情報保護學會論文誌
第17卷 第4號, 2007. 8

$GF(2^m)$ 상의 저복잡도 고속-직렬 곱셈기 구조

조 용 석*
영동대학교

Low Complexity Architecture for Fast-Serial Multiplier in $GF(2^m)$

Yong-Suk Cho*
Youngdong University

요 약

본 논문에서는 $GF(2^m)$ 상의 새로운 저복잡도 고속-직렬 곱셈기 구조를 제안하였다. 고속-직렬 곱셈기는 유한체 $GF(2^m)$의 표준기저 상에서 동작하며, 직렬 곱셈기 보다는 짧은 지연시간에 결과를 얻을 수 있고, 병렬 곱셈기 보다는 적은 하드웨어로 구현할 수 있다. 이 고속-직렬 곱셈기는 회로의 복잡도와 지연시간 사이에 적절한 절충을 피할 수 있는 장점을 가지고 있다. 그러나 기존의 고속-직렬 곱셈기는 t 배의 속도를 향상시키기 위하여 $(t-1)m$ 개의 레지스터가 더 사용되었다. 본 논문에서는 레지스터 수를 증가시키지 않는 새로운 고속-직렬 곱셈기를 설계하였다.

ABSTRACT

In this paper, a new architecture for fast-serial $GF(2^m)$ multiplier with low hardware complexity is proposed. The fast-serial 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 fast-serial architecture is that a trade-off between hardware complexity and delay time can be achieved. But The traditional fast-serial architecture needs extra $(t-1)m$ registers for achieving the t times speed. In this paper a new fast-serial multiplier without increasing the number of registers is presented.

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

Ⅰ. 서 론

유한체(finite fields or Galois fields)는 암호화(cryptography), 오류정정부호(error correcting codes), 디지털 신호처리 등과 같은 여러 분야에서 널리 사용되고 있으며 그 중요성이 점점 커지고 있다. 특히 오류정정부호 중 실용적으로 널리 사용되고 있는 BCH 부호와 Reed-Solomon 부호, 그리고 최근 공개키 암호 알고리듬으로 관심이 집중되고 있는 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 정의되

접수일: 2007년 5월 14일; 채택일: 2007년 7월 18일
* 주저자, yscho@youngdong.ac.kr

며, 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 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, 쌍대기저(dual basis)를 이용한 Berlekamp[5],[6]의 곱셈 알고리즘과, 정규기저(normal basis)를 이용한 Massey와 Omura[7]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 표준기저(standard basis) 상에서 동작하는 곱셈기를 설계한다.

유한체 $GF(2^m)$ 상의 곱셈기는 병렬 곱셈기(parallel multiplier)와 직렬 곱셈기(serial multiplier)로 구현할 수 있다. 병렬 곱셈기는 한 클럭(clock) 내에 결과를 출력하는 회로이며, 직렬 곱셈기는 일반적으로 m 클럭만큼의 시간 지연 후에 결과를 출력한다. 병렬 곱셈기는 연산속도는 빠른 반면에 회로가 복잡하며, 직렬 곱셈기는 회로는 간단하지만 m 클럭만큼의 시간 지연이 생긴다[8].

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.

조용석 등[9]과 Paar 등[10]은 유한체 $GF(2^m)$이 부분체(subfield)를 가지는 경우, 이 부분체 상의 병렬 연산기들을 이용하여 그 확대체 상의 직렬 곱셈기를 구성하는 방법을 제안하였다. 이와 같은 곱셈기를 하이브리드 곱셈기(hybrid multiplier)라고 하는 데, 이 곱셈기는 기존의 병렬 곱셈기에 비해 적은 하드웨어로 구현할 수 있고 직렬 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.

