다항식기저를 이용한 $GF(2^m)$ 상의 디지트병렬/비트직렬 곱셈기

정회원 조 용 석*

Digit-Parallel/Bit-Serial Multiplier for $GF(2^m)$ Using Polynomial Basis

Yong-suk Cho* ***Regular Member***

요 약

본 논문에서는 $GF(2^m)$ 상에서 기존의 비트직렬 곱셈기에 비해 짧은 지연 시간을 갖는 새로운 디지트병렬/비트직렬 곱셈기를 제안한다. 제안된 곱셈기는 유한체 $GF(2^m)$의 다항식기저 상에서 동작하며, D 클럭 사이클마다 곱셈의 결과를 출력한다. 여기에서 D는 디지트의 크기이다. 제안된 곱셈기는 기존의 비트직렬 곱셈기 보다는 짧은 지연시간에 곱셈의 결과를 얻을 수 있고, 비트병렬 곱셈기 보다는 적은 하드웨어로 구현할 수 있다. 따라서 회로의 복잡도와 지연시간 사이에 적절한 절충을 꾀할 수 있는 장점을 가지고 있다.

Key Words : Finite fields, Galois fields, Multiplier, Error correcting codes, Cryptography

ABSTRACT

In this paper, a new architecture for digit-parallel/bit-serial $GF(2^m)$ multiplier with low latency is proposed. The proposed multiplier operates in polynomial basis of $GF(2^m)$ and produces multiplication results at a rate of one per D clock cycles, where D is the selected digit size. The digit-parallel/bit-serial multiplier 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.

I. 서 론

유한체(finite fields or Galois fields)는 오류정정부호(error correcting codes), 암호화(cryptography), 디지털 신호처리 등과 같은 여러 분야에서 널리 사용되고 있다. 특히 오류정정부호 중 BCH부호와 Reed-Solomon부호, 공개키 암호 알고리즘 중 타원곡선 암호시스템(Elliptic Curve Cryptosystem) 등은 모든 연산이 유한체 상에서 이루어진다. 따라서 유한체 상의 연산은 이들 시스템의 구현 시, 전체 회로의 규모와 성능에 절대적인 영향을 미친다[1]-[3].

유한체 $GF(2^m)$은 2^m개의 유한한 개수의 원소를 갖는 4칙 연산이 정의되는 체(field)이며, 2개의 원소 0과 1을 갖는 유한체 $GF(2)$의 확대체(extension field)이다. 이와 같은 2진체(binary field)에서는 덧셈과 뺄셈은 동일한 연산으로 XOR (exclusive OR) 연산으로 정의되며, 곱셈은 AND 연산으로 정의된다. 유한체 $GF(2^m)$의 원소들은 유한체 $GF(2)$의 계수로 이루어진 $m-1$차 이하의 다항식으로 표현할 수 있다. 이와 같은 다항식표현 방법에서 덧셈은 비트별 XOR로 쉽게 구현할 수

* 영동대학교 정보통신사이버경찰학과 (yscho@youngdong.ac.kr)
논문번호 : KICS2007-11-515, 접수일자 : 20007년 11월 17일, 최종논문접수일자 : 2008년 10월 21일

있는 반면에 곱셈과 나눗셈은 상당히 복잡하게 된다. 일반적으로 나눗셈은 지수승과 곱셈의 반복으로 구현할 수 있으므로 곱셈이 유한체 연산 중에서 가장 핵심이 되는 연산이 된다[4].

따라서 유한체 $GF(2^m)$ 상에서 곱셈을 효율적으로 실행하는 방법을 찾아내려는 연구들이 집중적으로 이루어지고 있다. 대표적인 것으로, 기존에 사용되던 다항식기저(polynomial basis) 대신에 쌍대기저(dual basis)를 이용한 Berlekamp[5]의 곱셈 알고리즘과, 정규기저(normal basis)를 이용한 Massey와 Omura[6]의 곱셈 알고리즘을 들 수 있다. 이 알고리즘들은 다항식 기저를 적절히 변환하여 소요되는 하드웨어 및 지연시간을 줄이고자 하는 방법들로, 이들의 개선에 관한 많은 연구들이 발표되고 있다. 그러나 쌍대기저나 정규기저를 이용하면 기저 변환이 필요하게 되는 단점이 있다. 본 논문에서는 다항식기저 상에서 동작하는 곱셈기를 설계한다.

