情報保護學會論文誌
第18卷 第1號, 2008. 2

GF(2^m)상에서 디지트 단위 모듈러 곱셈/제곱을 위한 시스톨릭 구조

이 진 호† ,김 현 성‡
경일대학교 컴퓨터공학부

Systolic Architecture for Digit Level Modular Multiplication/Squaring over GF(2^m)

Jin-Ho Lee† , Hyun-Sung Kim‡
School of Computer Engineering, Kyungil University

요 약

본 논문에서는 유한 필드 GF(2^m)상에서 모듈러 곱셈과 제곱을 동시에 수행하는 새로운 디지트 단위 LSB-우선 시스톨릭 구조를 제안한다. 디지트의 크기를 L이라고 할 경우, L×L 크기의 디지트 구조로 유도하기 위하여 기존의 곱셈과 제곱을 동시에 수행하는 알고리즘을 사용하고, 그 알고리즘에서 유도된 구조의 각 셀을 분리하고 인덱스 변환 시킨 후 병합하는 방법을 사용한다. 본 논문에서 제안된 구조는 암호 프로세서를 위한 기본 구조로 이용될 수 있고, 단순성, 규칙성, 병렬성으로 인해 VLSI 구현에 적합하다.

ABSTRACT

This paper presents a new digit level LSB-first multiplier for computing a modular multiplication and a modular squaring simultaneously over finite field GF(2^m). To derive L×L digit level architecture when digit size is set to L, the previous algorithm is used and index transformation and merging the cell of the architecture are proposed. The proposed architecture can be utilized for the basic architecture for the crypto-processor and it is well suited to VLSI implementation because of its simplicity, regularity, and concurrency.

Keywords : *Public-key cryptosystem, Finite fields, Digit-level architecture, Modular multiplier*

Ⅰ. 서 론

유한 필드상의 연산은 암호학, 에러 교정 코드, 디지털 신호 처리 등의 응용에서 아주 중요하다[1-9]. 이러한 유한체 중에서 특별한 관심을 가지는 유한체는 GF(2^m)이다. 유한필드 GF(2^m)은 2^m개의 원소를 가지고 각각의 원소들은 0과 1의 비트-스트링으로 구성된다. 이러한 속성 때문에 갈로아 필드 연산의 하드웨어 구현에 유한필드 GF(2^m)이 적당하다. 특히, 공개키 암호 시스템에서의 기본 연산은 GF(2^m)상에서 정규(Normal), 듀얼(Dual), 표준(Standard) 기저를 기반으로 개발되어 왔다. 이들 중 정규기저와 듀얼기저는 독특한 기저만의 장점을 가지지만 연산 전후에 기저의 변환이 필요하다는 단점을 지닌다. 하지만, 표준 기저는 기저의 변환이 필요하지 않는 장점이 있다. 본 논문에서는 표준 기저를

접수일: 2007년 10월 17일; 채택일: 2007년 12월 6일
† 주저자, jhlee@kiu.ac.kr
‡ 교신저자, kim@kiu.ac.kr

이용한다.

유한 필드 상에서 덧셈은 비트별 배타적 논리합(exclusive or) 연산으로 간단하다. 그러나 곱셈과 역원 연산(inversion) 및 지수 연산은 복잡하다. 이들 연산에서 가장 기본이 되는 연산은 모듈러 곱셈 연산이다. 필드 상에서 곱셈 연산 및 지수 연산에 대한 효율적인 하드웨어 및 소프트웨어 구현에 대해 많은 관심의 대상이 되었다. 특히 구조가 간단하고, 계산 시간이 짧으며, 좋은 성능을 가지는 모듈러 곱셈에 대한 회로 설계는 아주 중요한 과제로 연구되어 왔다[9-16].

