공통인수 후처리 방식에 기반한 고속 유한체 곱셈기

문상국*

Fast GF(2m) Multiplier Architecture Based on Common Factor Post-Processing Method

Sangook Moon*

요 약

비도 높은 암호용으로 연구된 유한체 곱셈 연산기는 크게 직렬 유한체 곱셈기, 배열 유한체 곱셈기, 하이브리드 유한체 곱셈기으로 분류되어 왔다. 직렬 유한체 곱셈기는 마스트로비토 (Mastrovito) [1]에 의하여 제안되어 유한체 곱셈기의 가장 기본적인 구조로 자리잡아 왔고, 이를 병렬로 처리하기 위해 m 배의 자원을 투자하여 m 배의 속도를 얻어낸 결과가 2차원 배열 유한체 곱셈기이며 [2], 이들 기존 방식의 장점만을 취하여 제안된 방식이 1999년 Paar에 의해 제안된 하이브리드 (hybrid) 곱셈기이다 [3]. 반면 이 하이브리드 곱셈기는 사용 가능한 유한체로서 유한체의 차수를 합성수로 사용해야 한다는 제약이 따른다. 본 논문에서는 마스트로비토의 곱셈기의 구조를 기본으로 하고, 수식적으로 공통인수를 끌어내어 후처리하는 기법을 유도하여 적용한다. 제안한 방식으로 설계한 새로운 유한체 곱셈기는 HDL로 구현하여 소프트웨어 측면 뿐 아니라 하드웨어 측면에서도 그 기능과 성능을 검증하였다. 제안된 방식에서 직렬 다항 기준식 (polynomial)을 t (t는 1보다 큰 양의 정수) 부분으로 나누어 적용하였을 경우 곱셈기는 t 배의 속도 향상을 보일 수 있다.

ABSTRACT

So far, there have been grossly 3 types of studies on GF(2m) multiplier architecture, such as serial multiplication, array multiplication, and hybrid multiplication. Serial multiplication method was first suggested by Mastrovito [1], to be known as the basic GF(2m) multiplication architecture, and this method was adopted in the array multiplier [2], consuming m times as much resource in parallel to extract m times of speed. In 1999, Paar studied further to get the benefit of both architecture, presenting the hybrid multiplication architecture [3]. However, the hybrid architecture has defect that only complex order of finite field should be used. In this paper, we propose a novel approach on developing serial multiplier architecture based on Mastrovito's, by modifying the numerical formula of the polynomial-basis serial multiplication. The proposed multiplier architecture was described and implemented in HDL so that the novel architecture was simulated and verified in the level of hardware as well as software. The implemented GF(2m) multiplier shows t times as fast as the traditional one, if we modularized the numerical expression by t number of parts.

키워드

암호, 유한체, 유한체 곱셈기, GF(2m), HDL

*목원대학교 정보전자영상공학부

접수일자 : 2004. 06. 01

Ⅰ. 서 론

기존에 연구된 유한체 곱셈 연산기에는 크게 직렬 유한체 곱셈기, 배열 유한체 곱셈기, 하이브리드 유한체 곱셈기가 존재하였다. 직렬 유한체 곱셈기는 마스트로비토에 의하여 제안되어 유한체 곱셈기의 가장 기본적인 구조로 자리 잡아 왔고 [1], 이를 병렬로 처리하기 위해 m 배의 자원을 투자하여 m 배의 속도를 얻어 낸 결과가 2차원 배열 유한체 곱셈기이다 [2]. 배열 유한체 곱셈기는 가격대 성능 비에 있어서 효율이 떨어지는 반면, 기존 두 방식의 장점만을 취하여 제안된 방식이 1999년 Paar에 의해 제안된 하이브리드 곱셈기이다 [3]. 이 하이브리드 유한체 곱셈기는 기존의 마스트로비토의 직렬 곱셈기 안에 기본적인 연산기로서 2차원 배열 곱셈기를 탑재한 방식으로, 하드웨어 자원도 직렬 곱셈기와 배열 곱셈기의 중간 이하가 되고 수행 속도도 직렬 곱셈기에 비해 상당히 빠르기 때문에 유한체 곱셈기의 연구에 역사상 중요한 위치를 차지한다. 하지만, 사용 가능한 유한체로서 유한체의 차수가 합성수인 합성수 유한체를 사용하기 때문에 암호학적인 안전도가 떨어져 많은 응용 분야에의 적용에 힘들다. 본 논문에서는 기존에 연구되었던 여러 가지 유한체 곱셈기들의 구조와 장단점을 살리면서 수식적인 측면으로 접근하여 공통인수 후처리 방식에 기반한 고속 유한체 곱셈방식이라는 새로운 유한체 곱셈 방식과 곱셈기 구조를 제안하여 기존 연구 결과와 비교, 분석해 본다.

