情報保護學會論文誌
第17卷 第1號, 2007. 2

유한체 상에서 고속 연산을 위한 직렬 곱셈기의 병렬화 구조

조 용 석
영동대학교

Parallelized Architecture of Serial Finite Field Multipliers for Fast Computation

Yong-Suk Cho
Youngdong University

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 직렬 유한체 곱셈기에 비해 짧은 지연시간을 갖는 새로운 직렬 곱셈기 구조를 제안하였다. 제안한 곱셈기는 유한체의 곱을 표현하는 다항식을 여러 개로 분리한 다음, 이 다항식들을 동시에 처리하는 방식을 사용하여 직렬 곱셈기의 속도를 향상시켰다. 이 곱셈기는 유한체 $GF(2^m)$의 표준기저 상에서 동작하며, 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.

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, a new bit serial structure for a 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 standard 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.

Keywords : *Finite fields, Multiplier, Error-control coding, Cryptography*

Ⅰ. 서 론

유한체(finite fields or Galois fields) 상의 연산은 오류정정 부호, 디지털 신호처리, 암호화 등의 여러 분야에서 널리 사용되고 있다. 특히 오류정정 부호 중 실용상 널리 사용되고 있는 BCH 부호 및 Reed-Solomon 부호와 같은 블록부호, 그리고 최근 공개키 암호 알고리즘으로 관심이 집중되고 있는 타원곡선 암호시스템(Elliptic Curve Cryptosystem) 등은 모든 연산이 유한체 상에서 이루어진다. 따라서 유한체 상의 연산은 이들 분야의 시스템 설계 시 전체 시스템의 규모나 성능에 절대적인 영향을 미친다[1]-[4].

접수일: 2006년 9월 26일; 채택일: 2006년 12월 11일
주저자, yscho@youngdong.ac.kr

유한체 $GF(2^m)$은 2^m개의 원소(elements)를 가지고 있으며, 이 원소들을 표현하는 방법으로는 원시원(primitive element) α의 $m-1$차 이하 다항식으로 표현하는 다항식표현(polynomial representation) 방법과 α의 지수로 표현하는 지수표현(exponential or power representation) 방법으로 나눌 수 있다. 지수표현을 이용하면 곱셈과 나눗셈은 각각 2진수의 덧셈과 뺄셈으로 대치되므로 쉽게 수행할 수 있는 반면에 덧셈이 복잡해지며, 다항식표현을 이용하면 덧셈은 각 비트 별 2원합(modulo-2 sum)으로 간단하게 수행되지만 곱셈과 나눗셈이 어려워지는 문제점이 있다. $GF(2^m)$의 차수 m이 작은 경우에는 지수표현을 이용한 연산이 더 쉬운 반면, m이 커지면 지수표현 보다는 다항식표현을 이용한 연산이 더 적은 하드웨어로 구현할 수 있으며 고속처리가 가능하다[5].

따라서 다항식표현을 이용한 곱셈 및 나눗셈을 효율적으로 실행하는 방법을 찾아내려는 연구들이 집중적으로 이루어지고 있다[6]. 대표적인 것으로, 일반적으로 사용되는 표준기저(standard basis) 대신에 쌍대기저(dual basis)를 이용한 Berlekamp[7],[8]의 곱셈 알고리즘과, 정규기저(normal basis)를 이용한 Massey와 Omura[9],[10]의 곱셈 알고리즘을 들 수 있다. 이 알고리즘들은 다항식 기저를 적절히 변환하여 소요되는 하드웨어 및 지연시간을 줄이고자 하는 방법들이다.

유한체 $GF(2^m)$ 상의 곱셈기는 조합회로를 사용한 병렬 곱셈기(parallel multiplier)와 순서회로를 사용한 직렬 곱셈기(serial multiplier)로 구현할 수 있다. 병렬 곱셈기는 한 클럭 사이클 내에 결과를 출력하는 회로이며, 직렬 곱셈기는 일반적으로 m클럭만큼의 시간 지연 후에 결과를 출력한다. 유한체 $GF(2^m)$ 상에서 병렬 곱셈기는 일반적으로 $O(m^2)$의 공간 복잡도를 가지며 직렬 곱셈기는 $O(m)$의 공간 복잡도를 갖는다. 즉, 병렬 곱셈기는 연산속도는 빠른 반면에 회로가 복잡해지며, 직렬 곱셈기는 회로는 간단하지만 m클럭만큼의 시간 지연이 생긴다.