Yeh[11] 등은 다항식 기저 표현(polynomial basis representation)으로 GF(2^m) 곱셈기를 비트단위 시스톨릭 어레이 구조로 설계하였는데, VLSI 구현에 편리한 구조이지만 두 개의 제어신호를 가진 구조이다. Wang[12] 등은 유한 필드 상에서의 MSB-first 방법으로 비트단위 시스톨릭 곱셈기를 설계하였다. 이 곱셈기는 VLSI로 구현하기 좋은 구조이다. 그 외에도 Hsu[13] 등은 다항식 기저 표현뿐만 아니라, 듀얼, 및 정규 기저 표현에 근거한 표현으로 비트단위 GF(2^m) 곱셈기를 설계하였다. Yoo 등은 모듈러 곱셈과 제곱을 동시에 수행하는 비트단위 시스톨릭 구조를 설계하였다[14]. Yoo 등의 시스톨릭 곱셈/제곱기는 곱셈기보다 회로는 조금 더 복잡하지만 수행시간은 같아서 모듈러 지수연산에 아주 효율적으로 이용될 수 있음을 제시하였다. Kim 등은 Yoo 등의 구조에 기반 한 효율적인 모듈러 지수기를 제안하였다[15].

디지트 구조는 데이터를 일정한 크기의 디지트 크기로 나누고, 나누어진 데이터들은 디지트 단위로 처리되고 전송된다. 데이터 크기가 m비트이고 디지트 크기가 L비트이면 디지트 개수는 $N=\lceil m/L \rceil$ 이 된다. Guo 등은 MSB우선 디지트 병렬 시스톨릭 곱셈기를 제안하였다[16]. 이 곱셈기의 처리기 지연시간을 개선하기 위해서 Kim등은 LSB우선 디지트 단위 시스톨릭 곱셈기를 설계하였다[17].

본 논문에서는 모듈러 곱셈과 제곱을 동시에 수행하는 디지트 단위 시스톨릭 모듈러 곱셈/제곱기 구조를 제안한다. 본 논문에서 제안한 구조는 기존의 LSB-first 방식의 곱셈 알고리즘의 특성과 디지트 구조의 장점을 결합한 특징을 갖는다. 본 논문에서 제안한 구조를 기반으로 모듈러 지수기를 설계한다면 기존의 Kim 등의 디지트 구조에 비교하여 보다 효율성을 제시할 수 있을 것이다.

본 논문의 구성은 다음과 같다. 2장에서는 Yoo 등이 제안한 곱셈과 제곱을 동시에 수행하는 곱셈알고리즘과 비트단위 시스톨릭 구조에 대해서 살펴본다. 3장에서는 Yoo 등의 알고리즘에서 디지트 단위의 시스톨릭 구조를 제안한다. 4장에서는 본 논문에서 제안한 구조와 기존의 구조 간 비교분석을 제시한다. 마지막으로 5장에서 결론을 맺는다.

Ⅱ. 관련연구

본 장에서는 Yoo 등이 제안한 모듈러 곱셈과 제곱을 동시에 수행하는 알고리즘과 이를 위한 비트단위 시스톨릭 구조를 살펴본다[14].

2.1. 곱셈과 제곱 동시 수행 알고리즘

모듈러 곱셈을 이용하여 모듈러 지수 연산을 수행하는 가장 일반적인 방법은 이진 제곱과 곱셈(binary square and multiply) 알고리즘이다. 이 알고리즘에는 left-to-right 방식과 right-to-left 방식이 있다. 특히, right-to-left 방식에서는 각 반복마다 제곱 연산과 곱셈을 동시에 수행할 수 있다. 따라서 곱셈과 제곱 연산을 동시에 수행하는 하드웨어는 지수 연산에 아주 효율적으로 이용될 수 있다.

