저복잡도 디지트병렬/비트직렬 다항식기저 곱셈기

정회원 조 용 석*

Low Complexity Digit-Parallel/Bit-Serial Polynomial Basis Multiplier

Yong-Suk Cho* ***Regular Member***

요 약

본 논문에서는 $GF(2^m)$ 상에서 새로운 저복잡도 디지트병렬/비트직렬 곱셈기를 제안한다. 제안된 곱셈기는 $GF(2^m)$의 다항식기저에서 동작하며, D 클럭 사이클마다 곱셈의 결과를 출력한다. 여기에서 D는 임의로 선택할 수 있는 디지트의 크기이다. 디지트병렬/비트직렬 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. 그러나 기존의 디지트병렬/비트직렬 곱셈기는 속도를 향상시키기 위하여 더 많은 하드웨어를 사용하였다. 본 논문에서는 하드웨어 복잡도를 낮춘 새로운 디지트병렬/비트직렬 곱셈기를 설계한다.

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 complexity 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 digit-parallel/bit-serial architecture is that a trade-off between hardware complexity and delay time can be achieved. But the traditional digit-parallel/bit-serial multiplier needs extra hardware for high speed. In this paper a new low complexity efficient digit-parallel/bit-serial multiplier is presented.

Ⅰ. 서 론

유한체(finite fields or Galois fields)는 암호화(cryptography), 오류정정부호(error correcting codes), 디지털 신호처리 등과 같은 여러 분야에서 널리 사용되고 있다. 특히 오류정정부호 중 BCH 부호와 Reed-Solomon 부호, 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 연산 중에서 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 방법들이 집중적으로 연구되고 있다.

유한체 연산의 효율성을 결정하는 중요한 요소 중 하나는 유한체의 원소들을 표현하는데 사용하는

* 영동대학교 정보통신사이버경찰학과 (yscho@youngdong.ac.kr)
논문번호 : KICS2009-08-346, 접수일자 : 2009년 8월 11일, 최종논문접수일자 : 2010년 3월 17일

기저(basis)의 선택이다. 따라서 기존에 사용되던 다항식기저(polynomial basis)를 적절히 변환하여 소요되는 하드웨어 및 지연시간을 줄이고자 하는 많은 방법들이 발표되고 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. 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 복잡하고, 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[8]과 Paar 등[9]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 하는데, 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.

디지트병렬/비트직렬(digit-parallel/bit-serial) 곱셈기[10]는 이러한 제약이 없이 모든 유한체에 적용이 가능하며, 기존의 직렬 곱셈기의 긴 지연시간과 병렬 곱셈기의 높은 회로 복잡도 사이에 적절한 절충이 가능한 곱셈기이다. 이 디지트병렬/비트직렬 곱셈기는 데이터를 일정한 길이의 디지트로 나누고, 디지트 내부는 비트직렬 곱셈기를 사용하고 전체적으로는 디지트병렬 방식으로 곱셈을 처리한다. 따라서 디지트의 길이인 D 클럭 만에 곱셈의 결과를 얻을 수 있다. 그러나 기존의 디지트병렬/비트직렬 곱셈기는 속도를 향상시키기 위하여 비트직렬 곱셈기에 비하여 $(d-1)m$ 개의 레지스터를 더 사용하여야 하기 때문에 하드웨어가 복잡해지는 단점을 가지고 있다. 여기에서 디지트 개수 d는 $\lceil m/D \rceil$이다.

본 논문에서는 비트직렬 곱셈기와 동일한 $3m$ 개의 레지스터만을 사용하는, 즉 $(d-1)m$ 개의 레지스터를 절약할 수 있는 새로운 디지트병렬/비트직렬 곱셈기 구조를 제안한다. 제안된 곱셈기는 기존의 디지트병렬/비트직렬 곱셈기보다 훨씬 간단한 하드웨어로 구현할 수 있는 장점을 가지고 있다.

본 논문의 구성은, 먼저 Ⅱ.에서 유한체 상의 비트직렬 곱셈 알고리즘을 이용하여 회로의 복잡도와 지연시간 사이의 적절한 절충을 꾀할 수 있는 디지트병렬/비트직렬 곱셈기를 설계한다. Ⅲ.에서는 레지스터 수를 증가시키지 않는 새로운 저복잡도 디지트병렬/비트직렬 곱셈기를 설계하고 Ⅳ.에서 실험 및 비교를, 그리고 Ⅴ.에서 결론을 맺는다.