이러한 문제점을 해결하기 위하여 회로의 복잡도와 지연 시간 사이의 적절한 절충을 꾀하는 방법들이 발표되고 있다. 즉 기존의 직렬 곱셈기보다는 짧은 지연시간에 결과를 얻을 수 있으며, 병렬 곱셈기보다는 적은 하드웨어로 구현할 수 있는 방법들이 연구되고 있다[11],[12].

조용석 등[13]은 유한체 $GF(2^m)$이 부분체를 가지는 경우, 이 부분체 상의 병렬 연산기들을 이용하여 그 확대체 상의 직렬 곱셈기를 구성하는 방법을 제안하였으며, Paar 등[14]은 이 방법을 확장시켰다. 그러나 이러한 하이브리드 곱셈기(hybrid multiplier)는 유한체의 차수가 합성수일 때에만 적용이 가능하다는 제약이 따른다.

최근 문상국 등[15]은 유한체의 표준기저 상에서 임의의 두 원소의 곱을 표현한 다항식을 t개로 분리하여 각각을 병렬로 처리하는 방식으로 t배의 속도를 향상 시킬 수 있는 직렬 곱셈기를 제안하였다. 본 논문에서는 이 방법을 개선하여, 문헌 [15]의 곱셈기보다 더 간단한 하드웨어로 구현할 수 있으며 더 고속으로 동작하는 새로운 직렬 유한체 곱셈기 구조를 제안한다. 제안한 곱셈기는, 사용하는 유한체의 위수가 합성수이어야만 한다는 하이브리드 곱셈기의 단점을 해결한 것으로, 모든 유한체를 선택할 수 있는 장점을 가지고 있다.

본 논문의 구성은, 먼저 Ⅱ에서 유한체 상의 직렬 곱셈 알고리즘을 분석하고 Ⅲ에서는 회로의 복잡도와 지연시간 사이의 적절한 절충을 꾀할 수 있는 새로운 직렬 곱셈기를 설계한다. Ⅳ에서는 제안된 곱셈기의 성능을 기존의 곱셈기와 비교하고 Ⅴ에서 결론을 맺는다.

Ⅱ. 유한체 상의 직렬 곱셈기

유한체는 덧셈과 곱셈에 대해서 결합법칙, 교환법칙, 분배법칙이 성립하고, 덧셈에 대한 항등원과 역원, 곱셈에 대한 항등원과 역원을 가지는 유한개의 원소로 구성된 집합이다. 유한체 상에서 임의의 두 원소의 덧셈을 하드웨어로 구성하면, 각 비트를 XOR (Exclusive OR) 연산한 것이 된다. 그러나 유한체 상의 곱셈 연산은 보다 복잡한 과정으로 이루어진다.

α를 유한체 $GF(2^m)$의 원시원(primitive element)이라 할 때 영원(zero element)을 제외한 2^m-1개의 모든 원소들은 α의 멱(power)으로 표현할 수 있다. 또 이 α는 식 (1)과 같은,

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

차수가 m인 원시다항식(primitive polynomial)의 근(root)이므로, 즉 $p(\alpha) = 0$이므로

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

가 된다. 따라서 유한체 $GF(2^m)$의 각 원소들은 차수가

$m-1$ 이하인 α의 다항식으로 표현할 수 있다. 즉 유한체 $GF(2^m)$ 상의 임의의 한 원소 U는 다음과 같이 쓸 수 있다.

