타원곡선 암호시스템에 사용되는 최적단위 연산항을 기반으로 한 기저체 연산기의 하드웨어 구현

(A Hardware Implementation of the Underlying Field Arithmetic Processor based on Optimized Unit Operation Components for Elliptic Curve Cryptosystems)

조 성 제 † 권 용 진 ††
(Seong-Je Jo) (Yong-Jin Kwon)

요 약 1985년 N. Koblitz와 V. Miller가 각각 독립적으로 제안한 타원곡선 암호시스템(ECC : Elliptic Curve Cryptosystems)은 보다 짧은 비트 길이의 키만으로도 다른 공개키 시스템과 동일한 수준의 안전도를 유지할 수 있다는 장점으로 인해 IC 카드와 같은 메모리와 처리능력이 제한된 하드웨어에도 이식가능 하다. 또한 동일한 유한체 연산을 사용하면서도 다른 타원곡선을 선택할 수 있어서 추가적인 보안이 가능하기 때문에 고수준의 안전도를 유지하기 위한 차세대 암호 알고리즘으로 각광 받고 있다.

본 논문에서는 효율적인 타원곡선 암호시스템을 구현하는데 있어 가장 중요한 부분 중 하나인 타원곡선 상의 점을 고속으로 연산할 수 있는 전용의 기저체 연산기 구조를 제안하고 실제 구현을 통해 그 기능을 검증한다. 그리고 기저체 연산의 면밀한 분석을 통해 역원 연산기의 하드웨어 구현을 위하여 최적인 단위 연산항의 도출에 기반을 둔 효율적인 방법론을 제시하고, 이를 바탕으로 현실적인 제한 조건하에서 구현 가능한 수준의 게이트 수를 가지는 고속의 역원 연산기 구조를 제안한다.

또한, 본 논문에서는 제안된 방법론을 바탕으로 실제 구현된 설계회로가 기존 논문에 비해 게이트 수는 약 8.8배가 증가하지만, 승법연산 속도는 약 150배, 역원연산 속도는 약 480배 정도 향상되는 우수한 연구결과가 얻어짐을 보인다. 이것은 병렬성을 적용함으로서 당연히 얻어지는 속도면에서의 이득을 능가하는 성능으로, 본 논문에서 제안한 구조의 우수성을 입증하는 결과이다. 실제로, 승법 연산기의 속도에 관계없이 역원연산의 수행시간은 $\lfloor \log_2(m-1) \rfloor \times$(*clock cycle for one multiplication*)으로 최적화가 되며, 제안한 구조는 임의의 유한체 F_{2^m}에 적용 가능하다.

제안한 전용의 연산기는 암호 프로세서 설계의 기초자료로 활용되거나, 타원곡선 암호 시스템 구현시 직접 co-processor 형식으로 임베드 되어 사용할 수 있을 것으로 사료된다.

키워드 : 타원곡선암호, GF(2^m)상에서 승법에 대한 역원연산, 하드웨어 설계, 기저체 연산

Abstract In recent years, the security of hardware and software systems is one of the most essential factor of our safe network community. As elliptic Curve Cryptosystems proposed by N. Koblitz and V. Miller independently in 1985, require fewer bits for the same security as the existing *cryptosystems, for example RSA, there is a net reduction in cost, size, and time.* In this thesis, we propose an efficient hardware architecture of underlying field arithmetic processor for Elliptic Curve Cryptosystems, and a very useful method for implementing the architecture, especially multiplicative inverse operator over $GF(2^m)$ onto FPGA and futhermore VLSI, where the method is based on optimized unit operation components.

We optimize the arithmetic processor for speed so that it has a resonable number of gates to implement. The proposed architecture could be applied to any finite field F_{2^m}. According to the simulation result, though the number of gates are increased by a factor of 8.8, the multiplication speed

† 비 회 원 : 한국항공대학교 항공통신정보공학과
sjjo@mail.hangkong.ac.kr
†† 종신회원 : 한국항공대학교 전자정보통신컴퓨터공학부 교수
yjkwon@tikwon.hangkong.ac.kr
논문접수 : 2001년 1월 5일
심사완료 : 2001년 9월 20일

We optimize the arithmetic processor for speed so that it has a resonable number of gates to implement. The proposed architecture could be applied to any finite field F_{2^m}. According to the simulation result, though the number of gates are increased by a factor of 8.8, the multiplication speed and inversion speed has been improved 150 times, 480 times respectively compared with the thesis presented by Sarwono Sutikno et al. [7].

