論文2003-40SD-12-11

ECC 연산을 위한 가변 연산 구조를 갖는 정규기저 곱셈기와 역원기

(Scalable multiplier and inversion unit on normal basis for ECC operation)

李 燦 豪 *, 李 定 鎬 *

(Chanho Lee and Jeongho Lee)

요 약

타원곡선 암호(Elliptic Curve Crypto- graphy : ECC)는 기존의 어떤 공개키 암호 시스템보다 우수한 비트 당 안전도를 제공하고 있어 최근 큰 관심을 끌고 있다. 타원곡선 암호 시스템은 보다 작은 키 길이를 갖고 있어 시스템의 구현에 있어서 작은 메모리 공간과 적은 처리 전력을 필요로 하므로 다른 암호화 방식에 비해 임베디드 어플리케이션에 적용하는데 유리하다. 본 논문에서는 제곱 연산이 용이한 정규기저로 표현된 유한체에서의 곱셈기를 구현하였다. 이 곱셈기는 타원곡선 암호에서 사용되는 GF(2^{193}) 상에서 구현하였고, Massey와 Omura가 제시한 병렬 입력-직렬 출력 곱셈기의 구조를 변형하여 출력의 크기와 설계면적을 조절할 수 있다. 또한 제안한 곱셈기를 적용하여 정규기저 역원기를 구현하였다. 곱셈기와 역원기는 HDL을 이용하여 설계하고, 0.35 ㎛ CMOS 셀 라이브러리를 이용하여 구현하였으며 시뮬레이션을 통해 동작과 성능을 검증하였다.

Abstract

Elliptic curve cryptosystem(ECC) offers the highest security per bit among the known publick key system. The benefit of smaller key size makes ECC particularly attractive for embedded applications since its implementation requires less memory and processing power. In this paper, we propose a new multiplier structure with configurable output sizes and operation cycles. The number of output bits can be freely chosen in the new architecture with the performance-area trade-off depending on the application. Using the architecture, a 193-bit normal basis multiplier and inversion unit are designed in GF(2^m). It is implemented using HDL and 0.35㎛ CMOS technology and the operation is verified by simulation.

Keyword : ECC, multiplier, inversion unit, finite field operation, normal basis

Ⅰ. 서 론

최근 '정보'를 보다 중요하게 인식하게 되면서 보안에 대한 관심이 높아지고 있다. 시간이 지남에 따라 보다 높은 안전도를 필요로 하게 되고 원하는 안전도를 만족시키기 위해 암호키 길이도 길어지게 되었다. 하지만 길어진 키 길이는 연산 시간과 연산기 구조의 복잡도 및 면적의 증가를 초래하게 되었고, 보다 짧은 키 길이로 원하는 안전도를 얻을 수 있는 암호 알고리즘에 관심을 갖게 되었다.

타원곡선 암호는 기존의 어떤 공개키 암호 시스템보다 우수한 비트 당 안전도를 제공한다[1]. 예를 들어

* 正會員, 崇實大學校 情報通信電子工學部
(Soongsil University, School of Electronic Engineering)

※ 본 연구는 숭실대학교 교내연구비 지원으로 이루어졌습니다.

接受日字:2003年1月27日, 수정완료일:2003年11月28日

1024 비트 키 길이를 갖는 RSA와 160비트 키 길이를 갖는 타원곡선 암호 시스템은 유사한 안전도를 갖는다[2]. 보다 작은 키 길이는 시스템의 구현에 있어서 보다 작은 메모리 공간과 보다 적은 처리 전력을 필요로 하므로 타원곡선 암호 시스템을 임베디드 어플리케이션에 적용하는데 유리하게 한다[1]. 이러한 타원곡선 암호에서의 연산은 그 바탕을 유한체에 두고 있다.

여러 타원곡선 암호와 관련된 표준들에서 안전도를 고려한 160 비트 이상의 유한체에서 정의된 권장 타원곡선들을 제시하고 있다. 160 비트 정도의 키 길이를 갖는 타원곡선 암호는 향후 약 10년 정도는 안전하다고 볼 수 있다[3]. 193비트의 키 길이를 갖는 타원곡선 암호는 향후 약 20년 정도는 안전할 것으로 예상된다.