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

여기에서 다음과 같은 m개의 서로 독립인 원소들을 유한체 $GF(2^m)$의 표준기저(standard basis)라고 한다[5].

$$\{1, \alpha, \alpha^2, \cdots\cdots, \alpha^{m-2}, \alpha^{m-1}\} \tag{4}$$

유한체 $GF(2^m)$ 상의 임의의 두 원소 A와 B를 식 (3)과 같이 표현하면 다음과 같이 된다.

$$A = a_0 + a_1\alpha + \cdots\cdots + a_{m-1}\alpha^{m-1}, \ a_i \in GF(2) \tag{5}$$

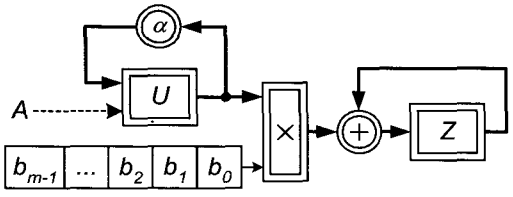
$$B = b_0 + b_1\alpha + \cdots\cdots + b_{m-1}\alpha^{m-1}, \ b_i \in GF(2) \tag{6}$$

이 두 원소의 곱을 Z라 하면 Z는 다음과 같이 쓸 수 있다.

$$\begin{aligned} Z &= A \cdot B \\ &= A \cdot (b_0 + b_1\alpha + b_2\alpha^2 + \cdots\cdots + b_{m-1}\alpha^{m-1}) \\ &= b_0A + b_1[A\alpha] + b_2[A\alpha^2] + \cdots\cdots + b_{m-1}[A\alpha^{m-1}] \end{aligned} \tag{7}$$

식 (7)을 살펴보면, 두 원소의 곱 Z는 임의의 한 원소 A에 α를 곱해 가면서 B의 계수들과 차례로 곱하여 계속 더하는 것이다. 따라서 LFSR (Linear Feedback Shift Register)을 이용하여 식 (7)을 구현하면 〔그림 1〕과 같은 직렬 곱셈기를 설계할 수 있다[13].

〔그림 1〕에서 굵은 선은 m비트 버스이고, □는 m비트 레지스터를, ⊕는 m개의 XOR 게이트를, ⊠은 m개의 AND 게이트를, ⓐ는 $GF(2^m)$의 원시원 α를 곱하는 회로(multiply-by-α)를 나타내고 있다.



〔그림 1〕 $GF(2^m)$ 상의 직렬 곱셈기

유한체 $GF(2^m)$에서 식 (5)와 같은 임의의 한 원소 A에 α를 곱하면 다음과 같이 된다.

$$A \cdot \alpha = a_0\alpha + a_1\alpha^2 + a_2\alpha^3 + \cdots\cdots + a_{m-2}\alpha^{m-1} + a_{m-1}\alpha^m \tag{8}$$

여기에 식 (2)를 대입하여 정리하면

$$A \cdot \alpha = a_{m-1} + (a_0 + a_{m-1}p_1)\alpha + (a_1 + a_{m-1}p_2)\alpha^2 + \cdots\cdots + (a_{m-2} + a_{m-1}p_{m-1})\alpha^{m-1} \tag{9}$$

가 된다. 따라서 식 (9)를 이용하면 α를 곱하는 회로를 구현할 수 있다.

〔그림 1〕 회로의 동작은 초기 상태에서 레지스터 Z는 클리어시키고 임의의 두 원소 A와 B를 각각 레지스터 U와 B에 로드시킨다. 그리고 각 레지스터를 m번 쉬프트 시키면 레지스터 Z에 두 원소의 곱 Z가 저장된다. 따라서 m클럭 시간에 곱셈의 결과를 얻을 수 있다.

