Journal of Knowledge Information Technology and Systems(JKITS), Vol. 9, No. 5, pp. 571~578, October 2014

Journal of The Korea Knowledge Information Technology Society

ISSN 1975-7700

http://www.kkits.or.kr

A LSB-first Word-Level Normal Basis Multiplier over $GF(2^m)$

Yong-Suk Cho[1], Jae-Yeon Choi[*2]

[1]*Department of Information & Communication Security, Youngdong University*
[2]*Department of Information & Communication Engineering, Namseoul University*

ABSTRACT

Finite field multipliers are the basic building blocks in many applications such as error-control coding, cryptography and digital signal processing. Hence, the design of efficient dedicated finite field multiplier architectures can lead to dramatic improvement on the overall system performance. In this paper, LSB-first word-level normal basis multiplier with low latency in Galois field is presented. To speed up multiplication processing, we divide the product polynomial into several parts and then process them in parallel. The proposed multiplier operates in normal basis of $GF(2^m)$ and is faster than bit serial ones but with lower area complexity than bit parallel ones. The most significant feature of the proposed architecture is that a trade-off between hardware complexity and delay time can be achieved.

© 2014 KKITS All rights reserved

KEYWORDS: Finite fields, Galois fields, Normal Basis, Multipliers, Cryptography

ARTICLE INFO: Received 7 August 2014, Revised 10 October 2014, Accepted 10 October 2014.

1. 서 론

유한체(finite field or Galois field) 상의 연산은 오류정정 부호이론 및 암호이론 등의 분야에서 널리 응용되고 있다[1]. 특히 오류정정부호 중 BCH 부호나 Reed-Solomon 부호와 같은 블록부호와 AES(Advanced Encryption Standard), ECC(Elliptic Curve Cryptosystem)와 같은 암호 알고리즘에서는 모든 연산이 유한체 $GF(2^m)$ 상에서 이루어진다.

*Corresponding author is with the Department of Information & Communication Engineering, Namseoul University, 91 Daehak-ro Seonghwan-eup Sebuk-gu Cheonan-si Chungcheongnam-do KOREA.
E-mail address: cjy@nsu.ac.kr

www.kci.go.kr

따라서 유한체 $GF(2^m)$ 상의 연산을 얼마만큼 효율적으로 수행하는가 하는 문제는 이들 시스템의 구현 시 전체 하드웨어 량에 직접적으로 영향을 미치는 매우 기본적이고 중요한 요소가 된다[2-3].

유한체 상의 연산에 있어서 효율성을 결정하는 중요한 요소 중 하나는 유한체 원소들을 표현하는데 사용하는 기저(basis)의 선택이다. 주로 사용되는 기저로는 다항식기저(polynomial basis), 쌍대기저(dual basis)[4], 정규기저(normal basis)[5] 등이 있다. 이 중에서 정규기저는 제곱 연산이 한 비트 순회치환(cyclic shift)만으로 계산될 수 있어서 하드웨어 구현 시 비용이 소요되지 않는 장점 때문에 여러 가지 타원곡선 암호시스템의 표준에서 널리 사용되고 있다[6-7]. 본 논문에서는 정규기저 상에서 동작하는 곱셈기를 설계한다.

유한체 $GF(2^m)$의 곱셈기는 비트병렬(bit-parallel) 방식, 비트직렬(bit-serial) 방식, 워드레벨(word-level) 방식의 3가지로 구현할 수 있다.

비트병렬 곱셈기는 한 클럭(clock) 내에 곱셈의 결과를 출력하는 회로이다. 비트병렬 곱셈기는 조합논리 회로로 구현된 경우에는 논리 게이트를 통과하는 지연(delay) 후에 결과를 출력하며, LUT(Look-Up Table)를 이용한 경우에는 메모리 액세스에 걸리는 지연 후에 결과를 출력한다. 이러한 병렬형의 곱셈기는 연산속도는 빠른 반면에 회로가 복잡하게 된다. 따라서 유한체의 차수 m이 매우 큰 암호 분야의 응용에는 적합하지 않다[8-9].