타원곡선 암호 시스템에서 사용되는 유한체 연산 중에서 가장 빈번하게 사용되는 연산은 덧셈과 곱셈인데, 덧셈에 비해서 곱셈이 보다 복잡하고 연산 시간이 오래 걸린다. 이진 확장체에서 유한체의 덧셈과 유한체의 뺄셈은 원소의 표현법에 상관없이 같고 비트별 XOR 연산으로 수행된다. 유한체의 곱셈은 원소의 표현법에 따라 계산 과정이 다르고 곱셈기의 복잡도 또한 서로 다르다.

원소의 표현법에는 다항식 기저 표현법(polynomial basis representation)과 정규 기저 표현법(normal basis representaion) 두 가지가 있다. 다항식 기저 표현은 곱셈기의 확장성이 용이하고 소프트웨어적인 구현에는 적합하지만 하드웨어적인 구현에는 부적합하다. 또한 역원 연산이 복잡하다. 정규 기저 표현은 정규 기저가 갖는 유한체 연산에서의 장점 때문에 하드웨어적인 구현에 적합하다[4]. 정규 기저에서 제곱 연산이 오른쪽으로 한 비트 회전 이동 시킨 것과 같아 제곱연산이 쉬운 장점이 있고 역원계산 또한 이러한 점을 이용하여 쉽게 구현할 수 있다. 대신 곱셈기에 있어서는 확장성이 떨어지는 단점이 있다. 163비트 곱셈기 구현을 위해 계산한 이진 함수는 155비트나 193비트에서는 전혀 사용할 수 없고 새로이 계산해야 한다.

정규기저 방식의 대표적인 곱셈기구조에는 Massey와 Omura가 제안한 방식[5]과 G.L. Feng이 제안한 방식[6], 그리고 C.C. Wang의 병렬 Massey-Omura 곱셈기[7]가 있다. Massey-Omura 곱셈기는 병렬 입력-직렬 출력 구조를 갖고, Feng 곱셈기는 직렬 입력-병렬 출력 구조를 갖는다. Wang의 곱셈기는 병렬 입출력 구조로 되어 있다.

$GF(2^m)$에 대해서 m 값이 큰 경우 Massey-Omura 곱셈기와 Feng 곱셈기의 경우 최종 결과가 나올 때까지 m 클럭이 소요되므로 연산시간 면에서, Wang의 병렬 곱셈기는 이진 함수가 m 개 사용되므로 면적 면에서 ECC 연산 프로세서에 사용하기에는 부적합하다.

본 논문에서는 Massey와 Omura가 제시한 병렬 입력-직렬 출력 구조의 곱셈기를 변형하여 출력 비트를 조절하여 전체 연산 시간과 설계 면적을 조절할 수 있는 설계 방식을 제안한다. 제한된 연산 시간과 설계 면적에 있어 적절한 수준을 만족하는 곱셈기를 설계하고자 하는 경우 본 논문에서 제안한 방식이 유용하게 사용될 수 있을 것이다. 곱셈기는 $GF(2^{193})$ 상에서 구현되었고 두 개의 193 비트 길이를 갖는 입력 값이 쉬프트 레지스터에 들어가 매 클럭 8비트 씩 연산하여 출력하고 25 클럭만에 최종 계산 값을 얻을 수 있다. 또한 제시한 곱셈기를 이용하여 역원기를 구현하였다.

II. Massey-Omura 곱셈기

$GF(2^m)$의 정규기저는 다음과 같은 형태의 일차 독립인 m 개의 $GF(2^m)$의 원소들의 집합으로 구성된다.

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

정규 기저에서 $GF(2^m)$의 임의의 원소는 내부적인 연산을 위해

$$A = a_0\alpha^{2^0} + a_1\alpha^{2^1} + \cdots + a_{m-1}\alpha^{2^{m-1}} \quad (2)$$

과 같이 유일하게 표현되고, 이를 m 개의 비트열

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

로 나타낸다.