Ⅱ. 유한체 곱셈기

Ⅱ-1. 직렬 (serial) 유한체 곱셈기

주어진 $GF(2^m)$ 상의 임의의 두 원소를 표준 기저로 표현한 다항식이 각각 다음과 같이 주어진다고 가정하자.

$$A(x) = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \ldots + a_1x + a_0$$
$$B(x) = b_{m-1}x^{m-1} + b_{m-2}x^{m-2} + \ldots + b_1x + b_0 \quad (1)$$

두 원소의 곱은 다음 식(2)와 같이 정리될 수 있다.

$$Z(x) = A(x)B(x)$$
$$= A(x)\sum_{i=0}^{m-1} b_ix^i = \sum_{i=0}^{m-1} b_i(x^iA(x)) \quad (2)$$

이는 마스트로비토의 x 곱셈 회로를 사용하여 최대 m-1차수 다항식으로 줄게 되며 다음 그림 1과 같이 구현된다. 이것이 마스트로비토의 직렬 유한체 곱셈기이다. 마스트로비토의 직렬 유한체 곱셈기는 적은 면적을 차지하면서 클럭 사이클을 주기로 쉬프트 레지스터에 그림과 같은 규칙으로 값을 주고받아 최종적으로 m 사이클 후에 원하는 유한체 곱셈 결과를 얻게 된다.

Ⅱ-2. 유한체 곱셈기의 성능 향상

가. 제안하는 곱셈기의 구조

제안하는 곱셈기의 구조는 마스트로비토의 직렬 곱셈기의 기본 구조를 바탕으로 한다. 유한체 원소 표현 방식도 표준 기저 방식을 사용하며 주어진 유한체로서 $GF(2^m)$을 사용한다. 직렬 유한체 곱셈기에서와 마찬가지로, 주어진 유한체의 임의의 두 원소를 각각 A(x), B(x) 라고 하면 곱셈 결과 값인 $Z(x)$는 다음 식과 같이 나타낼 수 있다.

$$Z(x) = A(x)\sum_{i=0}^{m-1} b_ix^i = \sum_{i=0}^{m-1} b_i(x^iA(x)) \bmod P(x)$$
$$= [b_0A(x) + \ldots + b_{m-2}x^{m-2}A(x) + b_{m-1}x^{m-1}A(x)] \bmod P(x) \quad (3)$$

제안되는 곱셈기의 성능 향상을 위해서 가장 핵심적인 부분은 기준이 되는 식을 분리하는 것이다. 결론적으로 말해서 식(3)을 t 등분으로 분리하여 각각 자원을 할당함으로써 t 배의 성능 향상을 유도하는 것이다. 곱셈기의 자세한 구조를 설명하기 위하여 $t=2$인 경우, 즉 성능을 두 배 향상시켰을 경우의 구조를 예를 들어 본다.

