적으로 이용될 수 있다. 또한 셀룰러 오토마타는 간단하고도 규칙적이며, 모듈화 하기 쉽고 계층화 하기 쉬운 구조이므로 VLSI 구현에도 효율적으로 활용될 수 있다.

I. 서론

최근 인터넷의 급속한 확산과 더불어 인터넷 상에서의 안전한 정보전송을 위해 정보보호에 대한 필요성이 대두되어 여러가지 보안기술이 개발되고 있는 실정이다. 따라서 정보보호의 핵심 기술이라 할 수 있는 암호화 시스템의 구현의 중요성이 점점 더 크게 부각되고 있다. 지난 30여년간 암호화 시스템 등 여러 분야에서 유한 필드에 대한 연구가 이루어 졌으며[1], Diffie-Hellman key exchange, ElGamal과 같은 대부분의 공개키 암호화 시스템에서는 유한 필드 상의 모듈러 지수(modular exponetiation)연산을 기본으로 하고 있다[2][3]. 이러한 모듈러 지수 연산기에서는 이의 구현을 위해 모듈러 곱셈(modular multiplication) 연산기를 기본 구조로서 사용하고 있다. 또한 타원 곡선 암호화 시스템에서는 정수배의 곱셈 연산을 기본으로 하고 있다[4]. 곱셈기를 구현하기 위한 알고리즘으로는 LSB 곱셈 알고리즘[5]과 MSB 곱셈 알고리즘[6] 및 몽고메리(montgomery) 알고리즘[7] 등이 있다.

본 논문에서는 셀룰러 오토마타(CA:Cellular Automata)를 이용하여[8][9], GF(2^m)상에서 효과적인 지수승 계산을 위하여 AB^2 연산을 빠르게 처리할 수 있는 새로운 구조를 제안한다. 지금까지 AB^2 연산을 위해 개발된 연구 결과들은 대부분이 시스톨릭 구조상에서 연구되었으며, CA를 이용한 AB^2 연산은 본 논문에서 처음으로 시도되었다. 지금까지 연구된 결과들은 다음과 같다.

먼저 시스톨릭 구조 상에서는 $m \times m$ 셀을 사용하여 $3m$ 클럭 사이클에 AB^2 연산을 하고 있으며[10], $(m \times m)/2$ 셀을 사용하여 $2m+m/2$ 클럭 사이클에 AB^2 연산을 하고 연산을 하고 있다[11].

본 논문에서는 처음으로 CA를 이용하여 m개의 셀을 사용하여 m 클럭 사이클만에 AB^2 연산을 수행하는 연산기를 설계함으로써 지금까지의 연구 결과보다 시간적으로나 공간적으로 월등히 개선된 구조를 제시하였다.

1) This work was supported by grant No. 2000-2-51200-001-2 from the Basic Research Program of the Korea Science & Engineering Foundation.

II. 셀룰러 오토마타

CA는 규칙적으로 상호 연결된 많은 셀들로 구성되어져 있는 유한 상태 머신(finite state machine)이다[8,9]. 각 셀들의 다음 상태는 각 셀들과 연결된 이웃의 현재 상태 값에 따라 달라지게 된다. 다음은 2-상태, 3-이웃 1-차원의 CA의 법칙에 대한 한 예이다.

이웃의 상태 :	111	110	101	100	011	010	001	000
다음 상태 :	0	1	0	1	1	0	1	0

(법칙 90)

법칙 90을 살펴보면, 자신의 왼쪽 이웃과 오른쪽 이웃의 상태 값을 XOR 하여 그 결과값으로 자신의 상태를 갱신하고 있음을 알 수 있다.

n개의 셀을 가지는 CA의 현재 상태는 n-벡터 $x=(x_0,x_1, \ldots ,x_{n-1})$로 나타낼 수 있다. 여기서 x_i는 셀 i의 값이며, x_i는 GF(2)의 원소이다. 선형 CA에서 다음의 상태는 특성 행렬과 현재 상태의 벡터를 곱함으로써 결정될 수 있는데, 이 때 특성 행렬은 CA의 전체적인 법칙을 나타낸다. x_t를 시간 t에서의 CA의 상태라 하고(하나의 열벡터로 간주), 특성 행렬을 T라하면, 시간 t+1에서의 CA의 상태는 다음과 같이 표현된다.

$$x^{t+1} = T \cdot x^t$$

여기서의 산술 연산은 GF(2) 상에서 일어난다.