정규기저에서 원소의 제곱은 해당 비트열을 한 비트 씩 오른쪽으로 회전이동 시킨 것과 같다.

$$A^2 = (a_{m-1} a_0 \cdots a_{m-2}) \quad (4)$$

$GF(2^m)$의 임의의 두 원소를 정규 기저로 표현한 것을 A, B라 하고, 두 원소 A, B의 곱셈 결과를 C라고 하면 A, B, C는 다음과 같이 쓸 수 있다.

$$A = (a_0 a_1 \cdots a_{m-1})$$

$$B = (b_0 b_1 \cdots b_{m-1})$$
$$C = (c_0 c_1 \cdots c_{m-1}) \quad (5)$$

결과 C는 A와 B의 계수들의 이진 함수들로 이루어지므로 C의 계수는 식 (6)처럼 표현된다.

$$c_{m-1} = f(a_0, a_1, \cdots, a_{m-1}; b_0, b_1, \cdots, b_{m-1}) \quad (6)$$

정규 기저에서 제곱은 계수들의 오른쪽 순환 치환과 같으므로

$$C^2 = A^2 \cdot B^2 (c_{m-1} c_0 \cdots c_{m-2})$$
$$= (a_{m-1} a_0 \cdots a_{m-2}) \cdot (b_{m-1} b_0 \cdots b_{m-2}) \quad (7)$$

C^2에서 c_{m-2}는 C에서 c_{m-1}을 구하는데 사용된 이진함수를 통해 계산되는 것을 볼 수 있다. 따라서 C의 모든 계수는 다음과 같이 동일한 이진 함수를 사용하여 얻을 수 있다.

$$\begin{aligned} c_{m-1} &= f(a_0, a_1, \cdots, a_{m-1}; b_0, b_1, \cdots, b_{m-1}) \\ c_{m-2} &= f(a_{m-1}, a_0, \cdots, a_{m-2}; b_{m-1}, b_0, \cdots, b_{m-2}) \\ &\vdots \qquad\qquad \vdots \\ c_0 &= f(a_1, a_2, \cdots, a_0; b_1, b_2, \cdots, b_0) \end{aligned} \quad (8)$$

식 (8)에서 나타난 바와 같이 Massey-Omura 곱셈기는 오른쪽으로 한 비트 쉬프트 되는 m-bit 순환 치환 레지스터들과 이진함수로 이루어진다. 이러한 Massey-Omura 곱셈기는 m 클럭이 지나야 계산이 끝나게 되므로 m 값이 큰 경우 연산 시간이 매우 길어지는 단점이 있다.

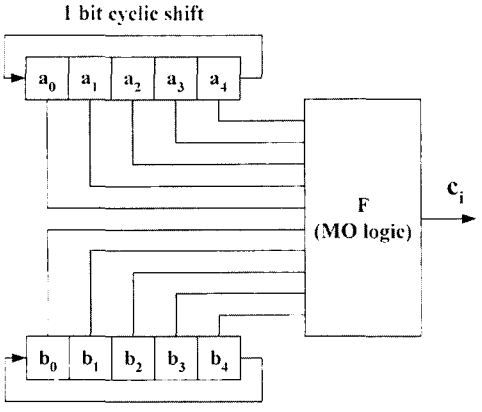


그림 1. Massey-Omura 곱셈기
Fig. 1. Massey-Omura multiplier.

Massey-Omura의 변형된 구조로 속도를 개선시키기 위해 이진 함수(f)를 m 개 사용함으로써 병렬로 곱셈을 처리할 수 있는 곱셈기가 C.C. Wang에 의해 사용되었지만 이것은 m 값이 큰 경우 m 개의 이진함수 때문에 설계 면적이 상당히 커지는 단점이 있다. m=5일 때 Massey-Omura 곱셈기와 병렬 Massey-Omura 곱셈기의 구조를 살펴보면 각각 <그림 1, 2>와 같다[5, 8]. 병렬 Massey-Omura 곱셈기는 f 함수의 면적이 5배가 됨을 알 수 있다.

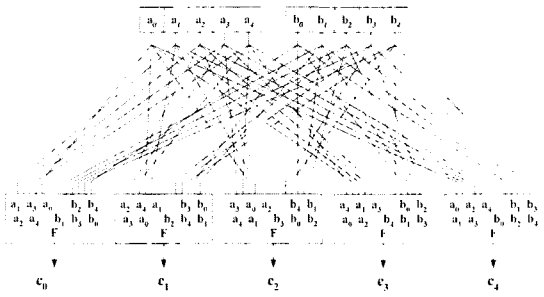


그림 2. 병렬 Massey-Omura 곱셈기
Fig. 2. Parallel Massey-Omura multiplier.

