논문 2017-12-02

GF(2^m) 상의 여분 표현을 이용한 낮은 지연시간의 몽고메리 AB^2 곱셈기

(Low-latency Montgomery AB^2 Multiplier Using Redundant Representation Over GF(2^m))

김 태 완, 김 기 원*
(Tai Wan Kim, Kee-Won Kim)

Abstract : Finite field arithmetic has been extensively used in error correcting codes and cryptography. Low-complexity and high-speed designs for finite field arithmetic are needed to meet the demands of wider bandwidth, better security and higher portability for personal communication device. In particular, cryptosystems in $GF(2^m)$ usually require computing exponentiation, division, and multiplicative inverse, which are very costly operations. These operations can be performed by computing modular AB multiplications or modular AB^2 multiplications. To compute these time-consuming operations, using AB^2 multiplications is more efficient than AB multiplications. Thus, there are needs for an efficient AB^2 multiplier architecture. In this paper, we propose a low latency Montgomery AB^2 multiplier using redundant representation over $GF(2^m)$. The proposed AB^2 multiplier has less space and time complexities compared to related multipliers. As compared to the corresponding existing structures, the proposed AB^2 multiplier saves at least 18% area, 50% time, and 59% area-time (AT) complexity. Accordingly, it is well suited for VLSI implementation and can be easily applied as a basic component for computing complex operations over finite field, such as exponentiation, division, and multiplicative inverse.

Keywords : Finite fields, Montgomery multiplication, Systolic array, Cryptography

Ⅰ. 서 론

유한체의 연산은 오류 정정 부호 (error correcting codes) 및 암호학 (cryptography)에서 중요한 역할을 한다 [1-3]. 유한체 응용에서 일반적으로 곱셈, 멱승, 곱셈에 대한 역원, 나눗셈 등의 연산이 필요하다. 이러한 연산 중에서 곱셈은 가장 중요한 연산이다. 이는 유한체 상의 다른 복잡한 연산들, 즉 멱승, 역원, 나눗셈은 모듈러 AB 또는 AB^2 곱셈의 반복적인 실행을 통하여 계산 가능하기 때문이다. 복잡하고 시간 소모가 큰 연산들의 빠른 계산을 위해 빠른 속도의 유한체 곱셈기 설계를 위한 알고리즘이 필요하다.

유한체 연산의 효율성은 원소 표현의 기저(base)선택에 의존적이다. 유한체의 기저들은 정규(normal), 듀얼 (dual), 다항식 (polynomial) 기저 등이 있다. 각 기저들은 각각의 특징들을 가지고 있으며, 본 논문에서는 여분 표현 (redundant representation) 기반의 곱셈기를 설계할 것이다.

정수 상에서 효율적인 모듈러 곱셈을 위해 몽고메리 곱셈 알고리즘이 제안되었다 [4]. 그리고 Koc과 Acar [5]에 의해 유한체 $GF(2^m)$의 곱셈으로 확장되었다. 유한체 $GF(2^m)$상의 몽고메리 곱셈은 고속의 구조 설계를 위해 효율적인 방법을 제공하

*Corresponding Author (nirkim@dankook.ac.kr)
Received: Nov. 14 2016, Revised: Dec. 19 2016, Accepted: Dec. 27 2016.
T.W. Kim: Pusan National University
K.W. Kim: Dankook Univertisy
※ 이 논문은 2015년도 정부 (교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2015R1D1A1A01059739).

ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2017.12.1.11

며 VLSI 구현에도 적합하다 [6-13]. Kim과 Lee [10]는 셀룰러 오토마타 (cellular automata)를 이용하여 몽고메리 AB^2 곱셈기를 제안하였다. Kim과 Jeon [11]은 다항식 기저 (plynomial basis) 기반의 몽고메리 AB 곱셈기를 제안하였다. Choi와 Lee [12]는 여분 표현 기반의 몽고메리 AB 곱셈기를 제안하였다. Kim과 Jeon [13]은 순차적인 입력을 통해 공간 복잡도를 감소시킨 다항식 기저 (plynomial basis) 기반의 몽고메리 AB 곱셈기를 제안하였다. Lee와 Kim [14]은 Choi와 Lee [12]의 곱셈기보다 효율적인 여분 표현 기반의 몽고메리 AB 곱셈기를 제안하였다.

