論文 91-28B-10-7

GF(2^m) 상의 유한체 승산기 설계 및 비교

(A Design and Comparison of Finite Field Multipliers over GF(2^m))

金 在 汶* 李 晩 榮*

(Jae Moon Kim and Man Young Rhee)

要 約

본 논문에서는 쌍대기저(dual basis), 정규기저(normal basis), 및 부분체(subfield)를 이용한 세가지 다른 유한체 승산기에 대해 제시한다. 첫째, Berlekamp의 비트 직렬승산알고리듬을 기초로하여 확장된 쌍대기저 승산기를 제안한다. 둘째, 정규기저표현에 의한 Massey-Omura 승산기의 자세한 설명과 설계방법에 대해 기술한다. 세째, 부분체 GF($2^{m/2}$)을 이용한 GF(2^m)상의 승산알고리듬을 제안한다. 특히, 이 세가지 승산기를 유한체 GF(2^4)에 대해 설계하였고 복잡도면에서 다른 승산기와 비교한다. 비교결과 확장된 쌍대기저 승산기가 가장 적은 Gate로 구성되고, 부분체를 이용한 승산기의 경우에는규칙성, 간단성, 및 모듈성을 가지므로 차수(order)가 크고($m \geq 8$) 부분체가 m/2일때 다른 승산기에 비해 회로설계면에서 용이함을 알 수 있다.

Abstract

Utilizing dual basis, normal basis, and subfield representation, three different finite field muitipliers are presented in this paper. First, we propose an extended dual basis multiplier based on Berlekamp's bit-serial multiplication algorithm. Second, a detailed explanation and design of the Massey-Omura multiplier based on a normal basis representation is described. Third, the mu.tiplication algorithm over GF(2^n) utilizing subfield is proposed. Especially, three different multipliers are designed over the finite field GF(2^4) and the complexity of each multiplier is compared with that of others. As a result of comparison, we recognize that the extended dual basis multiplier requires the smallest number of gates, whereas the subfield multiplier, due to its regularity, simplicity, and modularlity, is easier to implement than the others with respect to higher ($m \geq 8$) order and m/2 subfield order.

*正會員, 漢陽大學校 電子通信工學科
(Dept. of Elec. Comm. Eng., Hanyang Univ.)
接受日字 : 1991年 5月 6日
(※ 이 논문은 대우재단 (Post-Graduate 장학재단)의 후원으로 수행되었음.)

I. 서 론

최근 유한체(finite field) 연산은 암호이론(cryptography), 부호이론(coding theory) 및 스위칭 이론(switching theory)등에서 폭넓게 이용되고 있다.[1,2] 이중 오류정정부호의 경우 유한체 GF(2^m)상에서 이루어 지는 연산은 실제 부호기 및 복호기 설계시 전

체 시스템의 규모와 성능에 절대적인 영향을 미치므로 회로경로연결(wire routing), 시스템구조의 복잡성(complexity), 및 동시성(concurrence)등의 문제점을 개선하기 위한 연구가 활발히 진행중이다.

특히, 유한체 GF(2^m)상의 연산중 가장 많은 계산량이 요구되는 승산은 적은 규모와 높은 성능을 갖고 VLSI화에 적합케할 목적으로 연구되고 있으며 이러한 이유로 기저(basis)를 달리하는 다양한 연산방법들이 도입되고 있다.

유한체상의 원소를 표현하는 기저에는 크게 표준기저, 정규기저, 쌍대기저가 있다. 가장 관용적인 표준기저에 의한 승산알고리듬으로는 B.A.Laws 등이 제안한 Cellular-Array 승산기[4]와 C.S.Yeh등의 Systolic승산기[5]등이 있으며 정규기저표현에 의한 승산알고리듬에는 C.Wang등이 Massey-Omura 승산기를 VLSI화 시킨것[6]과 G.L.Feng이 제안한 방법[8] 등이 잘 알려져 있다. 그리고 쌍대기저를 이용한 승산알고리듬에는 E.R.Berlekamp에 의한 비트 직렬승산(bit-serial multiplication) 알고리듬[9]이 있다.

본 논문에서는 우선 Berlekamp에 의한 비트 직렬승산알고리듬을 확장하여 쌍대기저로 표현된 두 원소간에 이루어지는 승산알고리듬에 대해 제안한다. 그리고 정규기저에의한 승산기와 부분체를 이용한 새로운 알고리듬을 소개하고 m=4인 유한체 GF(2^4)상의 승산기를 각각에 대해 구성하였다. 그리고 B.A.Laws와 C.S.Yeh등이 제안한 승산기와 본 논문에서 다룬 세가지 승산기를 소요되는 게이트수 및 연산처리시간으로 비교하였다. 그 결과 확장된 직렬승산알고리듬과 부분체를 이용한 승산기가 회로소자 및 연산처리시간에서 우수함을 보였다.