고속-직렬(fast-serial) 곱셈기[11],[12]는 이러한 제약이 없이 모든 유한체에 적용이 가능하며, 기존의 직렬 곱셈기의 긴 지연시간과 병렬 곱셈기의 높은 회로 복잡도 사이에 적절한 절충이 가능한 곱셈기이다. 이 고속-직렬 곱셈기는 유한체의 표준기저 상에서 임의의 두 원소의 곱을 표현한 다항식을 여러 개로 분리한 다음, 이 다항식들을 동시에 처리하는 방식으로 속도를 향상 시키는 직렬 곱셈기이다. 그러나 기존의 고속-직렬 곱셈기는 t배의 속도를 향상시키기 위하여 $(t-1)m$개의 레지스터를 더 사용하여야 하기 때문에 하드웨어가 복잡해지는 단점을 가지고 있다.

본 논문에서는 레지스터의 수를 증가시키지 않는 새로운 고속-직렬 곱셈기 구조를 제안하였다. 제안된 곱셈기는 기존의 고속-직렬 곱셈기보다 훨씬 간단한 하드웨어로 구현할 수 있는 장점을 가지고 있다.

본 논문의 구성은, 먼저 Ⅱ.에서 유한체 상의 직렬 곱셈 알고리즘을 이용하여 회로의 복잡도와 지연시간 사이의 적절한 절충을 꾀할 수 있는 고속-직렬 곱셈기를 설계한다. Ⅲ.에서는 레지스터 수를 증가시키지 않는 새로운 저복잡도 고속-직렬 곱셈기를 설계하고 기존의 곱셈기와 비교한다. 그리고 Ⅳ.에서 결론을 맺는다.

Ⅱ. $GF(2^m)$ 상의 고속-직렬 곱셈기

유한체 $GF(2^m)$은 2^m 개의 원소를 가지고 있으며 그 원소들은 2진 계수를 갖는 $m-1$차 이하의 다항식으로 표현할 수 있다. 즉 유한체 $GF(2^m)$은 $GF(2)$ 상의 m차원 벡터공간(vector space)이 되며, 여기에서 선형 독립인 m개의 벡터를 기저(basis)라고 한다.

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

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

유한체 $GF(2^m)$의 원시원을 α라고 하고, 이 α를 식 (1)과 같은 원시다항식의 근(root)으로 정의한다. 그러면 유한체 $GF(2^m)$의 영원을 제외한 2^m-1개의 모든 원소들은 다음과 같이 원시원 α의 멱(power)으로 표현

할 수 있다.

$$GF(2^m) = \{0, \alpha^0, \alpha^1, \alpha^2, \cdots\cdots, \alpha^{2^m-2}\} \quad (2)$$

이러한 표현방식을 지수표현(exponential representation) 또는 멱표현(power representation)이라고 한다.

원시원 α는 식 (1)과 같은 원시다항식의 근이므로, 즉 $p(\alpha) = 0$이므로

$$\alpha^m = p_0 + p_1\alpha + p_2\alpha^2 + \cdots\cdots + p_{m-1}\alpha^{m-1} \quad (3)$$

가 된다. 따라서 식 (3)을 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.

$$\begin{aligned} U &= u_0 + u_1\alpha + \cdots\cdots + u_{m-1}\alpha^{m-1} \quad (4) \\ &= \sum_{i=0}^{m-1} u_i\alpha^i, \quad u_i \in GF(2) \end{aligned}$$

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

$$\{1, \alpha, \alpha^2, \cdots\cdots, \alpha^{m-2}, \alpha^{m-1}\} \quad (5)$$

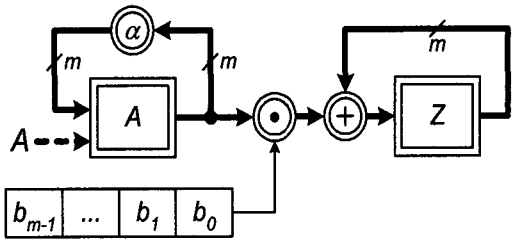
이상과 같이 유한체 $GF(2^m)$의 원소들은 지수표현과 다항식표현의 2가지 방법으로 표현할 수 있다. 지수표현을 이용하면 곱셈과 나눗셈은 각각 2진수의 덧셈과 뺄셈으로 대치되므로 쉽게 수행할 수 있는 반면에 덧셈이 복잡해지며, 다항식표현을 이용하면 덧셈은 각 비트별 2원합(modulo-2 sum)으로 간단하게 수행되지만 곱셈과 나눗셈이 어려워지는 문제점을 가지고 있다. $GF(2^m)$의 차수 m이 작은 경우에는 지수표현을 이용한 연산이 더 쉬운 반면, m이 커지면 지수표현 보다는 다항식표현을 이용한 연산이 더 적은 하드웨어로 구현할 수 있으며 고속처리가 가능하다.