유한체 곱셈기의 성능을 두 배 향상시키기 위해, 식(3)을 지수가 짝수인 부분과 지수가 홀수인 부분으로 나누면 아래와 같이 두 개의 식이 생기게 된다.

$$Z_{even}(x) = [b_0A(x) + b_2x^2A(x) + \ldots$$
$$+ b_{m-3}x^{m-3}A(x) + b_{m-1}x^{m-1}A(x)] \bmod P(x) \quad (4)$$

$$Z_{odd}(x) = [b_1xA(x) + b_3x^3A(x) + \ldots$$
$$+ b_{m-4}x^{m-4}A(x) + b_{m-2}x^{m-2}A(x)] \bmod P(x) \quad (5)$$

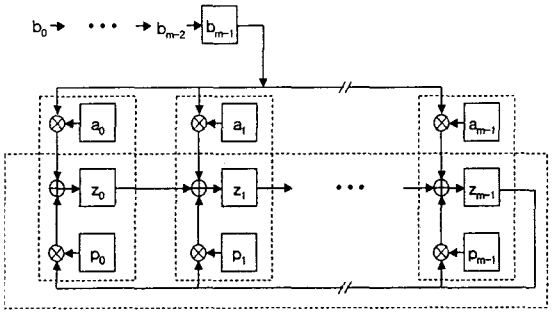

그림 1. 마스트로비토의 직렬 유한체 곱셈기
Fig. 1. Mastrovito's serial GF(2m) multiplier

식(4)에서 $Z_{even}(x)$의 식은 마스트로비토의 직렬 곱셈 알고리즘과 지수 부분을 제외한 나머지는 완전히 같다. 지수는 2 씩 증가하고 단지 지수가 짝수인 부분만 이 식에 포함되어 있어 항 (term)의 개수가 $\lceil \frac{m}{2} \rceil$ 개라는 것이 다를 뿐이다. 하지만, 식(5)는 약간 다르다. 이를 직렬 곱셈 알고리즘과 같은 형태로 만들기 위해 x를 밖으로 끌어내어 다음과 같이 변형하였다.

$$Z_{odd}(x) = x[\, b_1 A(x) + b_3 x^2 A(x) + \ldots + b_{m-4} x^{m-3} A(x) + b_{m-2} x^{m-1} A(x)] \bmod P(x) \quad (6)$$

식(6)의 대괄호 안의 부분은 계수 부분과 항의 개수가 $\lceil \frac{m-1}{2} \rceil$ 이라는 점을 제외하면 직렬 곱셈 알고리즘과 같게 된다.

여기서, 이제 직렬 곱셈 알고리즘과 다르게 2씩 증가하는 지수 부분을 처리하기 위해 $x^2A(x)$의 식을 유도해 본다.

$$x^2 A(x) = x^2(a_0 + a_1 x + \ldots + a_2 x^{m-2} + a_{m-1} x^{m-1}) = (a_0 x^2 + a_1 x^3 + \ldots + a_{m-2} x^m + a_{m-1} x^{m+1}) \quad (7)$$

여기서, 결과의 지수를 $m-1$ 이하로 줄이기 위하여 다음 식을 사용한다.

$$x^m = p_0 + p_1 x + \ldots + p_{m-2} x^{m-2} + p_{m-1} x^{m-1}$$
$$x^{m+1} = p_0 x + p_1 x^2 + \ldots + p_{m-2} x^{m-1} + p_{m-1} x^m \quad (8)$$

여기서 계산을 간편하게 하기 위하여 주어진 소수 다항식의 특성을 이용한다. SEG에서는 높은 암호학적인 복잡도의 특성을 나타내는 소수 다항식으로서 3항 다항식이나 5항 다항식을 권고한다 [4]. 이 3항 다항식이나 5항 다항식 공통으로 해당되는 특성은 계수 p_{m-1}이 0이라는 것이다. 따라서 이 성질을 이용하면 식(8)은 다음과 같이 간이화되어 이 식의 차수도 역시 $m-1$차 이하로 줄어든다.