Ⅱ. 쌍대기저(dual basis)를 이용한 승산기

Berlekamp는 Trace 개념을 도입한 쌍대기저를 이용하여 GF(2^m)상의 임의의 한 원소와 이미 알고 있는 한 원소간의 승산을 행하는 비트 직렬승산알고리듬에 대해 제안하였다.[9]

이 승산 알고리즘은 피승수(multiplicand)는 쌍대기저로 승수(multiplier)는 표준기저로 각각 표현되어 승산이 이루어 지는 것으로 Reed-Solomon 부호등의 부호기 및 복호기 설계시 hardware량을 대폭 감소시킬 수 있다.

본 절에서는 Berlekamp의 비트 직렬승산알고리즘을 확장하여 쌍대기저로 표현된 임의의 두원소 사이에 이루어 지는 승산알고리듬에 대해 제안하고 예로써 GF(2^4)상에서의 승산기를 설계한다. 우선, 비트직렬승산알고리듬에는 Trace에 관한 개념이 도입되므로 그 내용을 간략히 기술하면 다음과 같다.

[정의 1] 유한체 GF(p^m) 내의 임의의 원소 β에 대한 Trace, $Tr(\beta)$를

$$Tr(\beta)=\sum_{i=0}^{m-1}\beta p^i \qquad (1)$$

로 정의한다. 여기서 p는 소수(prime), m은 양의정수(positive integer)이다.

[정의 2] 두개의 기저 $\{\mu_i\}$, $\{\lambda_j\}$가 다음 조건을 만족할때 두개의 기저를 각각에 대한 쌍대기저 또는 상보기저(complementary basis)라 한다.

$$Tr\{\mu_i\lambda_j\}=\begin{cases}1, & i=j\\ 0, & i\neq j\end{cases} \qquad (2)$$

[추 론] GF(p^m) 상에서 원시 다항식(primitive polynomial)의 근을 α라 할때 $\{\alpha^k\}$, $0\leq k\leq m-1$을 GF(p^m)의 표준기저라 하고 GF(p^m) 상의 임의의 원소 Z를 쌍대기저로 표현하면

$$Z=z_0\lambda_0+z_1\lambda_1+\cdots+z_{m-1}\lambda_{m-1}=\sum_{i=0}^{m-1}z_i\lambda_i \qquad (3)$$

식(3)의 양변에 α^k를 곱하여 Trace를 취하면

$$Z\cdot\alpha^k=(\sum_{i=0}^{m-1}z_i\lambda_i)\cdot\alpha^k$$

$$Tr(Z\cdot\alpha^k)=Tr(\sum_{i=0}^{m-1}z_i\lambda_i\alpha^k)=\sum_{i=0}^{m-1}z_iTr(\lambda_i\alpha^k)=z_k \qquad (4)$$

따라서, 식(3)을 이용하면

$$Z=\sum_{j=0}^{m-1}z_j\lambda_j=\sum_{j=0}^{m-1}Tr(Z\alpha^j)\lambda_j \qquad (5)$$

로 표현되어 임의의 원소 Z를 쌍대기저로 나타낼 수 있다.

1. Berlekamp의 비트 직렬승산알고리듬

Berlekamp에 의해 유한체 GF(2^m)내에서 이미 알고있는 원소 G와 임의의 한 원소 Z를 곱하는 승산알고리듬이 제안되었는데 그 내용은 다음과 같다. 우선, G를 GF(2^m)의 원시원(primitive element) α의 멱(power)으로 표현된 표준기저로, Z를 쌍대기저로 나타내면 각각

$$G=g_0+g_1\alpha+g_2\alpha^2+\cdots+g_{m-1}\alpha^{m-1} \qquad (6)$$

$$Z=z_0\lambda_0+z_1\lambda_1+\cdots+z_{m-1}\lambda_{m-1} \qquad (7)$$

이다. 이때 두 원소의 곱 $Y=G\cdot Z$를 식(4)와 (5)를이

용하여 쌍대기저로 나타내면

$$Y=\sum_{K=0}^{m-1} y_K\lambda_K=\sum_{K=0}^{m-1} Tr(GZ\alpha^k)\lambda_K,\ 0\leq k\leq m-1 \quad (8)$$

