유한필드상에서 몽고메리 알고리즘을 이용한 곱셈기 설계
(New Multiplier using Montgomery Algorithm over Finite Fields)

하 경 주*, 이 창 순*
(Kyung-Ju Ha, Chang-Soon Lee)

요약 유한 필드 GF(2^m) 상에서의 곱셈은 Diffie-Hellman key exchange, ElGamal 과 같은 공개키 암호시스템에서의 기본적인 연산이다. 본 논문에서는 셀룰러 오토마타를 이용하여 GF(2^m) 상에서 몽고메리 곱셈을 m 클럭 사이클만에 처리하는 새로운 구조를 제시하였다. 본 논문에서 제시된 몽고메리 곱셈기는 모듈러 지수기, 나눗셈기, 곱셈의 역원기등을 효율적으로 구현하는데 활용될 수 있다. 또한 셀룰러 오토마타는 간단하고도 규칙적이며, 모듈화 하기 쉽고 계층화 하기 쉬운 구조이므로 VLSI 구현에도 효율적으로 활용될 수 있다.

Abstract Multiplication in Galois Field GF(2^m) is a primary operation for many applications, particularly for public key cryptography such as Diffie-Hellman key exchange, ElGamal. The current paper presents a new architecture that can process Montgomery multiplication over GF(2^m) in m clock cycles based on cellular automata. It is possible to implement the modular exponentiation, division, inversion [1]architecture, etc. efficiently based on the Montgomery multiplication proposed in this paper. Since cellular automata architecture is simple, regular, modular and cascadable, it can be utilized efficiently for the implementation of VLSI.

1. 서론

현대 사회가 점차 고도 정보화 사회로 발전되어 감에 따라, 음성, 화상, 데이터 등 다양한 종류의 정보를 교환하고 저장하는 대량 정보통신 시스템이 구축되어가고 있다. 이러한 시스템이 사회 전반에 걸쳐 일반화되기 위해서는 시스템의 신뢰성과 안전성이 필수 불가결한 요소이며, 특히 시스템 내부 또는 각 시스템 상호간 통신에서의 각종 정보에 대한 보안기술은 그 거래를 가능하게 하는 장점을 가져다 준 반면 정보보호의 측면에서는 개인 정보 유출과 도용 등의 불안을 안고 있다.

이와 같이 고도 정보화 사회의 필수 불가결한 요소인 정보보호에 대한 필요성으로 여러 가지 보안기술이 개발되고 있으며, 정보보호의 핵심 기술이라 할 수 있는 암호

* 경산대학교 정보과학부
* 경산대학교 정보과학부

시스템 구현의 중요성이 점점 더 크게 부각되고 있다.

공개키 암호 시스템의 암호 알고리즘을 이루고 있는 기본 연산은 유한 필드 상에서의 지수승(exponentiation) 연산이며, 지수승 연산의 기본은 모듈러 곱셈연산이다. 곱셈기를 구현하기 위한 알고리즘으로는 LSB 우선 곱셈 알고리즘[1]과 MSB 우선 곱셈 알고리즘[2] 및 몽고메리(montgomery) 알고리즘[3] 등이 있다.

본 논문에서는 몽고메리 알고리즘을 이용하여 지수기의 기본 연산이 되는 곱셈기를 설계한다. 지금까지 몽고메리 곱셈기는 주로 시스톨릭 구조상에서 연구되어져 왔는데, 본 논문에서는 셀룰러 오토마타를 이용하여 빠른 시간에 효율적으로 몽고메리 곱셈을 수행할 수 있는 구조를 제시한다.

지금까지 시스톨릭 구조 상에서는 $GF(2^m)$상에서 $3m$시간에 몽고메리 곱셈을 수행하였으며[6], 본 논문에서는 셀룰러 오토마타를 이용하여 m 시간에 몽고메리 곱셈을 수행하였다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 본 논문에서 사용하는 있는 셀룰러 오토마타에 대해 알아보고, 3장에서는 셀룰러 오토마타를 이용한 몽고메리 곱셈기를 제시한다. 4장에서는 본 논문에서 제시한 구조와 지금까지의 연구결과의 성능을 비교분석하며, 5장에서 결론을 맺는다.