반면에, 비트직렬 곱셈기는 일반적으로 m 클럭 만큼의 시간 지연 후에 결과를 출력한다. 비트병렬 곱셈기는 연산속도는 빠른 반면에 회로가 복잡하며, 비트직렬 곱셈기는 회로는 간단하지만 m 클럭 만큼의 시간 지연이 생긴다[10-11].

이러한 문제점을 해결하기 위하여 회로의 복잡도와 지연 시간 사이의 적절한 절충을 꾀하는 방법이 워드레벨 곱셈기이다.

워드레벨 곱셈기는 워드직렬 비트병렬(word-serial bit-parallel) 곱셈기와 워드병렬 비트직렬(word-parallel bit-serial) 곱셈기로 나눌 수 있다. 워드직렬 비트병렬 곱셈기는 유한체의 임의의 두 원소의 곱을 나타내는 다항식을 일정한 길이의 워드 단위로 분할하여 각 워드 내부는 비트병렬 곱셈기를 사용하고 전체적으로는 직렬로 처리하는 방식이다[12-13]. 워드병렬 비트직렬 곱셈기는 워드 내부는 비트직렬 곱셈기를 사용하고 전체적으로는 병렬로 처리하는 방식이다[14-15].

본 논문에서는 유한체 $GF(2^m)$의 정규기저에서 동작하는 LSB 우선 워드레벨 곱셈기 구조를 제안한다. 제안한 곱셈기는 유한체의 곱을 표현하는 다항식을 여러 개로 분리한 다음, 이 다항식들을 동시에 처리하는 방식을 사용하여 속도를 향상시킨 것이다. 제안된 곱셈기는 기존의 직렬 곱셈기보다는 짧은 지연시간에 결과를 얻을 수 있고, 병렬 곱셈기보다는 적은 하드웨어로 구현할 수 있다. 제안한 곱셈기는 회로의 복잡도와 지연시간 사이에 적절한 절충을 꾀할 수 있는 장점을 가지고 있다.

본 논문의 구성은 먼저 2.에서 유한체 $GF(2^m)$의 정규기저에서 동작하는 LSB 우선 비트직렬 곱셈 알고리즘을 분석하고, 3.에서는 LSB 우선 워드레벨 정규기저 곱셈기를 설계한다. 또한 실제 예로써 $GF(2^5)$에서 3 클럭만에 곱셈의 결과를 출력하는 워드레벨 정규기저 곱셈기를 설계한다. 그리고 4.에서 결론을 맺는다.

2. LSB 우선 비트직렬 정규기저 곱셈기

유한체 $GF(2^m)$에서 다음과 같은 m개의 서로 독립인 원소들을 유한체 $GF(2^m)$의 정규기저라고 한다.

www.kci.go.kr

$$\{\beta^{2^0},\ \beta^{2^1},\ \beta^{2^2},\cdots\cdots,\ \beta^{2^{m-1}}\} \tag{1}$$

유한체 $GF(2^m)$의 임의의 한 원소 A를 정규기저를 이용하여 표현하면

$$A = \sum_{i=0}^{m-1} a_i \beta^{2^i}, \quad a_i \in GF(2) \tag{2}$$
$$= a_0\beta^{2^0} + a_1\beta^{2^1} + \cdots + a_{m-1}\beta^{2^{m-1}}$$

가 된다. 여기에서 $\beta^{2^m} = \beta$이므로, 식 (2)와 같은 임의의 한 원소를 제곱하면

$$A^2 = a_{m-1}\beta + a_0\beta^2 + \cdots + a_{m-2}\beta^{2^{m-1}} \tag{3}$$

가 된다. 즉, 정규기저에서 제곱 연산은 오른쪽으로 한 번 순회치환(cyclic shift)하면 구현할 수 있다.