예를 들어 원시다항식이 $p(x) = 1 + x^2 + x^5$인 유한체 $GF(2^5)$에서 〔그림 1〕과 같은 직렬 곱셈기를 설계하여 보자. $GF(2^5)$ 상의 임의의 한 원소 A를

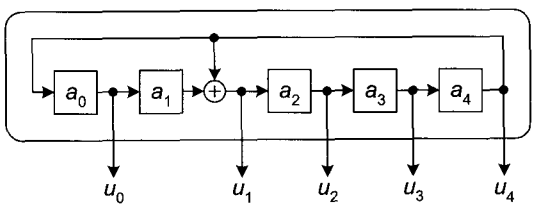
$$A = a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3 + a_4\alpha^4 \tag{10}$$

라 하고, 여기에 원시원 α를 곱하면

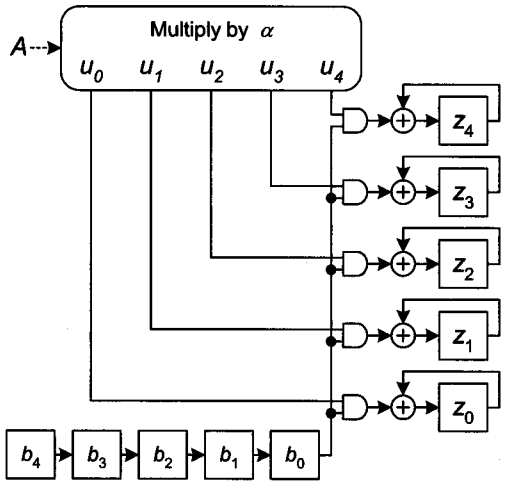
$$\begin{aligned} A \cdot \alpha &= a_0\alpha + a_1\alpha^2 + a_2\alpha^3 + a_3\alpha^4 + a_4(1 + \alpha^2) \\ &= a_4 + a_0\alpha + (a_1 + a_4)\alpha^2 + a_2\alpha^3 + a_3\alpha^4 \end{aligned} \tag{11}$$

가 된다. 따라서 식 (11)을 이용하면 〔그림 2〕와 같이 $GF(2^5)$ 상의 임의의 한 원소 A에 α를 곱하는 (multiply-by-α) 회로를 설계할 수 있다.

〔그림 2〕를 이용하여 〔그림 1〕과 같은 $GF(2^5)$ 상의 직렬 곱셈기를 설계하면 〔그림 3〕과 같이 된다.



〔그림 2〕 $GF(2^5)$ 상에서 α를 곱하는 회로



〔그림 3〕 $GF(2^5)$ 상의 직렬 곱셈기

Ⅲ. 고속 직렬 유한체 곱셈기

〔그림 1〕과 같은 유한체 상의 직렬 곱셈기는 m클럭 시간 후에 곱셈의 결과가 나온다. 이를 고속화하기 위하여 식 (7)을 t개로 분할하여 각 부분을 동시에 처리하면 t배의 속도를 향상시킬 수 있다.

곱셈기의 자세한 구조를 설명하기 위하여 $t=2$인 경우, 즉 속도를 2배 향상시켰을 경우의 곱셈기를 설계하고 이를 바탕으로 t배속 곱셈기를 설계한다.

3.1 2배속 직렬 유한체 곱셈기

2배속 직렬 유한체 곱셈기를 설계하기 위하여, 식 (7)을 α의 지수가 짝수인 항과 홀수인 항으로 나누면 다음과 같이 쓸 수 있다.

$$Z = Z_{even} + Z_{odd} \tag{12}$$

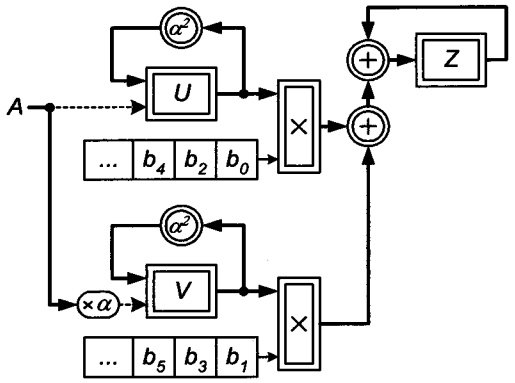
여기에서 Z_{even}과 Z_{odd}는 다음과 같이 된다.

$$Z_{even} = b_0 A + b_2 A\alpha^2 + b_4 A\alpha^4 + \cdots\cdots \tag{13}$$

$$Z_{odd} = b_1 A\alpha + b_3 A\alpha^3 + b_5 A\alpha^5 + \cdots\cdots \tag{14}$$

여기에서 m이 홀수이면 Z_{even}과 Z_{odd}의 항의 개수가 같지 않게 되므로 Z_{odd}의 마지막 항에 0을 삽입하여 항의 개수를 일치시킨다.