유한체 $GF(2^m)$ 상의 임의의 두 원소 A와 B를 식 (4)와 같이 다항식표현으로 나타내면 다음과 같이 된다.

$$A = a_0 + a_1\alpha + \cdots\cdots + a_{m-1}\alpha^{m-1}, \quad a_i \in GF(2) \quad (6)$$

$$B = b_0 + b_1\alpha + \cdots\cdots + b_{m-1}\alpha^{m-1}, \quad b_i \in GF(2) \quad (7)$$

이 두 원소의 곱을 Z라 하면 Z는



〔그림 1〕 $GF(2^m)$ 상의 직렬 곱셈기

$$\begin{aligned} Z &= A \cdot B \quad (8) \\ &= A \cdot (b_0 + b_1\alpha + b_2\alpha^2 + \cdots\cdots + b_{m-1}\alpha^{m-1}) \end{aligned}$$

가 된다. 또한 식 (8)을 다시 정리하면 다음과 같이 쓸 수 있다.

$$Z = b_0A + b_1[A\alpha] + b_2[A\alpha^2] + \cdots\cdots + b_{m-1}[A\alpha^{m-1}] \quad (9)$$

식 (9)를 살펴보면, 두 원소의 곱 Z는 임의의 한 원소 A에 α를 곱해 가면서 B의 계수들과 차례로 곱하여 계속 더하는 것이다. 따라서 식 (9)를, LFSR (Linear Feedback Shift Register)을 이용하여 구현하면 〔그림 1〕과 같은 직렬 곱셈기를 설계할 수 있다[9].

〔그림 1〕에서 굵은 선은 m비트 버스이고, □는 m비트 레지스터를, ⊕는 m개의 2입력 XOR 게이트를, ⊙은 m개의 2입력 AND 게이트를, ⓐ는 $GF(2^m)$의 원시원 α를 곱하는 회로를 나타내고 있다. 〔그림 1〕 회로의 동작은 초기상태에서 레지스터 Z는 클리어시키고 임의의 두 원소 A와 B를 각각 레지스터 A와 B에 로드시킨다. 그리고 각 레지스터를 m번 쉬프트시키면 레지스터 Z에 두 원소의 곱 Z가 저장된다. 따라서 m 클럭 시간에 곱셈의 결과를 얻을 수 있다.

〔그림 1〕과 같은 유한체 상의 직렬 곱셈기는 m 클럭 시간 후에 곱셈의 결과가 나온다. 이를 고속화하기 위하여 식 (9)를 t개로 분할하여 각 부분에 하드웨어 자원을 할당하면 t배의 속도를 향상시킬 수 있다. 식 (9)를 t개로 분할하면 다음과 같이 정리할 수 있다.

$$\begin{aligned} Z_0 &= b_0A + b_tA\alpha^t + b_{2t}A\alpha^{2t} + \cdots\cdots \quad (10) \\ Z_1 &= b_1(A\alpha) + b_{t+1}(A\alpha)\alpha^t + b_{2t+1}(A\alpha)\alpha^{2t} + \cdots\cdots \\ Z_2 &= b_2(A\alpha^2) + b_{t+2}(A\alpha^2)\alpha^t + b_{2t+2}(A\alpha^2)\alpha^{2t} + \cdots\cdots \\ &\vdots \\ Z_{t-1} &= b_{t-1}(A\alpha^{t-1}) + b_{2t-1}(A\alpha^{t-1})\alpha^t \\ &\quad + b_{3t-1}(A\alpha^{t-1})\alpha^{2t} + \cdots\cdots \end{aligned}$$

문헌 [12]에서는 식 (10)을 이용하여 [그림 2]와 같은 t배속 고속-직렬 곱셈기를 설계하였다. [그림 2]와 같은 곱셈기는 $\lceil m/t \rceil$ 클럭 시간에 곱셈의 결과를 얻을 수 있다.