유한체 $GF(2^m)$의 임의의 두 원소 A와 B의 곱 C는 다음과 같이 정리할 수 있다.

$$C = A \cdot B = A\left(\sum_{i=0}^{m-1} b_i \beta^{2^i}\right) \tag{4}$$
$$= A(b_0\beta + b_1\beta^{2^1} + \cdots + b_{m-1}\beta^{2^{m-1}})$$
$$= b_0(A\beta) + b_1(A\beta^{2^1}) + \cdots\cdots$$
$$+ b_{m-1}(A\beta^{2^{m-1}})$$

식 (4)의 양변을 $1/2^{m-1}$승을 하면

$$C^{1/2^{m-1}} \tag{5}$$
$$= b_0 A^{1/2^{m-1}}\beta^{1/2^{m-1}} + b_1 A^{1/2^{m-1}}\beta^{1/2^{m-2}}$$
$$+ \cdots\cdots + b_{m-1}(A^{1/2^{m-1}}\beta)$$
$$= b_0(A\beta)^{1/2^{m-1}} + b_1(A^{1/2^1}\beta)^{2/2^{m-1}}$$
$$+ \cdots\cdots + b_{m-1}(A^{1/2^{m-1}}\beta)$$

가 되고 다음과 같이 다시 정리할 수 있다.

$$C^{1/2^{m-1}} \tag{6}$$
$$= (\cdots((b_0A\beta)^{1/2} + b_1A^{1/2}\beta)^{1/2}\cdots\cdots$$
$$+ b_{m-2}A^{1/2^{m-2}}\beta)^{1/2} + b_{m-1}A^{1/2^{m-1}}\beta$$

식 (6)은 입력 A를 계속해서 1/2 승을 하면서 입력 B의 계수를 LSB부터 차례대로 β와 곱하고 그 결과를 계속해서 1/2승을 하면서 m 클럭까지 더해 가는 것이다. 두 원소의 곱 C는 식 (6)의 결과 값을 2^{m-1}승 하면 구할 수 있다. 따라서 식 (6)을 이용하면 <그림 1>과 같이 비트직렬 정규기저 곱셈기를 설계할 수 있다.

<그림 1>에서 굵은 선은 m 비트 버스이고, ▭는 m 비트 레지스터를, ⊕는 m 개의 2입력 XOR 게이트를, ⊙은 m 개의 2입력 AND 게이트를, Ⓑ는 $GF(2^m)$의 한 원소 β를 곱하는 상수 곱셈기를 나타내고 있다. 또한 점선으로 된 사각형은 2의 멱승을 수행하는 회로로 정규기저 상에서는 결선만 바꾸는 것이다. 정규기저 표현에서 1/2 승은 각 계수를 왼쪽으로 한 번 순회치환하면 되고 2^{m-1}승은 오른쪽으로 $m-1$번 순회치환하면 된다.

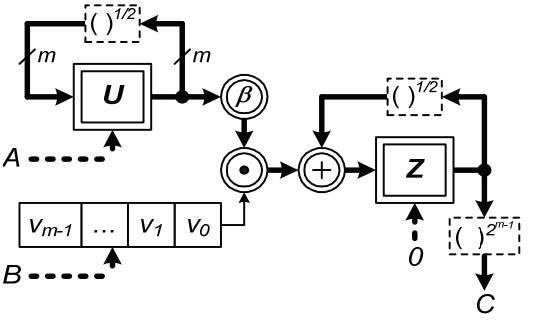

그림 1. $GF(2^m)$ 상의 비트직렬 정규기저 곱셈기
Figure 1. Bit-serial normal basis multiplier over $GF(2^m)$

www.kci.go.kr

<그림 1>의 회로는 초기 상태에서 레지스터 Z는 클리어시키고 레지스터 U에는 입력 A를, 레지스터 V에는 입력 B를 로드시킨다. 그리고 각 레지스터를 m번 치환시키면 레지스터 Z에 결과가 저장된다. 따라서 m클럭 시간 후에 곱셈의 결과를 얻을 수 있다.