$$x^{m+1} = p_0 x + p_1 x^2 + \ldots + p_{m-2} x^{m-1} \quad (9)$$

이제 식(8)(9)를 식(7)에 대입하여 정리하면 다음과 같은 식을 얻는다.

$$x^2 A(x) = a_{m-2} p_0 + (a_{m-2} p_1 + a_{m-1} p_0) x + (a_{m-2} p_2 + a_{m-1} p_1 + a_0) x^2 + \cdots + (a_{m-2} p_{m-1} + a_{m-1} p_{m-2} + a_{m-3}) x^{m-1} \quad (10)$$

위 식이 2배속 유한체 곱셈기의 구조의 핵심이 되는 부분이고 이를 그림 2에 나타내었다. 이 회로를 x^2 곱셈 회로라고 명명하기로 한다.

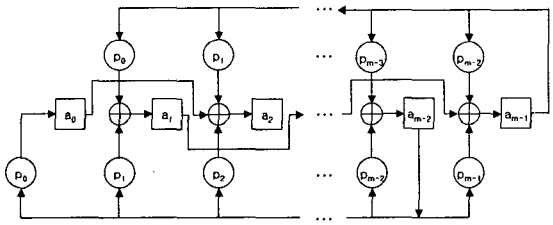

그림 2. x2 곱셈 회로의 구조
Fig. 2. Block diagram of the x2-multiplying circuit.

이제 식(4)의 $Z_{even}(x)$에 x^2 곱셈 회로를 적용해 보자. 여기서 x의 지수를 분석해 보면 직렬 유한체 곱셈기에서와 비슷하게 반복적으로 x^2 곱셈 회로가 필요하다는 것을 알 수가 있다. 따라서 항의 개수만큼, 즉 $\lceil \frac{m}{2} \rceil$ 번만큼 연산이 반복되면 $Z_{even}(x)$의 결과를 얻을 수 있다. 그림 3에 $Z_{even}(x)$의 값을 연산하는 회로의 블록도를 보인다.

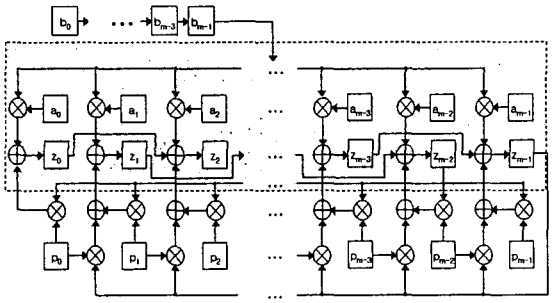

그림 3. Zeven(x)회로의 블록도
Fig. 3. Block diagram of $Z_{even}(x)$ circuit.

식(6)의 $Z_{odd}(x)$ 값을 얻기 위해서는 괄호 안의 연산은 $Z_{even}(x)$ 에서와 마찬가지로 $\lceil \frac{m-1}{2} \rceil$ 번만큼 연산을 반복해 주면 된다. 여기서 괄호 밖의 x를 처리해 주어야 하는데, 이것은 마스트로비토의 직렬 곱셈기에서 중요한 역할을 수행하였던 x 곱셈 회로를 적용하면 된다. 일반적으로 m이 홀수인 소수이기 때문에 $Z_{even}(x)$ 연산의 항 수가 $Z_{odd}(x)$ 연산의 항 수보다 하나가 많다. 이 마지막 사이클을 낭비하지 않기 위해서 $Z_{even}(x)$ 연산의 마지막 사이클에 해당하는 시간에 $Z_{odd}(x)$ 괄호 안의 연산 결과를 x 곱셈 회로를 사용하면서 한 사이클 실행해 주면 $Z_{even}(x)$의 결과 값과 $Z_{odd}(x)$의 결과 값과 같은 시간에 산출되기 때문에 이 두 값을 XOR 해 줌으로써 최종적으로 유한체 곱셈 결과를 $\lceil \frac{m}{2} \rceil$ 사이클만에 얻을 수 있다.