2. 셀룰러 오토마타

셀룰러 오토마타는 규칙적으로 상호 연결된 많은 셀들로 구성되어져 있는 유한 상태 머신(finite state machine)이다[4, 5]. 각 셀들의 다음 상태는 각 셀들과 연결된 이웃의 현재 상태 값에 따라 달라지게 된다. 이와 같이 셀룰러 오토마타는 상태의 변화에 관여하는 이웃의 수와 이들 이웃을 이용하여 상태를 갱신하기 위해 사용되는 함수인 법칙으로 구성된다. 이 때 이웃이란 자기 자신을 포함하여 셀의 상태 갱신에 직접적으로 영향을 미칠 수 있는 셀을 의미한다. 다음은 2-상태, 3-이웃 1-차원 셀룰러 오토마타의 두 가지 법칙에 대한 예이다.

이웃상태	111	110	101	100	011	010	001	000	
다음상태	0	1	0	1	1	0	1	0	법칙90
다음상태	1	0	0	1	0	1	1	0	법칙150

여기서 이웃의 상태는 시간 t에 3개의 이웃이 가질 수 있는 가능한 8개의 상태이며, 이를 나타내는 3개의 비트 중 가운데 비트가 자신의 상태를 나타내고, 이의 왼쪽, 오른쪽 비트는 각각 왼쪽, 오른쪽 이웃의 상태를 나타낸다. 법칙 90과 법칙 150은 시간 t+1에 i번째 셀이 가지는 상태를 나타내고 있다. 여기서 90과 150의 의미는 다음 상태 8비트를 십진수로 나타낸 수이다. 법칙 90을 살펴보면, 자신의 왼쪽 이웃과 오른쪽 이웃의 상태 값을 XOR 하여 그 결과 값으로 자신의 상태를 갱신하고 있으며, 법칙 150은 자신의 왼쪽, 오른쪽 이웃 그리고 자신의 상태 값을 XOR 한 결과값을 자신의 다음 상태로 갱신하고 있음을 알 수 있다. 따라서 $q_i(t)$를 시간 t에서 i번째 셀의 상태 값이라 했을 때, 법칙 90과 150은 다음과 같은 수식으로 표현될 수 있다.

법칙 90 :

$$q_i(t+1) = q_{i-1}(t) \oplus q_{i+1}(t)$$

법칙 150 :

$$q_i(t+1) = q_{i-1}(t) \oplus q_i(t) \oplus q_{i+1}(t)$$

여기서 $\oplus$은 XOR 연산을 나타내며, q_{i-1}은 q_i의 왼쪽 이웃을 q_{i+1}은 q_i의 오른쪽 이웃을 나타낸다.

셀룰러 오토마타의 구조에서 경계 조건(boundary condition)을 고려해야하는데, 본 논문에서는 가장 왼쪽 셀의 왼쪽 이웃과 가장 오른쪽 셀의 오른쪽 이웃을 모두 0으로 간주하는 NBCA(Null Boundary CA)를 사용한다.