3. LSB 우선 워드레벨 정규기저 곱셈기

비트직렬 곱셈기는 곱셈이 완료되어 결과를 출력할 때까지 m클럭 시간이 걸린다. 이와 같은 지연을 줄이기 위한 것이 워드레벨 곱셈기이다. 비트직렬 곱셈기를 고속화하기 위하여 식 (4)에서 곱하는 두 원소 중 하나인 원소 B를 다음과 같이 w비트씩 묶어서 $d(=\lceil m/w \rceil)$개로 분할한다.

$$\begin{aligned} C &= A \cdot B \\ &= A(b_0\beta^{2^0} + b_1\beta^{2^1} + \cdots + b_{w-1}\beta^{2^{w-1}}) \\ &+ A(b_w\beta^{2^w} + b_{w+1}\beta^{2^{w+1}} + \cdots + b_{2w-1}\beta^{2^{2w-1}}) \\ &+ \cdots\cdots \\ &+ A(b_{(d-1)w}\beta^{2^{(d-1)w}} + \cdots + b_{m-1}\beta^{2^{m-1}}) \end{aligned} \tag{7}$$

여기에서 C를 다음과 같이 정의하면

$$C \equiv C_0 + C_1 + \cdots + C_{d-1} \tag{8}$$

C의 각각의 항은 다음과 같이 정리할 수 있다.

$$C_0 = A(b_0\beta + b_1\beta^{2^1} + \cdots + b_{w-1}\beta^{2^{w-1}}) \tag{9}$$

$$C_1 = A(b_w\beta^{2^w} + \cdots + b_{2w-1}\beta^{2^{2w-1}}) \tag{10}$$

$$\vdots$$

$$C_{d-1} = A(b_{(d-1)w}\beta^{2^{(d-1)w}} + \cdots + b_m\beta^{2^m}) \tag{11}$$

식 (9)를 살펴보면 식 (4)와 동일한 구조를 가지고 있으며, 식 (10)은 β대신에 β^{2^w}를 대입하면 역시 식 (4)와 동일한 구조가 된다. 또한 식 (11)도 β대신에 $\beta^{2^{(d-1)w}}$를 대입하면 역시 식 (4)와 동일한 구조가 된다. 따라서 식 (9)~(11)을 식 (6)과 같이 정리하면 다음과 같이 된다.

$$\begin{aligned} & C_0^{\,1/2^{w-1}} \\ &= (\cdots((b_0A\beta)^{1/2} + b_1A^{\,1/2}\beta)^{1/2} \cdots\cdots \\ &\quad + b_{w-2}A^{\,1/2^{w-2}}\beta)^{1/2} + b_{w-1}A^{\,1/2^{w-1}}\beta \end{aligned} \tag{12}$$

$$\begin{aligned} & C_1^{\,1/2^{w-1}} \\ &= (\cdots((b_wA\beta^{2^w})^{1/2} + b_{w+1}A^{\,1/2}\beta^{2^w})^{1/2} + \\ &\quad \cdots\cdots + b_{2w-1}A^{\,1/2^{w-1}}\beta^{2^w} \\ &\quad \vdots \end{aligned} \tag{13}$$

$$\begin{aligned} & C_{d-1}^{\,1/2^{w-1}} \\ &= (\cdots((b_{(d-1)w}A\beta^{2^{(d-1)w}})^{1/2} + \\ &\quad \cdots\cdots + b_{2w-1}A^{\,1/2^{w-1}}\beta^{2^{(d-1)w}} \end{aligned} \tag{14}$$

식 (12)~(14)를 이용하면 <그림 2>와 같은 LSB 우선 워드레벨 정규기저 곱셈기를 설계할 수 있다. <그림 2>의 회로는 초기 상태에서 레지스터 Z는 클리어시키고 레지스터 U에는 입력 A를 로드시킨다. 레지스터 V에는 입력 B를 w비트씩 분할하여 로드시킨다. 첫 번째 클럭에서 각 워드의 첫 번째 비트인 b_0, b_w, b_{2w}, ……, $b_{(d-1)w}$가 입력되고, 두 번째 클럭에서는 각 워드의 두 번째 비트가, ……, 그리고 w번째 클럭에서는 각 워드의 마지막 비트가 입력된다. m개의 비트가 모두 입력되면 레지스터 V의 끝에는 0으로 채워진다. 따라서 w클럭 시간 후에 곱셈의 결과를 얻을 수 있다.