가 된다. 따라서, 식(8)을 이용하면 GF(2^m) 상의 직렬승산이 가능하게 된다. 승산결과 Y의 각 계수 y_k는 식(8)로 부터

$$\begin{aligned} y_0&=Tr(G\cdot Z)\\ y_1&=Tr(G\cdot(Z\alpha))\\ y_2&=Tr(G\cdot(Z\alpha^2))\\ &\vdots\\ y_{m-1}&=Tr(G\cdot(Z\alpha^{m-1})) \end{aligned} \quad (9)$$

와 같이 구할 수 있다. 식(9)로 부터 알 수 있듯이 Y의 각 계수들은 $Tr(G\cdot Z)$로 부터 Z대신 $Z\alpha$를 대입하면 차례로 얻을 수 있다. 이러한 승산과정은 그림 1과 같다.

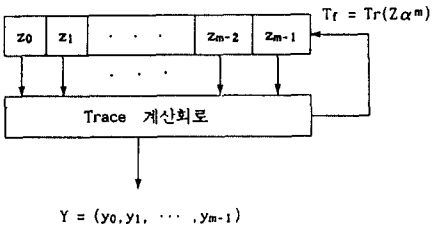

그림 1. Barlekamp의 비트 직렬승산기
Fig. 1. Berlekamp's bit-serial multiplier.

2. 확장된 비트 직렬승산기 구성

본 절에서는 Berlekamp의 비트 직렬승산알고리듬을 쌍대기저로 표현된 임의의 두 원소간의 승산으로 확장하였다. GF(2^m) 상에서 이 승산기의 블럭도는 그림 2와 같다.

GF(2^m) 상의 임의의 두 원소 A, B에 대해 쌍대기저 A′와 표준기저 B를 각각

$$\begin{aligned} A'&=a'_0\lambda_0+a'_1\lambda_1+\cdots+a'_{m-1}\lambda_{m-1}\\ B&=b_0+b_1\alpha+\cdots+b_{m-1}\alpha^{m-1} \end{aligned} \quad (10)$$

라 놓고 A′와 B의 승산 M=A′B을 쌍대기저표현으로

$$M=m_0\lambda_0+m_1\lambda_1+\cdots+m_{m-1}\lambda_{m-1} \quad (11)$$

라 할때, M의 k번째 요소 m_k, $0\leq k\leq m-1$은

$$m_k=Tr(M\alpha^k)=Tr(A'B\alpha^k),\ 0\leq k\leq m-1 \quad (12)$$

이를 m=4인 GF(2^4) 상에서의 승산기로 구성하면 다

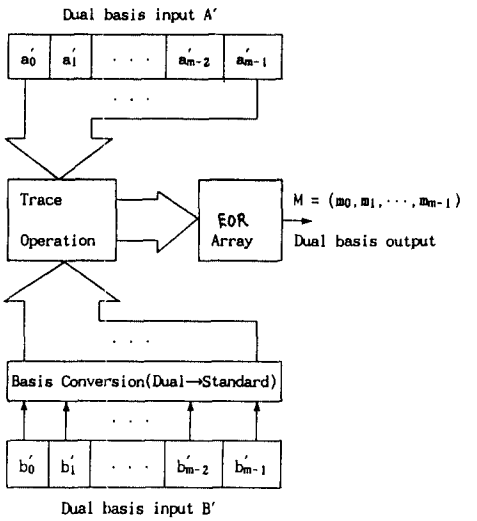

그림 2. 확장된 GF(2^m) 상의 비트직렬승산기의 블럭도
Fig. 2. Block diagram of the extended bit-sireal multiplier over GF(2^m).

음과 같다. 먼저, GF(2^4) 상의 원시다항식(primitive polynomial) p(x)를

$$p(x)=x^4+x^3+1 \quad (13)$$

이라 하면 임의의 한 원소 Z의 Trace는

$$\begin{aligned} Tr(Z)&=Tr(z_0+z_1\alpha+z_2\alpha^2+z_3\alpha^3)\\ &=z_0Tr(1)+z_1Tr(\alpha)+z_2Tr(\alpha^2)+z_3Tr(\alpha^3)\\ &=z_1+z_2+z_3 \end{aligned} \quad (14)$$

가 된다. 식(14)를 이용하면 GF(2^4)의 쌍대기저 $\{\lambda_k\}$, $0\leq k\leq 3$은 $\{\lambda_0,\lambda_1,\lambda_2,\lambda_3\}=\{\alpha^{12},\alpha^{11},\alpha^{10},\alpha^{13}\}$가 됨을 알 수 있다.