유한체 $GF(2^m)$ 상의 곱셈기는 비트병렬 곱셈기(bit-parallel multiplier)와 비트직렬 곱셈기(bit-serial multiplier)로 구현할 수 있다. 비트병렬 곱셈기는 한 클럭(clock) 내에 곱셈의 결과를 출력하는 회로이며, 비트직렬 곱셈기는 일반적으로 m 클럭만큼의 시간 지연 후에 결과를 출력한다. 비트병렬 곱셈기는 연산속도는 빠른 반면에 회로가 복잡하며, 비트직렬 곱셈기는 회로는 간단하지만 m 클럭만큼의 시간 지연이 생긴다.

이러한 문제점을 해결하기 위하여 회로의 복잡도와 지연 시간 사이의 적절한 절충을 꾀하는 방법들이 발표되고 있다. 즉 기존의 비트직렬 곱셈기보다는 짧은 지연시간에 결과를 얻을 수 있으며, 비트병렬 곱셈기보다는 적은 하드웨어로 구현할 수 있는 방법들이 연구되고 있다.

조용석 등[7]은 유한체 $GF(2^m)$이 부분체(subfield)를 가지는 경우, 이 부분체 상의 비트병렬 연산기들을 이용하여 그 확대체 상의 직렬 곱셈기를 구성하는 방법을 제안하였으며, Paar 등[8]은 이 방법을 확장시켰다. 이와 같은 곱셈기를 하이브리드 곱셈기(hybrid multiplier)라고 하는 데, 이 곱셈기는 기존의 비트병렬 곱셈기에 비해 적은 하드웨어로 구현할 수 있고 비트직렬 곱셈기 보다는 빠른 시간에 곱셈의 결과를 얻을 수 있다. 그러나 이 하이브리드 곱셈기는 유한체의 차수(order) m이 합성수(composite number)일 때에만 적용이 가능하다는 제약이 따른다.

본 논문에서는 이러한 제약이 없이 모든 유한체에 적용이 가능하며, 기존의 비트직렬 곱셈기의 긴 지연시간과 비트병렬 곱셈기의 높은 회로 복잡도 사이에 적절한 절충이 가능한 새로운 디지트병렬/비트직렬 곱셈기를 설계한다. 디지트병렬/비트직렬 곱셈기는 데이터를 일정한 길이의 디지트로 나누고, 디지트 내부는 비트직렬 곱셈기를 사용하고 전체적으로는 디지트병렬 방식으로 곱셈을 처리한다. 데이터의 길이가 m 비트이고 디지트의 길이를 D 비트라고 하면 디지트 개수 d는 $\lceil m/D \rceil$ 가 된다. 디지트병렬/비트직렬 곱셈기는 D 클럭 만에 곱셈의 결과를 얻을 수 있다.

본 논문의 구성은, 먼저 Ⅱ.에서 유한체 상의 비트직렬 곱셈 알고리즘을 분석하고, Ⅲ.에서는 회로의 복잡도와 지연시간 사이의 적절한 절충을 꾀할 수 있는 새로운 디지트병렬/비트직렬 곱셈기를 설계한다. 그리고 Ⅳ.에서 결론을 맺는다.

Ⅱ. 유한체 $GF(2^m)$ 상의 비트직렬 곱셈기

모든 유한체 $GF(2^m)$은 영원(zero element), 단위원(unit element), 원시원(primitive element)을 가지고 있으며, 다음과 같은 차수가 m인 원시다항식(primitive polynomial)을 최소한 1개 이상 가지고 있다[3].

$$p(x) = 1 + p_1 x + \cdots\cdots + p_{m-1} x^{m-1} + x^m \quad (1)$$
$$, \; p_i \in GF(2)$$

유한체 $GF(2^m)$의 원시원을 α라고 하고, 이 α를 식 (1)과 같은 원시다항식의 근(root)으로 정의하면, $p(\alpha) = 0$이므로

$$\alpha^m = 1 + p_1 \alpha + \cdots\cdots + p_{m-1} \alpha^{m-1} \quad (2)$$

가 된다. 따라서 식 (2)를 이용하면 유한체 $GF(2^m)$의 0이 아닌 모든 원소들은 차수가 $m-1$ 이하인 α의 다항식으로 표현할 수 있다. 이러한 표현방식을 다항식표현(polynomial representation)이라고 한다. 즉 유한체 $GF(2^m)$ 상의 임의의 한 원소 U는 다음과 같이 쓸 수 있다.