[그림 1]과 [그림 2]를 비교하면 t배의 속도를 향상시키기 위하여 A_1부터 A_{t-1}까지 $t-1$개의 m비트 레지스터가 더 사용되었으며, α^2부터 α^t까지 $2(t-1)$개의 상수(constant) 곱셈기와 $(t-1)m$개의 2입력 AND 게이트, 그리고 $(t-1)m$개의 2입력 XOR 게이트가 더 사용되었음을 알 수 있다.

Ⅲ. $GF(2^m)$ 상의 저복잡도 고속-직렬 곱셈기

[그림 2]와 같은 고속-직렬 곱셈기에서 레지스터 수를 줄이기 위하여 식 (10)을 식 (11)과 같이 변형한다. 식 (11)을 이용하면 [그림 3]과 같은 저복잡도 고속-직렬 곱셈기를 설계할 수 있다.

$$
\begin{aligned}
Z_0 &= b_0[A] + b_t[A\alpha^t] + b_{2t}[A\alpha^{2t}] + \cdots\cdots \\
Z_1 &= b_1[A]\alpha + b_{t+1}[A\alpha^t]\alpha + b_{2t+1}[A\alpha^{2t}]\alpha + \cdots\cdots \\
Z_2 &= b_2[A]\alpha^2 + b_{t+2}[A\alpha^t]\alpha^2 + b_{2t+2}[A\alpha^{2t}]\alpha^2 + \cdots\cdots \\
&\vdots \\
Z_{t-1} &= b_{t-1}[A]\alpha^{t-1} + b_{2t-1}[A\alpha^t]\alpha^{t-1} \\
&\quad + b_{3t-1}[A\alpha^{2t}]\alpha^{t-1} + \cdots\cdots
\end{aligned}
\tag{11}
$$

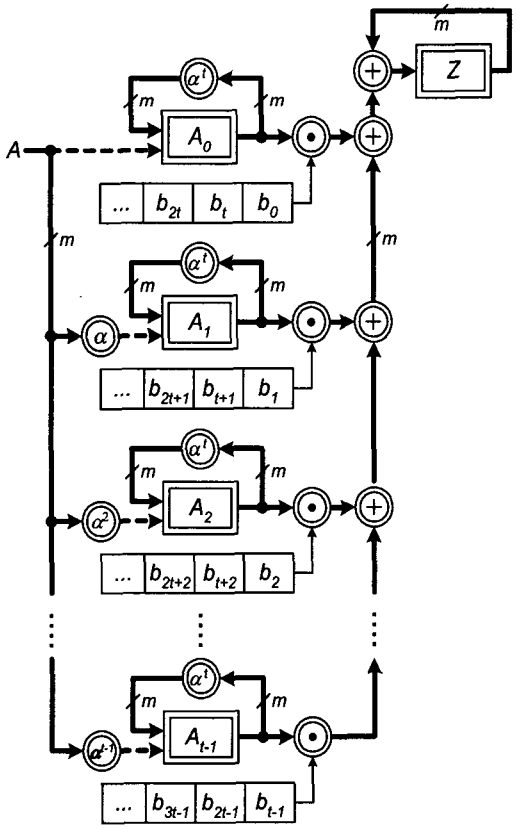
[그림 2]와 [그림 3]을 비교하면 $t-1$개의 m비트 레지스터가 생략되었으며, $t-1$개의 α^t 곱셈기가 생략되었음을 알 수 있다.