식 (7)과 식 (13)을 비교해 보면, 식 (13)에서는 임의의 한 원소 A에 α대신 α^2을 곱해 가면서 B의 짝수 항 계수들과 차례로 곱하여 계속 더하는 것이다. 또한



〔그림 4〕 $GF(2^m)$ 상의 2배속 직렬 곱셈기

식 (14)는 다음과 같이 다시 쓸 수 있다.

$$Z_{odd} = b_1(A\alpha) + b_3(A\alpha)\alpha^2 + b_5(A\alpha)\alpha^4 + \cdots\cdots \tag{15}$$

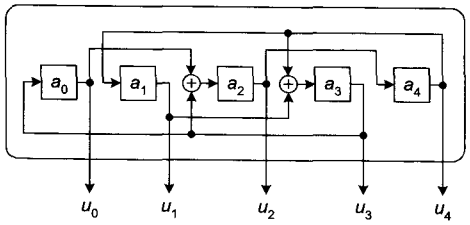
식 (15)를 살펴보면 식 (13)에서 A대신 $A\alpha$로 바꾸고 B의 홀수 항 계수들과 차례로 곱하면 식 (13)과 동일한 구조가 된다. 따라서 식 (13)과 식 (15)를 이용하면 〔그림 4〕와 같은 2배속 직렬 곱셈기를 구성할 수 있다.

〔그림 1〕과 마찬가지로 〔그림 4〕의 회로에서도 굵은 선은 m비트 버스를, □는 m비트 레지스터를, ⊕는 m개의 XOR 게이트를, ⊠은 m개의 AND 게이트를 나타낸다. 〔그림 1〕과 비교하여 한 가지 다른 점은, 입력되는 임의의 한 원소 A에 α대신 α^2을 곱하는 것이다.

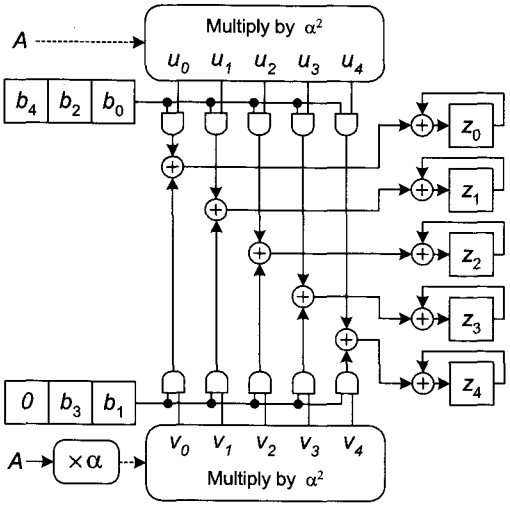
〔그림 4〕의 회로는 초기 상태에서 레지스터 Z는 클리어 시키고 위쪽 레지스터에는 A를 로드 시키고, 아래쪽 레지스터에는 $A\alpha$를 로드 시킨다. 그리고 각 레지스터를 $\lceil m/2 \rceil$번 쉬프트 시키면 레지스터 Z에 두 원소를 곱한 결과가 저장된다. 따라서 $\lceil m/2 \rceil$ 클럭 시간에 곱셈의 결과를 얻을 수 있다.

〔그림 4〕의 곱셈기를 〔그림 1〕의 곱셈기와 비교해보면, m비트 레지스터와 m개의 XOR 게이트, m개의 AND 게이트, 그리고 α와 α^2을 곱하는데 소요되는 몇 개의 XOR 게이트가 더 사용되었음을 알 수 있다. 그러나 곱셈에 소요되는 시간을 약 절반으로 줄일 수 있다.