$$U = u_0 + u_1 \alpha + \cdots\cdots + u_{m-1} \alpha^{m-1} \quad (3)$$
$$= \sum_{i=0}^{m-1} u_i \alpha^i, \quad u_i \in GF(2)$$

여기에서 다음과 같은 m개의 서로 독립인 원

소들을 유한체 $GF(2^m)$의 다항식기저(polynomial basis)라고 한다.

$$\{1, \alpha, \alpha^2, \cdots\cdots, \alpha^{m-2}, \alpha^{m-1}\} \quad (4)$$

유한체 $GF(2^m)$ 상의 임의의 두 원소 A와 B를 식 (3)과 같이 다항식표현으로 나타내면

$$A = a_0 + a_1\alpha + \cdots\cdots + a_{m-1}\alpha^{m-1} \quad (5)$$

$$B = b_0 + b_1\alpha + \cdots\cdots + b_{m-1}\alpha^{m-1} \quad (6)$$

가 되며, 이 두 원소의 곱을 Z라 하면 Z는

$$\begin{aligned} Z &= A \cdot B \\ &= A \cdot (b_0 + b_1\alpha + \cdots\cdots + b_{m-1}\alpha^{m-1}) \end{aligned} \quad (7)$$

가 된다. 또한 식 (7)을 다시 정리하면 다음과 같이 쓸 수 있다.

$$\begin{aligned} Z = {} & b_0A + b_1[A\alpha] + b_2[A\alpha^2] \\ & + \cdots\cdots + b_{m-1}[A\alpha^{m-1}] \end{aligned} \quad (8)$$

식 (8)을 살펴보면, 두 원소의 곱 Z는 임의의 한 원소 A에 α를 계속 곱해 가면서 B의 계수들을 차례로 곱하여 더하는 것이다. 따라서 식 (8)을 이용하면 그림 1과 같은 비트직렬 곱셈기를 설계할 수 있다.

그림 1에서 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 원시원 α를 곱하는 상수곱셈기를 나타내고 있다. 그림 1 회로의 동작은 초기 상태에서 레지스터 Z는 클리어시키고 임의의 두 원소 A와 B를 각각 레지스터 A와 B에 로드시킨다. 그리고 각 레지스터를 m번 쉬프트 시키면 레지스터 Z에 두 원소의 곱 Z가 저장된다. 따라서 m 클럭 시간에 곱셈의 결과를 얻을 수 있다.

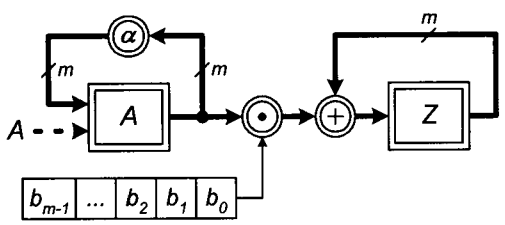

그림 1. $GF(2^m)$ 상의 비트직렬 곱셈기

Ⅲ. $GF(2^m)$ 상의 디지트병렬/비트직렬 곱셈기

그림 1과 같은 비트직렬 곱셈기는 m 클럭 시간 후에 곱셈의 결과가 나온다. 이를 고속화하기 위하여 식 (7)에서 원소 B를 다음과 같이 D 비트씩 묶어서 $d = \lceil m/D \rceil$ 개로 분할한다.

$$B = \sum_{i=0}^{d-1} B_i(\alpha^D)^i \quad (9)$$

여기에서 B_i는 다음과 같이 쓸 수 있다.

$$B_i = \sum_{j=0}^{D-1} (b_{iD+j})\alpha^j \quad (10)$$

따라서 식 (9)를 식 (7)에 대입하면

$$\begin{aligned} Z = A \cdot B &= A \cdot \left(\sum_{i=0}^{d-1} B_i\alpha^{iD}\right) \\ &= \sum_{i=0}^{d-1} (A\alpha^{iD})B_i \end{aligned} \quad (11)$$

가 된다. 여기에서 식 (11)을 풀어쓰면 다음과 같이 쓸 수 있다.

$$\begin{aligned} Z = {} & AB_0 + (A\alpha^D)B_1 + (A\alpha^{2D})B_2 \\ & + \cdots\cdots + (A\alpha^{(d-1)D})B_{d-1} \end{aligned} \quad (12)$$