예를 들어 원시다항식이 $p(x) = 1 + x^2 + x^5$ 인 유한체 $GF(2^5)$에서 3배속 고속-직렬 곱셈기를 설계하여 보자. 두 원소의 곱을 Z라 하면

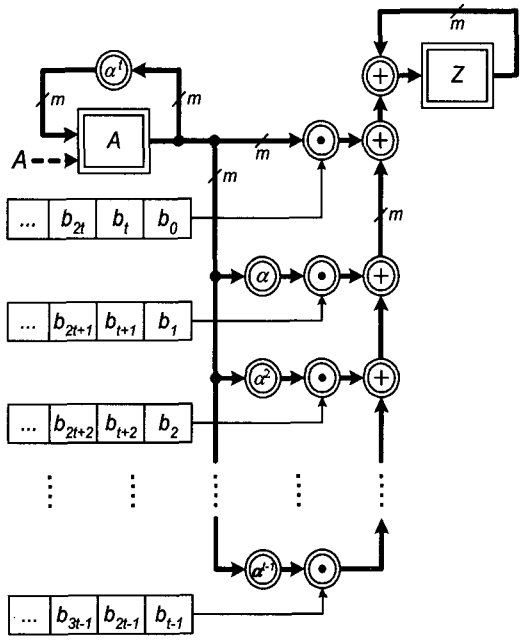
$$
\begin{aligned}
Z &= A \cdot (b_0 + b_1\alpha + b_2\alpha^2 + b_3\alpha^3 + b_4\alpha^4) \\
&\equiv Z_0 + Z_1 + Z_2
\end{aligned}
\tag{12}
$$

가 된다. 여기에서 Z_0, Z_1, Z_2은 각각 다음과 같이 쓸 수 있다.

$$
\begin{aligned}
Z_0 &= A \cdot (b_0 + b_3\alpha^3) \\
&= b_0[A] + b_3[A\alpha^3]
\end{aligned}
\tag{13}
$$



[그림 2] $GF(2^m)$ 상의 t 배속 고속-직렬 곱셈기



[그림 3] $GF(2^m)$ 상의 저복잡도 t 배속 고속-직렬 곱셈기

$$Z_1 = A\cdot(b_1\alpha + b_4\alpha^4) \quad (14)$$
$$= b_1[A]\alpha + b_4[A\alpha^3]\alpha$$

$$Z_2 = A\cdot(b_2\alpha^2 + b_5\alpha^5) \quad (15)$$
$$= b_2[A]\alpha^2 + 0[A\alpha^3]\alpha^2$$

그러므로 식 (13), 식 (14), 식 (15)를 이용하여 $GF(2^5)$ 상에서 3배속 고속-직렬 곱셈기를 설계하면 〔그림 4〕와 같이 된다. 〔그림 4〕의 회로는 $\lceil 5/3 \rceil = 2$ 클럭 시간에 곱셈의 결과를 얻을 수 있다.

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

$$A\alpha = a_0\alpha + a_1\alpha^2 + a_2\alpha^3 \quad (16)$$
$$+ a_3\alpha^4 + a_4(1+\alpha^2)$$
$$= a_4 + a_0\alpha + (a_1+a_4)\alpha^2$$
$$+ a_2\alpha^3 + a_3\alpha^4$$

$$A\alpha^2 = a_3 + a_4\alpha + (a_0+a_3)\alpha^2 \quad (17)$$
$$+ (a_1+a_4)\alpha^3 + a_2\alpha^4$$

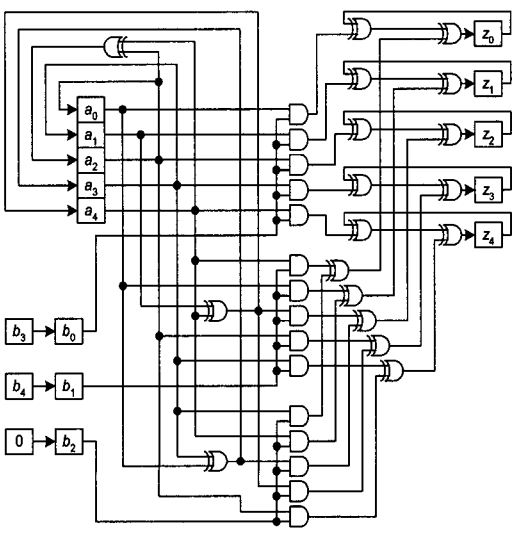
$$A\alpha^3 = a_2 + a_3\alpha + (a_2+a_4)\alpha^2 \quad (18)$$
$$+ (a_0+a_3)\alpha^3 + (a_1+a_4)\alpha^4$$

식 (16), 식 (17), 식 (18)을 이용하면 〔그림 4〕에서와 같이, $GF(2^5)$ 상의 상수 곱셈기를 설계할 수 있다. 〔그림 4〕에서 보듯이 α, α^2, α^3곱셈기를 3개의 2입력 XOR 게이트로 구현할 수 있다.