www.kci.go.kr

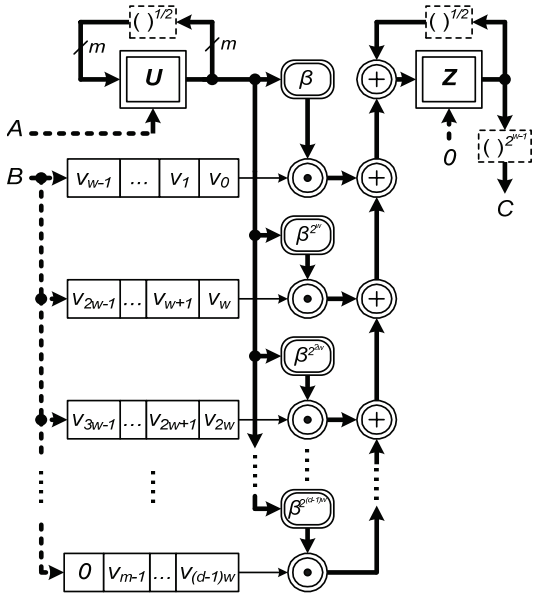

그림 2. $GF(2^m)$ 상의 LSB 우선 워드레벨 정규기저 곱셈기
Figure 2. LSB-first word-level normal basis multiplier over $GF(2^m)$

<그림 2>의 곱셈기를 <그림 1>의 곱셈기와 비교해보면, $(d-1)m$개의 2입력 XOR 게이트, $(d-1)m$개의 2입력 AND 게이트, 그리고 β^{2^w}, $\beta^{2^{2w}}$, ……, $\beta^{2^{(d-1)w}}$를 곱하는 상수 곱셈기가 더 사용되었음을 알 수 있다. 여기에서 d는 $\lceil m/w \rceil$이다.

표 1. 유한체 곱셈기들의 복잡도 비교
Table 1. Complexities Comparison between Finite Field Multipliers

구 분	비트병렬 곱셈기	비트직렬 곱셈기	제안된 곱셈기
REG	0	$3m$	$3m$
AND	m^2	m	dm
XOR	m^2-1	m	dm
상수 곱셈기	0	1	dm
CLOCK	1	m	w $(1 \le w \le m)$

그러나 <그림 2>의 곱셈기는 w 클럭 시간에 곱셈의 결과를 얻을 수 있다. 따라서 워드 길이 w를 적절히 선택하면 회로의 복잡도와 지연시간 사이에 적절한 절충이 가능하게 된다. <표 1>에 기존의 곱셈기와 제안한 곱셈기의 복잡도를 비교하였다.

실제 예로서 원시다항식이 $p(x)=1+x^2+x^5$인 유한체 $GF(2^5)$에서 $w=3$인 워드레벨 곱셈기를 설계하여 보자. $GF(2^5)$의 임의의 두 원소 A와 B의 곱 C는 다음과 같이 된다.

$$\begin{aligned} C &= A \cdot B \\ &= b_0(A\beta)+b_1(A\beta^{2^1})+b_2(A\beta^{2^2}) \\ &\quad + b_3(A\beta^{2^3})+b_4(A\beta^{2^4}) \end{aligned} \tag{15}$$

m이 5이고 w가 3이므로 d는 $\lceil m/w \rceil = \lceil 5/3 \rceil$, 즉 2가 된다. 따라서 식 (15)를 식 (8)과 같이 나누면 다음과 같이 된다.

$$C_0 = b_0 A\beta + b_1 A\beta^2 + b_2 A\beta^{2^2} \tag{16}$$

$$C_1 = b_3 A\beta^{2^3} + b_4 A\beta^{2^4} + 0\,A\beta^{2^5} \tag{17}$$

여기에서 식 (16)과 식 (17)을 식 (12)~(14)와 같이 다시 정리하면 다음과 같이 된다.

$$\begin{aligned} C_0^{1/2^2} &= ((b_0 A\beta)^{1/2} + b_1 A^{1/2^1}\beta)^{1/2} \\ &\quad + b_2 A^{1/2^2}\beta \end{aligned} \tag{18}$$

$$\begin{aligned} C_1^{1/2^2} &= ((b_3 A\beta^{2^3})^{1/2} + b_4 A^{1/2^1}\beta^{2^3})^{1/2} \\ &\quad + 0\,A^{1/2^2}\beta^{2^3} \end{aligned} \tag{19}$$