$GF(2^m)$상에서 멱승, 역원, 및 나눗셈 연산을 수행할 때, AB 계산을 이용하는 것보다 AB^2 계산을 이용하는 것이 효율적이며, AB^2 곱셈 연산을 고속으로 구현하기 위해 다양한 구조의 곱셈기들이 제안되었다 [15-23]. 이러한 곱셈기 구조 중에서 시스톨릭 어레이 (systolic array)는 동일한 셀들을 인근의 셀들과 정규적으로 연결하며, 셀들은 특정한 계산을 수행 후, 그 결과를 인근의 셀들에게 고속으로 전송한다. 시스톨릭 어레이의 구조는 정규적인 회로 형태를 가지며 유한체 $GF(2^m)$상의 빠른 연산 구현에 적합하다.

Wei [15]와 Wang-Guo [16]는 표준 기저 기반의 비트-병렬 AB^2 시스톨릭 곱셈기를 제안하였다. 이들이 제안한 곱셈기들은 다소 넓은 칩 면적을 가진다. Liu 등 [17]은 기약 AOP (all one polynomial) 기반의 셀룰러 구조를 사용한 AB^2 곱셈기를 제안하였다. Lee 등 [18]은 기약 AOP 기반의 비트 병렬 시스톨릭 AB^2 곱셈기를 제안하였다. Ting 등[19]은 공간 복잡도를 줄인 3차원 형태의 $C+AB^2$을 계산하는 시스톨릭 구조를 제안하였다. Lee 등[20]은 인터리브드 (interleaved) 계산 방법을 사용한 비트 병렬 시스톨릭 AB^2 곱셈기를 제안하였다. Kim과 Lee [21]는 AB^2 곱셈을 위해 낮은 복잡도를 가지는 병렬 및 직렬 시스톨릭 구조를 제안하였다. 또한 그들은 문헌 [22]에서 AB^2 곱셈을 위해 효율적인 병렬 입출력 시스톨릭 구조를 제안하였고, 문헌 [23]에서 셀룰러 오토마타 (cellular automata)를 이용하여 AB^2 곱셈 구조를 제안하였다.

본 논문의 구성은 다음과 같다. 2장에서는 $GF(2^m)$상의 몽고메리 곱셈과 여분 표현 기반 곱셈을 고찰한다. 3장에서는 여분표현을 이용한 몽고메리 AB^2 곱셈 알고리즘을 제안하고 곱셈기를 설계한다. 4장은 제안한 곱셈기의 공간 및 시간 복잡도를 분석한다. 마지막으로 5장에서 결론을 맺는다.

Ⅱ. 관련 연구

본 장에서는 유한체상의 몽고메리 곱셈, 여분표현, 여분기저 기반 곱셈 알고리즘에 대해 고찰한다.

1. 유한체상의 몽고메리 곱셈

정수 상에서 효율적인 모듈러 곱셈을 위해 몽고메리 곱셈 알고리즘이 제안되었고, Koc과 Acar에 의해 유한체 $GF(2^m)$의 곱셈으로 확장되었다 [4-5]. $GF(2^m)$ 상에서 몽고메리 곱셈을 살펴보면 다음과 같다. α와 β는 $GF(2^m)$상의 두 원소이고 $\theta=\alpha\beta \bmod G$ 라고 하자. 여기서 G는 $GF(2^m)$상의 기약 다항식이다. 몽고메리 잉여 (Montgomery residue) A와 B는 각각 $A=\alpha r \bmod G=\sum_{j=0}^{m-1}a_j x^j$와 $B=\beta r \bmod G=\sum_{j=0}^{m-1}b_j x^j$와 이다. 여기서 r은 $\gcd(r,G)=1$을 만족하는 몽고메리 인자 (Montgomery factor)이다. r^{-1}을 r의 역수 (multiplicative inverse)라고 하면, $rr^{-1}+GG'=1$을 만족하는 $GF(2^m)$상의 원소 G'이 존재함을 알 수 있다. 그러면, $GF(2^m)$상의 몽고메리 곱셈은 $Q=ABr^{-1} \bmod G$이다. 몽고메리 인자의 선택에 따라 몽고메리 곱셈 알고리즘은 간단하고 효율적인 하드웨어로 구현될 수 있다. 몽고메리 잉여 A와 B를 사용하면 몽고메리 곱셈은 $Q=(\alpha r)(\beta r)r^{-1} \bmod G=\theta r \bmod G$ 이다. 여기서 Q는 $\theta(=\alpha\beta)$의 몽고메리 잉여이다. 일반적으로 몽고메리 잉여로의 변환은 연산을 시작하기 전에 피연산자들을 한 번 몽고메리 잉여로 바꾸고, 연속적인 곱셈이나 다른 연산을 수행하고 마지막에 원래의 표현으로 변환하면 된다. 마지막 변환 작업은 $Qr^{-1} \bmod G$연산을 수행하면 된다.

