효율적인 공간 복잡도의 LFSR 곱셈기 설계

(Design of an LFSR Multiplier with Low Area Complexity)

정 재 형* 이 성 운** 김 현 성*
(Jae-Hyung Jung Sung-Woon Lee Hyun-Sung Kim)

요 약 본 논문에서는 GF(2^m) 상에서 효율적인 공간 복잡도를 가진 LFSR(Linear Feedback Shift Register) 구조 기반의 모듈러 곱셈기를 제안한다. 먼저, 공개키 암호화 시스템의 기본 연산인 모듈러 지수승을 위한 지수승 알고리즘을 살펴보고 이를 위한 기본 구조를 제안한다. 특히, 본 논문은 이러한 지수기를 설계하기 위한 기본 구조로서 효율적인 모듈러 곱셈기를 제안한다. 제안된 구조는 기약다항식으로 모든 계수가 1인 속성의 AOP(All One Polynomial)를 이용하며 구조복잡도 면에서 기존의 구조들보다 훨씬 효율적이다.

핵심주제어 : 암호화 프로세서, 유한필드, AOP, 모듈러 곱셈기

Abstract This paper proposes a modular multiplier based on LFSR (Linear Feedback Shift Register) architecture with efficient area complexity over GF(2^m). At first, we examine the modular exponentiation algorithm and propose it's architecture, which is basic module for public-key cryptosystems. Furthermore, this paper proposes on efficient modular multiplier as a basic architecture for the modular exponentiation. The multiplier uses AOP (All One Polynomial) as an irreducible polynomial, which has the properties of all coefficients with '1' and has a more efficient hardware complexity compared to existing architectures.

Key Words : Crypto-processor, Finite fields, All one polynomial, Modular multiplier

I. 서 론

공개키 암호화 시스템을 구현하기 위한 다양한 모듈러 연산기가 제안 되었다[1-8]. 많은 연구에서 공간 및 시간 복잡도 향상을 위한 여러 가지 모듈러 연산기가 제안되었다[5-8]. 1997년 Fenn등은 GF(2^m)상에서 LFSR (Linear Feedback Shift Register) 구조를 이용한 두가지 형태의 모듈러 곱셈기를 설계하였다[7]. Fenn의 구조는 효율적인 구조 복잡도를 가진 기약 다항식 AOP (All One Polynomial)에 기반 한 모듈러 곱셈기이다. 2002년 Kim은 Fenn이 제안한 구조를 향상시키기 위한 다양한LFSR 구조를 설계하였다[8].

본 논문에서는 Fenn이 제안한 구조 복잡도를 줄이기 위한 효율적인 LFSR 곱셈기를 제안한다. 본 논문의 구조는 Fenn의 구조와 마찬가지로 기약 다항식 AOP의 속성을 이용한 LFSR 구조의 곱셈기이다.

제안한 구조는 Fenn의 구조와 비교하여 단지 추가적인 2개의 클럭사이클이 필요하지만 XOR와 AND 게이트의

* 경일대학교 컴퓨터공학부
** 경북대학교 컴퓨터공학과

개수를 1/2 정도 줄일 수 있었다. 본 논문에서 제안한 구조는 제한적인 하드웨어 구조가 요구되는 여러 가지 응용을 위한 기본 구조로 사용될 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 유한필드 에 대한 기본적인 정의와 기약다항식으로서의 AOP 속성에 대하여 설명하고 3장에서는 지수 알고리즘과 지수기를 살펴본다. 4장에서는 모듈러 곱셈 알고리즘을 유도하고 Fenn의 곱셈기와 본 논문에서 제안한 곱셈기를 살펴본다. 5장에서는 기존의 구조와 제안된 구조의 여러 가지 특성들을 비교하고 분석한다. 끝으로 6장에서 결론을 맺는다.