예를 들어 원시다항식이 $p(x)=1+x^2+x^5$인 유한체 $GF(2^5)$에서 〔그림 4〕와 같은 2배속 직렬 곱셈기를 설계하여 보자. 식 (10)과 같은 $GF(2^5)$ 상의 임의의 한 원소 A에 α^2을 곱하면 다음과 같이 된다.



〔그림 5〕 $GF(2^5)$ 상에서 α^2을 곱하는 회로



〔그림 6〕 $GF(2^5)$ 상의 2배속 직렬 곱셈기

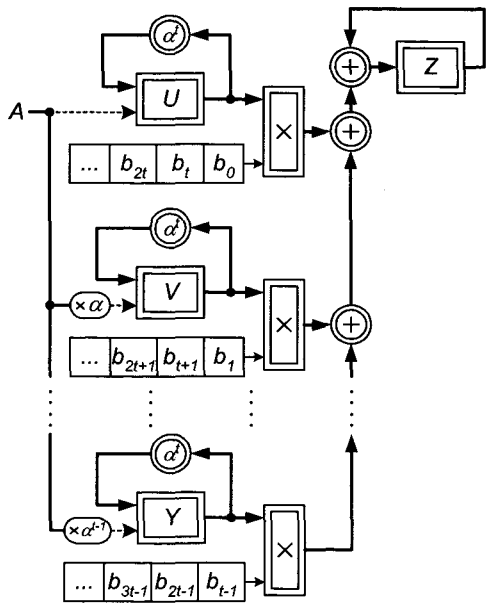
$$\begin{aligned} A\alpha^2 &= a_0\alpha^2 + a_1\alpha^3 + a_2\alpha^4 \\ &\quad + a_3(1+\alpha^2) + a_4(\alpha+\alpha^3) \\ &= a_3 + a_4\alpha + (a_0+a_3)\alpha^2 \\ &\quad + (a_1+a_4)\alpha^3 + a_2\alpha^4 \end{aligned} \tag{16}$$

따라서 식 (16)을 이용하면 〔그림 5〕와 같이 $GF(2^5)$ 상의 임의의 한 원소 A에 α^2을 곱하는(multiply-by-α^2) 회로를 설계할 수 있다.

〔그림 5〕를 이용하여 〔그림 4〕와 같은 $GF(2^5)$ 상의 2배속 직렬 곱셈기를 설계하면 〔그림 6〕과 같이 된다.

〔그림 6〕에서 Multiply-by-α^2은 〔그림 5〕와 같은 회로이고, $\times\alpha$는 〔그림 2〕와 같이 $GF(2^5)$ 상의 임의의 한 원소 A에 α를 병렬로 곱하는 회로이다.

〔그림 6〕의 회로는 초기 상태에서 레지스터 Z는 클리어 시키고 위쪽의 레지스터에는 A를 로드 시키고, 아래쪽의 레지스터에는 $A\alpha$를 로드 시킨다. 그리고 각 레지스터를 $\lceil 5/2 \rceil = 3$번 쉬프트 시키면 레지스터 Z에 두 원소의 곱이 저장된다. 따라서 3 클럭 시간에 곱셈의 결과를 얻을 수 있다.



〔그림 7〕 $GF(2^m)$ 상의 t배속 직렬 곱셈기

3.2 t배속 직렬 유한체 곱셈기

2배속 직렬 곱셈기의 구조를 t배속 직렬 곱셈기 구조로 확장하기 위하여 식 (7)을 t개로 분할하면 다음과 같이 정리할 수 있다.

$$\begin{aligned} Z_0 &= b_0 A + b_t A\alpha^t \\ &\quad + b_{2t} A\alpha^{2t} + \cdots\cdots \\ Z_1 &= b_1(A\alpha) + b_{t+1}(A\alpha)\alpha^t \\ &\quad + b_{2t+1}(A\alpha)\alpha^{2t} + \cdots\cdots \\ Z_2 &= b_2(A\alpha^2) + b_{t+2}(A\alpha^2)\alpha^t \\ &\quad + b_{2t+2}(A\alpha^2)\alpha^{2t} + \cdots\cdots \\ &\vdots \\ Z_{t-1} &= b_{t-1}(A\alpha^{t-1}) \\ &\quad + b_{2t-1}(A\alpha^{t-1})\alpha^t \\ &\quad + b_{3t-1}(A\alpha^{t-1})\alpha^{2t} + \cdots\cdots \end{aligned} \tag{17}$$