The designed underlying arithmetic processor can be also applied for implementing other crypto-processor and various finite field applications.

Key words : Elliptic curve cryptography, Multiplicative inverse over $GF(2^m)$, Hardware design, Underlying field arithmetic operation.

1. Introduction

최근 들어 통신망상의 보안이 원활한 정보 통신 세계를 구축하기 위해 중요한 요소가 됨에 따라 정보의 암호 기술 또한 활발히 연구되고 있다. 초기의 암호 기술은 간단한 대치 및 전치방식을 사용했으나, 근래 컴퓨터의 계산능력이 비약적으로 증대되고 암호를 해독할 수 있는 새로운 알고리즘 및 수학적 이론들의 등장으로 인해 원하는 수준의 보안을 유지하기 위해 제안되는 암호 방식은 점점 더 복잡해지고 있다. 그러나 암호 방식의 복잡도가 증가함에 따라, 이를 응용하고 사용하기 위해 암호 알고리즘을 실제로 구현하는 것이 또 다른 문제로 대두되었다. 즉, 계산량이 커지고 암·복호화 시간이 오래 걸리며, 시스템을 구현하기 위해서는 사양이 높은 하드웨어가 필요하게 되었다.

1985년 N. Koblitz와 V. Miller가 각각 독립적으로 제안한 타원곡선 암호시스템(ECC : Elliptic Curve Cryptosystem)은 보다 짧은 비트 길이의 키만으로도 다른 공개키 시스템과 동일한 수준의 안전도를 유지할 수 있는 장점으로 인해 IC 카드와 같은 메모리와 처리능력이 제한된 하드웨어에도 이식가능 하다[1]. 또한 동일한 유한체 연산을 사용하면서도 다른 타원곡선을 선택할 수 있어서 추가적인 보안이 가능하기 때문에 고수준의 안전도를 유지하기 위한 차세대 암호 알고리즘으로 각광 받고 있다[2].

타원곡선 암호는 이미 차세대 암호 기법으로 인정받고 있으며, HP, Motorola, NTT, Fujitsu 등 세계 주요 IT 업체들이 타원곡선 암호기술을 기반으로 한 새 표준 암호 기술을 개발 중이다[3]. 또한 외국에서는 타원곡선 암호와 관련된 연구 및 소프트웨어, 하드웨어 구현이 다수 이루어지고 있다. 그러나 국내에서는 아직 타원곡선 암호에 대한 연구가 미비한 실정이며, 개발 역시 소프트웨어를 사용한 구현 사례만이 극소수 있을 뿐, 하드웨어 구현 사례는 없다. 따라서 본 연구에서는 타원곡선이 정의된 기저체 상의 연산을 고속으로 수행할 수 있는 전용 프로세서의 개발을 통해 IC Card, 이동 통신 단말기 등과 같은 낮은 사양의 하드웨어 환경에서도 강력한 보안이 필요한 여러 분야로의 응용을 가능하게 하고, 나아가 국내 암호 프로세서의 IP 구축을 목적으로 하고 있다.

본 논문에서는 먼저 타원곡선 암호의 수학적인 배경에 대해서 설명한다. 다음으로 구현의 핵심이 되는 기저체 연산을 분석하고, 체 원소의 표현법에 따른 연산 방법의 차이에 대해서 고찰한다. 이러한 분석을 통해 얻은 하드웨어로의 효율적인 구현을 위한 알고리즘과 방법론을 제시한 뒤, 최종적으로 도출된 이론을 바탕으로 연산기를 설계하는 과정 및 구현 결과에 대해서 기술한다.

2. Mathematical Background

타원곡선을 암호시스템에 적용하기 위해서는 우선 적절한 타원곡선을 선택하고 그에 따른 타원곡선 군을 구해야 한다. 기저체 상의 타원곡선을 나타내기 위해서 여러 가지 좌표계를 사용할 수 있으나, Affine 좌표계나 Projective 좌표계를 주로 사용한다. Affine 좌표계를 사용하면 점 연산 시 곱셈과 덧셈의 횟수가 줄어들지만 역원을 구하는 과정이 필요하다. 반면 Projective 좌표계를 사용하면 역원 연산은 없으나 곱셈과 덧셈의 횟수가 증가한다. 타원곡선을 표현함에 있어서 좌표계의 선택은 역원을 얼마나 효율적으로 구할 수 있는가에 달려 있다. 역원을 구하는 과정 역시 여러 단계의 곱셈 및 덧셈으로 이루어져 있기 때문에 전체적인 연산량을 비교 검토한 후에 좌표계를 선택하는 것이 바람직하다. 유한체 상에서 역원을 구하는 효율적인 알고리즘이 최근 다수 발표되었고 본 연구에서도 역원을 구하는 효율적인 하드웨어 알고리즘을 제시하고 있으므로, Affine 좌표계에 타원곡선을 나타내는 방법을 선택하였다.