GF(2^4) 상의 임의의 한 원소 B의 표준기저와 쌍대기저를 각각

$$\begin{aligned} B&=b_0+b_1\alpha+b_2\alpha^2+b_3\alpha^3\\ B'&=b'_0\lambda_0+b'_1\lambda_1+b'_2\lambda_1+b'_3\lambda_3 \end{aligned} \quad (15)$$

로 표현할때, 식(4)와 (15)로 부터

$$\begin{aligned} b_k&=Tr\{B'\lambda_k\},\ 0\leq k\leq 3\\ &=Tr((b'_0\lambda_0+b'_1\lambda_1+b'_2\lambda_2+b'_3\lambda_3)\lambda_k)\\ &=b'_0Tr(\lambda_0\lambda_k)+b'_1Tr(\lambda_1\lambda_k)+b'_2Tr(\lambda_2\lambda_k)\\ &\quad+b'_3Tr(\lambda_3\lambda_k) \end{aligned} \quad (16)$$

를 얻을 수 있다. 따라서, $\{\lambda_0,\lambda_1,\lambda_2,\lambda_3\}=\{\alpha^{12},\alpha^{11},\alpha^{10},\alpha^{13}\}$을 식(16)에 대입하면

$$b_0=b_0'+b_1'$$
$$b_1=b_0'+b_2'+b_3'$$
$$b_2=b_1'+b_3'$$
$$b_3=b_1'+b_2' \qquad (17)$$

이다. 그림 2에서 basis conversion은 승수와 피승수가 모두 쌍대기저일때 한 원소를 표준기저로 변환시키는 역할을 하며 식(17)에 의해 수행된다. 그림 3은 이상을 이용하여 GF(2⁴)상의 승산기를 구성한 것이다. 여기서 T_f는

$$T_f = Tr(A'\alpha^4) = Tr(A'(1+\alpha^3))$$
$$= a_0'+a_3' \qquad (18)$$

이다.

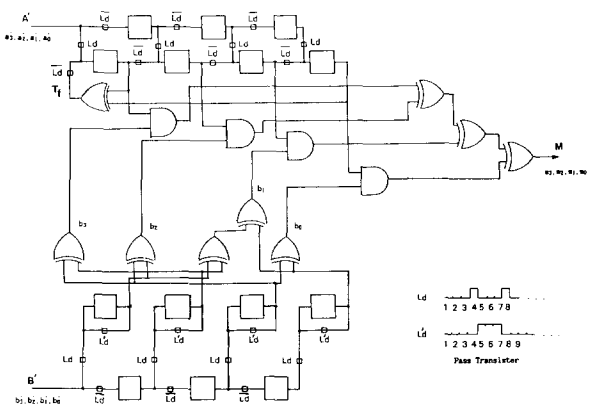

그림 3. GF(2⁴)상의 확장된 쌍대기저 승산기
Fig. 3. The extended dual basis multilplier over GF(2⁴).

그림 3에 의한 승산기 동작은 쌍대기저로 표현된 A′와 B′가 동시에 레지스터에 입력되고, 각각의 Pass Transister는 High Level에서는 "ON," Low Level에서는 "OFF" 동작을 하여 해당값을 통과 또는 단락시킨다($\bar{L}_d$는 L_d의 반전). B′는 EOR Gate(b_0, b_1, b_2, b_3)에 의해 표준기저로 변환되고 A′와의 승산이 이루어져 그 결과 M을 출력한다.

Ⅲ. 정규기저(normal basis)를 이용한 승산기

Massey와 Omura는 유한체 GF(2^m) 내의 두원소간의 승산을 정규기저 $\{\alpha, \alpha^2, \alpha^{2^2}, \cdots, \alpha^{2^{m-1}}\}$ 표현에 의해 수행되는 승산기에 대해 개발하였다.[7] 정규기저에 의한 연산중 가장 특징적인것은 제곱(square) 및 제곱근(square root)을 단순한 순회치환(cyclic shift)에 의해 얻을 수 있다는 것이다.

1. Massey-Omura 승산기[6]

GF(2^m)상에서 임의의 두 원소 A와 B의 정규기저 표현을 각각

$$A = a_0\alpha + a_1\alpha^2 + \cdots + a_{m-1}\alpha^{2^{m-1}}$$
$$B = b_0\alpha + b_1\alpha_2 + \cdots + b_{m-1}\alpha^{2^{m-1}} \qquad (19)$$

이라 할때, 두 원소 A와 B의 승산 결과를 M이라 하면

$$M = A\cdot B = (a_0\alpha + a_1\alpha^2 + \cdots + a_{m-1}\alpha^{2^{m-1}})$$
$$\cdot(b_0\alpha + b_1\alpha^2 + \cdots + b_{m-1}\alpha^{2^{m-1}})$$
$$= m_0\alpha + m_1\alpha^2 + \cdots + m_{m-1}\alpha^{2^{m-1}} \qquad (20)$$