Ⅱ. GF(2^m) 상의 디지트병렬/비트직렬 곱셈기

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. 모든 유한체 $GF(2^m)$은 영원(zero element), 단위원(unit element), 원시원(primitive element)을 가지고 있으며, 다음과 같은 차수가 m 인 원시다항식(primitive polynomial)을 최소한 1개 이상 가지고 있다[4].

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

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

$$\alpha^m = 1 + p_1 \alpha + \cdots\cdots + p_{m-1} \alpha^{m-1} \tag{2}$$

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 두 원소 A와 B는

다음과 같이 쓸 수 있다.

$$A = a_0 + a_1\alpha + \cdots\cdots + a_{m-1}\alpha^{m-1} \quad (3)$$

$$B = b_0 + b_1\alpha + \cdots\cdots + b_{m-1}\alpha^{m-1} \quad (4)$$

여기에서, 이 두 원소의 곱을 Z라 하면 Z는

$$\begin{aligned} Z &= A \cdot B \\ &= A \cdot (b_0 + b_1\alpha + \cdots\cdots + b_{m-1}\alpha^{m-1}) \end{aligned} \quad (5)$$

가 된다. 또한 식 (5)를 다시 정리하면 다음과 같이 쓸 수 있다.

$$\begin{aligned} Z = {} & b_0 A + b_1[A\alpha] + b_2[A\alpha^2] \\ & + \cdots\cdots + b_{m-1}[A\alpha^{m-1}] \end{aligned} \quad (6)$$

식 (6)을 살펴보면, 두 원소의 곱 Z는 임의의 한 원소 A에 a를 계속 곱해 가면서 B의 계수들을 차례로 곱하여 더하는 것이다. 따라서 식 (6)을 이용하면 그림 1과 같은 비트직렬 곱셈기를 설계할 수 있다.

그림 1과 같은 비트직렬 곱셈기는 m 클럭 시간 후에 곱셈의 결과가 나온다. 이를 고속화하기 위하여 식 (5)에서 원소 B를 다음과 같이 D 비트씩 묶어서 $d = \lceil m/D \rceil$ 개로 분할한다.

$$B = \sum_{i=0}^{d-1} B_i (\alpha^D)^i \quad (7)$$

여기에서 B_i는 다음과 같이 쓸 수 있다.

$$B_i = \sum_{j=0}^{D-1} (b_{iD+j})\alpha^j \quad (8)$$

따라서 식 (7)을 식 (5)에 대입하면 다음과 같이 된다.

$$\begin{aligned} Z = A \cdot B &= A \cdot \left(\sum_{i=0}^{d-1} B_i \alpha^{iD} \right) \\ &= \sum_{i=0}^{d-1} (A\alpha^{iD}) B_i \end{aligned} \quad (9)$$

또한 식 (9)를 풀어쓰면 다음과 같이 쓸 수 있다.

$$\begin{aligned} Z = {} & AB_0 + (A\alpha^D)B_1 + (A\alpha^{2D})B_2 \\ & + \cdots\cdots + (A\alpha^{(d-1)D})B_{d-1} \end{aligned} \quad (10)$$

식 (10)의 첫 번째 항에 식 (8)을 대입하면

$$\begin{aligned} AB_0 &= A\left(\sum_{j=0}^{D-1} b_j \alpha^j \right) \\ &= A(b_0 + b_1\alpha + \cdots\cdots + b_{D-1}\alpha^{D-1}) \end{aligned} \quad (11)$$

가 된다. 식 (11)은 식 (5)와 동일한 구조를 가지고 있으므로 식 (11)은 그림 1과 같은 비트직렬 곱셈기로 구현할 수 있다. 또한 식 (10)의 두 번째 항은 A 대신 $A\alpha^D$를 대입하면 식 (11)과 동일한 구조가 된다. 따라서 식 (10)를 이용하면 그림 2와 같은 디지트병렬/비트직렬 곱셈기를 설계할 수 있다[10].