LSB-first 곱셈 알고리즘의 특성을 이용하여 곱셈 $M(x)=A(x)B(x) \bmod P(x)$와 제곱 연산 $S(x)=A(x)A(x) \bmod P(x)$의 계산과정을 전개하면 아래와 같다.

$$
\begin{aligned}
M(x) &= A(x)B(x) \bmod P(x) \\
&= b_0A(x)+b_1[A(x)x \bmod P(x)] \\
&\quad +b_2[A(x)x^2 \bmod P(x)]+\ldots+b_{m-1}[A(x)x^{m-1} \bmod P(x)] \\
S(x) &= A(x)A(x) \bmod P(x) \\
&= a_0A(x)+a_1[A(x)x \bmod P(x)] \\
&\quad +a_2[A(x)x^2 \bmod P(x)]+\ldots+a_{m-1}[A(x)x^{m-1} \bmod P(x)]
\end{aligned}
$$

위의 두 식에서 [ ]안에 있는 계산은 같음을 알 수 있다. 따라서 곱셈과 제곱 연산을 동시에 수행할 때 공통부분을 한번만 계산하면 효율적이다. 즉, 위의 식에서 곱셈과 제곱 연산은 식 $A^{(i)}(x)=A^{(i-1)}(x)x \bmod P(x)$을 공통으로 포함한다. 따라서 위의 식을 다음과 같이 곱셈과 제곱 연산을 동시에 수행하는 순환식으로 쉽게 유도할 수 있다($1 \le i \le m$).

$A^{(i)}(x)=A^{(i-1)}(x)x \bmod P(x)$

$M^{(i)}(x)=M^{(i-1)}(x)+b_{i-1}A^{(i-1)}(x)$

$S^{(i)}(x)=S^{(i-1)}(x)+a_{i-1}A^{(i-1)}(x)$

여기서 $A^{(0)}(x)=A(x)$이고, $S^{(0)}(x)=0$이다. 위의 식에서 $i=m$일 때, $M^{(m)}=M(x)=A(x)B(x) \bmod P(x)$와 $S^{(m)}=S(x)=A(x)A(x) \bmod P(x)$으로 모듈러 곱셈 $M(x)$과 제곱 $S(x)$를 동시에 계산할 수 있다. 위 3개의 순환식은 병렬로 수행될 수 있다. 이 순환식으로부터 비트별 LSB-first 모듈러 곱셈/제곱을 동시에 계산하는 알고리즘을 유도한다.

[비트별 LSB-first 모듈러 곱셈/제곱 알고리즘]

입력 : $A=(a_{m-1}, a_{m-2}, ..., a_1, a_0)$

$B=(b_{m-1}, b_{m-2}, ..., b_1, b_0)$

$P=(1, p_{m-1}, p_{m-2}, ..., p_1, p_0)$

출력 : $M^{(m)}=M(x)=A(x)B(x) \bmod P(x)$,

$S^{(m)}=S(x)=A(x)A(x) \bmod P(x)$

초기치: $M^{(0)}=(M^{(0)}_{m-1}, M^{(0)}_{m-2}, \cdots, M^{(0)}_1, M^{(0)}_0)$

$=(0, 0, ..., 0, 0)$

$S^{(0)}=(S^{(0)}_{m-1}, S^{(0)}_{m-2}, \cdots, S^{(0)}_1, S^{(0)}_0)$

$=(0, 0, ..., 0, 0)$

$A^{(0)}=(A^{(0)}_{m-1}, A^{(0)}_{m-2}, \cdots, A^{(0)}_1, A^{(0)}_0, A^{(0)}_{-1})$

$=(a_{m-1}, a_{m-2}, ..., a_1, a_0, 0)$

$A^{(i)}_{-1}\equiv 0,\ 1\le i\le m$

순환식 : for $i=1$ to m

for $j=1$ to m

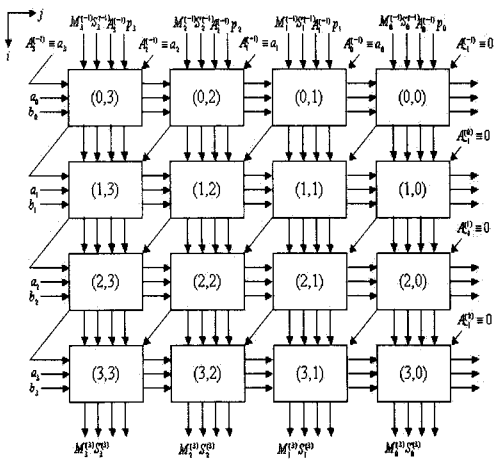
$A^{(i)}_{m-j}=A^{(i-1)}_{m-1-j}+A^{(i-1)}_{m-1}\,p_{m-j}$