3. $GF(2^m)$상에서 셀룰러 오토마타를 이용한 몽고메리 곱셈기 설계

3.1 $GF(2^m)$상에서 몽고메리 곱셈을 위한 알고리즘

본 절에서는 GF(2^m)상에서 일반적인 몽고메리 곱셈을 위한 알고리즘에 대해 살펴본다[7]. 몽고메리 곱셈 알고리즘에서는 GF(2^m)상에서 $a(x)b(x)$ mod $n(x)$ 대신 $a(x)b(x)r(x)^{-1}$ mod $n(x)$을 계산하는데, 여기서 $r(x)$는 GF(2^m)상의 원소이다. 정수상에서 비슷한 방식으로 곱셈을 하는 알고리즘이 몽고메리에 의해 제안되었다[7]. 이러한 몽고메리의 기법은 GF(2^m)상에서도 역시 응용될 수 있다[7]. 그리고 $r(x)$를 x^m으로 선택했을 때 매우 유용함이 밝혀져 있다. 따라서 $r(x)$가 유한필드상의 원소라고 할 때, $n(x)=(n_m\ n_{m-1}\ \ldots\ n_1n_0)$, $r(x)=(r_{m-1}\ \ldots\ r_1r_0)$로 나타낼 수 있다. 몽고메리 곱셈 방법에서 $r(x)$와 $n(x)$는 서로 소이다. 즉, $gcd(r(x), n(x))=1$이다.이것은 $n(x)$가 $r(x)$로 나누어지지 않음을 만족시켜야 한다. 하지만, $n(x)$는 필드 GF(2)상에서의 기약 다항식이므로 이 조건을 항상 만족한다. 또한 $r(x)$와 $n(x)$가 서로 소이기 때문에, 다음의 성질을 만족하는 $r^{-1}(x)$와 $n'(x)$가 존재한다.

$$r(x)r^{-1}(x) + n(x)n'(x)=1$$

여기서 $r^{-1}(x)$는 $r(x)$ mod $n(x)$의 역원이다. 다항식 $r^{-1}(x)$와 $n'(x)$는 확장된 유클리디언 알고리즘을 사용하여 계산할 수 있다[8]. 다항식 $a(x)$와 $b(x)$의 몽고메리 곱셈은 다음과 같이 정의된다.

$$c(x)= a(x)b(x)r^{-1}(x) \text{ mod } n(x)$$

이것은 다음 알고리즘을 이용하여 계산할 수 있다[7].

알고리즘 3.1 : MMM($a(x)$, $b(x)$, $n(x)$) Bit-Level Algorithm for Montgomery Multiplication

입력 : $a(x)$, $b(x)$, $n(x)$
출력 : $c(x)= a(x)b(x)x^{-m}$ mod $n(x)$
단계 1 : $c(x)=0$
단계 2 : for $i=0$ to $m-1$
단계 3 : $c(x)=c(x)+a_ib(x)$
단계 4 : $c(x)=c(x)+c_0n(x)$
단계 5 : $c(x)=c(x)/x$

3.2 셀룰러 오토마타를 이용한 몽고메리 곱셈기 설계

본 절에서는 셀룰러 오토마타를 이용하여 알고리즘 3.1에서 나타난 GF(2^m)상에서 비트 레벨 몽고메리 곱셈을 빠른 시간에 계산할 수 있는 구조를 제시한다.
알고리즘 3.1을 구현하기 위한 기본적인 연산들은 다음과 같다.

연산 3.1 : $c(x)=c(x)+a_ib(x)$
연산 3.2 : $c(x)=c(x)+c_0n(x)$
연산 3.3 : $c(x)=c(x)/x$

우선 연산 3.1($c(x)=c(x)+a_ib(x)(0\le i\le m-1)$)을 수행하기 위해 $a_ib(x)$연산을 먼저 살펴보자. $a_ib(x)(0\le i\le m-1)$ 연산을 위해서는, $b(x)$ 레지스터의 m 비트가 m개의 AND 게이트에 각각 입력으로 들어가고, AND 게이트의 나머지 하나의 입력은 a_i로 한다. 그 다음 그 결과와 $c(x)$를 XOR 하여 그 결과를 다시 $c(x)$에 저장한다. 이를 위한 구조도가 그림 3.1에 나타나 있다. 그림 3.1은 $i(0\le i\le m-1)$번째 클럭에서의 연산을 보여주고 있다.

그림 3.1 연산 3. 1($c(x)=c(x)+a_ib(x)$ for $0\le i\le m-1$)을 위한 구조도.

다음으로 연산 3.2($c(x)=c(x)+c_0n(x)$)의 수행에 대해 살펴본다. 이를 위해서는, $c(x)$의 LSB가 1이면, 즉 $c_0=1$이면 (c_{m-1}, ..., c_2, c_1, c_0)와 (n_{m-1}, ..., n_2, n_1, n_0)의 각 비트가 XOR 연산을 수행하여야한다. 그리고 그 결과 값은 $c(x)$에 저장된다. 이를 위한 구조도가 그림 3.2에 나타나있다.