로 표현할 수 있다. 여기서 M의 각 계수 $m_k, 0\leqq k\leqq m-1$은 a_k와 $b_k, 0\leqq k\leqq m-1$의 동일 2진 함수 f와 정규기저의 순회치환 특성에 의해 결정된다. 따라서 M의 모든 계수는

$$m_{k-1} = f(a_0, a_1, \cdots, a_{m-1};\ b_0, b_1, \cdots, b_{m-1})$$
$$m_{k-2} = f(a_{m-1}, a_0, \cdots, a_{m-2};\ b_{m-1}, b_0, \cdots, b_{m-2})$$
$$\vdots$$
$$m_0 = f(a_1, a_2, \cdots, a_0;\ b_1, b_2, \cdots, b_0) \qquad (21)$$

와 같이 차례로 얻을 수 있다. 순회치환 레지스터를 이용한 Massey-Omura의 승산기에 대한 블럭도는 그림 4와 같다.

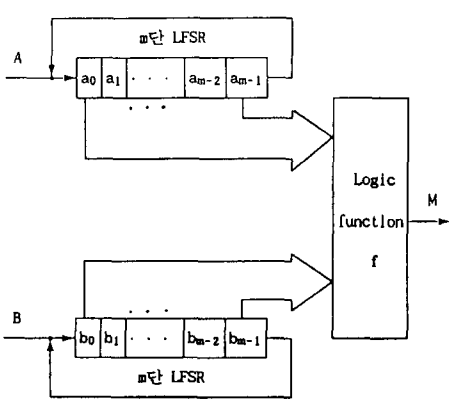

그림 4. 정규기저를 이용한 승산기의 블럭도
Fig. 4. Block diagram of multiplier using normal basis.

여기서, 승산기내에서 구성되는 논리함수 f에 대해 살펴 보자. GF(2⁴)상의 원시다항식을 $p(x)=x^4+x^3+1$로 선택하였을 때 식(20)과 식(21)로 부터 m_k, $0\leq k\leq 3$은 각각

$$\begin{aligned}m_3&=a_2b_2+a_3b_2+a_2b_3+a_3b_1+a_1b_3+a_3b_0+a_0b_3+a_1b_0\\&\quad+a_0b_1\\m_2&=a_1b_1+a_2b_1+a_1b_2+a_2b_0+a_0b_2+a_2b_3+a_3b_2+a_0b_3\\&\quad+a_3b_0\\m_1&=a_0b_0+a_1b_0+a_0b_1+a_1b_3+a_3b_1+a_1b_2+a_2b_1+a_3b_2\\&\quad+a_2b_3\\m_0&=a_3b_3+a_0b_3+a_3b_0+a_0b_2+a_2b_0+a_0b_1+a_1b_0+a_2b_1\\&\quad+a_1b_2\end{aligned}\tag{22}$$

와 같다. 따라서 식(22)를 식(21)과 비교하면 2진 함수 f는

$$\begin{aligned}&f(a_0,a_1,\cdots,a_{m-1};b_0,b_1,\cdots,b_{m-1})\\&=a_2b_2+a_3b_2+a_2b_3+a_3b_1+a_1b_3+a_3b_0+a_0b_3+a_1b_0\\&\quad+a_0b_1\end{aligned}\tag{23}$$

가 되고 2진 함수 f의 입력값을 차례로 순회치환하므로써 승산결과값을 얻어낼 수 있다. GF(2⁴)상의 두 원소간에 이루어 지는 Massey-Omura 승산기는 그림 5과 같다.

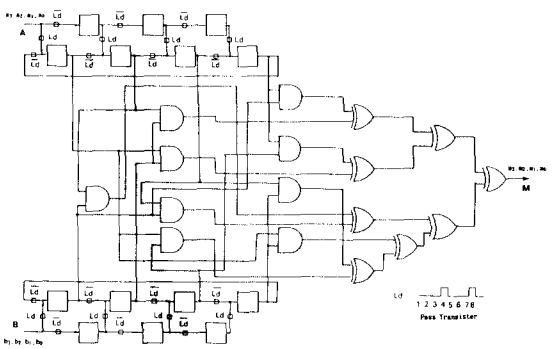

그림 5. 정규기저를 이용한 GF(2⁴)상의 승산기
Fig. 5. Multiplier over GF(2⁴) using normal basis.