II. 유한필드

유한필드는 Galois 필드(GF)로도 불린다. 비록 유한필드가 많은 소수의 지수 차수에 대해서 존재하지만, 암호학에서 주로 사용되는 필드는 소수 q에 대한 소수 유한필드 GF(q)와 양수 m에 대한 이진 유한필드 GF(2^m)이다. 유한필드 GF(2^m)은 길이가 m인 2^m개의 가능한 비트 스트링으로 구성된다[3].

$$GF(2^m)=\{(a_{m-1}\ a_{m-2}\ \cdots\ a_1\ a_0)|a_i\in GF(2),\ 0\le i\le m-1\}$$

여기서 GF(2)는 GF(2^m)의 하부필드(Sub-field)라 불리고, 유한필드 GF(2^m)은 GF(2)의 확장필드라고 한다. 필드에서의 원소들을 표현하기 위해서는 정규기저 (Normal Basis) 표기법, 이원기저(Dual Basis) 표기법, 다항식기저(Polynomial Basis) 표기법 등의 세 가지 표기법이 있다. 그러나 정규기저 표기법과 이원기저 표기법을 이용한 연산에서는 연산전후에 기저변환 단계를 거쳐야 하는 문제가 있다. 본 논문에서는 다항식기저 표기법으로 필드상의 원소들을 표현한다. 다항식기저 표기법에서의 GF(2^m)의 각 원소는 다음과 같이 m차수 미만의 다항식으로 표현된다.

$$a(x)=a_{m-1}x^{m-1}+a_{m-2}x^{m-2}+\cdots+a_1x+a_0,$$

$$a_i\in GF(2),\ 0\le i\le m-1.$$

GF(2^m)상에서 연산 후 연산 결과를 필드의 원소로 만들기 위해서는 차수 m의 기약 다항식(Irreducible Polynomial)이 필요하고, 이 기약 다항식을 이용한 모듈러 연산이 필요하다.

GF(2)의 원소를 계수로 갖는 m차의 기약 다항식을 $f(x)$ 할 때, 다항식의 계수가 모두 '1' 인 다항식 $f(x)=x^m+x^{m-1}+x^{m-2}+\cdots+x+1$을 AOP(All One Polynomial) 라 한다. 이 방정식의 근을 α라고 두면, AOP는 $\alpha^{m+1}+1=0$, (m+1은 소수) 의 속성을 가진다[6]. 100보다 작은 m에 대해서 m이 2, 4, 10, 12, 18, 28, 36, 52, 58, 60, 66, 82 일 때 기약 다항식으로서의 AOP를 만족한다.

즉, 본 논문에서는 AOP의 속성을 이용하여 GF(2^m)보다 하나 확장된 GF(2^{m+1})상에서 곱셈 연산이 수행된다 GF(2^{m+1})상의 한 원소 A는 $A=A_m\alpha^m+A_{m-1}\alpha^{m-1}+A_{m-2}\alpha^{m-2}+\cdots+A_1\alpha+A_0$, ($A_m$=0)으로 표현된다. 여기서, $A_i=a_i+A_m$, $0\le i\le m-1$ 이다. 또한, 기저 { 1, α, α^2,…, α^{m-1}, α^m } 은 GF(2^m)상의 표준기저에서 한 차원 확장된 기저이다. 이러한 속성은 곱셈연산을 수행하는데 있어서 효율적인 모듈러 감소를 제공할 수 있다.

III. 모듈러 지수 연산

본 장에서는 지수 알고리즘을 살펴보고 지수 연산기를 제안한다.