타원곡선 군을 얻는 방법은 다음과 같다. K를 임의의 체라 하자. 그러면 아래의 non-homogeneous(affine) Weierstrass 방정식을 만족하는 해 $(x, y) \in K^2$와 무한원점(point at infinity) O는 K상의 타원곡선을 정의한다.

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \quad \text{단, } a_i \in K$$

이때 체 K의 표수가 2가 아니라면 $y^2=x^3+ax^2+bx+c$로 변형할 수 있다(만약 표수 $K>3$이면 $y^2=x^3+bx+c$로 변형할 수 있다). 그리고 체 K의 표수가 2인 경우에 이 식은 아래 식 중 어느 하나로 변환할 수 있다.

$$y^2+a_3y=x^3+a_4x+a_6$$

또는

$$y^2+xy=x^3+a_2x^2+a_6.$$

만약 E가 이 곡선상에 있는 점들의 집합이라면, 덧셈 연산을 잘 정의함으로써 집합 $E\cup\{O\}$는 가환군(abelian group)이 될 수 있으며 이것을 $E(K)$로 나타낸다. 이 가환군은 항등원인 무한원점 O와 함께 타원곡선 군이 된다.

이제 타원곡선 상에 있는 점들간의 가법 연산을 정의한다. 다음과 같이 가법 연산을 정의 하면 타원곡선 상의 점 집합은 가환군(abelian group)을 이룬다.

정의 E를 실수상의 타원곡선 이라 하고, P와 Q를 타원곡선 E상에 있는 2개의 점이라 한다. 여기에서 아래와 같은 방법에 의해 P의 음 $-P$와 합 $P+Q$를 정의한다.

(1) 만약 P가 무한원점 O이라면, $-P$를 O라 하고, $P+Q$를 Q라 정의한다. 즉, 무한원점 O가 여기에서 생각하고 있는 점의 군에 있어서 가법의 단위원인 zero원에 상당한다고 생각할 수 있다. 이것으로부터 뒤에 기술하는 부분에서는 두 점 P와 Q의 어느 것도 무한원점이 아니라고 가정한다.

(2) $-P$는 P와 동일한 x 좌표값을 가지며, P의 y 좌표의 음의 값을 가진다. 즉, $-(x,y)=(x,-y)$라 한다. 점 (x,y)가 곡선상의 점이라면 점 $(x,-y)$도 곡선상의 점이 되는 것은 명백하다.

(3) 만약 2점 P와 Q가 서로 다른 x좌표값을 가질 때, 직선 $l=\overline{PQ}$ 와 곡선은 반드시 또 하나의 교점 R을 가지는 것은 명백하다(단, 직선이 점 P에 있어서 곡선과 접하지 않는 것으로 한다. 만약 점 P에서 접하고 있을 경우에는 $R=P$가 되고, 점 Q에서 접하고 있을 때에는 $R=Q$로 한다). 이때 $P+Q$를 $-R$, 즉 제 3번째 교점의 x축에 대한 대칭 상 이라고 정의한다.

(4) 만약 $Q=-P$일 때(즉, Q가 P에 대해서 동일한 x 좌표값을 가지고 옵인 y좌표값을 가짐), $P+Q=O$(무한원점)라 정의한다.

(5) 나머지 생각할 수 있는 경우는 $P=Q$일 때이다. 이 경우 l을 곡선상의 점 P에서의 접선이라 하고, R을 곡선과 접선 l의 접점 이외의 유일한 교점이라 하면, $P+Q=-R$이라고 정의한다(만약 생각하고 있는 접선이 곡선과 2중으로 접하고 있는, 즉 P가 변곡점(point of inflection)인 경우에는 R을 P라 한다).