마지막 사이클의 x 곱셈 회로를 적용시키는 회로는 그림 4에서처럼 몇 개의 추가적인 mux를 사용함으로써 가능하다. x^2 제곱 회로와 x 제곱 회로 중 동작하는 하나를 선택하기 위해서는 $m-1$ 개의 2-to-1 mux가 필요하고 소수 다항식에 따라 ω_t-1개의 mux가 필요하다. 여기서 ω_t는 주어진 소수 다항식의 Hamming weight 를 뜻하는데, 이 정수는 3 혹은 5로 권유되며 소수 다항식의 차수 m에 비해 무시할 만큼 작은 정수이다.

그림 1에서 보이듯이 기존 직렬 유한체 곱셈기에서 가장 자원이 많이 할당되는 부분은 레지스터 쪽인데, 크게 보아 $A(x)$, $Z(x)$ 두 부분의 m 비트 레지스터 블록으로 구성된다고 볼 수 있다. 제안하는 구조는 전체 회로 크기의 거의 1/3에 해당하는 면적을 차지하는 $A(x)$ 레지스터를 공유할 수 있기 때문에 t 배의 성능을 얻기 위해 t 배의 자원을 할당하지 않는다는 장점이 있다.

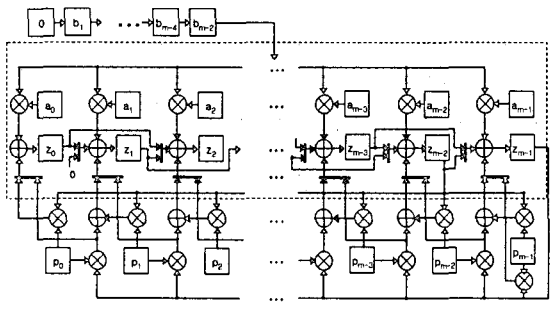

그림 4. Zodd(x) 회로의 블록도
Fig. 4. Block diagram of $Z_{odd}(x)$ circuit.

제안된 곱셈기의 구현에는 연산 블록별로 해당되는 제어 mux와 x^t 제곱 연산기가 필요하다. 이 구조의 또 하나의 장점은, 기존의 직렬 곱셈기와 거의 비슷한 최대 임계 경로 (critical path delay)를 가지므로 고속 연산에 적합하다.

제안하는 곱셈 방식은 하이브리드 유한체 곱셈기와 마찬가지로 느리면서 작은 구조도 아니고, 빠르면서 매우 큰 구조도 아니다. 따라서 기존의 면적 대 성능 비가 가장 우수한 하이브리드 유한체 곱셈기 구조와 비교하는 것이 타당할 것이다. 하드웨어 소모 면에서는 하이브리드 유한체 곱셈기에 비해 불리한 측면이 있지만, 제안된 곱셈기 구조는 암호학적인 복잡도가 높은 소수 유한체에서 사용이 가능하다는 장점이 있다. 아래에 t=2 일 경우 기존 구조와 제안된 구조를 비교하고 그 성능을 평가한다.

III. 제안된 곱셈기의 성능 평가

III-1. 성능 비교

비교는 각 곱셈기가 직렬 곱셈기 성능의 두 배의 성능을 내는 경우인 $m=2k$인 경우에 대하여 비교하였다. 자원이 얼마나 추가적으로 할당되는지 계산하기 위해서 그림 1의 점선 안의 내용에 초점을 맞추었다. 두 곱셈기 구조 모두 입력되는 $B(x)$의 계수는 2비트씩 동일하며, SEG 1에서 권유되는 소수 다항식의 Hamming weight인 3 혹은 5를 사용함으로써 160 비트 이상이 권유되는 소수 다항식의 차수에 비하여 무시할 만큼 작기 때문에 그림 1의 점선 밖의 하드웨어 부담은 동일하다고 가정하였다.