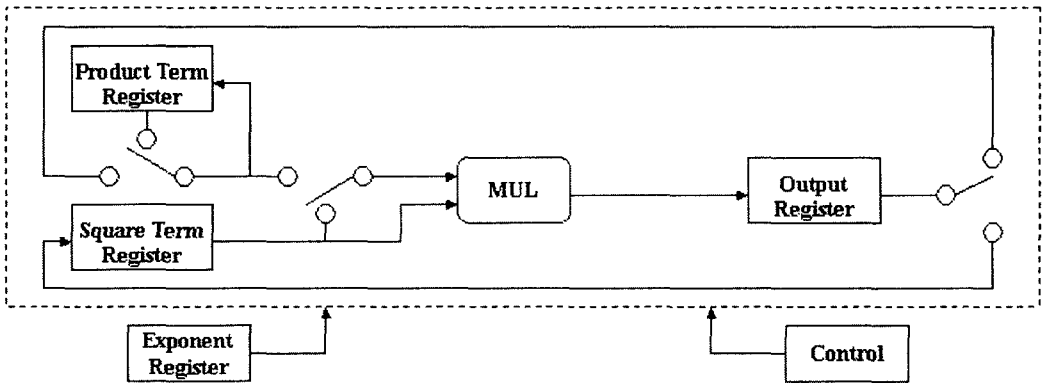

그림 1: GF(2^m)상에서 제안된 지수 연산기

3.1 지수 알고리즘

유한필드 상에서 Diffie-Hellman 키 교환 방식, 디지털 서명 알고리즘과 ElGamal 암호화 방식과 같이 잘 알려진 알고리즘을 응용한 타원 곡선 (Elliptic Curve) 기반의 공개키 암호화 시스템의 구현에 있어서 GF(p)나 GF(2^m) 상에서 지수 연산이 필요하다. 효율적인 지수 연산을 위해서 지수의 처리 방식에 따라 LSB (Least Significant Bit)와 MSB (Most Significant Bit) 우선 방식의 알고리즘이 있다[8]. LSB 알고리즘은 다음과 같다.

[알고리즘1] LSB 우선 지수 알고리즘

입력 : A , E , $f(x)$
출력 : $C = A^E \bmod f(x)$
단계1 : $T = A$
단계2 : if (e_0 == 1) $C = T$ else $C = \alpha^0$
단계3 : for i = 1 to m - 1
단계4 : $T = TT \bmod f(x)$
단계5 : if (e_i == 1) $C = CT \bmod f(x)$

알고리즘의 단계4와 단계5에서 제곱 연산과 곱셈 연산이 각각 필요하다. 이 연산은 제곱기와 곱셈기를 이용하거나, 하나의 곱셈기를 이용하여 한번은 곱셈을 또 다른 한번은 제곱을 연산하는 연산기를 통하여 효율적인 연산을 수행할 수 있다.

3.2 제안된 지수 연산기

본 논문에서 제안한 지수 연산기는 그림 1과 같다. 그림 1은 앞서 설명한 LSB 알고리즘을 수행하기 위하여 하나의 곱셈기가 제곱과 곱셈을 동시에 수행할 수 있는 효율적인 구조이다. 다음 장에서는 그림 1의 MUL(Multiplier) 구조를 위한 효율적인 모듈러 곱셈 알고리즘과 이에 따른 구조를 제안한다.

Ⅳ. 모듈러 곱셈

본 장에서는 연산을 위한 AOP 알고리즘을 유도하고, 이 알고리즘에 기반 한 각 구조를 살펴본다. 먼저 Fenn등의 LFSR 구조에 기반 한 모듈러 곱셈기와 본 논문에서 제안한 곱셈기를 살펴본다.

4.1 모듈러 곱셈 알고리즘

Kim은 AOP를 기약다항식으로 이용한 모듈러 곱셈기를 구성하기 위한 알고리즘을 제안하였다[8]. 본 절에서는 AOP 알고리즘의 유도과정을 살펴보고 새로운 알고리즘을 제안한다.

A =				A_4	A_3	A_2	A_1	A_0
× B =				B_4	B_3	B_2	B_1	B_0
				A_4B_0	A_3B_0	A_2B_0	A_1B_0	A_0B_0
			A_4B_1	A_3B_1	A_2B_1	A_1B_1	A_0B_1	
		A_4B_2	A_3B_2	A_2B_2	A_1B_2	A_0B_2		
	A_4B_3	A_3B_3	A_2B_3	A_1B_3	A_0B_3			
A_4B_4	A_3B_4	A_2B_4	A_1B_4	A_0B_4				
P_8	P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0

그림 2: 모듈러 곱셈

α^4	α^3	α^2	α^1	α^0
A_4B_0	A_3B_0	A_2B_0	A_1B_0	A_0B_0
A_3B_1	A_2B_1	A_1B_1	A_0B_1	A_4B_1
A_2B_2	A_1B_2	A_0B_2	A_4B_2	A_3B_2
A_1B_3	A_0B_3	A_4B_3	A_3B_3	A_2B_3
A_0B_4	A_4B_4	A_3B_4	A_2B_4	A_1B_4
P_4	P_3	P_2	P_1	P_0

그림 3: 모듈러 곱셈 알고리즘

α^3	α^2	α^1	α^0
P_4	P_4	P_4	P_4
A_3B_0	A_2B_0	A_1B_0	A_0B_0
A_2B_1	A_1B_1	A_0B_1	A_4B_1
A_1B_2	A_0B_2	A_4B_2	A_3B_2
A_0B_3	A_4B_3	A_3B_3	A_2B_3
A_4B_4	A_3B_4	A_2B_4	A_1B_4
p_3	p_2	p_1	p_0

그림 4: 모듈러 곱셈 알고리즘

AB곱셈기 설계를 위한 GF(2^4)상에서의 곱셈 알고리즘

은 그림 2와 같다.