$M^{(i)}_{m-j}=M^{(i-1)}_{m-j}+b_{i-1}A^{(i-1)}_{m-j}$

$S^{(i)}_{m-j}=S^{(i-1)}_{m-j}+a_{i-1}A^{(i-1)}_{m-j}$

2.2. 곱셈과 제곱 동시 구조

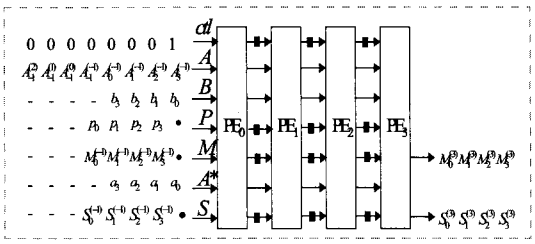
본 절에서는 앞 절에서 언급한 곱셈 알고리즘으로부터 시스톨릭 곱셈기를 설계한다. 곱셈 알고리즘으로부터 [그림 1]의 비트 단위 데이터 의존그래프를 유도하고 해석적이고 체계적인 절차로 시스톨릭 어레이를 설계할 수 있다[18]. 시스톨릭 어레이의 설계에 대한 설명을 간단히 하기 위해서 예를 들어 유한 필드 GF(2^4)상에서의 시스톨릭 곱셈기의 설계를 설명한다. 일반적인 GF(2^m)상의 시스톨릭 어레이의 설계는 같은 방법으로 확장하여 적용하면 된다.



〔그림 1〕. 비트 단위 데이터 의존 그래프(m=4)

일반적으로 GF(2^m)상에서 곱셈 알고리즘의 계산 점들이 최대로 병렬성을 살려서 수행될 때 걸리는 시간스텝은 $3m$이다.

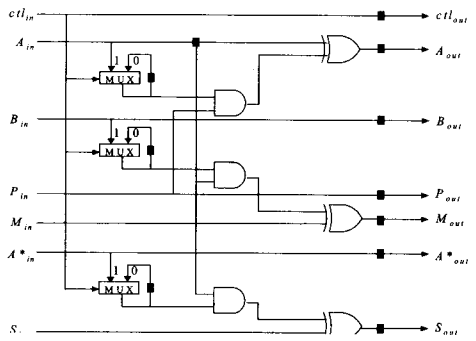
[그림 2]는 [그림 1]의 데이터 의존 그래프를 j축으로 투영하여 얻은 시스톨릭 어레이로 구현한 결과이다. [그림 3]은 [그림 2]의 곱셈/제곱기 각각의 PE 구조를 상세히 보여준다. 일반적으로 GF(2^m)상의 시스톨릭 곱셈/제곱기는 [그림 2]에서 PE의 개수가 m개로 증가되고, 각 PE의 구조는 [그림 3]과 같다. 여기서 '■'는 1비트 래치를 의미한다.



〔그림 2〕. GF(2^4)상의 시스톨릭 곱셈/제곱기

Ⅲ. 디지트 단위 모듈러 곱셈/제곱기

본 장에서는 Yoo 등이 제안한 모듈러 곱셈과 제곱을 동시에 수행하는 알고리즘에 기반 한 디지트 단위 시스톨릭 모듈러 곱셈/제곱기를 제안한다. 먼저, [그림 1]에 기반 한 디지트 단위 데이터 의존 그래프를 제시하고, 제시된 데이터 의존 그래프에 기반 한 시스톨릭 모듈러

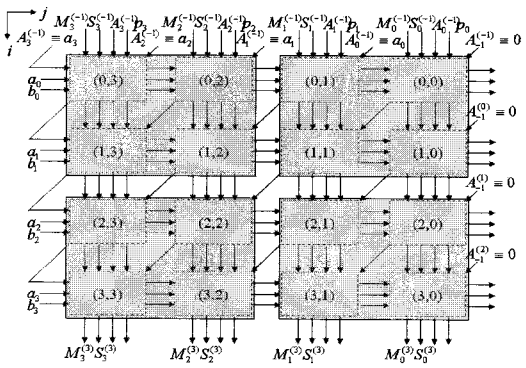


〔그림 3〕. 시스톨릭 곱셈/제곱 기본구조

곱셈/제곱기 구조를 제안한다.