$GF(2^m)$상에서 몽고메리 AB^2 곱셈을 살펴보면 다음과 같다. α와 β는 $GF(2^m)$상의 두 원소이고 $\delta=\alpha\beta^2 \bmod G$라고 하자. 몽고메리 AB^2 곱셈은 $P=AB^2r^{-2} \bmod G$이다. 몽고메리 잉여 A와 B를 사용하면 $P=(\alpha r)(\beta r)^2 r^{-2} \bmod G=\delta r \bmod G$이다 [10].

2. 여분 표현 (Redundant Presentation)과 곱셈

x는 n번째 원시근이면, 유한체 $GF(2^m)$의 원소

A는 $A=a_0+a_1x+a_2x^2+\cdots+a_{n-1}x^{n-1}$와 같이 표현된다. 여기서 $a_i \in GF(2), 0 \le i \le n-1$이다. n은 m보다 작은 정수이며, 집합 $\{1, x, x^2, \ldots, x^{n-1}\}$은 여분 기저 (redundant basis)이다 [24-25]. 유한체 $GF(2^m)$에서 $m+1$이 소수이고 2가 법 $m+1$에 관하여 원시근일 때, 타입 I ONB (optimal normal basis)가 존재한다. 여기서 $n=m+1$이다 [24].

$A=\sum_{i=0}^{n-1}a_ix^i$와 $B=\sum_{i=0}^{n-1}b_ix^i$가 여분기저 $\{1, x, x^2, \cdots, x^{n-1}\}$로 표현된 원소들이라고 하자. 여분기저의 특성 $x^n=1$에 따라, A와 B의 곱셈 결과는 식 (1)과 같다.

$$\begin{aligned} AB &= \left(\sum_{i=0}^{n-1}a_ix^i\right)\left(\sum_{i=0}^{n-1}b_ix^i\right) \\ &= \sum_{i=0}^{n-1}\sum_{j=0}^{n-1}a_{\langle i-j\rangle}b_jx^i \\ &= \sum_{i=0}^{n-1}\sum_{j=0}^{n-1}a_jb_{\langle i-j\rangle}x^i, \end{aligned} \tag{1}$$

여기서 $<y>$는 $y \bmod n$을 의미한다.

Ⅲ. 제안하는 몽고메리 AB^2 시스톨릭 어레이

이장에서는 여분 기저 기반의 몽고메리 AB^2 알고리즘을 제안하고 이를 이용하여 시스톨릭 곱셈기를 제안한다.

1. 제안하는 몽고메리 AB^2 곱셈 알고리즘

$A=\sum_{i=0}^{n-1}a_ix^i$와 $B=\sum_{i=0}^{n-1}b_ix^i$가 여분기저 $\{1, x, x^2, \cdots, x^{n-1}\}$로 표현된 유한체상의 원소들이라고 하자. 본 논문에서는 낮은 지연시간의 곱셈기 설계를 위해, $k=\lfloor n/2 \rfloor$라 두고 몽고메리 인자 $r=x^k=x^{\lfloor n/2 \rfloor}$을 사용한다. 그러면 몽고메리 AB^2은 식 (2)와 같다.

$$P=AB^2r^{-2} \tag{2}$$

B의 제곱은 식 (3)과 같다.

$$B^2=\left[\sum_{i=0}^{n-1}b_ix^i\right]^2=\sum_{i=0}^{n-1}b_ix^{2i} \tag{3}$$

여분 기저의 특성 $x^n=1$ 때문에, 식 (3)의 B^2은 식 (4)과 같다. 여기서 $\langle y\rangle$는 $y \bmod n$ 이다.

$$\begin{aligned} B^2 &= b_0+b_1x^2+\cdots+b_{n-1}x^{2(n-1)} \\ &= b_0+b_1x^2+\cdots+b_{(n-1)/2}x^{n-1}+b_{(n+1)/2}x^{n+1} \\ &\quad +b_{(n+3)/2}x^{n+3}+\cdots+b_{n-1}x^{2(n-1)} \\ &= b_0+b_1x^2+\cdots+b_{(n-1)/2}x^{n-1} \\ &\quad +b_{(n+1)/2}x+b_{(n+3)/2}x^3+\cdots+b_{n-1}x^{n-2} \\ &= b_0+b_{(n+1)/2}x+b_1x^2+b_{(n+3)/2}x^3+\cdots \\ &\quad +b_{n+1}x^{n-2}+b_{(n-1)/2}x^{n-1} \\ &= b_{\langle 0/2\rangle}+b_{\langle 1/2\rangle}x+b_{\langle 2/2\rangle}x^2+b_{\langle 3/2\rangle}x^3 \\ &\quad +\cdots+b_{\langle (n-2)/2\rangle}x^{n-2}+b_{\langle (n-1)/2\rangle}x^{n-1} \\ &= \sum_{i=0}^{n-1}b_{\langle i/2\rangle}x^i \end{aligned} \tag{4}$$