법칙 90에 대한 특성 행렬을 구해보면 다음과 같다.

$$\begin{bmatrix} 0 & 1 & 0 & 0 & \ldots & 0 & 0 \\ 1 & 0 & 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & 0 & 1 & \ldots & 0 & 0 \\ 0 & 0 & 1 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & 1 & \ldots & 0 & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & 0 & \ldots & 1 & 0 \end{bmatrix}$$

위의 예에서 i번째 행 j번째 열에 있는 행렬의 요소 1은 i번째 셀이 j번째 셀과 이웃에 대한 의존도가 있음을 나타낸다.

III. GF(2^m)상에서 지수승 연산을 위한 알고리즘

본 장에서는 GF(2^m) 상에서 $M(x)^E \bmod P(x)$를 구하는 일반적인 알고리즘에 대해 설명한다.

$A(x)$와 $B(x)$를 GF(2^m) 상의 한 원소라 하자. 그리고 $P(x)$는 차수 m의 원시 기약 다항식(primitive irreducible polynomial)이라 하자[1]. 그러면 세 개의 다항식 $A(x)$, $B(x)$, $P(x)$는 다음과 같다.

$$A(x)=a_{m-1}x^{m-1}+\ldots+a_1x^1+a_0 \quad (1)$$
$$B(x)=b_{m-1}x^{m-1}+\ldots+b_1x^1+b_0 \quad (2)$$
$$P(x)=x^m+p_{m-1}x^{m-1}+\ldots+p_1x^1+p_0 \quad (3)$$

먼저 $M(x)^E \bmod P(x)$의 계산은 지수 E(= [e_{m-1}, e_{m-2}, ... ,e_1, e_0])의 처리 방식에 따라 LSB 우선 방법과 MSB 우선 방법으로 나뉘어 지는데, 연산의 방법은 다음과 같다.

LSB 우선 지수승 : e_0 부터 e_{m-1} 순으로 연산

$$M^E = M^{e_0}(M^2)^{e_1}(M^4)^{e_2}\ldots(M^{2^{m-1}})^{e_{m-1}}$$

MSB 우선 지수승 : e_{m-1} 부터 e_0 순으로 연산

$$M^E = (..((M^{e_{m-1}})^2 M^{e_{m-2}})^2 \ldots M^{e_1})^2 M^{e_0}$$

본 장에서는 MSB 우선 곱셈 방법에 대해 일반적인 알고리즘[13]에 대해 살펴 보고, 다음 장에서 이의 계산을 효율적으로 할 수 있는 CA를 이용한 AB2 연산기 구조를 제시한다.

알고리즘 1:MSB 우선 곱셈에 의한 지수승 알고리즘
입력 : $A(x)$, $B(x)$, $P(x)$
출력 : $C(x)=A(x)^E \bmod P(x)$
단계 1 : if e_{m-1}==1 $C(x)=A(x)$ else $C(x)=\alpha^0$
단계 2 : for $i=m-2$ to 0
단계 3 : if e_i ==1 $C(x)=A(x)C(x)^2 \bmod P(x)$
else $C(x)=\alpha^0 C(x)^2 \bmod P(x)$

알고리즘 1을 구현하기 위한 일반적인 방법은 단계 3에서 사용되는 AB^2의 연산기를 설계하여 지수승 연산 구현에 이용하는 것이다. 본 논문의 다음장에서는 이러한 AB^2의 곱셈 연산기를 CA를 이용하여 m-클럭 사이클에 구현하는 구조를 제시한다.

IV. CA를 이용한 AB^2 연산기

본 장에서는 GF(2^m) 상에서 $M(x)^E \bmod P(x)$를 MSB 우선 방식으로 구함에 있어 필수적인 요소인 AB^2(알고리즘 1의 단계 3)의 연산을 CA를 이용하여 빠른 시간에 계산 할 수 있는 구조를 제시하고자 한다.

먼저 III장의 식(1), (2), (3) 으로부터 다음과 같은 식을 유도할 수 있다.

$$B(x)^2=b_{m-1}x^{2m-2}+b_{m-2}x^{2m-4}+\ldots+b_1x^2+b_0 \quad (4)$$

$$\begin{aligned} &A(x)B(x)^2 \bmod P(x) \\ &=A(x)(b_{m-1}x^{2m-2}+b_{m-2}x^{2m-4}+\ldots+b_1x^2+b_0) \bmod P(x) \\ &=(A(x)b_{m-1}x^{2m-2}+A(x)b_{m-2}x^{2m-4}+\ldots+A(x)b_1x^2+A(x)b_0) \bmod P(x) \\ &=[A(x)b_{m-1}x^{2m-4}+A(x)b_{m-2}x^{2m-6}+\ldots+A(x)b_1]x^2 \bmod P(x) + A(x)b_0 \\ &=\{\ldots[A(x)b_{m-1}x^2 \bmod P(x) + A(x)b_{m-2}]x^2 \bmod P(x) + \ldots + A(x)b_1\}x^2 \\ &\bmod P(x) + A(x)b_0 \end{aligned} \quad (5)$$

위의 식(5)를 구현하기 위한 구체적인 알고리즘은 다음과 같다.