그림 1과 그림 2에서 굵은 선은 m 비트 버스이고, □는 m 비트 레지스터를, ⊕는 m 개의 2입력

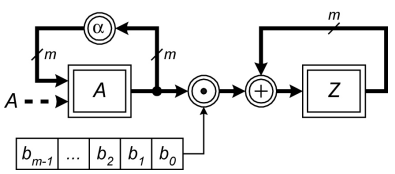

그림 1. $GF(2^m)$ 상의 비트직렬 곱셈기

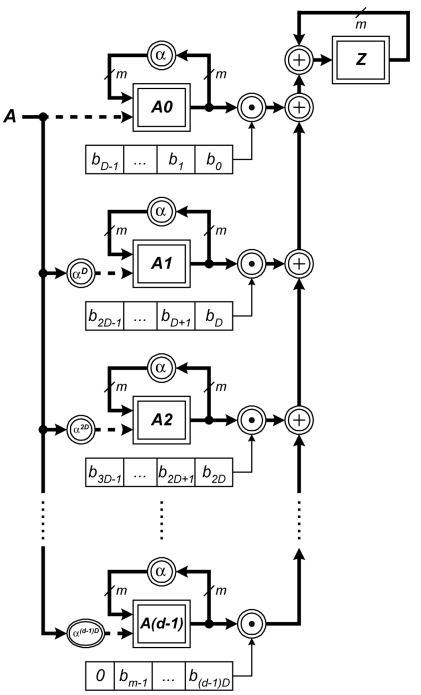

그림 2. $GF(2^m)$ 상의 디지트병렬/비트직렬 곱셈기

XOR 게이트를, ⊙은 m 개의 2입력 AND 게이트를, ⓐ는 a^i를 곱하는 상수 곱셈기를 나타내고 있다. 그림 2의 회로는 초기 상태에 레지스터 Z를 클리어시키고 첫 번째 레지스터에는 A를 로드시키고, 두 번째 레지스터에는 Aa^D를, 세 번째 레지스터에는 Aa^{2D}를, ……, 마지막 레지스터에는 $Aa^{(d-1)D}$을 로드시킨다. 그리고 각 레지스터를 D 번 쉬프트 시키면 레지스터 Z에 두 원소를 곱한 결과가 저장된다. 따라서 D 클럭 시간에 곱셈의 결과를 얻을 수 있다.

그림 2의 곱셈기를 그림 1의 곱셈기와 비교해보면, $(d-1)$ 개의 m 비트 레지스터와 $(d-1)m$ 개의 2입력 XOR 게이트, $(d-1)m$ 개의 2입력 AND 게이트, 그리고 $(d-1)$ 개의 a 곱셈기와 a^D, a^{2D}, ……, $a^{(d-1)D}$의 상수 곱셈기가 더 사용되었음을 알 수 있다. 그러나 D 클럭 시간에 곱셈의 결과를 얻을 수 있다. 따라서 디지트 길이 D를 적절히 선택하면 회로의 복잡도와 지연시간 사이에 적절한 절충을 도모할 수 있게 된다.

Ⅲ. 저복잡도 디지트병렬/비트직렬 곱셈기 설계

(그림 2)와 같은 디지트병렬/비트직렬 곱셈기에서 레지스터 수를 절약하기 위하여, 식 (10)의 첫 번째 항을 다음과 같이 변형시킨다.

$$\begin{aligned} AB_0 &= A(b_0+b_1\alpha+b_2\alpha^2+\cdots\cdots+b_{D-1}\alpha^{D-1}) \\ &= [A]b_0+[(A\alpha)]b_1+[(A\alpha^2)]b_2 \\ &\quad +\cdots\cdots+[(A\alpha^{D-1})]b_{D-1} \end{aligned} \tag{12}$$

같은 방식으로 두 번째, 세 번째, 그리고 마지막 항을 변형하면 다음과 같이 된다.

$$\begin{aligned} (A\alpha^D)B_1 &= (A\alpha^D)(b_D+b_{D+1}\alpha+b_{D+2}\alpha^2 \\ &\quad +\cdots\cdots+b_{2D-1}\alpha^{D-1}) \\ &= [A\alpha^D]b_D+[(A\alpha)\alpha^D]b_{D+1} \\ &\quad +[(A\alpha^2)\alpha^D]b_{D+2} \\ &\quad +\cdots\cdots+[(A\alpha^{D-1})\alpha^D]b_{2D-1} \end{aligned} \tag{13}$$

$$\begin{aligned} (A\alpha^{2D})B_2 &= (A\alpha^{2D})(b_{2D}+b_{2D+1}\alpha+b_{2D+2}\alpha^2 \\ &\quad +\cdots\cdots+b_{3D-1}\alpha^{D-1}) \\ &= [A\alpha^{2D}]b_{2D}+[(A\alpha)\alpha^{2D}]b_{2D+1} \\ &\quad +[(A\alpha^2)\alpha^{2D}]b_{2D+2} \\ &\quad +\cdots\cdots+[(A\alpha^{D-1})\alpha^{2D}]b_{3D-1} \end{aligned} \tag{14}$$