Ⅳ. 부분체를 이용한 승산기

본절에서는 부분체를 갖는 유한체의 승산을 그 부분체에서의 연산으로 처리하는 승산알고리듬에 대해 소개하고 직접 부분체 GF(2²)을 이용한 GF(2⁴)상의 승산기를 예로서 구성한다.

1. 부분체에 의한 표현

일반적으로 GF(2^m)에 있어서 m이 합성수(composite)일때, 즉

$$m=e_0e_1e_2\cdots e_{s-1}\tag{24}$$

으로 표현될 때 GF(2^m)은 GF(2^{e_i}), $0\leq i\leq s-1$를 그 부분체로 갖는다. 따라서 m=4인 경우 $4=2\cdot2\cdot1$이되므로 GF(2²)은 GF(2⁴)의 부분체가 될 수 있고 GF(2⁴)의 모든 원소는 GE(2²)상의 다음과 같은 기저로 표현 가능하다.

$$\{1,\beta\}\tag{25}$$

여기서 $\beta\in$GF(2⁴)이며 반드시 $\beta\notin$GE(2²) 이어야 한다. 그러면 GF(2⁴)의 임의의 한 원소 δ는 식(25)에 의해

$$\delta=\delta_0+\delta_1\beta,\ \delta_0,\delta_1\in GF(2^2)\tag{26}$$

로 나타낼 수 있다. 또한 부분체의 연산을 Hardware적으로 구현하는데 있어서 보다 간단화하기 위해

$$\beta^2=d_0+d_1\beta,\quad d_0,d_1\in GF(2^2)\tag{27}$$

에서 $d_1=1$로 하면 식(32)는 선형화 다항식(linearlized polynomial)과 GF(2²)의 원소 d_0로 이루어진 2차 affine 다항식의 형태를 갖게되므로 β의 결정이 용이하게 된다. 따라서

$$\beta^2+\beta+d_0=0\tag{28}$$

가 되고 $\beta=X$라 놓으면 β는 결국

$$X^2+X+d_0=0\tag{29}$$

의 근이라 할 수 있고 식(29)가 GF(2⁴)에서 근을 갖기 위해서는

$$Tr_2^2(do)=1\tag{30}$$

를 만족해야 한다. 여기서 $Tr_q^n(k)=\sum_{i=0}^{n-1}X^{q^i}$이다.

그리고 $\gamma^2+\gamma+1=0$이고 $Tr_2^2(\gamma)=1$이 되는 GF(2⁴)의 임의의 한 원소 γ를 찾으면 $\gamma=\alpha^5$이 된다. 따라서

$$\beta^2+\beta+\alpha^5=0\tag{31}$$

으로 놓으면 식(31)을 만족하는 β는 $\beta=\alpha^7$이 되어 GF(2⁴)의 부분체 GF(2²)상의 기저는

$$\{1,\alpha^7\}\tag{32}$$

이 된다.

2. 부분체를 이용한 승산

부분체 GF(2²)의 기저로 표현된 GF(2⁴)의 두 원소 A, B를 각각

$$A = a_0 + a_1\beta$$
$$B = b_0 + b_1\beta \quad a_0, a_1, b_0, b_1 \in GF(2^2) \tag{33}$$

이라 할때 두 원소 A, B의 승산 결과 M은

$$M = A \cdot B$$
$$= (a_0 + a_1\beta)(b_0 + b_1\beta)$$
$$= m_0 + m_1\beta, \quad m_0, m_1 \in GF(2^2) \tag{34}$$

가 되고 m_0와 m_1은 각각

$$m_0 = a_0b_0 + a_1b_1\gamma, \; m_1 = a_0b_1 + a_1b_0 + a_1b_1 \tag{35}$$

이다. 그리고 GF(2⁴) 내의 임의의 한원소 $C = c_0 + c_1\beta$는 c_0와 c_1이 GF(2²)의 원소이므로

$$C = c_0 + c_1\beta$$
$$= (x_0 + x_1\gamma) + (y_0 + y_1\gamma)\beta \tag{36}$$

로 표현할 수 있다. 여기서 $x_i, y_i \in GF(2), 0 \leq i \leq 1$이고, γ는 $p(x) = x^2 + x + 1$의 근이다. 결국 식(34), (35)에서 m_0, m_1은 GF(2²) 상의 계산으로 구할 수 있으며 이때문에 부분체에 의한 연산이 가능하다.

그림 6은 부분체 GF(2²)을 이용한 GF(2⁴) 상의 승산기에 대한 블럭도를 나타낸 것이다.