Ⅲ. 가변 연산 구조의 정규기저 곱셈기

1. 가변 연산 구조

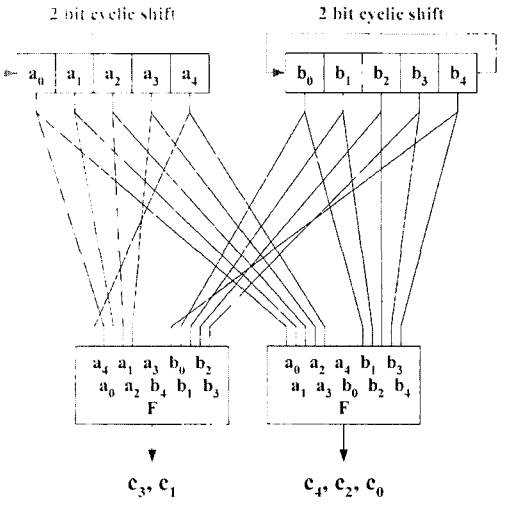
큰 m 값에 대해서 Massey-Omura 곱셈기에서의 길어지는 계산시간과 병렬 곱셈기에서의 커지는 면적을 서로 절충하면 필요에 따라 면적과 성능을 조절하고 구현 효율을 증가시킬 수 있는 방법을 제안한다.

식 (8)을 살펴보면 c_{m-1}부터 c_0까지 차례로 입력 A, B의 계수들이 오른쪽으로 순환 치환되어 이진함수의 입력으로 연결되는 것을 볼 수 있다. 식 (8)을 n (1≤n ≤m)에 대해 n개의 이진함수(f_n)를 사용하여 다시 정리하면 다음과 같다.

$$\begin{aligned} &(c_{m-1}, \cdots, c_{m-n}) = \{f(a_0, \cdots, a_{m-1}; b_0, \cdots, b_{m-1}), \cdots \\ &\quad , f(a_{m-n+1}, \cdots, a_{m-n}; b_{m-n+1}, \cdots, b_{m-n})\} \\ &(c_{m-n-1}, \cdots, c_{m-2n}) \\ &= \{f(a_{m-n}, \cdots, a_{m-n-1}; b_{m-n}, \cdots, b_{m-n-1}), \cdots, \\ &\quad , f(a_{m-2n+1}, \cdots, a_{m-2n}; \ b_{m-2n+1}, \cdots, b_{m-2n})\} \\ &\qquad\qquad \vdots \end{aligned}$$

$$(c_{m-kn-1}, \cdots, c_{m-n+r}) =$$
$$\{f(a_{m-kn}, \cdots, a_{m-kn-1}; b_{m-kn}, \cdots, b_{m-kn-1}), \cdots$$
$$, f(a_{m-n+r+1}, \cdots, a_{m-n+r}; b_{m-n+r+1}, \cdots, b_{m-n+r})\} \quad (9)$$

여기서 $k-1 = \lceil m/n \rceil$ 이다.



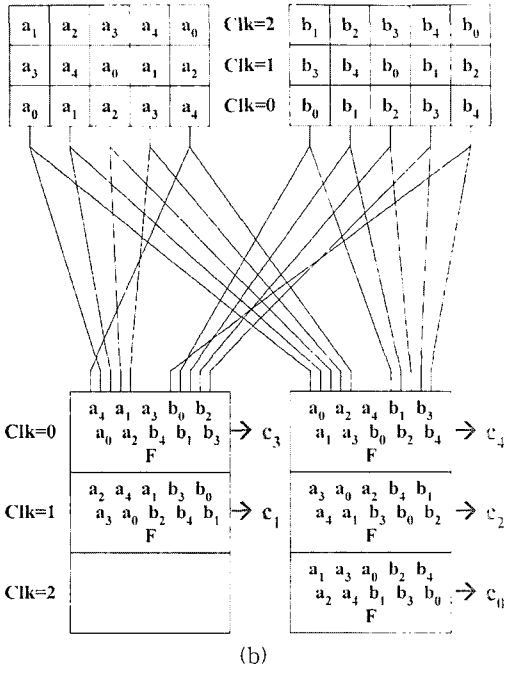


그림 3. GF(2^5) 곱셈기 (a) 구조 (b) 매 클럭 레지스터와 출력값 변화

Fig. 3. Multiplier in GF(2^5) (a) Structure (b) Contents and output value of a register each cycle.