3.1. 디지트 단위 데이터 의존 그래프

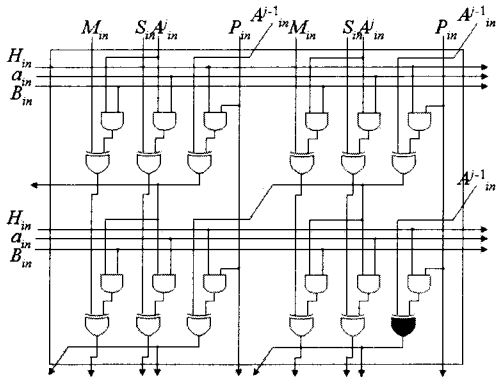
본 소절에서는 [그림 1]의 비트 단위 데이터 의존 그래프의 계산점들을 원하는 디지트로 묶어서 하나의 계산점을 만들어 디지트 단위의 데이터 의존 그래프로 변형한다. 본 절에서는 논의를 단순히 하기 위해서 2-디지트를 위한 데이터 의존 그래프를 [그림 4]와 같이 제시한다.



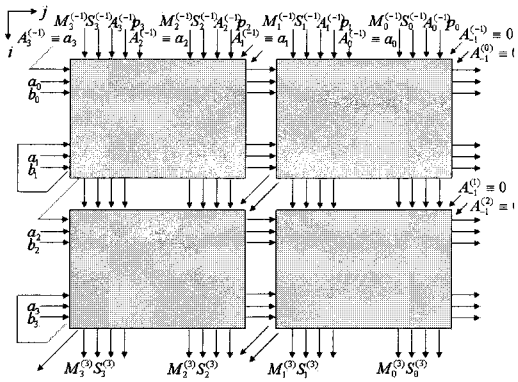
〔그림 4〕. 디지트 단위 데이터 의존 그래프-1

즉, [그림 1]의 계산점을 2X2로 계산점들을 묶어서 [그림 4]의 음영 블럭으로 표시된 하나의 계산점으로 만들면 디지트의 크기는 2가 되고 계산점의 수는 $Nodes=m/2$가 된다. [그림 5]는 [그림 4]의 하나의 계산점을 위한 기본 구조를 보여준다.

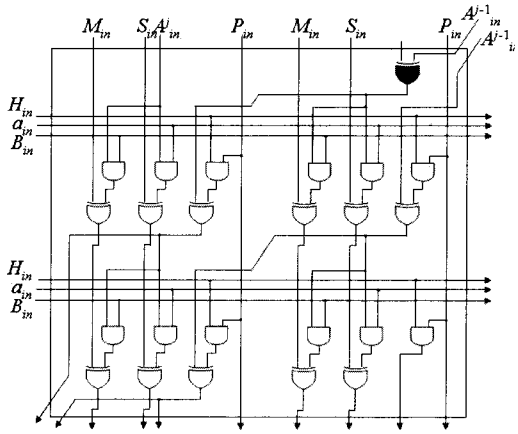
[그림 4]의 데이터 의존 그래프에는 $(i, j-1)$에서 (i, j) 계산점으로 역방향 자료 흐름이 존재한다. 이러한 역방향 자료 흐름은 선형 시스톨릭 어레이의 j방향 매핑에 있어서 비효율성을 제시한다. 이러한 문제를 해결하기