곱셈 후 연산의 결과에 모듈러 연산을 적용하면 그림 3과 같다.

즉, 곱셈 후 그림 2의 왼쪽에 강조된 부분의 값에 따라서모듈러 감소 연산이 적용되어야 하고, α^{m+1}이 기약다항식으로서 사용되면 모듈러 감소는 그림 3에서 보여주는 바와 같이 계산된다. 여기서, 모듈러 감소 연산은 그림 2에서 왼쪽 강조된 부분이 그림 3에서 보여준 것처럼 오른쪽으로 치환 됨으로 계산된다.

그러나 그림 3의 결과값 역시 확장된 기저상의 연산이므로, 한번의 추가적인 모듈러 감소 연산이 필요하며 연산은 그림 4와 같은 새로운 모듈러 곱셈 알고리즘을 유도할 수 있다.

AOP를 이용한 모듈러 곱셈은 일반필드에서 한차원 확장된 필드에서 연산이 필요하고 연산 후 추가적인 한번의 모듈러 감소 연산이 필요하다. 그림 4는 그러한 속성을 고려한 모듈러 곱셈 알고리즘이며, 그림 4의 음영부분이 추가적인 모듈러 감소 연산을 나타낸다.

4.2 Fenn의 곱셈기

Fenn등은 모듈러 곱셈을 위해서 그림 3의 곱셈 알고리즘을 이용하고 LFSR 구조에 기반 한 두 가지 비트순차 모듈러 곱셈기(AOPM과 MAOPM)를 제안하였다[7].

AOPM은 일반 필드에서 한 차원 확장된 필드의 입력과 출력을 갖는다. 그래서 곱셈 연산을 수행한 후 추가적인 모듈러 감소 연산이 필요하다. 그러나 MAOPM은 일반 필드상의 입력과 출력을 갖는 구조이다. 본 논문에서는 MAOPM에 초점을 맞춰서 살펴본다. 그림5의 MAOPM은 GF(2^4) 상의 구조를 보여준다. 이 구조에서는 추가적인 모듈러 감소 연산을 위해서 그림 5의 오른쪽 부분에서 보여주는 것과 같이 3개의 2-AND(2-input AND)와 1개의 3-XOR(3-input XOR) 게이트를 필요로 한다.

AOPM과 MAOPM은 모듈러 곱셈을 수행하기 위하여 전체 $2m$+1과 $2m$-1개의 클럭사이클이 각각 필요하다.

4.3 새로운 곱셈기

제안한 구조는 그림5의 MAOPM의 구조 복잡도를 줄이는데 목적이 있다. Fenn등은 AOP의 속성을 이용한 곱셈기를 제안하기 위하여 제시된 필드보다 한차원 확장된 구조상에서 모듈러 곱셈을 수행하는 AOPM 구조를 제안하였다. 그러나 그 구조는 한차원 확장된 필드의 곱셈 결과 값을 반환하므로 연산 후 추가적인 모듈러 감소 연산이 필요하다.