식 (2)에서 $D=B^2r^{-1}$라고 두면 D는 식 (5)와 같다.

$$\begin{aligned} D &= B^2r^{-1}=B^2x^{-k} \\ &= \left[\sum_{i=0}^{n-1}b_{\langle i/2\rangle}x^i\right]x^{-k} \\ &= \sum_{i=0}^{n-1}b_{\langle i/2\rangle}x^{i-k} \\ &= \sum_{i=0}^{n-1}b_{\langle (i+k)/2\rangle}x^i \end{aligned} \tag{5}$$

예를 들어 $GF(2^4)$를 고려할 때, $n=5$이며 $x^5=1$이므로 $B=b_0+b_1x+b_2x^2+b_3x^3+b_4x^4$의 제곱인 B^2는 다음과 같다. 여기서 $n=5$일 때 $\langle 1/2\rangle=3$이다.

$$\begin{aligned} B^2 &= (b_0+b_1x+b_2x^2+b_3x^3+b_4x^4)^2 \\ &= b_0+b_1x^2+b_2x^4+b_3x^6+b_4x^8 \\ &= b_0+b_1x^2+b_2x^4+b_3x+b_4x^3 \\ &= b_0+b_3x+b_1x^2+b_4x^3+b_2x^4 \\ &= b_{\langle 0/2\rangle}+b_{\langle 1/2\rangle}x+b_{\langle 2/2\rangle}x^2 \\ &\quad +b_{\langle 3/2\rangle}x^3+b_{\langle 4/2\rangle}x^4 \end{aligned} \tag{6}$$

$n=5$일 때 $k=\lfloor n/2 \rfloor=2$이다. 따라서 $D=B^2r^{-1}=B^2x^{-k}$는 다음 식과 같다.

$$\begin{aligned} D &= [b_{\langle 0/2\rangle}+b_{\langle 1/2\rangle}x+b_{\langle 2/2\rangle}x^2 \\ &\quad +b_{\langle 3/2\rangle}x^3+b_{\langle 4/2\rangle}x^4]x^{-2} \\ &= b_{\langle (0+2)/2\rangle}+b_{\langle (1+2)/2\rangle}x+b_{\langle (2+2)/2\rangle}x^2 \\ &\quad +b_{\langle (3+2)/2\rangle}x^3+b_{\langle (4+2)/2\rangle}x^4]x^{-2} \\ &= b_1+b_4x+b_2x^2+b_0x^3+b_3x^4 \\ &\equiv d_0+d_1x+d_2x^2+d_3x^3+d_4x^4 \end{aligned} \tag{7}$$

$D=B^2r^{-1}$이므로 식 (2)에서 P는 식 (8)과 같다.

$$P=AB^2r^{-2}=A(B^2r^{-1})r^{-1}=ADr^{-1} \tag{8}$$

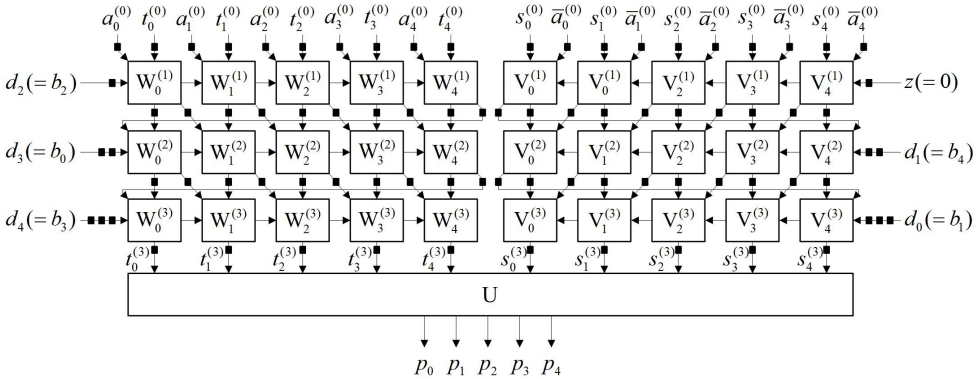
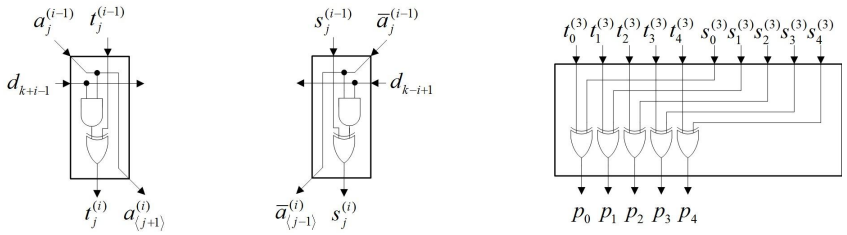