식 (17)을 이용하여 〔그림 4〕와 같은 구조를 갖는 t배속 직렬 곱셈기를 설계하면 〔그림 7〕과 같이 된다. 〔그림 7〕과 같은 곱셈기는 $\lceil m/t \rceil$ 클럭 시간에 곱셈의 결과를 얻을 수 있다.

〔표 1〕 유한체 곱셈기들의 성능 비교 ($m = 2k$인 경우)

	직렬 곱셈기	하이브리드 곱셈기[14]	문헌 [15]의 곱셈기	제안된 곱셈기
AND	m	$2m$	$2m$	$2m$
XOR	m	$(3/2)m$	$3m$	$2m$
REG	$2m$	$2m$	$3m$	$3m$
MUX	0	0	$2m$	0
Clock	m	$m/2$	$m/2+1$	$m/2$
Delay	(i)	(ii)	(iii)	(iv)

(i) Register → AND → XOR
(ii) Register → $GF(2^2)$ parallel multiplier (2 AND → 2 XOR) → XOR
(iii) Register → AND → MUX → XOR
(iv) XOR → Register → AND → XOR

Ⅳ. 제안한 곱셈기의 성능 평가

〔표 1〕에 $t = 2$일 경우 기존의 구조와 제안된 구조를 비교하고 그 성능을 평가하였다. 〔그림 1〕의 α를 곱하는 회로와 그림 3의 α^2를 곱하는 회로는 전체 회로에 비하여 무시할 수 있을 만큼 작기 때문에, 각 곱셈기에서의 하드웨어 부담은 동일하다고 가정하고 비교 대상에서 제외시켰다[15].

문헌 〔15〕에서는 식 (14)를 다음과 같이 변형하고

$$Z_{odd} = \alpha [b_1 A + b_3 A\alpha^2 + b_5 A\alpha^4 + \cdots\cdots] \quad (18)$$

이를 구현할 때 괄호 안의 부분을, 식 (13)과 같이 α^2을 곱하는 회로로 구현한 다음, 다시 α를 곱하는 회로를 사용하여 괄호 밖의 α를 곱하는 방법을 사용하였다. 이 과정에서 α를 곱하는 회로와 α^2를 곱하는 회로 중 하나를 선택하기 위하여 2-to-1 MUX를 사용하였으며, 한 클럭 사이클을 추가로 사용하였다. 그러므로 문헌 〔15〕의 곱셈기는 Z_{even}회로와 Z_{odd} 회로에 각각 m개의 AND 게이트와 m개의 XOR 게이트가 필요하며, Z_{even}과 Z_{odd}를 더하여 최종 곱셈 결과를 구하기 위하여 m개의 XOR 게이트가 소요된다. 즉 표 1과 같이 총 $2m$개의 AND 게이트와 $3m$개의 XOR 게이트가 필요하게 된다.

따라서 제안된 곱셈기와 문헌 〔15〕의 곱셈기를 비교하여 보면, 〔표 1〕에서 보듯이 제안된 곱셈기가 XOR의 개수도 m개만큼 적게 사용되었으며 MUX도 필요하지 않기 때문에 훨씬 적은 하드웨어로 구현할 수 있다. 또한 문헌 [15]의 곱셈기는 B의 홀수 항 계수들을 곱하는 회로를 구현하는데 있어서 한 클럭 사이클을 더 사용하기 때문에 제안된 곱셈기가 더 고속으로 곱셈을 수행할 수 있는 장점을 가지고 있다.