따라서 클럭 마다 오른쪽으로 n 비트 순환치환 시키는 LFSR과 n개의 이진함수를 사용하면 클럭 당 n 비트씩 계산하고 $\lceil m/n \rceil$ 클럭 만에 최종 계산 결과를 얻을 수 있는 곱셈기를 구현할 수 있다. m=5, n=2일 때의 경우의 곱셈기 구조를 <그림 3(a)>에, 매 클럭 LFSR에 들어있는 값들과 이진 함수와의 연결 상태 및 출력 결과를 <그림 3(b)>에 보였다.

<그림 2>와 <그림 3(b)>를 비교해서 보면 이진함수로의 입력이 두 가지가 서로 같은 것을 확인할 수 있다. 따라서 계산되는 결과는 같다. 참고로 m=5인 경우 <그림 1>의 Massey-Omura 방식은 5 클럭, <그림 2>의 병렬 Massey-Omura 방식은 1 클럭 <그림 3>의 가변구조방식은 3 클럭이 최종 결과를 위해 필요하다. 반면에 면적은 가변 구조 방식이 병렬 Massey-Omura 방식의 40%, Massey-Omura 방식의 2배 정도가 요구된다.

2. GF(2^{193}) 정규기저 곱셈기 설계

앞의 Ⅲ장에서 제시한 방식으로 GF(2^{193})의 정규 기저 곱셈기를 설계하였다. 유한체의 생성 다항식으로는 IEEE 표준 1363[9]에서 제시한 m=193일 때의 삼항 다항식을 사용하였다.

$$f(x) = x^{193} + x^{15} + 1 \quad (10)$$

제안된 가변구조 방식과 기존의 방식으로 구현된 곱셈기를 비교하기 위해 각 구조에 대해 Synopsys tool을 이용하여 합성하고 0.35 ㎛ CMOS 공정을 이용하여 게이트수, 동작 주파수 그리고 단위 시간당 데이터 처

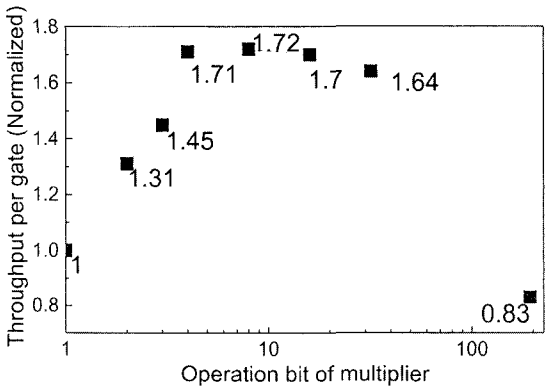


그림 4. GF(2^{193})의 정규 기저 곱셈기들의 구현 효율

Fig. 4. Implementation efficiency of normal basis multipliers in GF(2^{193}).

리 능력(throughput)을 비교하였다. 그리고 사용된 면적에 대한 throughput을 구하여 구현 효율을 비교하였다. <그림 4>에 가변 구조 곱셈기의 처리 bit수를 변화시키며 구현 효율을 구한 그림이 나타나 있다. n=8에서 최대 효율을 보임을 알 수 있고 n=4~16 bit 사이에서는 큰 효율 저하 없이 사용할 수 있다. n=1인 경우가 Massey-Omura 구조이고 n=193인 경우가 병렬 Massey-Omura 구조이다.

n=8인 경우에 대해 <표 1>에서 Massey-Omura 구조와 병렬 Massey-Omura 구조와 비교한 결과를 나타내었다. 설계한 곱셈기는 두 개의 193비트 입력을 받아 매 클럭 마다 8 비트씩 계산하고 최종 결과까지 25 클럭을 소요한다. 게이트 수는 편의상 정규화 시켜서 비교하였고 최대 출력량은 실제 수치와 정규화 된 수치를 같이 비교하였다. 앞서 예상했던 대로 Massey-Omura 곱셈기에 비해 병렬 Massey-Omura 곱셈기가 195배의 면적, 약 163배의 출력량을 보인 반면 본 논문에서 제안한 구조는 4배의 면적, 약 7배의 출력량을 보인다. n이 변화할 때 193 bit 레지스터는 항상 존재하므로 n이 작은 경우에는 레지스터 면적이 차지하는 비중이 커서 n=8인 경우 순수 연산기 부분이 8배 정도 증가하여도 전체 면적은 4배만 증가한다. n=193인 경우는 내부에서 구동해야하는 부하가 커져서 버퍼가 크게 증가하여 면적이 n보다 더 크게 증가한다. 구현 효율을 보면 가변구조의 경우가 병렬 Massey-Omura에 비해 2배 이상 큰 값을 갖는다. 이는 동일한 throughput을만들기 위해 가변구조의 곱셈기를 병렬로 연결해 구현할 경우 병렬구조에 비해 절반 이하의 면적만 필요하다는 것을 의미한다.