그림 6에서 GF(2²)상의 승산에 대한 연산은

$$z_0 + z_1\gamma = (x_0 + x_1\gamma)(y_0 + y_1\gamma)$$
$$= (x_0y_0 + x_1y_1) + (x_1y_0 + x_1y_0 + x_1y_1)\gamma \tag{37}$$

이므로 $z_0 = x_0y_0 + x_1y_1$, $z_1 = x_0y_1 + x_1y_0 + x_1y_1$가 되어 그림 7과 같은 논리회로로 구성할 수 있다. 또한 그림

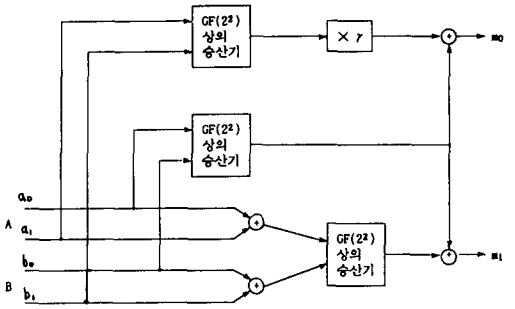

그림 6. 부분체 GF(2²)을 이용한 GF(2⁴) 상의 승산기 블럭도

Fig. 6. Block diagram of multiplier over GF(2⁴) using subfield GF(2²).

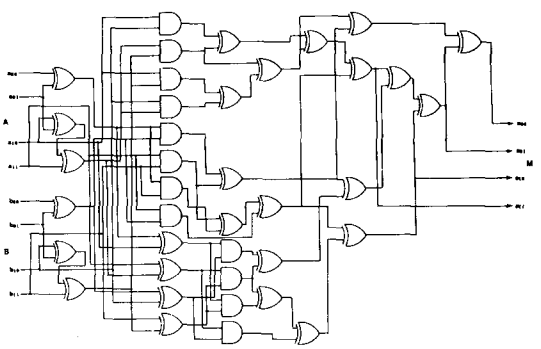

그림 7. 부분체 GF(2²)을 이용한 GF(2⁴) 상의 승산기

Fig. 7. Multiplier over GF(2⁴) using subfield GF(2²).

표 1. GF(2⁴) 상에서 각 승산기 비교표

Table 1. Comparison table of multipliers for GF(2⁴).

Multiplier Item	Systolic Multipliers[5] (1D)	Systolic Multipliers[5] (2D)	Cellular-Array[4] Multiplier	Fig. 3 (Dual Basis)	Fig. 6[6] (Normal Basis)	Fig. 8 (Standard Basis)
1. I/O format	serial	parallel	parallel	serial	serial	parallel
2. Register	42	128	12	14	14	—
3. EOR	8	32	24	9	8	27
4. AND	12	32	16	4	8	12
5. Switch	4	—	—	—	—	—
6. pass-transister	—	—	—	22	22	—
7. Clock Time	8	8	about 8 gate delay	7	7	about 11 gate delay

7에서 γ에 대한 승산은

$$\gamma \cdot Z = \gamma(z_0 + z_1\gamma)$$
$$= z_1 + (z_0 + z_1)\gamma \quad (38)$$

이므로 EOR게이트 1개로 구현할 수 있다.

본 절에서는 GF(2^m) 상에서 수행되는 승산기를 m=4인 경우에 대해 본 논문에서 다룬 세가지 경우와 타 논문의 승산기를 소요소자 및 연산처리시간면에서 비교하였고 그 결과를 표 1에 나타내었다.

먼저, 표 1에서와 같이 C.S. Yeh등에 의해 제안된 두가지형태의 Systolic형 승산기는 본 논문에서의 것보다 많은 레지스터와 게이트들이 필요하며 연산처리속도 역시 떨어진다. 그리고 B.A. Laws등이 제안한 Cellular-Array 승산기는 연산처리속도가 가장 빠르나 게이트수는 그림 3과 그림 5에서의 것보다 많이 소요된다.

한편, 쌍대기저를 이용한 승산기는 표준기저로 재표현하지 않을때 가장적은 규모로 회로를 구성할 수 있다. 정규기저에 의한 승산기는 특정 원시다항식에서만 효율적이지 역원기나 재산기등을 구성시 많은 Pass Transister가 요구된다. 부분체를 이용한 승산기는 연산처리속도가 빠르고 부분체연산부, 변환 및 역변환부 등으로 모율화가 가능하다는 장점을 갖는다. 특히 이 승산기는 m이 크고 역원기나 재산기 구성시 회로규모를 크게 감소시킬 수 있으나 부분체를 m/2로 하는 경우 이외에는 오히려 연산기의 조각화(granularity)가 생겨 비효율적이다.