그림 1. 제안하는 $GF(2^4)$상의 AB^2 곱셈기

Fig. 1 The proposed AB^2 multiplier over $GF(2^4)$



(a) $W_j^{(i)}$ cell (b) $V_j^{(i)}$ cell (c) U cell

그림 2. 자세한 셀 구조

Fig. 2 The detailed cell

$P=ADr^{-1}$는 식 (9)와 같이 표현된다.

$$\begin{aligned} P&= ADr^{-1}=ADx^{-k}\\ &= d_0Ax^{-k}+d_1Ax^{-k+1}+\cdots+d_{k-1}Ax^{-1}\\ &\quad +d_kA+\cdots+d_{n-2}Ax^{k-1}+d_{n-1}Ax^k \end{aligned} \tag{9}$$

식 (9)에서 마지막 오른쪽 항을 보면, P는 x의 지수가 양수인 것과 음수인 것으로 구분된다. P를 다음과 같이 정의 하자.

$$P=S+T \tag{10}$$

$$S=\sum_{j=0}^{k-1} d_jAx^{-k+j} \tag{11}$$

$$T=\sum_{j=k}^{n-1} d_jAx^{-k+j} \tag{12}$$

여분기저에서 A에 x와 x^{-1}을 각각 곱한 것을 고려하자. 여분기저의 특성상 $x^n=1$이고 $x^{-1}=x^{n-1}$이므로, Ax와 Ax^{-1}은 다음 식과 같다.

$$Ax=\sum_{j=0}^{n-1} a_jx^{j+1}=\sum_{j=0}^{n-1} a_{\langle j-1\rangle}x^j \tag{13}$$

$$Ax^{-1}=\sum_{j=0}^{n-1} a_jx^{j-1}=\sum_{j=0}^{n-1} a_{\langle j+1\rangle}x^j \tag{14}$$

$A^{(i)}$와 $\overline{A}^{(i)}$를 $A^{(i)}=Ax^i$와 $\overline{A}^{(i)}=Ax^{-i}$라고 두면, $A^{(i)}$와 $\overline{A}^{(i)}$는 다음 식과 같이 표현된다.

$$A^{(i)}=\sum_{j=0}^{n-1} a_j^{(i)}x^j=Ax^i \tag{15}$$

$$\overline{A}^{(i)}=\sum_{j=0}^{n-1} \overline{a}_j^{(i)}x^j=Ax^{-i} \tag{16}$$

여기서 $A^{(i)}=\overline{A}^{(i)}=A$이다.

식 (15)와 (16)의 $A^{(i)}$와 $\overline{A}^{(i)}$는 다음과 같이 다시 표현할 수 있다.

표 1. $GF(2^m)$상의 비트-병렬 시스톨릭 AB^2 곱셈기들의 비교

Table 1. Comparison of bit-parallel systolic AB^2 multipliers over $GF(2^m)$

	Liu et al.[17]	Lee et al.[18]	Ting et al.[19]	Proposed multiplier
Area complexity				
AND_2	m^2+2m+1	m^2+2m+1	m^2	m^2+3m+2
XOR_2	m^2+2m+1	m^2+2m+1	m^2+m	m^2+4m+3
Latch	$3m^2+6m+3$	$3m^2+6m+3$	$3m^2+4m-1$	$2.25m^2+9.5m+8$
Total transistors	$66.68m^2+133.36m+6.68$	$66.68m^2+133.36m+6.68$	$66.68m^2+76m-16$	$54.68m^2+220.04m+177.36$
Time complexity				
Cell delay	0.418	0.418	0.418	0.418
Latency	$2m+2$	$m+1$	$m+1$	$0.5m+2$
Total delay	$0.836m+0.836$	$0.418m+0.418$	$0.418m+0.418$	$0.209m+0.836$
AT complexity	$55.74m^3+167.23m^2$ $+167.23m+55.74$	$27.87m^3+83.62m^2$ $+83.62m+27.87$	$27.87m^3+59.64m^2$ $+25.08m-6.69$	$11.43m^3+91.70m^2$ $+221.02m+148.27$

$$A^{(i)} = A^{(i-1)}x = \sum_{j=0}^{n-1} a_{\langle j-1\rangle}^{(i-1)} x^j \tag{17}$$

$$\overline{A}^{(i)} = \overline{A}^{(i-1)}x^{-1} = \sum_{j=0}^{n-1} \overline{a}_{\langle j+1\rangle}^{(i-1)} x^j \tag{18}$$