식 (12)의 첫 번째 항에 식 (10)을 대입하여 다시 쓰면

$$\begin{aligned} AB_0 &= A\left(\sum_{j=0}^{D-1} b_j\alpha^j\right) \\ &= A(b_0 + b_1\alpha + \cdots\cdots + b_{D-1}\alpha^{D-1}) \end{aligned} \quad (13)$$

가 된다. 식 (13)은 식 (7)과 동일한 구조를 가지고 있다. 그러므로 식 (13)은 그림 1과 같은 비트직렬 곱셈기로 구현할 수 있다. 또한 식 (12)의 두 번째 항은 A 대신 $A\alpha^D$를 대입하면 식 (13)과 동일한 구조가 된다. 따라서 식 (12)를 이용하면 그림 2와 같은 디지트병렬/비트직렬 곱셈기를 설계할 수 있다.

그림 1에서와 마찬가지로 그림 2에서도 굵은 선은 m 비트 버스이고, □는 m 비트 레지스터를, ⊕는 m 개의 2입력 XOR 게이트를, ⊙은 m 개의 2입력 AND 게이트를, ⓐ는 α^i를 곱하는 상수곱셈기를 나타내고 있다.

그림 2의 회로는 초기 상태에서 레지스터 Z를

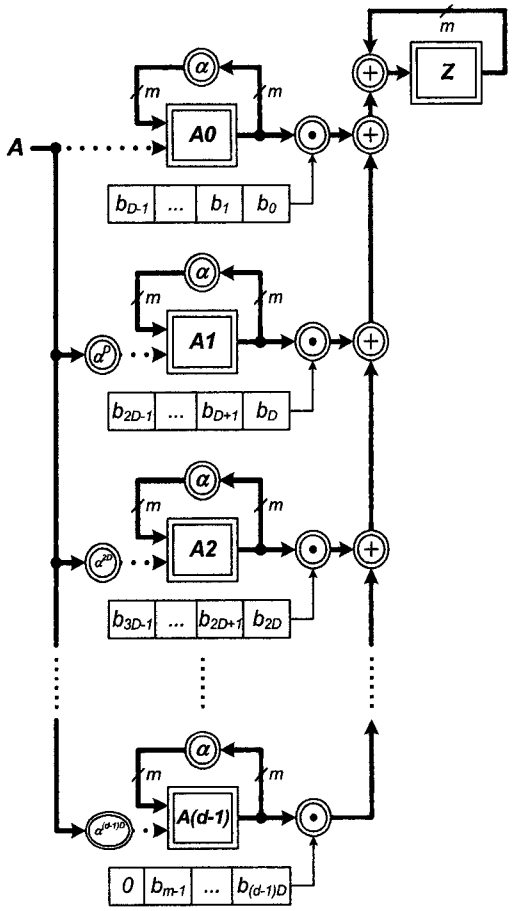

그림 2. $GF(2^m)$ 상의 디지트병렬/비트직렬 곱셈기

클리어시키고 첫 번째 레지스터에는 A를 로드시키고, 두 번째 레지스터에는 $A\alpha^D$를, 세 번째 레지스터에는 $A\alpha^{2D}$를, ……, 마지막 레지스터에는 $A\alpha^{(d-1)D}$을 로드시킨다. 그리고 각 레지스터를 D번 쉬프트 시키면 레지스터 Z에 두 원소를 곱한 결과가 저장된다. 따라서 D클럭 시간에 곱셈의 결과를 얻을 수 있다.

그림 2의 곱셈기를 그림 1의 곱셈기와 비교해 보면, $(d-1)$개의 m비트 레지스터와 $(d-1)m$개의 2입력 XOR 게이트, $(d-1)m$개의 2입력 AND 게이트, 그리고 $(d-1)$개의 α 곱셈기와 α^D, α^{2D}, ……, $\alpha^{(d-1)D}$ 곱셈기가 더 사용되었음을 알 수 있다. 여기에서 d는 $\lceil m/D \rceil$이다. 그러나 그림 2의 곱셈기는 D클럭 시간에 곱셈의 결과를 얻을 수 있다. 따라서 디지트 길이 D를 적절히 선택하면 회로의 복잡도와 지연시간 사이에 적절한 절충을 도모할 수 있게 된다. 표 1에 기존의 구조와 제안된 구조를 비교하였다.