Ⅳ. 결 론

본 논문에서는 유한체 GF(2^m)상의 승산기를 m=4인 경우에 대해 쌍대기저, 정규기저와 부분체를 이용한 표준기저로 각각 구성하고 비교검토하였다. 앞에서도 각 부분의 모듈화에 따른 단순화 및 소요되는 회로소자의 감소면에서 크게 개선시켰다.

우선, Berlekamp의 비트 직렬승산알고리듬을 기초로한 승산기의 경우, 서로 다른 기저로 표현된 두 원소간의 승산을 기저변환기에 의해 쌍대기저로 통일시켜 승산을 수행시켰다. 따라서, Reed-Solomon 부호의 직렬부호기에 대응하는 복호기 구성시 기저의 변환없이 직접 승산이 가능하게 된다.

또한, 정규기저표현에 의한 Massey-Omura 승산기는 C.C. Wang등에 의해 VLSI화가 용이하도록 설계되었고, 순회치환특성에 따른 제곱연산을 효과적으로 이용하여 역원기와 재산기 구성이 가능하도록 하였으나, 원시다항식의 선택과 전체적인 연산속도면에서 단점을 지니고 있다.

한편, 표준기저로 표현된 GF(2^m)내의 승산을 부분체에서의 연산으로 처리하는 승산 알고리듬은 m/2인 부분체를 이용하는 경우가 가장 효과적이고 각 부분체 연산 부분의 모듈화가 가능하다. 그러나 m/2이외의 부분체는 오히려 전체 연산기의 조각화(granularity)가 생겨 연산기 규모를 커지게하는 단점이 있다. 이 승산기는 m이 큰 값(m≧8)을 가질 때 소요되는 회로소자의 갯수를 크게 줄일 수 있으며 부분체를 이용한 연산은 승산에서 보다 역원 및 제산을 수행하는 경우 다른 알고리듬에 비해 그 우수성을 보일 것이며 이에대한 연구는 현재 본 논문을 기초로 진행중이다.

參 考 文 獻

[1] M.Y. Rhee, *Error Correcting Coding Theory*, McGraw-Hill, New York, 1989.

[2] R.J. McEliece, *Finite Field for Computer Scientists and Engineers*, Kluwer Academic Publishers, 1987.

[3] E.R. Berlekamp, *Algebraic Coding Theory*, McGraw-Hill, New York, 1968.

[4] B.A. Laws, C.K. Rushford, "A Cellular-Array Multiplier for GF(2^m), " *IEEE Trans. Computers*, vol. C-20, no: 12, pp. 1573-1578, Dec. 1971.

[5] C.S. Yeh, I.S. Reed and T.K. Trung, "Systolic multipliers for finite field GF(2^m)," *IEEE Trans. Comput.*, vol. C-33, pp. 357-360, Apr. 1984.

[6] C.C. Wang, T.K. Trung, H.M. Shao, L.J. Deutsch, J.K. Omura and I.S. Reed., "VLSI Architecture for Computing Multiplications and Inverses in GF(2^m)," *IEEE Trans. Comput.*, vol. C-34, pp. 709-717, Aug. 1985.

[7] F.J. McWilliams, and N.J. Sloane, *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam, the Netherland, 1977.

[8] G.L. Feng, "A VLSI Architecture for Fast Inversion in GF(2^m)," *IEEE Trans. Comput*, vol. 38, no. 10, Oct. 1989.

[9] I.S. Hsu, T.K. Truong, L.J. Deutsch, and I.S. Reed, "A comparison of VLSI Architecture of Finite Multipliers using Dual, Normal, or Standard Bases," *IEEE Trans. Comput.*, vol. C-37, pp. 735-739, 1988.

[10] I.S. Hsu, I.S. Reed, T.K. Truong, K.E. Wang, C.S. Yeh and L.J. Deutsch, "The VLSI Implementation of a Reed-Solomon Encoder Using Berlekamp's Bit-Serial Multiplier Algorithm," *IEEE Trans. Comput.*, vol. C-33, no. 10, Oct. 1984.

[11] M. Morii and M. Kasahara, "Efficient Construction of Gate Circuit for Computing Multiplicative Inverses over GF(2^m)," Trans. IEICE, Japan, vol. E-72, no. 1, Jan. 1988.

著 者 紹 介

金 在 汶 (正會員) 第28卷 A編 第6號 參照
현재 한양대학교 전자통신공학과 박사과정

李 晩 榮 (正會員) 第26卷 第2號 參照
현재 한양대학교 전자통신공학과 명예교수