표 1. $GF(2^{193})$상에서의 정규기저 곱셈기의 동작 특성 비교

Table 1. Characteristics of multipliers based on normal basis on $GF(2^{193})$.

Type	MO[1]	CMO[2]	PMO[3]
Total Number of Gates(normalized)	1	4.2	194.9
Maximum Clock Frequency (㎒)	227	217	192
Maximum Throughput (normalized) (Mop/s)[4]	1.18 (1)	8.69 (7.36)	192.31 (162.97)
implementation efficiency (throughput/gate counts) (normalized)	1	1.72	0.83

[1]MO: 직렬 Massey-Omura 곱셈기
[2]CMO: 제안한 Massey-Omura 곱셈기
[3]PMO: 병렬 Massey-Omura 곱셈기
[4]Mop/s : 1초당 백만 번 연산

<그림 5>는 곱셈기의 simulation 결과 파형으로, α^7과 α^{120}에 해당하는 비트열이 각각 ina와 inb에 병렬로 입력되어 outc(rtl)에 functional simulation 결과가 outc(syn)에 timing simulation 결과가 출력되었다. 결과는 α^{127}로 동일함을 확인할 수 있다.

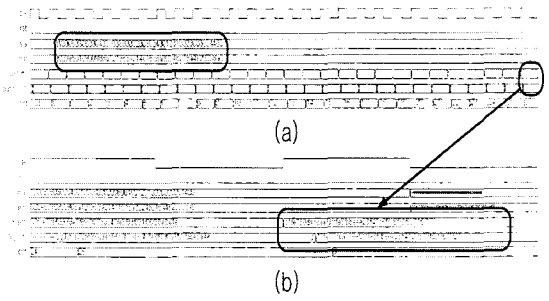


그림 5. Simulation 결과 파형 (a) 입력 및 출력 (b) 출력 부분 확대

Fig. 5. Simulation waveform (a) Input and output (b) Output in detail.

3. $GF(2^{193})$ 정규기저 역원기 설계

앞서 구현한 곱셈기를 이용하여 정규기저 유한체 역원기를 구현하였다. 구현한 역원기는 G.L.Feng의 역원 알고리즘을 적용하였으며 그 블록다이어그램은 <그림 6>과 같다. 역원기에 대해서도 역원기에 사용한 곱셈기

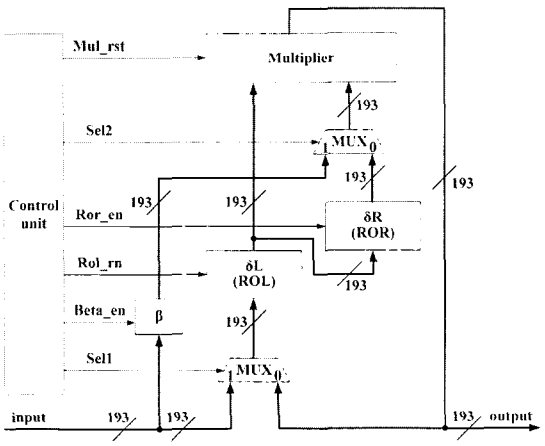


그림 6. $GF(2^{193})$에서의 역원기의 블록 다이어그램

Fig. 6. Block diagram of inversion unit in $GF(2^{193})$.

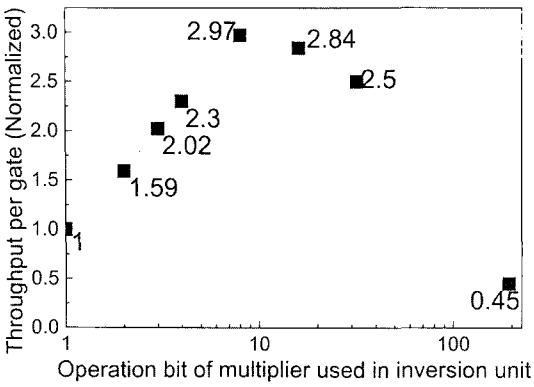


그림 7. $GF(2^{193})$ 정규기저 역원기에 사용된 곱셈기의 연산 비트에 따른 구현 효율

Fig. 7. Implementation efficiency of normal basis inversion units in $GF(2^{193})$.