〔그림 5〕. 디지트 단위 기본구조-1



〔그림 6〕. 디지트 단위 데이터 의존 그래프-2

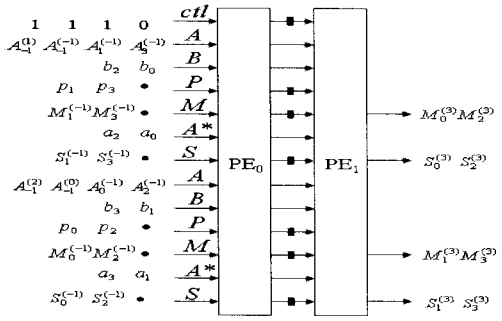


〔그림 7〕. 디지트 단위 기본구조-2

위해서는 [그림 5]의 검정색으로 표시된 XOR게이트의 지연 연산을 통해서 해결 가능하다. 즉 [그림 5]의 기본 구조를 [그림 7]의 기본구조로 변경함으로서 보다 효율적인 데이터 흐름을 가지는 [그림 6]과 같은 데이터 의존 그래프를 얻을 수 있다. [그림 6]의 데이터 의존 그래프는 단방향 데이터 흐름을 가지므로 선형 시스톨릭 어레이로 유도하기 위해서 j방향으로 매핑 될 수 있다.

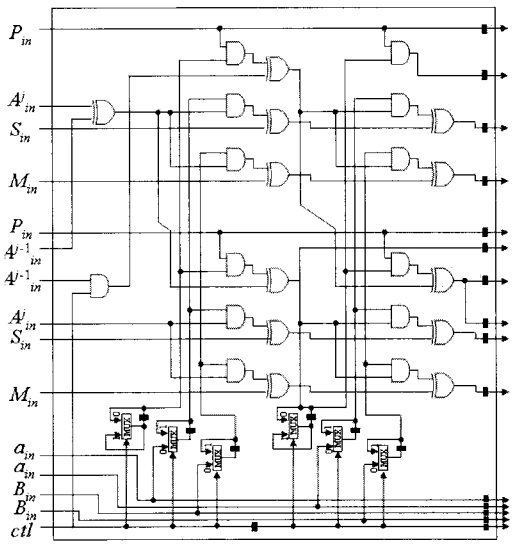
3.2. 디지트 단위 시스톨릭 구조

본 절에서는 데이터 의존 그래프로부터 디지트 순차 시스톨릭 곱셈/제곱기를 제안한다. 디지트의 크기를 L이라고 할 경우, 먼저 LXL 크기의 디지트 단위 데이터 의존 그래프를 알고리즘으로부터 유도하고, 선형 시스톨릭 주로를 유도하기 위해서 데이터 의존 그래프의 각 셀을 분리하고 인덱스 변환 시킨 후 병합하는 방법을 사용한다. [그림 6]의 데이터 의존 그래프로부터 j축으로 투영하면 데이터 의존 그래프는 $m/(L=2)$개의 PE로 매핑시켜 컷셋(cut-set) 시스톨릭화 기법을 적용하면 [그림 8]과 같은 시스톨릭 구조를 얻을 수 있다[18].



[그림 8]. GF(2^4)상의 2-디지트 단위 시스톨릭 구조

[그림 9]를 기본구조로 하는 [그림 9]의 2-디지트 단위 시스톨릭 곱셈/제곱기는 연속적인 입력에 대해 $3m/2$ 시간스텝의 초기 지연 후 $m/2$시간스텝마다 하나의 결과를 얻을 수 있다. 일반적으로 GF(2^m)상에서 L디지트 단위 곱셈 알고리즘의 계산 점들이 최대로 병렬성을 살려서 수행될 때 걸리는 초기 지연시간은 시간스텝은 $3m/L$이다. 하지만 GF(2^m)상에서 L디지트 단위 시스톨릭 구조로 일반화를 유도할 경우에는 [그림 7]의 검정색으로 표시된 XOR게이트의 수도 추가적인 확장이 필요하다. 즉 L디지트 단위 시스톨릭 구조일 경우 L-1개



[그림 9]. 2-디지트 단위 기본구조

의 추가적인 검정색으로 표시된 XOR 게이트들이 필요하다.