〔표 1〕에 기존의 직렬 곱셈기와 문헌 〔12〕의 고속-직렬 곱셈기, 그리고 본 논문에서 제안한 저복잡도 고속-직렬 곱셈기의 하드웨어와 클럭 수를 비교하였다. 〔표 1〕에서 볼 수 있듯이 제안된 저복잡도 고속-직렬 곱셈기는 문헌 〔12〕의 곱셈기에 비해, $(t-1)m$개의 레지스터가 더 적게 사용되었으며, 상수 곱셈기도 $t-1$개 적게 사용되어 훨씬 적은 하드웨어로 구현할 수 있음을 알 수 있다.

2입력 AND 게이트 1개의 지연을 T_A라 하고, 2입력 XOR 게이트 1개의 지연을 T_X라고 하면, 〔그림 1〕과 같은 직렬 곱셈기의 임계 경로 지연(critical path delay)은 $T_A + T_X$가 된다. 또한 〔그림 2〕와 같은 고속-직렬 곱셈기[12]의 임계 경로 지연은 $T_A + (\lceil \log_2(t+1) \rceil) T_X$가 된다.

$\alpha^i (1 \le i \le t-1)$를 곱하는 상수 곱셈기의 지연을 T_C라 하면, 제안된 저복잡도 고속-직렬 곱셈기의 임계 경로 지연은 $T_C + T_A + (\lceil \log_2(t+1) \rceil) T_X$가 된다. 상수 곱셈기의 지연 T_C는 원시다항식의 항수에 따라 달라지지만 주로 많이 사용되는 삼항(trinomial) 및 오항(pentanomial) 원시다항식의 경우에는 〔그림 4〕에서와 같이 대략 T_X 또는 $2T_X$ 정도가 된다.



〔그림 4〕 $GF(2^5)$ 상의 3배속 고속-직렬 곱셈기

Ⅳ. 결 론

본 논문에서는, 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 있는 고속-직렬 곱셈기를 구현하는데 있어서 기존의 방법에 비해 훨씬 간단한 하드웨어로 구현할 수 있는 새로운 구조를 제안하였다.

〔표 1〕 유한체 곱셈기들의 성능 비교

	직렬 곱셈기	문헌 〔12〕의 곱셈기	제안된 곱셈기
2입력 AND 게이트 수	m	tm	tm
2입력 XOR 게이트 수	m	tm	tm
레지스터 수	$3m$	$(t+2)m$	$3m$
상수 곱셈기 수	1	$2t-1$	t
클럭 수	m	$\lceil m/t \rceil$	$\lceil m/t \rceil$

제안된 저복잡도 고속-직렬 곱셈기는 사용하는 유한체의 위수가 합성수이어야 한다는 하이브리드 유한체 곱셈기[9],[10]가 가지는 제약이 필요 없이 모든 유한체를 선택할 수 있으며, $(t-1)m$개의 2입력 AND 게이트와 $(t-1)m$개의 2입력 XOR 게이트, 그리고 $t-1$개의 상수 곱셈기를 추가하면 기존의 직렬 곱셈기에 비해 t배 빨리 곱셈의 결과를 얻을 수 있는 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] 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.

[4] R. J. McEliece, *Finite Fields for Computer Scientist and Engineers*, Kluwer Academic, 1987.

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

[6] 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.

[7] 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.

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

[9] 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.

[10] 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.

[11] 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.

[12] 조용석, "유한체 상에서 고속 연산을 위한 직렬 곱셈기의 병렬화 구조," *정보보호학회논문지*, 제17권, 제1호, pp.33-39, 2007. 2.

〈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·사이버경찰학과 부교수
<관심분야> 정보보호, 오류정정부호, 네트워크 보안