의 연산 비트에 따른 구현 효율을 살펴보았다. 그 결과는 <그림 7>과 같이 n=8에서 최대 구현 효율을 보였고 이는 병렬 구조에 비해 6.6배의 값이다. 따라서 역원기에서는 그 효율이 더욱 증가함을 알 수 있다.

Ⅳ. 결론

본 논문에서는 타원곡선 암호 시스템에서 사용할 수 있는 가변구조의 193비트 정규기저 유한체 곱셈기를 구현하고 이를 이용하여 193비트 정규기저 유한체 역원기를 구현하였다. 본 논문에서 제시한 곱셈기의 구조는 m 값이 큰 경우에 제한된 설계 면적과 동작시간에 대해서 가변적으로 주어진 조건을 적절히 만족시킬 수 있다는 장점이 있다. 기본적으로 타원곡선에서 사용하는 유한체의 크기는 160비트 이상이므로 이러한 특징은 정규기저 곱셈기를 설계하는데 매우 유용하다. 가변구조 곱셈기를 이용하면 응용 목적에 따라 원하는 throughput과 면적을 적절히 조절하여 구현할 수 있고 그 구현 효율은 기존 구조에 비해 곱셈기는 최대 2배, 역원기는 최대 6배의 향상이 있음을 확인하였다.

참 고 문 헌

[1] K.H. Leung, K.W. Ma, W.K. Wong, P.H.W. Leong, FPGA implementation of a microcoded elliptic curve cryptographic processor, Field Programmable Custom Computing Machines, 2000 IEEE Symposium on , 2000, pp. 68-76.

[2] SEC1, Elliptic Curve Cryptography, v.1.0, pp. 62, Sep.20, 2000.

[3] TTAS.KO-12.0015, 부가형 전자서명 방식 표준-제3부: 타원곡선을 이용한 인증서 기반 전자서명 알고리즘, pp. 8-10, pp. 61-70

[4] S. Sutikno, R. Effendi, A. Surya, Design and Implementation of Arithmetic Processor For F_{2^m} for Elliptic Curve Cryptosystems, IEEE APCCAS 1998, pp. 647-650.

[5] J.L. Massey and J.K. Omura, Computational method and apparatus for finite field architecture, U.S. PAtent Application, submitted 1981.

[6] G.L. Feng, A VLSI Architecture for Fast Inversion in $GF(2^m)$, IEEE Trans. Computer, vol. 38, no. 10, pp. 1383-1386, Oct. 1989.

[7] C.C. Wang, T.K. Trung, H.M. Shao, L.J. Deutsch, J.K. Omura, I.S. Reed, VLSI architectures for computing multiplications and inverses in $GF(2^m)$, IEEE Trans. Compt., vol C-34, pp. 709-717, Aug. 1985.

[8] Y.R. Shayan, T. Le-Ngoc, The least complex parallel Massey-Omura multiplier and its LCA and VLSI designs, Circuits, Devices and Systems, IEE Proceedings G , vol. 136, issue 6, pp. 345-349, Dec. 1989.

[9] IEEE std 1363-2000, IEEE standard specifications for public-key cryptography, pp. 110.

저 자 소 개

李 燦 豪(正會員)

1987년 2월 : 서울대학교 전자공학과 졸업(공학사). 1989년 2월 : 서울대학교 대학원 전자공학과 졸업(공학 석사). 1994년 6월 : University of California, Los Angeles 전자공학과 졸업(공학 박사). 1994년 8월~1995년 2월 : 삼성전자 반도체연구소 선임연구원. 1995년 3월~현재 : 숭실대학교 전자공학과 부교수. <주관심분야 : 채널 코덱의 VLSI 구현, 저전력 프로세서 설계, 암호프로세서 설계, On-chip-network, SOC 설계 방법론 등.>

李 定 鎬(正會員)

2001년 2월 : 숭실대학교 전자공학과 졸업(공학사). 2003년 2월 : 숭실대학교 대학원 전자공학과 졸업(공학 석사). 2001년 2월~현재 : (주) 화음소 근무중. <주관심분야 : 암호프로세서, 마이크로세서, DSP 프로세서 설계.>