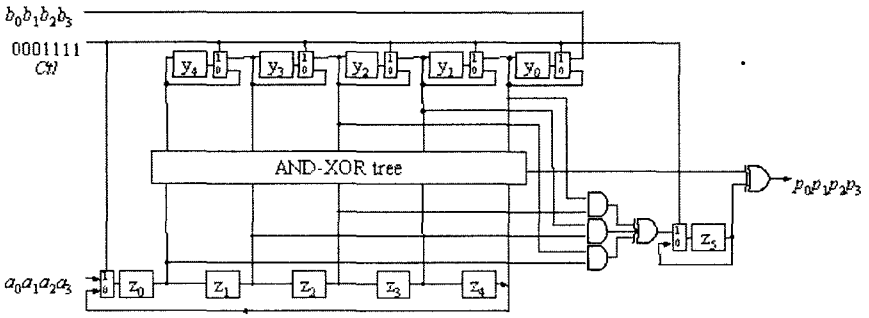

그림 5: Fenn의 MAOPM

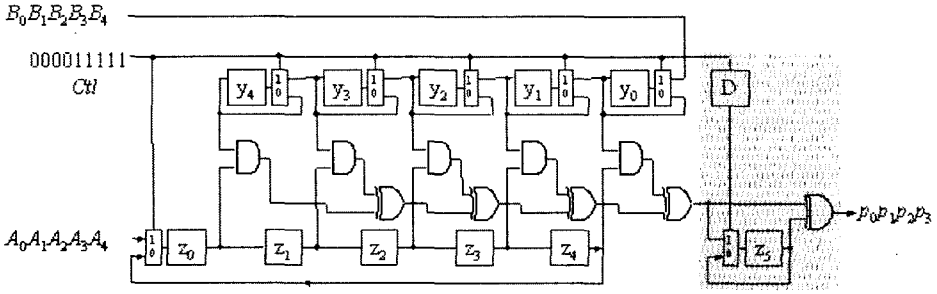

그림 6: 제안한 곱셈기

Fenn등의 연구에서는 AOPM에서 MAOPM으로 변환하는 과정에서 공간 복잡도를 추가함으로서 시간 복잡도를 줄이는데 연구의 초점을 맞췄다. 그러나 본 연구에서는 반대로 시간복잡도의 추가로 효율적인 공간 복잡도를 갖는 모듈러 곱셈기를 제안한다. 본 논문에서 제안한 곱셈기는 기존의 구조에 비해서 2 클럭사이클이 추가로 필요하다.

그림6은 본 논문에서 제안한 모듈러 곱셈기를 보여준다. 그림6의 오른쪽 음영 부분이 그림4의 알고리즘에서의 추가적인 모듈러 감소를 수행하기 위한 부분이다. 즉, 그림6의 구조는 입력의 마지막 스텝에 모든 레지스터 값이 초기화되고, 그 시점에 모듈러 감소를 위한 결과값 (P_4)이 계산되고 Z_5에 저장된다. 이 값은 그 이후의 클럭 사이클의 결과값과 XOR 연산을 통하여 추가적인 모듈러 감소 연산을 수행할 수 있다. 곱셈기의 수행에 있어서 입력 상태와 처리 상태의 구별을 위해 하나의 제어 신호가 필요하다. 제어 신호는 입력을 위한 m+1비트의 '1'과 연산을 위한 m비트의 '0'을 필요로 한다. 즉, 본 논문에서 제안한 구조는 모듈러 곱셈 연산 수행을 위해서 AOPM과 같은 전체 $2m$+1개의 클럭사이클을 필요로 한다.

V. 분석 및 시뮬레이션

본 장에서는 본 논문에서 제안한 곱셈기 구조와 기존의 비트순차 LFSR 구조들과 여러 가지 특성을 비교하고 분석하고 시뮬레이션 결과를 제시한다.

5.1 분석

본 절에서는 4.2절에서 상세히 살펴본 Fenn의 모듈러 곱셈기와 논문 [8]에서 Kim이 제안한 모듈러 곱셈기를 기반으로 본 논문에서 제안한 구조와 비교 분석한다. 표 1은 여러 가지 모듈러 곱셈기를 여러 가지 특성에 기반 한 비교를 보여준다.

Fenn의 구조에 대해서는 4.2절에서 살펴보았으므로 여기서는 생략한다. Kim의 구조는 Inner-product에 기반 한 모듈러 곱셈기이다. Kim의 구조는 Fenn의 구조에 비교하여 상당히 구조 복잡도를 줄일 수 있었지만, 이 구조 역시 AOPM과 마찬가지로 추가적인 모듈러 감소 연산이 필요하다는 문제를 가지고 있다.

그러므로, 표 1에서 보여준 바와 같이 본 논문에서 제안한 구조가 기존의 구조에 비교하여 보다 효율적인 하드웨어 복잡도를 가짐을 알 수 있다.

5.2 시뮬레이션

제안한 구조의 논리적인 검증을 위하여 먼저 C언어로 알고리즘의 검증을 수행하였고, Altera사의 MAX+PLUSII를 이용하여 구조 시뮬레이션 하였다. 그림 7은 본 논문에서 제안한 구조의 시뮬레이션 결과를 보여준다.