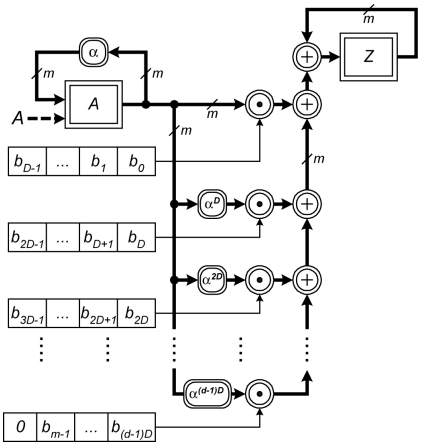

그림 3. $GF(2^m)$상의 저복잡도 디지트병렬/비트직렬 곱셈기

$$\begin{aligned} (A\alpha^{(d-1)D})B_{d-1} &= [A\alpha^{(d-1)D}]b_{(d-1)D} \\ &\quad +[(A\alpha)\alpha^{(d-1)D}]b_{(d-1)D+1} \\ &\quad +[(A\alpha^2)\alpha^{(d-1)D}]b_{(d-1)D+2} \\ &\quad +\cdots\cdots+[(A\alpha^{D-1})\alpha^{(d-1)D}]b_{dD-1} \end{aligned} \tag{15}$$

식 (13)은 임의의 한 원소 A에 a를 계속 곱하는 항에 상수인 a^D를 곱한 다음 B의 계수인 b_D, b_{D+1}, b_{D+2}, ……, b_{2D-1}를 곱하는 것이다. 즉 식 (13)은 식 (12)와 비교하면 곱하는 B의 계수가 다르고 a^D를 곱하는 것만 다를 뿐 나머지는 동일하다. 식 (14)와 식 (15)는 곱하는 상수가 각각 a^{2D}와 $a^{(d-1)D}$인 것만 다를 뿐 식 (13)과 동일한 구조이다. 따라서 식 (12)~(15)를 이용하면 그림 3과 같은 저복잡도 디지트병렬/비트직렬 곱셈기를 설계할 수 있다.

그림 3을 그림 2와 비교하면, $(d-1)m$ 개의 레지스터가 절약되었으며 a를 곱하는 상수 곱셈기가 $(d-1)$ 개 적게 사용되어 훨씬 적은 하드웨어로 구현할 수 있음을 알 수 있다.

Ⅳ. 실험 및 비교

제안된 곱셈기가 정상적으로 동작하는지를 확인하기 위하여, 원시다항식이 $p(x)=1+x^2+x^5$ 인 유한체 $GF(2^5)$에서 $D=3$인 경우의 디지트병렬/비트직렬 곱셈기를 설계하여 보자. 임의의 두 원소 A와 B의 곱 Z는 다음과 같이 쓸 수 있다.

$$\begin{aligned} Z &= A\cdot(b_0+b_1\alpha+b_2\alpha^2+b_3\alpha^3+b_4\alpha^4) \\ &= A\cdot(b_0+b_1\alpha+b_2\alpha^2) \\ &\quad +A\cdot\alpha^3(b_3+b_4\alpha+0) \end{aligned} \tag{16}$$

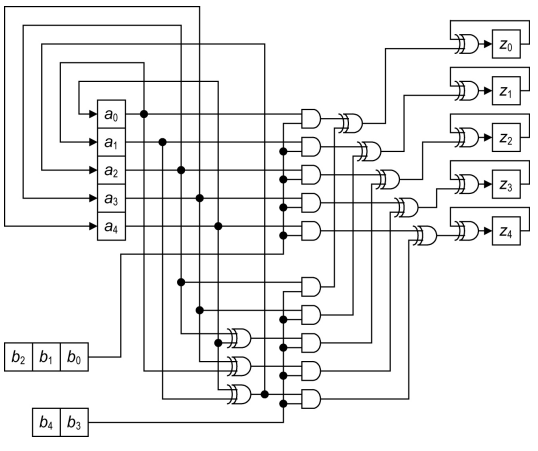

그림 4. $GF(2^5)$ 상에서 $D=3$ 일 때 저복잡도 디지트병렬/비트직렬 곱셈기

식 (16)을 이용하여 그림 3과 같은 곱셈기를 설계하면 그림 4와 같이 된다. 그림 4의 회로는 $D=3$ 클럭 시간에 곱셈의 결과를 얻을 수 있다.