이와 같은 정의를 바탕으로, K의 표수가 2와 3이 아닌 경우에 $P+Q$의 좌표값을 유도해 보자. 기저체 K의 표수가 2도 아니고 3도 아닌 경우, 타원곡선은 무한원점 O와 다음 식을 만족하는 $x,y\in K$인 점 (x,y)의 집합을 나타낸다.

$$y^2=x^3+a_4x+a_6.$$

우선, (x_1,y_1), (x_2,y_2), (x_3,y_3)을 각각 P, Q, $P+Q$의 좌표 값이라고 하자. x_3과 y_3을 x_1, y_1, x_2, y_2를 사용해서 나타내면 $P+Q$의 좌표값을 구하는 것이 된다. $y=\alpha x+\beta$를 두 점 P, Q를 지나는 직선(여기에서 생각하는 (3)의 직선은 y축에 평행하지 않다)의 방정식이라 한다. 이때 $\alpha=(y_2-y_1)/(x_2-x_1)$, $\beta=y_1-\alpha x_1$ 임은 명백하다. 곡선상의 두 점 P와 Q에 의한 직선 상의 점, 즉 점 $(x,\alpha x+\beta)$은 $(\alpha x+\beta)^2=x^3+a_4x+a_6$를 만족할 때, 그리고 그때에 한해서 곡선상에도 있다. 이것으로부터 3차 방정식 $x^3-(\alpha x+\beta)^2+a_4x+a_6$의 근의 개수만큼 교점이 존재하는 것을 알 수 있다. 이제, $(x_1,\alpha x_1+\beta)$과 $(x_2,\alpha x_2+\beta)$은 각각 P, Q라 하는 곡선상의 두 점인 것을 알고 있으므로, 앞의 방정식의 근 중 두 개의 근이 x_1과 x_2임을 알 수 있다. 또한, 모닉 다항식의 해의 합은, 다항식에서 두 번째로 높은 차수인 항의 계수에 음을 취한 값과 동일하므로, 이 경우에 구하고자 하는 세 번째의 해는 $x_3=\alpha^2-x_1-x_2$가 된다. 이것으로부터, x_3에 대해서 x_1, x_2, y_1, y_2를 이용해 구하는 것이 가능하며, 따라서 $P+Q=(x_3,-(\alpha x_3+\beta))$도 구해진다. 결과적으로 $P+Q$는 다음과 같다.

$$\begin{aligned} x_3 &= \left(\frac{y_2-y_1}{x_2-x_1}\right)^2-x_1-x_2 \\ y_3 &= -y_1+\left(\frac{y_2-y_1}{x_2-x_1}\right)(x_1-x_3) \end{aligned}$$

가 된다.

그리고 경우(5)의 $P=Q$인 경우에 대해서도 α를 점 P의 미분계수 dy/dx로 하는 것 이외에는 동일하다. 타원곡선 식의 음함수 미분(implicit differentiation)으로부터 식 $\alpha=(3x_1^2+a_4)/2y_1$이 유도되며, P의 2배의 좌표값을 나타낸 식

$$\begin{aligned} x_3 &= \left(\frac{3x_1^2+a_4}{2y_1}\right)^2-2x_1 \\ y_3 &= -y_1+\left(\frac{3x_1^2+a_4}{2y_1}\right)(x_1-x_3) \end{aligned}$$

가 유도된다.

마찬가지로, K의 표수가 2인 경우 $P+Q$의 좌표값을 유도하면 다음과 같다. 세부적인 유도과정은 [4]를 참조한

다.

$$x_3 = \left(\frac{y_1+y_2}{x_1+x_2}\right)^2+\frac{y_1+y_2}{x_1+x_2}+x_1+x_2+a_2 \quad P\neq Q$$
$$= x_1^2+\frac{a_6}{x_1^2} \quad P=Q$$

$$y_3 = \left(\frac{y_1+y_2}{x_1+x_2}\right)(x_1+x_2)+x_3+y_1 \quad P\neq Q$$
$$= x_1^2+\left(x_1+\frac{y_1}{x_1}\right)x_3+x_3 \quad P=Q$$

유도된 $P+Q$의 좌표 값에 대한 식은, K의 표수가 2이고 타원곡선이 nonsupersingular curve인 경우에 대한 것이다. 본 연구에서는 하드웨어로의 구현을 목적으로 하고 있기 때문에 이후로는 기저체 K를 표수가 2인 $GF(2^m)$형태로 생각한다.