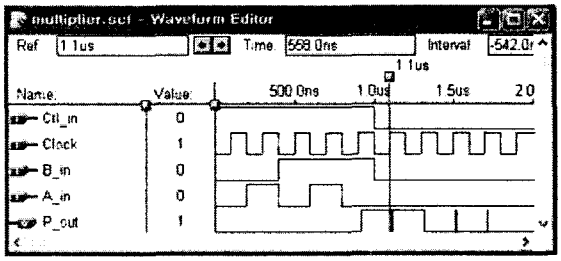

그림 7: 시뮬레이션 결과

표 1. 모듈러 곱셈기 비교

구조 / 항목	Fenn의 MAOPM	Kim	제안한 구조
기약다항식	AOP	AOP	AOP
연산 후 결과	GF(2^m)	GF(2^{m+1})	GF(2^m)
추가적인 모듈러 연산	필요없음	필요함	필요없음
레지스터	2(m+1)	2(m+1)	2m+4
AND	2m-1	m+1	m+1
XOR	2m-2	m	m+1
MUX	m+2	2	m+3
Latency	2m-1	2m+1	2m+1

VI. 결론

본 논문에서는 새로운 모듈러 곱셈 알고리즘을 제안하였고 LFSR 구조에 기반 한 모듈러 곱셈기를 설계하였다. 제안한 모듈러 곱셈기는 기약 다항식으로서 AOP를 이용한 효율적인 하드웨어 복잡도를 가진 구조이다. 표 1에서 보여 준 것처럼 제안한 곱셈기는 기존의 곱셈기와 비교해 보다 효율적인 구조 복잡도를 가짐을 확인할 수 있다. 제안한 곱셈기는 공개키 기반의 암호화 프로세서 설계에 효율적으로 이용할 수 있을 것으로 기대된다.

참 고 문 헌

[1] W.Diffie and M.E.Hellman, "New directions in cryptography," *IEEE Trans. on Info. Theory*, Vol. 22, pp. 644-654, Nov. 1976

[2] T.ElGamal. "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. on Info. Theory*, Vol. 31(4). pp. 469-472, July 1985

[3] R. Lidl, H.Niederreiter, and P.M.Cohn, *Finite Fields (Encyclopedia of Mathematics and Its Applications)*, Cambridge University Press, 1997.

[4] I.S.Reed and T.K.Truong, "The use of finite fields to compute convolutions," *IEEE Trans. Inform Theory*, Vol. IT-21, pp. 208-213, Mar. 1975

[5] 김현성, 유기영, "유한필드상에서의 곱셈기 설계 방법," 한국정보과학회 정보보호 연구회지, Vol. 1, No. 1, pp. 12-19, 2001년 4월

[6] 김현성, 유기영, "유한필드상에서의 AOP 곱셈기 설계," 한국정보과학회 정보보호 연구회지, Vol. 3, No. 1, pp. 12-20, 2003년 4월

[7] S.T.J.Fenn, M.G.Parker, M.Benaissa, and D. Taylor, "Bit-serial multiplication in GF(2^m) using irreducible all-one polynomials," *IEE Proc.-Comput. Digit. Tech.*, Vol. 144, No. 6, pp. 391-393, Nov. 1997

[8] H.S.Kim, *Bit-Serial AOP Arithmetic Architecture for Modular Exponentiation*, Ph.D. Thesis, Kyungpook Nat'l Univ., 2002.

정 재 형 (Jae-Hyung Jung)

1999년 3월~현재 경일대학교 컴퓨터공학과

관심분야 : 정보보안, 암호 프로세서 설계, IDS

이 성 운 (Sung-Woon Lee)

1994년 2월 전남대학교 전산학과 졸업

1996년 전남대학교 전산학과 석사과정 졸업

2001년 3월~현재 경북대학교 박사과정

2002년 9월~현재 경일대학교 교양학부 교수

관심분야 : 정보보안, 인증 프로토콜

김 현 성 (Hyun-Sung Kim)

1996년 2월 경일대학교 컴퓨터공학과 공학사

1998년 2월 경북대학교 컴퓨터공학과 공학석사

2002년 2월 경북대학교 컴퓨터공학과 공학박사

2002년 3월~현재 경일대학교 컴퓨터공학과 교수

관심분야 : 정보보안, 암호 알고리즘, 암호 프로세서 설계,IDS, PKI