Ⅳ. 비교 및 분석

본 장에서는 본 논문에서 제안한 구조와 기존의 곱셈기와의 비교를 제시한다. Guo 등과 Kim 등은 모듈러 곱셈을 위한 디지트 단위 시스톨릭 곱셈기를 제안하였다[16, 17]. [표 1]은 이들 구조와 본 논문에서 제안한 구조와의 비교를 제시한다. 보다 가시적인 비교를 위해서 Guo 등[16]이 제안한 3-입력 XOR 게이트와 4-입력 XOR 게이트는 각각 2개의 2-입력 XOR 게이트와 3개의 2-입력 XOR 게이트로 구성된다고 가정한다[19].

그러면 처리기 복잡도에서 XOR게이트와 MUX의 수는 기존의 구조에 비해서 1.5배 많다. 하지만 기본적인 연산을 비교한다면 Guo 등과 Kim 등의 구조의 기본 연산은 단지 모듈러 곱셈인데 반해 본 논문에서 제안하는 시스톨릭 구조의 기본 연산은 모듈러 곱셈뿐만 아니라 모듈러 제곱 연산을 동시에 수행한다. 즉, 본 논문에서 제안하는 시스톨릭 곱셈/제곱기는 기존의 구조에 비해서 두 배의 연산을 수행하면서 곱셈과 제곱에 공통적으로 수행되는 연산을 단순화 할 수 있어서 하드웨어 복잡도는 1.5배정도 요구하므로 기존의 구조에 비해서 효율적이라고 할 수 있다. 계산의 최대 지연시간에 대해서도 Kim 등의 구조와 동일한 지연 시간을 가진다.

〔표 1〕. GF(2^m)상의 2-디지트 구조 간 속성 비교

속성 / 구조	전체 곱셈기의 복잡도	기본구조 복잡도	기본구조 최대 지연시간	기본 연산
Guo등 [16]	N 처리기 1 XOR_2 1 AND_2	8 XOR_2 10 AND_2 20 Latch 4 MUX	$2T_{AND2}+T_{MUX}$ $+5T_{XOR2}$	곱셈
Kim등 [17]	N 처리기	8 XOR_2 9 AND_2 22 Latch 4 MUX	$T_{AND2}+T_{MUX}$ $+3T_{XOR2}$	곱셈
제안한 곱셈/제곱기	N 처리기	12 XOR_2 13 AND_2 26 Latch 6 MUX	$T_{AND2}+T_{MUX}$ $+3T_{XOR2}$	곱셈 제곱

V. 결 론

본 논문에서는 GF(2^m)상에서 모듈러 곱셈/제곱을 동시에 수행하는 알고리즘을 이용하여 디지트 단위 시스톨릭 곱셈기를 제안하였다. 디지트 단위 시스톨릭 어레이구조를 제안하기 위해서 기존의 곱셈/제곱 알고리즘으로부터 데이터 의존 그래프를 유도하고, 유도된 그래프의 데이터 흐름을 정형화하기 위한 변형된 데이터 의존 그래프를 제시하였다. 이렇게 변형된 데이터 의존 그래프를 기반으로 컷-셋 시스톨릭화 기술을 적용하여 비트단위 순차 시스톨릭 구조보다 빠르고, 비트단위 병렬 시스톨릭 구조보다 적은 하드웨어를 사용하는 디지트 단위 시스톨릭 곱셈/제곱기를 제안하였다. 제안된 시스톨릭 곱셈/제곱기는 기존의 곱셈기의 시간과 공간간의 상충관계를 효율적으로 개선한 구조이다. 본 논문에서 제안된 구조는 VLSI구현에 적합하며 공개키 암호의 핵심이 되는 지수기의 구현을 위한 효율적인 기본구조로 사용될 수 있을 것이다.

참고문헌

[1] W. W. Peterson and E. J. Weldon, *Error-correcting codes*, Cambridge, MA: MIT Press, 1972.

[2] I. S. Reed and T. K. Truong,"The use of finite fields to compute convolutions", *IEEE Trans. Inform.* Theory, vol. IT-21, pp. 208-213, 1975.

[3] D. E. R. Denning, *Cryptography and data security*, Reading, MA: Addison-Wesley, 1983.