$GF(2^5)$의 임의의 한 원소 A에 $\beta(=\alpha^5)$와 $\beta^{2^w}=\beta^{2^3}$을 곱하여 정리하면 다음과 같이 된다.

www.kci.go.kr

$$A \cdot \beta = a_1\beta + (a_0 + a_3)\beta^2 + (a_3 + a_4)\beta^4 + (a_1 + a_2)\beta^8 + (a_2 + a_4)\beta^{16} \quad (20)$$

$$A \cdot \beta^{2^3} = (a_1 + a_2)\beta + (a_0 + a_4)\beta^2 + (a_0 + a_2)\beta^4 + a_4\beta^8 + (a_1 + a_3)\beta^{16} \quad (21)$$

따라서 식 (18)~(21)을 이용하면 ,<그림 3>과 같이 $GF(2^5)$에서 $w = 3$인 LSB 우선 워드레벨 정규기저 곱셈기를 설계할 수 있다. <그림 3>의 곱셈기는 3클럭만에 곱셈의 결과를 출력한다.

4. 결 론

본 논문에서는 유한체 $GF(2^m)$의 정규기저 상에서의 곱셈에 있어서, 곱하는 임의의 한 원소를 $w(1 \le w \le m)$ 비트씩 묶어서 d개로 나눈 다음, 각각의 항들을 동시에 병렬로 처리하는 방식을 사용하여 w클럭만에 곱셈의 결과를 얻을 수 있는 LSB 우선 워드레벨 정규기저 곱셈기를 설계하였다.

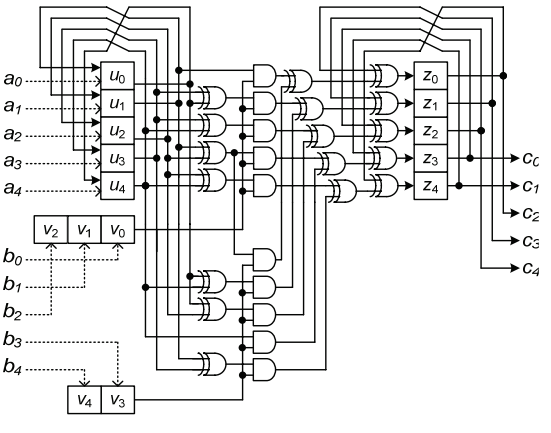

그림 3. $GF(2^5)$ 상의 LSB 우선 워드레벨 정규기저 곱셈기
Figure 3. LSB-first word-level normal basis multiplier over $GF(2^5)$

설계된 곱셈기는 비트직렬 곱셈기의 긴 지연시간과 비트병렬 곱셈기의 복잡한 회로 사이를 적절하게 절충함으로써, 비트직렬 곱셈기보다는 짧은 지연시간에 결과를 얻을 수 있으며 비트병렬 곱셈기보다는 적은 하드웨어로 구현할 수 있는 장점을 가지고 있다.

References

[1] R. Lidl, and H. Niederreiter, *Introduction to finite fields and their applications*, Cambridge University Press, 1994.

[2] Man Young Rhee, *Error-correcting coding theory*, Mcgraw-Hill, 1989.

[3] D. Hankerson, A. Menezes, and S. Vanstone, *Guide to elliptic curve cryptography*, Springer-Verlag, 2004.

[4] E. R. Berlekamp, *Bit-serial reed-solomon encoders*, IEEE Transactions on Information Theory, Vol. 28, No. 6, pp. 869-874, 1982.

[5] J. K. Omura, and J. L. Massey, *Computational method and apparatus for finite field arithmetic*, U.S. Patent #4,587,627, 1986.

[6] IEEE Std 1363-2000, *IEEE Standard Specifications for public-key cryptography*, 2000.

[7] National Institute of Standards and Technology, *Digital signature standard*, FIPS Publications 186-3, 2009.

[8] A. Reyhani-Masoleh, and M. A. Hasan, *A new construction of Massey-Omura parallel multiplier over GF(2^m)*, IEEE Transactions on Computers, Vol. 51, No. 5, pp. 511–520, 2002.