식 (17)과 (18)의 $A^{(i)}$와 $\overline{A}^{(i)}$를 사용하여, 식 (11)과 (12)의 T와 S는 다음 식과 같다. 여기서 식의 구조를 동일하게 구성하기 위해서, 아래 식에서 S에 $z\overline{A}^{(0)}$을 추가하였으며 $z=0$이다.

$$T = \sum_{j=k}^{n-1} d_j A x^{-k+j} = \sum_{j=0}^{k} d_{k+j} A^{(j)} \tag{19}$$

$$S = \sum_{j=0}^{k-1} d_j A x^{-k+j} = \sum_{j=0}^{k-1} d_j \overline{A}^{(k-j)} + z\overline{A}^{(0)} \tag{20}$$

식 (19)와 (20)으로부터 T와 S의 점화식 (recurrent equation)을 다음과 같이 유도할 수 있다. 여기서 $T^{(0)} = S^{(0)} = 0$이다.

$$T^{(i)} = T^{(i-1)} + d_{k+i-1}A^{(i-1)}, \text{ for } 2 \le i \le k+1, \tag{21}$$

$$S^{(i)} = \begin{cases} S^{(i-1)} + z\overline{A}^{(i-1)}, & \text{for } i=1 \\ S^{(i-1)} + d_{k-i+1}\overline{A}^{(i-1)}, & \text{for } 2 \le i \le k+1. \end{cases} \tag{22}$$

$\{A^{(i)}, T^{(i)}\}$와 $\{\overline{A}^{(i)}, S^{(i)}\}$는 서로 데이터 의존성이 없기 때문에 수식 {(17),(21)}과 {(18),(22)}는 동시에 계산이 가능하다. $T^{(k+1)}$과 $S^{(k+1)}$을 계산한 후에 AB^2은 마지막에 $P = S^{(k+1)} + T^{(k+1)}$을 계산하여 얻을 수 있다.

2. 제안하는 몽고메리 AB^2 시스톨릭 곱셈기

제안한 몽고메리 AB^2 곱셈 알고리즘을 이용해, $GF(2^4)$상의 비트-병렬 시스톨릭 곱셈기를 그림 1에서 제안한다. 여기서 "■"는 1-비트 지연소자(1-bit latch)이다. 제안한 곱셈기는 $n(n+1)/2$개 W 셀들, $n(n+1)/2$개 V 셀들, 1개의 U셀과 $4m^2+7m+3$개의 1-비트 지연소자들로 구성된다. 각 W 셀은 식 (17)과 (21)를 구현하기 위해 하나의 2-입력 AND 게이트와 하나의 2-입력 XOR 게이트로 구성되며 자세한 구조는 그림 2(a) 와 같다. 각 V 셀은 식 (18)과 식 (22)를 구현하기 위해 하나의 2-입력 AND 게이트와 하나의 2-입력 XOR 게이트로 구성되며 자세한 구조는 그림 2(b) 와 같다. U 셀은 $P = S^{(k+1)} + T^{(k+1)}$을 계산하기 위해 n개의 XOR로 구성되며, 자세한 구조는 그림 2(c)와 같다.

Ⅳ. 성능 비교 분석

본 장에서는 제안한 AB^2 곱셈기와 기존의 AB^2 곱셈기들의 성능을 분석하고 비교한다. 실제적인 자세한 비교를 위하여 "SAMSUNG STD 150 0.13μm 1.2V CMOS Standard Cell Library"'를 사용한다. 이 라이브러리에 기반하여 제안한 곱셈기와 기존의 곱셈기의 시간 및 공간 복잡도를 계산한다. A_{GATEn}이 n-입력 게이트의 트렌지스터 카운트 (transistor count)를 나타낸다면, A_{AND2} = 6.68, A_{XOR2} = 12.00, A_{LATCH} = 16.00이다. 또한 T_{GATEn}이 n-입력 게이트의 전파 지연 (propagation delay) 시간을 나타낸다면, T_{AND2} = 0.094ns, T_{XOR2} = 0.167ns, T_{LATCH} = 0.157ns이다.