[4] A. M. Odlyzko, "Discrete logarithms in finite fields and their cryptographic significance", *Proc. Eurocrypt84*, pp. 224-314, 1984.

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

[6] E. R. Berlekamp, *Algebraic coding theory*, New York: McGraw-Hill, 1968.

[7] A.J. Menezes, *Applications of finite fields*, Boston, MA: Kluwer Academic Publishers, 1993.

[8] R. Lidl, H. Niederreiter, and P. M. Cohn, *Finite fields(Encyclopedia of mathematics and its applications)*, Cambridge University Press, 1997.

[9] P. A. Scott, S.E. Tavares, and L.E. Peppard, "A fast VLSI multiplier for GF(2^m)", *IEEE Jour. of Selected Areas in Comm.*, vol. 4. pp. 62-66, 1986.

[10] C. C. Wang, T. K. Truong, H. M. Shao, L. J. Deutsch, J. K. Omura, and I. S. Reed, "VLSI architectures for computing multiplications and inverses in GF(2^m)", *IEEE Trans. Computer,* vol. C-34, pp. 709-717, 1985.

[11] C. S. Yeh, I. S. Reed, and T. K. Truong, "Systolic multipliers for finite fields GF(2^m)", *IEEE Trans. Computer*, vol. C-33, pp. 357-360, 1984.

[12] C. L. Wang and J. L. Lin, "Systolic array implementation of multipliers for finite fields GF(2^m)", *IEEE Trans. Circuits Systems*, vol. 38, pp. 796-800, 1991.

[13] I. S. Hsu. T. K. Truong, L.J. Deutsch, and I. S. Reed, "A comparison of VLSI architecture of finite field multipliers using dual, normal, standard bases", *IEEE Trans. Computer*, vol. 37, pp. 735-739, 1988.

[14] 유기영, 김정준, "유한필드 GF(2^m)상의 시스톨릭 곱셈/제곱기", 정보과학회논문지, 28(5), pp. 289-300, 2001.

[15] H. S. Kim and K. Y. Yoo, "Area efficient exponentiation using modular multiplier/squarer in GF(2^m)", *Lecture notes in computer science*, vol. 2108, pp. 262-267, 2001.

[16] J. H. Guo and C. L. Wang, "Digit-serial systolic multiplier for finite fields GF(2^m)", *IEE Proc. Comp. Digit. Tech.*, vol. 145, no. 2, pp. 143-148, 1998.

[17] 김기원, 이건직, 유기영, "GF(2^m)상에서 2-디지트 시리얼 시스톨릭 곱셈기 설계 및 분석", 한국정보과학회 가을 학술발표논문집, 27(2), pp. 605-607, 2000.

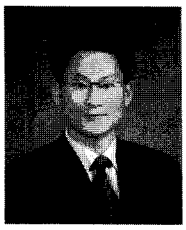
[18] S. Y. Kung, *VLSI array processors*, Englewood Cliffs, NJ:Prentice-Hall, 1988.

[19] N Weste and K. Eshraghian, *Principle of CMOS VLSI design : a system perspective*, Addison Wesley, Reading, MA, 1985.

〈著者紹介〉

이 진 호 (Jin-Ho Lee) 정회원

1974년 2월 : 영남대학교 전자공학과 공학사
1981년 2월 : 영남대학교 전자계산학과 공학석사
1996년 2월 : 영남대학교 전자계산학과 공학박사
1979년 3월 ~ 현재 : 경일대학교 컴퓨터공학부 교수
<관심분야> 프로그래밍언어, 정보보호



김 현 성 (Hyun-Sung Kim) 종신회원

1996년 2월 : 경일대학교 컴퓨터공학과 공학사
1998년 2월 : 경북대학교 컴퓨터공학과 공학석사
2002년 2월 : 경북대학교 컴퓨터공학과 공학박사
2002년 3월 ~ 현재 : 경일대학교 컴퓨터공학부 교수
<관심분야> 정보보호, 암호 프로토콜, 암호 프로세서 설계, 센서네트워크 보안