3. Optimal Normal Bases in $GF(2^m)$

타원곡선을 이용한 암호 시스템을 설계할 경우 가장 기본이 되는 연산은 타원곡선 상에 있는 점들간의 가법 연산이다. 그런데 유한체 상의 타원곡선 위에 있는 점들이 군을 형성할 수 있도록 정의된 가법연산은 2장에서 유도한 바와 같이 기저체 연산, 즉 타원곡선이 정의된 유한체의 연산으로 이루어져 있다. 따라서 타원곡선을 이용한 암호 시스템의 성능, 즉 암·복호화 속도는 기저체 연산기의 성능에 의해 결정된다. 유한체 상의 연산은, 근본적으로는 동일하지만, 체의 원소 표현법에 따라 세부적인 계산법이 달라진다. 체의 원소 표현법은 크게 polynomial basis representation과 normal basis representation이 있는데, 하드웨어 구현시에는 비트 단위의 연산이 간단해 지는, 표수 2인 체 상의 normal basis representation이 보다 적합하다. 구체적인 사항은 다음과 같다.

1. 만약 F_{2^m}이 type I ONB만을 가진다면 $f(x)=x^m+x^{m-1}+\cdots+x^2+x+1$이라 한다. 그렇지 않고, F_{2^m}이 type II ONB를 가진다면 다음의 점화식을 사용해 $f(x)=f_m(x)$를 계산한다.

$$f_0(x) = 1,$$
$$f_1(x) = x+1,$$
$$f_{i+1}(x) = xf_i(x)+f_{i-1}(x), \quad i\geq 1.$$

$f(x)$는 계수가 F_2의 원소이고 차수가 m인 다항식이다. 다항식의 집합 $\{x, x^2, x^{2^2}, \ldots, x^{2^{m-1}}\}$은 F_2상에 있는 F_{2^m}의 기저를 형성한다. 이 기저를 normal basis(정규기저)라 한다.

2. 다항식 x^{2^i}을 mod $f(x)$하여 생성된 binary vector를 i번째 열로 해서 $m\times m$ 행렬 A를 구성한다. 단, $0\leq i\leq m-1$ 이다. A의 각 엔트리는 F_2의 원소로 이루어져 있다.

3. A의 역행렬 A^{-1}를 구한다.

4. 다항식 $x\cdot x^{2^i}$을 mod $f(x)$하여 생성된 binary vector를 v라 하자. $v\cdot A^{-1}$를 i번째 열로 해서 $m\times m$ 행렬 T를 구성한다.

5. $\lambda_{ij}=T(j-i,-i)$라 한다. 단, $0\leq i,j\leq m-1$ 이다. 여기에서 $T(i,j)$는 T의 (i,j)엔트리를 나타낸다. 각각의 λ_{ij}는 F_2의 원소로 이루어져 있다. 단 하나의 j에 대해서 $\lambda_{0j}=1$가 성립하며, i가 $1\leq i\leq m-1$인 경우에는 정확히 2개의 j에 대해서 $\lambda_{ij}=1$이 성립한다. 그러므로 행렬 T에 있는 m^2개의 엔트리 중에서 $2m-1$개의 엔트리만이 1이며, 나머지는 모두 0이다. 이러한 이유 때문에 이 normal basis는 특별히 optimal normal basis라 한다.

이러한 표현법을 optimal normal basis representation 또는 ONB representation이라 부르며, 유한체 F_{2^m}에서의 연산기를 효율적으로 구현하기 위한 수학적인 기반이 된다.

4. Arithmetic Operations

이와 같이 ONB representation을 사용하면 유한체 F_{2^m} 상에 있는 임의의 두 원소간의 가법연산(Addition)은 간단히 비트별 XOR 가 되며, 제곱연산(Squaring)은 rotation이 된다. 또한 $a=(a_{m-1}a_{m-2}\cdots a_0)$와 $b=(b_{m-1}b_{m-2}\cdots b_0)$를 F_{2^m}의 원소라 하고 $a\cdot b=c=(c_{m-1}c_{m-2}\cdots c_0)$를 승법연산(Multiplication)의 결과라 하면, m-tuple c는 다음과 같이 나타내어진다.

$$c_k = \sum_{i=0}^{m-1}\sum_{j=0}^{m-1} a_{i+k}b_{j+k}\lambda_{ij}, \quad 0\leq k\leq m-1.$$

단, 여기서 각 첨자는 모듈로 m을 취한다.