그림 3.2 연산 3.2 ($c(x)=c(x)+c_0n(x)$)를 위한 구조도

다음으로 연산 3.3($c(x)=c(x)/x$)을 수행하기위해서는 m 개의 셀을 가지는 1차원 NBCA가 사용된다. NBCA에서는 나누기 연산을 위해 오른쪽으로 한비트 쉬프트 연산을 구현하게 되는데, 이를 위해 각 셀의 왼쪽 이웃의 값을 자신의 값으로한다. 이러한 특성을 나타내주는 $m\times m$ 특성행렬 T는 다음과 같다.

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

위와 같은 특성행렬을 가지는 셀룰러오토마타는 법칙 240을 가지게 되며, 이를 위한 구조도는 다음 그림 3.3과 같다.

그림 3.3 특성행렬 T를 가지는 NBCA의 구조도

그림 3.4에는 그림 3.1, 3.2, 3.3을 이용하여 몽고메리 곱셈을 수행하는 구조도가 나타나 있다. 여기서 그림 3.4는 $i(0\le i\le m-1)$번째 클럭에서의 연산을 나타내고 있으며, 여기서 첫번째 수행에 앞서 NBCA와 $a(x)$, $b(x)$, $n(x)$ 레지스터가 초기화 되어야한다. 그리고 그림 3.1과 3.2에서 $c(x)$레지스터는 NBCA로 대치될 수 있다. 각각의 초기값은 다음과 같다.

-NBCA의 초기값 : all 0
-$a(x)$ 레지스터의 초기값 : a_{m-1} ... a_2 a_1 a_0
-$b(x)$ 레지스터의 초기값 : b_{m-1} ... b_2 b_1 b_0
-$n(x)$ 레지스터의 초기값 : n_{m-1} ... n_2 n_1 n_0

그림 3.4 비트-레벨의 몽고메리 곱셈을 수행하는 구조도

그림 3.4의 구조를 사용하면, m 셀을 이용하여 $2m$개의 AND 게이트, $2m$ 개의 XOR 게이트 3개의 레지스터를 이용하여 m 클럭 사이클에 몽고메리 곱셈을 수행할 수 있다.

4. 성능분석

본 논문의 결과를 지금까지의 연구결과와 비교해 보면 표 4.1과 같다.

<표 4.1> 몽고메리 곱셈을 위한 연구결과 비교

구조	Systolic array	Cellular Automata
	Heo *et al* [6]	Proposed paper
수행연산	몽고메리 곱셈	
셀의 수	m	m
AND 게이트 수	$2m$	$2m$
XOR 게이트 수	$2m$	$2m$
MUX의 수	$2m$	0
control signal의 수	1	0
레지스터의 수	2	3
실행시간	$3m$	m

5. 결론

본 논문에서는 유한필드 상에서 효율적인 몽고메리 곱셈기를 설계하였다. 제시한 몽고메리 곱셈기는 m개의 셀과, $2m$개의 AND 게이트, $2m$ 개의 XOR 게이트, 3개의 레지스터를 이용하여 m 클럭사이클의 수행시간이 소요된다. 이는 지금까지 시스톨릭 구조 상에서 연구된 결과[6]보다 우수한 성능을 보이며, 제시된 구조는 지수승 연산과 곱셈에 대한 역원기, 나눗셈기 등에 사용될 수 있다. 따라서 본 논문에서 제시한 몽고메리 곱셈기는 효율적인 공개키 암호화 시스템 설계의 기본 구조로서 사용될 수 있다.

참고문헌

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

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

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

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

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

[6] Y.J. Heo, K.J. Lee, K.Y. Yoo, Design of systolic array for multiplication in GF(2^n)," ICCCS '98, Taegu, Korea 1998.

[7] Ç. K.KOÇ, T. ACAR, Montgomery Multiplication in GF(2^k), Kluwer Academic Publishers , Designs, Codes and Cryptography, 14(1), pp. 57-69, April 1998.

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