알고리즘 2 . AB^2 연산 알고리즘
입력 : $A(x)$, $B(x)$, $P(x)$
출력 : $A(x)B(x)^2 \bmod P(x)$
단계 1 : $M(x)=0$;
단계 2 : for $i= m-1$ to 0
단계 3 : $M(x)=M(x)\alpha^2 \bmod P(x) + A(x)b_i$

알고리즘 2의 단계 3에서 $M(x)^2 \bmod P(x)$ 연산과 $A(x)b_i$ 연산은 동시에 수행될 수 있다. 이를 구현하기 위한 기본 적인 연산들은 다음과 같다.

연산 1 : $M(x)\alpha^2$ 연산
연산 2 : modular reduction 연산
연산 3 : $A(x)b_i$ 연산

우선 연산 1을 수행하기 위하여 m개의 셀을 가지는 초기값을 0으로 가지는 CA를 이용한다. 그런데 여기서 연산 1은 2을 구현하기 위해 두 비트를 왼쪽으로 쉬프트하여야한다. 이를 위해 m셀 CA에서 각 셀의 다음 상태는 자신의 두번째 오른쪽 이웃의 이전 상태로 정의한다. 이러한 CA를 나타내는 $m \times m$ 특성 행렬 B는 다음과 같이 나타낼 수 있다.

$$B=\begin{bmatrix} 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 \end{bmatrix}$$

특성 행렬 B를 가지는 CA의 구조도는 그림 1과 같다.

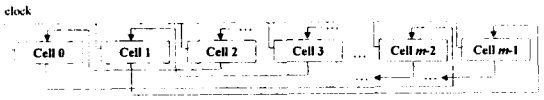

그림 1 : 특성행렬 B를 가지는 CA 구조도

연산 2(modular reduction)를 수행하기 위해서는 연산 1의 결과 두 비트가 왼쪽으로 쉬프트 되므로 두 번의 모듈러 reduction 연산이 필요하다.

연산1의 결과 CA로 부터 두 비트 왼쪽으로 쉬프트 된 결과값이 나오므로, 두 번의 modular reduction은 그림 2와 같이 구현될 수 있다.

연산 3을 수행하기 위해서는 $i(0 \le i \le m-1)$번째 클럭에 $A(x)$의 각 원소와 b_{m-i-1}원소를 곱해야 하므로 이는 m개의 AND 게이트를 이용하여 쉽게 구할 수 있다. 이에 대한 구조도는 그림 3에 나타나 있다. 단 여기서는 0 번째 클럭에서의 연산을 보여주고 있다.

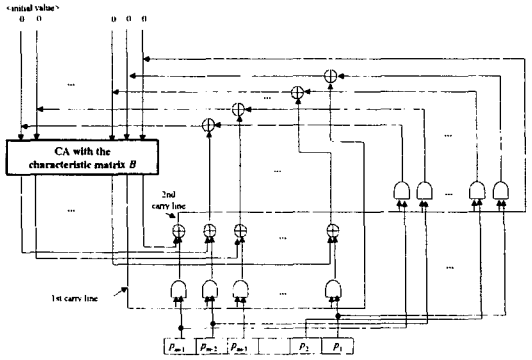

그림 2 : 연산2를 위한 구조도

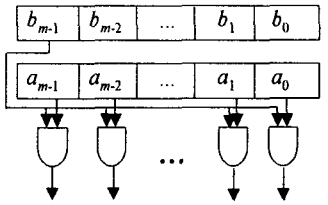

그림 3 : 연산 3을 위한 구조도

이제 연산 1, 연산 2 그리고 연산 3을 이용하여 알고리즘 2의 방법으로 AB^2을 구하는 방법에 대해 알아보자. 연산 2와 연산 3은 동시에 수행되므로 전체적인 구조도는 그림 4와 같다. 단 여기서는 0 번째 클럭에서의 연산을 보여주고 있으며, 각 레지스터 및 CA의 초기값은 다음과 같다.

- PBCA 초기값: all 0
- $A(x)$ 레지스터 초기값 : a_{m-1} ... a_2 a_1 a_0
- $B(x)$ 레지스터 초기값 : b_{m-1} ... b_2 b_1 b_0
- $P(x)$ 레지스터 초기값 : p_{m-1} ... p_2 p_1

V. 결론

본 논문에서는 CA를 이용하여, GF(2^m)상에서 효과적인 지수승 계산을 위하여 AB^2 연산을 빠르게 처리할 수 있는 구조를 제안한다.