Ⅴ. 결 론

본 논문에서는 유한체의 표준기저 상에서 임의의 두 원소의 곱을 표현하는 다항식을 t개로 분리하여 각각을 병렬로 처리하는 방식으로 t배의 속도를 향상 시킬 수 있는 새로운 직렬 곱셈기를 제안하였다.

제안된 곱셈기는 하이브리드 곱셈기[13],[14]와 유사하게, 직렬 곱셈기의 긴 지연시간과 병렬 곱셈기의 복잡한 회로 사이를 적절하게 절충함으로써, 직렬 곱셈기보다는 짧은 지연시간에 결과를 얻을 수 있으며 병렬 곱셈기보다는 적은 회로로 구현할 수 있다. 하드웨어 측면에서는 하이브리드 유한체 곱셈기에 비해 약간 복잡하지만 제안된 곱셈기 구조는 사용하는 유한체의 위수가 합성수 이어야 한다는 제약이 필요 없이 모든 유한체를 선택할 수 있다는 장점을 가지고 있다.

참고문헌

〔1〕 이만영, *BCH 부호와 Reed-Solomon 부호*, 민음사, 1988

〔2〕 S. B. Wicker and V. K. Bhargava, *Reed-Solomon Codes and Their Applications*, IEEE Press, 1994.

〔3〕 G. B. Agnew, R. C. Mullin, and S. A. Vanstone, "An Implementation of Elliptic Curve Cryptosystems over $F_{2^{155}}$," *IEEE Journal on Selected Areas in Communications*, Vol.11, No.5, pp.804-813, June 1993.

〔4〕 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.

〔5〕 R. J. McEliece, *Finite Fields for Computer Scientist and Engineers*, Kluwer Academic, 1987.

〔6〕 이형목, 김현성, 전준철, 유기영, "$GF(2^m)$상에서 AB^2 연산을 위한 세미시스톨릭 구조," *정보보호학회논문지*, 제12권, 제2호, pp.45-52, 2002. 4.

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

〔8〕 T. K. Truong, L. J. Deutsch, I. S. Reed, I. S. Hsu, K. Wang, and C. S. Yeh, "The VLSI Implementation of a Reed-Solomon Encoder Using Berlekamp's Bit-Serial Multiplier Algorithm," *IEEE Transactions on Computers*, Vol.33, No.10, pp.906-911, October 1984.

〔9〕 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.

〔10〕 김창한, 지성연, 장상운, 임종인, "타입 II 최적 정규기저를 갖는 유한체의 새로운 병렬곱셈 연산기," *정보보호학회논문지*, 제16권, 제4호, pp.85-89, 2006. 8.

〔11〕 정석원, 윤중철, 이선옥, "$GF(2^n)$에서의 직렬-병렬 곱셈기 구조," *정보보호학회논문지*, 제13권, 제3호, pp.27-34, 2003. 6.

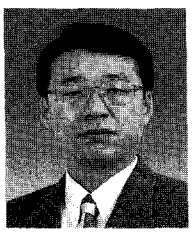
〔12〕 C. H. Kim, C. P. Hong, S. H. Kwon, "A Digit-Serial Multiplier for Finite Field $GF(2^m)$," *IEEE Transactions on VLSI Systems*, Vol.13, No.4, pp.476-483, April 2005.

〔13〕 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.

〔14〕 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.

〔15〕 S. Moon, Y. Lee, J. Park, B. Moon and Y. Lee, "A Fast Finite Field Multiplier Architecture for High-security Elliptic Curve Cryptosystems," *IEICE Transactions on Information and Systems*, Vol.E85-D, No.2, pp. 418-420, February 2002.

〈著者紹介〉



조 용 석 (Yong-Suk Cho) 정회원

1986년 2월: 한양대학교 전자통신과 학사
1988년 2월: 한양대학교 전자통신과 석사
1998년 8월: 한양대학교 전자통신과 박사
1989년 4월~1996년 2월: 한국전기통신공사 연구개발원
1996년 3월~현재: 영동대학교 정보통신·사이버경찰과 부교수
<관심분야> 정보보호, 오류정정부호, 네트워크 보안