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

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

따라서 식 (17)을 이용하면 그림 4에서와 같이, $GF(2^5)$ 상의 상수곱셈기를 설계할 수 있다. 그림 4에서 보듯이 a와 a^3을 곱하는 상수곱셈기는 3개의 2입력 XOR 게이트로 구현할 수 있다.

그림 4와 같이 설계된 곱셈기의 동작을 검증하기 위하여 ALTERA의 MAX+PLUS II 툴을 사용하여 VHDL로 구현하고 시뮬레이션 하였다. 그림 5는 시뮬레이션의 결과이다. 그림 5에서 입력 A를 0x12 ($10010=a^{30}$)로 하고 입력 B를 0x0A ($01010=a^6$)로 하면 3클럭 후에 출력 Z가 0x05 ($00101=a^5$)이 출력되는 것을 볼 수 있다.

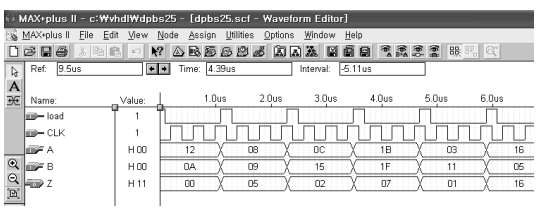

그림 5. 그림 4 곱셈기의 시뮬레이션 결과

표 1. 유한체 곱셈기들의 성능 비교

	하이브리드 곱셈기 [9]	문헌 [11] 곱셈기	문헌 [10] 곱셈기	제안된 곱셈기
레지스터	$3m$	$(d+2)m$	$(d+2)m$	$3m$
2입력 AND	dm	dm	dm	dm
2입력 XOR	$(d+1)m-D$	$(2d-1)m$	dm	dm
클럭	D ($m=dD$ 일 때)	D	D	D

표 1에 기존의 유한체 곱셈기들과 본 논문에서 제안한 저복잡도 디지트병렬/비트직렬 곱셈기의 하드웨어와 클럭 수를 비교하였다. 상수곱셈기는 각 곱셈기에서의 하드웨어 부담이 비슷하다고 가정하고 비교 대상에서 제외하였다. 표 1에서 볼 수 있듯이 제안된 저복잡도 디지트병렬/비트직렬 곱셈기는 다른 곱셈기에 비해 더 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 1과 같은 직렬 곱셈기의 임계경로지연(critical path delay)은 T_A+T_X가 된다. 또한 그림 2와 같은 문헌 [10]의 디지트병렬/비트직렬 곱셈기의 임계경로지연은 $T_A+\lceil \log_2(d+1) \rceil T_X$가 된다. $a^i\ (1 \le i \le d-1)$를 곱하는 상수 곱셈기의 지연을 T_C라 하면, 제안된 곱셈기의 임계경로지연은 $T_C+T_A+\lceil \log_2(d+1) \rceil T_X$가 된다.

V. 결 론

본 논문에서는, 직렬 곱셈기의 긴 지연시간과 병렬 곱셈기의 복잡한 회로 사이를 적절하게 절충함으로써, 비트직렬 곱셈기보다는 짧은 지연시간에 결과를 얻을 수 있으며 비트병렬 곱셈기보다는 적은 회로로 구현할 수 있는 디지트병렬/비트직렬 곱셈기를 구현하는데 있어서 기존의 방법에 비해 훨씬 간단한 하드웨어로 구현할 수 있는 새로운 구조를 제안하였다.

제안된 저복잡도 디지트병렬/비트직렬 곱셈기는 사용하는 유한체의 위수가 합성수이어야 한다는 하이브리드 곱셈기[8],[9]가 가지는 제약이 없이 모든 유한체를 선택할 수 있으며, $(d-1)m$ 개의 2입력 AND 게이트와 $(d-1)m$ 개의 2입력 XOR 게이트, 그리고 $d-1$ 개의 상수 곱셈기를 추가하면 기존의 비트직렬 곱셈기보다 빠른 D 클럭만에 곱셈의 결과를 얻을

수 있는 장점을 가지고 있다.

참 고 문 헌

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

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

〔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〕 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.

〔9〕 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.

〔10〕 조용석, "다항식기저를 이용한 $GF(2^m)$ 상의 디지트병렬/비트직렬 곱셈기," *한국통신학회논문지*, 제33권, 제11호, pp.892-897, 2008. 11.

〔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.

조 용 석 (Yong-Suk Cho) 정회원

1986년 2월 한양대학교 전자통신공학과 학사

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