본 논문에서 제시된 구조를 사용하면, m개의 셀과 $3m-2$ AND 게이트 그리고 m 2-input XOR 게이트, $m-1$ 3-input XOR 게이트, 3개의 레지스터를 사용하여 m 클럭 사이클만에 AB^2 연산을 수행할 수 있다.

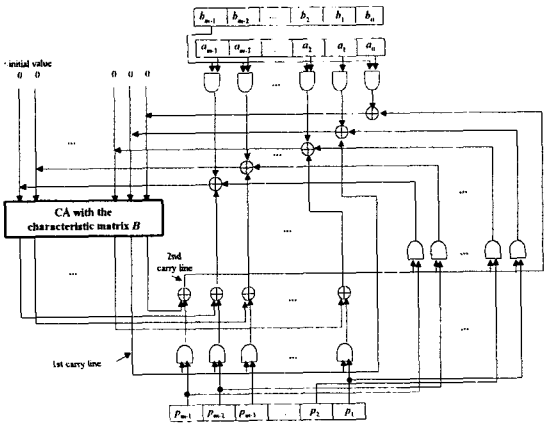

그림 4 : AB^2 연산을 위한 구조도

이는 MSB 우선 방식으로 AB^2 연산을 함에 있어 2비트 쉬프트 연산을 CA를 이용하여 1 클럭에 구현함으로써 가능하게 되었다.

본 논문의 결과를 지금까지의 연구결과와 비교해 보면 표 1과 같다.

표 1 : 지금까지의 연구결과 비교

구조	Systolic array		본 논문
	Wei in [10]	Wang in [11]	
연산	AB^2	AB^2	AB^2
셀의 개수	$m \times m$	$(m \times m)/2$	m
AND 게이트 수	$3m \times m$	$6(m \times m)/2$	$3m-2$
XOR 게이트 수	$3m \times m$	$6(m \times m)/2$	2-input : m 3-input : $m-1$
래치 수	$11m \times m$	$17(m \times m)/2$	0
MUX 수	0	0	0
제어 신호 수	0	0	0
레지스터 수	0	0	3
실행시간	$3m$	$2m+m/2$	m

표 1을 분석해 보면 시스톨릭 구조[10,11]과 비교시 셀의 개수, 게이트의 수, 수행시간 면에서 본 논문에서 제시한 구조가 월등히 좋은 결과를 보여주고 있음을 알 수 있다.본 논문에서 제시된 AB^2 연산기를 기반으로 하면 나눗셈, 지수 연산, 곱셈의 역원기 등을 효율적으로 구현할 수 있다.

참고문헌

[1] R.J. McEliece, Finite Fields for Computer Scientists and Engineerings, New York:Kluwer Academic, 1987

[2] W.Diffie and M.E.Hellman, New directions in cryptography, IEEE Trans. on Info. Theory, vol. 22, pp.644-654, Nov. 1976.

[3] T.ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms, IEEE Trans. on Info. Theory, vol. 31(4). pp. 469-472, July 1985.

[4] A.J. Menezes, Elliptic Curve Public Key Cryptosystems, Kluwer Academic Publishers, 1993.

[5] C.S. YEH, IRVING S. REED, T.K. TRUONG, Systolic Multipliers for Finite Fields GF(2m), IEEE TRANSACTIONS ON COMPUTERS, VOL. C-33, NO. 4, pp. 357-360, April 1984.

[6] C.L. Wang, J.L. Lin, Systolic Array Implementation of Multipliers for Finite Fields GF(2m), IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, VOL. 38, NO. 7, pp. 796-800, July 1991.

[7] P.L. Montgomery, Modular multiplication without trial division, Mathematics of Computation, 44(170):519-521, April, 1985.

[8] M. Delorme, J. Mazoyer, Cellular Automata, KLUWER ACADEMIC PUBLISHERS 1999.

[9] STEPHEN WOLFRAM, Cellular Automata and Complexity, Addison-Wesly Publishing Company, 1994.

[10] S.W.Wei, VLSI architectures for computing exponentiations, multiplicative inverses, and divisions in GF(2^m), IEEE Trans. On Circuit & System -II: Analog and Digital Signal Processing, vol. 44, NO. 10, pp. 847-855, Oct. 1997.

[11] C.L.Wang, and J.H.Guo, New Systolic Arrays for C+AB^2, inversion, and division in GF(2m), IEEE Trans. On Computer, vol. 49, NO. 10, Oct. 2000.

[12] A.J.Menezes, Applications of Finite Fields, Kluwer Academic, 1993.

[13] Knuth, THE ART OF COMPUTER PROGRAMMING, Vol. 2/Seminumerical Algorithms, ADDISON-WESLEY, 1969.