[9] C. K. Koc, and B. Sunar, *Low-complexity bit-parallel canonical and normal basis multipliers for a class of finite fields*, IEEE Transactions on Computers, Vol. 47, No. 3, pp. 353–356, 1998.

www.kci.go.kr

[10] T. Beth, and D. Gollmann, *Algorithm engineering for public key algorithms*, IEEE Journal on Selected Areas in Communications, Vol. 7, No. 4, pp. 458-466, 1989.

[11] G. B. Agnew, R. C. Mullin, I. M. Onyszchuck, and S. A. Vanstone, *An implementation for a fast public-key cryptosystem*, Journal of Cryptology, Vol. 3, pp. 63-79, 1991.

[12] L. Song, and K. K. Parhi, *Low-energy digit-serial/parallel finite field multipliers*, Journal of VLSI Signal Processing, Vol. 19, pp. 149-166, 1998.

[13] W. Tang, H. Wu, and M. Ahmadi, *VLSI Implementation of bit-parallel word-serial multiplier in GF(2^{233})*, Proceedings of The 3rd International IEEE-NEWCAS Conference, Xi'an, China, pp. 399–402, 2005.

[14] A. H. Namin, H. Wu, and M. Ahmadi, *Comb architectures for finite field multiplication in F_{2^m}*, IEEE Transactions on Computers, Vol. 56, No. 7, pp. 909-916, 2007.

[15] Y. S. Cho and J. Y. Choi, *A hardware implementation of word-parallel bit-serial polynomial basis multiplier*, GDC/IESH/ CGAG 2012, Communications in Computer and Information Science, Vol. 351, pp. 176-181, 2012.

유한체 $GF(2^m)$의 정규기저를 이용한 LSB 우선 워드 레벨 곱셈기

조용석[1], 최재연[2]

[1] *영동대학교 정보통신보안학과*

[2] *남서울대학교 정보통신공학과*

요 약

유한체 상의 곱셈기는, 오류제어부호, 암호 시스템, 디지털 신호처리 등과 같은 여러 분야에서 기본적인 구성 요소로 사용되고 있다. 그러므로 효율적인 구조를 갖는 유한체 상의 곱셈기를 설계하면 전체적인 시스템의 성능을 대폭 향상시킬 수 있다. 본 논문에서는 기존의 비트직렬 유한체 곱셈기에 비해 짧은 지연시간을 갖는 LSB 우선 워드레벨 정규기저 곱셈기 구조를 제안한다. 제안한 곱셈기는 유한체의 곱을 표현하는 다항식을 여러 개로 분리한 다음, 이 다항식들을 동시에 처리하는 방식을 사용하여 속도를 향상시킨 것이다. 제안된 곱셈기는 유한체 $GF(2^m)$의 정규기저 상에서 동작하며, 기존의 비트직렬 곱셈기보다는 짧은 지연시간에 결과를 얻을 수 있고, 비트병렬 곱셈기보다는 적은 하드웨어로 구현할 수 있다. 제안한 곱셈기는 회로의 복잡도와 지연시간 사이에 적절한 절충을 꾀할 수 있는 장점을 가지고 있다.

Yong-Suk Cho received the B.S., M.S., and Ph.D. degree in the Department of Electronic Communication Engineering from Hanyang University in 1986, 1988 and 1998, respectively. From 1989 to 1996, he was a researcher at Korea Telecom. He has been a professor in the Department of Information & Communication Security at Youngdong University since 1996. His current research interests include finite field arithmetic, cryptography, and error-control coding.

E-mail address: yscho@yd.ac.kr

www.kci.go.kr

Jae-Yeon Choi received the B.S., M.S., and Ph.D. degree in the Department of Electronic Communication Engineering from Hanyang University in 1985, 1987 and 1998, respectively. From 1987 to 1989, he was a researcher at Samsung Advanced Institute of Technology. He was a researcher at LG Information & Communication Research Center from 1989 to 1992. He has been a professor in the Department of Information & Communication Engineering at Namseoul University since 1996. His current research interests include finite field arithmetic, cryptography, and error-control coding.

E-mail address: cjy@nsu.ac.kr

www.kci.go.kr