표 1. 유한체 곱셈기들의 성능 비교

	비트병렬 곱셈기	비트직렬 곱셈기	제안된 곱셈기
REG	0	$3m$	$(d+2)m$
AND	m^2	m	dm
XOR	m^2-1	m	dm
상수곱셈기	0	1	$2d-1$
CLOCK	1	m	D

예를 들어 원시다항식이 $p(x)=1+x^2+x^5$인 유한체 $GF(2^5)$에서 $D=3$인 경우의 디지트병렬/비트직렬 곱셈기를 설계하여보자. 임의의 한 원소 B를 식 (9)와 같이 정리하면 다음과 같이 된다.

$$\begin{aligned} B &= b_0 + b_1\alpha + b_2\alpha^2 + b_3\alpha^3 + b_4\alpha^4 \\ &= (b_0 + b_1\alpha + b_2\alpha^2) \\ &\quad + (b_3 + b_4\alpha + 0\alpha^2)\alpha^3 \end{aligned} \tag{14}$$

따라서 식 (14)를 식 (12)에 대입하여 정리하면 다음과 같이 쓸 수 있다.

$$\begin{aligned} Z &= AB_0 + (A\alpha^3)B_1 \\ &= A(b_0 + b_1\alpha + b_2\alpha^2) \\ &\quad + A\alpha^3(b_3 + b_4\alpha + 0\alpha^2) \end{aligned} \tag{15}$$

$GF(2^5)$ 상의 임의의 한 원소 A에 α를 곱하여 정리하면 다음과 같이 쓸 수 있다.

$$\begin{aligned} A\alpha &= a_0\alpha + a_1\alpha^2 + a_2\alpha^3 \\ &\quad + a_3\alpha^4 + a_4(1+\alpha^2) \\ &= a_4 + a_0\alpha + (a_1 + a_4)\alpha^2 \\ &\quad + a_2\alpha^3 + a_3\alpha^4 \end{aligned} \tag{16}$$

또한 $GF(2^5)$ 상의 임의의 한 원소 A에 α^3을 곱하면 다음과 같이 된다.

$$\begin{aligned} A\alpha^3 &= a_0\alpha^3 + a_1\alpha^4 + a_2(1+\alpha^2) \\ &\quad + a_3(\alpha+\alpha^3) + a_4(\alpha^2+\alpha^4) \\ &= a_2 + a_3\alpha + (a_2 + a_4)\alpha^2 \\ &\quad + (a_0 + a_3)\alpha^3 + (a_1 + a_4)\alpha^4 \end{aligned} \tag{17}$$

따라서 식 (15), 식 (16), 식 (17)을 이용하면 $GF(2^5)$에서 $D=3$인 경우의 디지트병렬/비트직렬 곱셈기를 그림 3과 같이 설계할 수 있다.