표 1. 유한체 곱셈기들의 성능 비교 (m=2k인 경우)
Table 1. Performance comparison of GF multipliers (m=2k).

GF(2^m)	AND	XOR	REG	MUX	CLK	Delay
직렬곱셈기	m	m	$2m$	0	m	fastest (i)
하이브리드	$2m$	$(3/2)m$	$2m$	0	$m/2$	slow (ii)
제안된곱셈기	$2m$	$3m$	$3m$	$m+\omega_t-2$	$m/2+1$	fast (iii)

(i) Register -> AND -> XOR
(ii) Register -> GF(22) 배열 곱셈기 (2 AND -> 2 XOR) -> XOR
(iii) Register -> AND -> MUX -> XOR

표 1은 비교 결과를 나타낸다. 제안된 유한체 곱셈기 구조에서 성능을 기존 직렬 곱셈기 구조의 두 배로 했을 경우, $Z_{even}(x)$ 회로에는 m 개의 AND 게이트 (gate), m 개의 XOR 게이트, m 개의 짝수 차수에 대한 결과 레지스터가 있고 이와 비슷하게 $Z_{odd}(x)$ 회로에는 m 개의 AND 게이트, m 개의 XOR 게이트, x 곱셈 회로와 x^2 곱셈 회로를 구현하기 위한 $m-1+\omega_t-1=m+\omega_t-2$ 개의 MUX, m 개의 홀수 차수에 대한 결과 레지스터가 존재한다. 이에 더하여, 공통되는 $A(x)$의 계수를 담기 위한 m 개의 레지스터와 최종 곱셈 결과를 위한 m 개의 XOR 회로가 필요하다. 제안된 곱셈기의 최대 지연 경로는 직렬 곱셈기의 최대 지연 경로와 거의 비슷하다. 하이브리드 곱셈기는 두 배의 성능을 보일 때 $2m$ 개의 AND, $(3/2)m$ 개의 XOR, $2m$ 개의 레지스터가 필요하므로 제안하는 곱셈기가 면적 면에서 볼 때 하이브리드 곱셈기에 비해 $(3/2)m$ 개의 XOR 게이트와 약 m 개의 MUX, m 개의 레지스터만큼의 크기를 더 차지한다고 볼 수 있다. 최대 임계 지연 경로를 살피면 제안된 곱셈기의 구조는 직렬 곱셈기의 짧은 임계 경로에 한 번의 MUX 지연 시간이 더해지는데 비해 하이브리드 곱셈기는 AND 게이트에 해당하는 부분이 배열 곱셈기로 대치되어야 한다. 이 때 추가되는 회로는 n^2 개의 AND와 n^2-1 개의 XOR 회로가 되는데, 두 배 성능 곱셈기 구현 시에는 AND, XOR의 지연시간이 추가되고 성능 배수인 n이 커질수록 임계 경로가 n의 제곱에 비례해서 커지는 단점을 보이고 있다 [3]. 따라서 전체적으로 보았을 때 제안하는 곱셈기의 구조가 약간의 추가적인 하드웨어의 부담은 있지만 하이브리드 곱셈기와 대등한 성능을 나타내면서 유한체 $GF(2^m)$의 지수에 관계없이 사용이 가능하다는 면에서 우수하다고 할 수 있다.

제안된 곱셈기의 구조는 이론상 t 배의 성능을 높일 수 있도록 확장이 가능하고, 또한, 타원 곡선 연산의 응용 중 유한체 제곱 연산에도 적용이 가능할 뿐만 아니라 이제까지 개발된 유한체 제곱 연산 회로에 비해 면적 대 성능 비가 유리하다.