표 1은 기존의 비트-병렬 곱셈기들과 제안한 곱셈기들을 비교한 것이다. Liu 등 [17], Lee 등 [18]과 Ting 등 [19]의 AB^2 곱셈기들의 트랜지스

터 카운트는 각각 $66.68m2+133.36m+6.68$, $66.68m^2+133.36m+6.68$, $66.68m^2+76m-16$이다. 제안한 AB^2 곱셈기의 트랜지스터 카운트는 $54.68m^2+220.04m+177.36$이며, 기존의 곱셈기들에 비해 약 18% 가량 감소되었다.

Liu 등 [17], Lee 등 [18], Ting 등 [19]과 제안한 AB^2 곱셈기들의 셀 처리 시간은 모두 $T_{AND2}+T_{XOR2}+T_{LATCH}$으로 동일하다. Liu 등 [17]의 곱셈기의 지연 시간은 $2m+2$, Lee 등 [18]은 $m+1$, Ting 등 [19]은 $m+1$ 클록 사이클이다. 제안한 곱셈기의 지연 시간은 $0.5m+2$ 클록 사이클로 기존의 곱셈기에 비해 절반가량 지연 시간이 감소되었다. 셀 처리 시간과 지연 시간을 같이 고려하여 전체 처리 시간을 비교하면 제안한 곱셈기는 Liu 등 [17], Lee 등 [18]과 Ting 등 [19]의 곱셈기에 비해 각각 약 75%, 50%, 50% 가량 감소되었다.

제안한 곱셈기의 시간 및 공간 복잡도면에 대해 종합적인 분석을 위해, AT product 복잡도를 비교하면, 제안한 곱셈기는 Lin 등 [17], Lee 등 [18]과 Ting 등 [19]의 곱셈기에 비해 각각 약 79%, 59%, 59% 감소되었다.

V. 결 론

본 논문은 $GF(2^m)$상에서 여분기저 기반의 몽고메리 AB^2 시스톨릭 곱셈기를 제안하였다. 기존의 곱셈기들과의 비교를 통해 트랜지스터 개수, 셀 지연 시간 및 전체 처리 지연 시간이 낮아 효율적임을 보였다. 특히 제안한 AB^2 곱셈기는 기존의 곱셈기에 비해 전체 지연 시간이 짧아 빠른 속도를 요구하는 응용에 적합하다. 따라서 제안한 곱셈기들은 기존의 곱셈기들에 비해 높은 성능을 가지며 이는 오류 정정 부호 및 암호학에서의 중요한 연산인 지수, 역원 및 나눗셈 연산의 구조에 기본적인 구조에 사용되기에 적합하다. 또한 시스톨릭 어레이의 구조적 특성에 따라, 제안한 곱셈기들은 간단한 구조와 정규성으로 인하여 VLSI 구현에 적합하다.

References

[1] A.J. Menezes, P.C. van Oorschot, S.A. Vanstone, Handbook of Applied Cryptography, Boca Raton, FL, CRC Press, 1996.

[2] R.E. Blahut, Theory and Practice of Error Control Codes, Reading, MA, Addison-Wesley, 1983.

[3] N. Kobliz, "Elliptic curve cryptography," Math. Computation, Vol. 48, No. 177, pp. 203-209, 1987.

[4] P. Montgomery, "Modular multiplication without trial division," Mathematics of Computation, Vol. 44, No. 170, pp. 519-521, 1985.

[5] C.K. Koc, T. Acar, "Montgomery multiplication in $GF(2^k)$," Designs Codes and Cryptography, vol. 14, pp. 57-69, 1998.

[6] C.Y. Lee, J.S. Horng, I.C. Jou, "Low-complexity bit-parallel systolic Montgomery multipliers for special classes of $GF(2^m)$," IEEE Transactions on Computers, Vol. 54, No. 9, pp. 1061-1070, 2005.

[7] C.W. Chiou, C.Y. Lee, A.W. Deng, J.M. Lin, "Concurrent error detection in Montgomery multiplication over $GF(2^m)$," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E89-A, No. 2, pp. 566-574, 2006.

[8] A. Hariri, A. Reyhani-Masoleh, "Bit-serial and bit-parallel Montgomery multiplication and squaring over $GF(2^m)$," IEEE Transactions on Computers, Vol. 58, No. 10, pp. 1332-45, 2009.

[9] A. Hariri, A. Reyhani-Masoleh, "Concurrent error detection in Montgomery multiplication over binary extension fields," IEEE Transactions on Computers, Vol. 60, No. 9, pp. 1341-53, 2011.