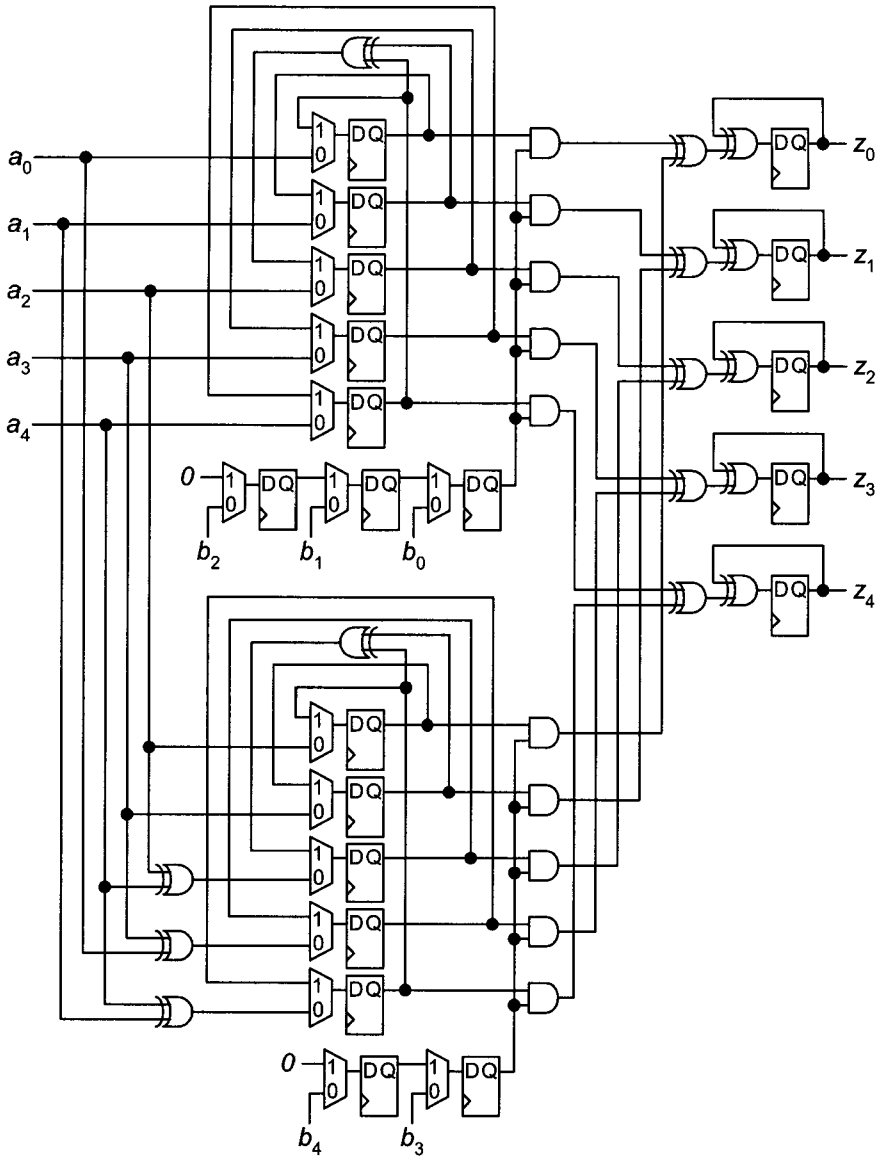

그림 3. $GF(2^5)$ 상의 $D=3$ 디지트병렬/비트직렬 곱셈기

Ⅳ. 결 론

본 논문에서는 유한체의 표준기저 상에서의 곱셈에 있어서 곱하는 임의의 한 원소를 D 비트씩의 디지트로 나눈 다음, 각각의 항들을 동시에 병렬 처리하는 방식을 사용하여 D 클럭만에 곱셈의 결과를 얻을 수 있는 새로운 디지트병렬/비트직렬 곱셈기를 제안하였다.

제안된 곱셈기는 하이브리드 곱셈기[7],[8]와 유사하게, 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 있다. 그러나 하이브리드 곱셈기는 사용하는 유한체의 차수가 합성수 이어야 한다는 제약이 있는 반면에 제안된 곱셈기 구조는 모든 유한체를 선택할 수 있다는 장점을 가지고 있다.

참 고 문 헌

[1] 이만영, *BCH 부호와 Reed-Solomon 부호*, 민음

사, 1988.

[2] M. Benaissa and W. M. Lim, "Design of Flexible $GF(2^m)$ Elliptic Curve Cryptography Processors," *IEEE Transactions on VLSI Systems*, Vol.14, No.6, pp.659-662, June 2006.

[3] S. Lin and D. Costello, *Error Control Coding: Fundamentals and Applications*, Pearson Prentice-Hall, 2004, 2nd ed.

[4] R. Lidl and H. Niederreiter, *Finite Fields*, Reading, Mass., Addison-Wesley, 1983.

[5] E. R. Berlekamp, "Bit-Serial Reed-Solomon Encoders," *IEEE Transactions on Information Theory*, Vol.28, pp.869-874, November 1982.

[6] 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 Transactions on Computers*, Vol.34, No.8, pp.709-716, August 1985.

[7] Yong Suk Cho and Sang Kyu Park, "Design of $GF(2^m)$ Multiplier Using Its Subfields," *Electronics Letters*, Vol.34, No.7, pp.650-651, April 1998.

[8] C. Paar, P. Fleischmann, P. Soria-Rodriguez, "Fast Arithmetic for Public-Key Algorithms in Galois Fields with Composite Exponents," *IEEE Transactions on Computers*, Vol.48, No.10, pp. 1025-1034, October 1999.

조 용 석 (Yong-suk Cho) 정회원

1986년 2월 한양대학교 전자통신공학과 졸업

1988년 2월 한양대학교 전자통신공학과 석사

1998년 8월 한양대학교 전자통신공학과 박사

1989년 4월~1996년 2월 한국전기통신공사 연구개발단 전임연구원

1996년 3월~현재 영동대학교 정보통신사이버경찰학과 부교수

<관심분야> 유한체연산, 오류정정부호, 암호시스템