III-2. 구현 결과 및 성능 평가

설계에 걸리는 시간을 단축하고 오류를 쉽게 수정할 수 있도록 RTL 수준에서 HDL [5]을 사용하여 하드웨어를 기술하고 합성을 수행하였다. 사용된 표준 셀(standard cell) 라이브러리는 삼성전자의 0.35um 공정의 std90이다. 합성 툴은 Synopsys Design Analyzer 에 통합되어 있는 Synopsys Design Compiler [6]를 사용하였으며 SDF (Standard Delay Format) 파일의 추출에는 CubicWare의 CubicDelay [7]를 사용하였다.

표 2는 최종 합성 결과로서 193 비트 2 배속 곱셈기에 대해 구현하여 수행한 결과이다. IC 카드용 하드웨어는 물론 고속 어플리케이션에 사용되기에 충분한 결과를 보였다. 또한 게이트 단위 면적을 보더라도 25 mm^2의 제한을 두는 IC 카드의 칩 면적 이내에서 활용되기에 충분한 결과를 보인다.

표 2. 193 비트 유한체 곱셈기의 합성 결과
Table 2. Synthesized result of 193-bit GF multiplier

	제안된 곱셈기
임계경로지연	0.7 ns
동작 주파수	1.42 GHz
단위게이트 수	4,563
사용공정	삼성전자 0.35um st90

IV. 결 론

본 논문에서는 Diffie-Hellman [8] 키 교환 방식을 바탕으로 하는 Elgamal [9] 타원 곡선 암호 시스템에서 적은 면적을 소비하면서도 빠른 암호화, 복호화를 수행할 수 있는 새로운 유한체 곱셈기 구조를 제안하고 제안된 알고리즘과 구조를 이용하여 HDL로 구현하여 그 성능을 검증하고 평가하였다. 제안하는 구조의 의도는 허용하는 만큼의 자원을 투자하여 성능을 극대화 하는데 초점을 맞추었다. 설계 결과는 적용 대상인 IC 카드와 같이 수행 연산 규모나 자원 규모가 제한되는 전자서명 [10]과 같은 어플리케이션에 충분히 적합하였다.

참고문헌

[1] E. D. Mastrovito, "VLSI Architectures for Computations in Galois Fields," Linkoping Studies in Science and Technology Dissertations, No. 242, 1991.

[2] C. Wang and J. Lin, "Systolic Array Implementation of Multipliers for Finite Fields GF(2m)," IEEE Transactions on Circuits and Systems, vol. 38, no. 7, pp. 796-800, July 1991.

[3] C. Paar, "Efficient VLSI Architectures for Bit-Parallel Computation in Galois Fields," Ph.D. thesis, Institute for Experimental Mathematics, University of Essen, Essen, Germany, June 1994.
[4] L. Song, "Low-Power VLSI Architectures for Finite Field Applications," Ph.D. thesis, UMI Microform 9935004, 1999.
[5] D. E. Thomas and Philip Moorby, The Verilog Hardware Description Language, Kluwer Academic Publishers, 1991.
[6] Design Compiler Reference Manual Fundamentals, Synopsys, Jan. 1997.
[7] Design Compiler Reference Manual Optimization and Timing Analysis, Synopsys, Jan. 1997.
[8] W. Diffie and M. Hellman, "New directions in cryptography," IEEE Transactions on Information Theory, pp. 644-654, Nov. 1976.
[9] T. Elgamal, "A Public key cryptosystem and a sugnature schme based on discrete logarithms," IEEE Transactions on Information Theory, vol. 31, pp. 469-472, 1985.
[10] National Institute of Standards and Technology, Digital Signature Standard, FIPS Publication 186-2, Feb. 2000.

저자소개

문상국(Sangook Moon)

1995 연세대학교 전자공학 학사
1997 연세대학교 전자공학 석사
2002 연세대학교 전자공학 박사
2002~2004 하이닉스반도체 선임연구원
2004~현재 목원대학교 전자정보보호공학부 전임강사
※관심분야 : 정보보호 VLSI 설계, Data encryption, 유비쿼터스 컴퓨팅 보안