[10] K.W. Kim, W.J. Lee, "Efficient cellular automata based Montgomery AB^2 multipliers over $GF(2^m)$," IETE Technical Review, Vol. 31, No. 1, pp. 92-102, 2014.

[11] K.W. Kim, J.C. Jeon, "Polynomial basis multiplier using cellular systolic architecture," IETE Journal of Research, Vol. 60, No. 2, pp. 194-199, 2014.

[12] S.H. Choi, K.J. Lee, "Low complexity semi-systolic multiplication architecture over $GF(2^m)$," IEICE Electron. Express, Vol. 11, No. 20, pp. 20140713, 2014.

[13] K.W. Kim, J.C. Jeon, "A semi-systolic Montgomery multiplier over $GF(2^m)$," IEICE

Electonics Express, Vol. 12, No. 21, pp. 20150769, 2015.

[14] H.H. Lee, K.W. Kim, “Efficient semi-systolic finite field multiplier using redundant basis,” International Journal of Computer, Electrical, Automation, Control and Information Engineering, Vol. 10, No. 10, pp. 1614-1618, 2016.

[15] S.W. Wei, “A systolic power-sum circuit for $GF(2^m)$,” IEEE Transactions on Computers, Vol. 43, No. 2, pp. 226-229, 1994.

[16] C.L. Wang, J.H. Guo, “New systolic arrays for $C+AB^2$, inversion, and division in $GF(2^m)$,” IEEE Transactions on Computers, Vol. 49, No. 10, pp. 1120-1125, 2000.

[17] C.H. Liu, N.F. Huang, C.Y. Lee, “Computation of AB^2 multiplier in $GF(2^m)$ using an efficient low-complexity cellular architecture,” IEICE Transactions on Fundamentals of Electronics, Vol. E83-A, No. 12, pp. 2657 - 2663, 2000.

[18] C.Y. Lee, E.H. Lu, L.F. Sun, “Low-complexity bit-parallel systolic architecture for computing AB^2+C in a class of finite field $GF(2^m)$,” IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, Vol. 48, No. 5, pp. 519-523, 2001.

[19] Y.R. Ting, E.H. Lu, J.Y. Lee, “Low complexity bit-parallel systolic architecture for computing $C+AB^2$ over a class of $GF(2m)$,” INTEGRATION, the VLSI journal, Vol. 37, No. 3, pp. 167 - 176, 2004.

[20] C.Y. Lee, A.W. Chiou, J.M. Lin, “Low-complexity bit-parallel systolic architectures for computing $A(x)B^2(x)$ over $GF(2_m)$,” IEEE Proceedings of Circuits Devices and Systtems, Vol. 153, No. 4, pp. 399-406, 2006.

[21] K.W. Kim, W.J. Lee, “Low-complexity parallel and serial systolic architectures for AB^2 multiplication in $GF(2_m)$,” IETE Technical Review, Vol. 30, No. 2, pp. 134-141, 2013.

[22] K.W. Kim, W.J. Lee, "An efficient parallel systolic array for AB^2 over $GF(2_m)$," IEICE Electronics Express, Vol. 10, No. 20, pp. 20130585, 2013.

[23] K.W. Kim, W.J. Lee, “Efficient cellular automata based Montgomery AB^2 multipliers over $GF(2_m)$,” IETE Technical Review, Vol 31, No. 1, pp. 92-102, 2014.

[24] G. Drolet, “A new representation of elements of finite fields yielding small complexity arithmetic circuits,” IEEE Transactions on Computers, Vol. 47, No. 9, pp. 938-946, 1998.

[25] H. Wu, M.A. Hasan, I.F. Blake, S. Gao, “Finite field multiplier using redundant representation,” IEEE Transactions on Computers, Vol. 51, No. 11, pp. 1306-1316, 2002.

Tai Wan Kim (김 태 완)

He is an associate professor at the Design at the Pusan National University. He has the Ph.D. degree in Digital Content from Kookmin University in 2011. And a M.F.A. degree in Design & Technology from Parsons School of Design, NY. His research interests include motion graphic, augmented reality characters, digital content security and emotional interaction.

Email: ktw623@pusan.ac.kr

Kee-Won Kim (김 기 원)

He is an assistant professor at the College of Convergence Technology at the Dankook University. He has the Ph.D. degree and M.S. in Computer Engineering from Kyungpook National University in 2006 and 2001, respectively, and a B.S. degree in Computer Science & Statistics from Kyungsung University. His primary research interests include information security, security protocol, and big data analysis.

Email: nirkim@dankook.ac.kr