위의 수식을 바탕으로 승법 연산기를 하드웨어로 설계하는 방법은 크게 두 가지가 있는데, 첫 번째는 serial하게 설계하는 것이고, 두 번째는 parallel하게 설계하는 것이다. 전자의 방법은 레퍼런스 [7]에서 구현되었으며 회로사이즈가 작은 대신 한번의 승법연산을 수행하는데 여러 클럭(약 m 클럭)이 소요되는 단점이 있다. 본 연구에서는 연산기를 속도면에서 최적화하고 제어회로를 단순화하여, trade-off를 고려한 전체적인 관점에서 퍼포먼스를 향상시킬 수 있도록 승법 연산기를 병렬적인 조합회로로 설계한다. 설계에 사용한 승법 연산기의 구조는 그림 1과 같다.

그림 1에서 입력은, 상기의 수식 중 λ 벡터에 의해 선택되는 a_{i+k} 와 b_{j+k} 각 비트의 m-tuple A와 B 이며, 출력은 c_k의 m-tuple C 이다. 또한, k번째 Unit Cell의 입력은 k 와 λ 벡터 값에 의해 결정되는 특정 a 비트와 특

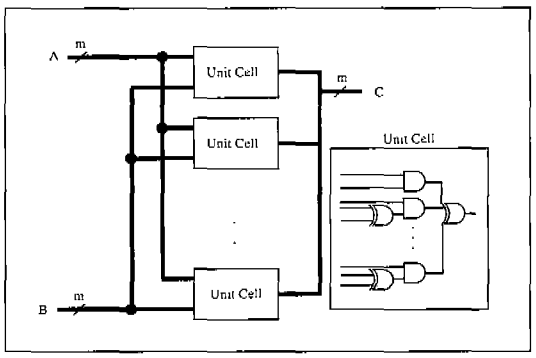

그림 1 F_{2^m}에 대한 승법 연산기의 구조

정 b 비트의 $\sum$ 이며(상기 수식 참고), 출력되는 한 비트는 결과 값 C 중 k 번째 비트에 해당한다.

5. Hardware Architecture for Multiplicative Inversion

임의의 체 K에서 정의된 승법연산에 대한 역원은 일반적으로 다음과 같이 나타낼 수 있다.

$$a \cdot a^{-1} = 1. \qquad 단,\ a \in K$$

따라서 a가 F_{2^m}의 원소일 때에도 이 식을 적용할 수 있다. 단, 이 경우 승법연산은 전술한 정의에 따르며, 승법연산에 대한 항등원 "1"은 m-tuple (11…1)이 된다. 위의 식에 페르마 정리를 적용하면,

$$a \cdot a^{-1} \equiv a^{2^m-1} \mod 2^m$$
$$a^{-1} \equiv a^{2^m-2} \mod 2^m$$

이 된다. 지수만 살펴보면 아래 식과 같이 된다.

$$-1 = 2^m - 2$$

이 식의 우변을 적당히 인수분해 할 수 있다면 역원연산은 제곱과 승산으로 계산될 수 있다. 인수분해 과정은 다음과 같다.

$$\begin{aligned} 2^m-2 &= 2(2^{m-1}-1) \\ &= 2(2^{\frac{m-1}{2}}-1)(2^{\frac{m-1}{2}}+1), \\ &\quad m-1\text{이 짝수일 경우.} \\ &= 2(2(2^{\frac{m-2}{2}}-1)(2^{\frac{m-2}{2}}+1)+1), \\ &\quad m-1\text{이 홀수일 경우.} \end{aligned}$$

이러한 인수분해 과정은 [5]에 따른 것이며, 각각의 항에 대해서 이러한 인수분해는 반복적으로 적용가능 하다. 특별히, 하드웨어 구현에서는 2의 멱승 연산이 rotation으로 간단하게 처리 될 수 있으므로, 2의 멱승 형태의 연산을 최대화시키는 위의 인수분해는 매우 바람직하다. 그러나 이러한 지수의 인수분해가 이루어졌다 하더라도, 이 수식을 하드웨어로 옮기는 것은 또 다른 문제이다. 즉, 하드웨어 구현의 특징이자 장점인 병렬성을 이용해 단순히 모든 항을 병렬로 계산한다면 역원 연산의 특성상 하드웨어 면적 및 지연이 지나치게 커져, 전체를 모두 병렬로 구현하는 것은 실제적인 구현에 적합하지 않다.

사실 역원 연산기의 설계에 대해서 이러한 논의를 하기 위해서는 승법 연산기의 개수와 실행시간 사이의 trade-off에 대해서 고찰할 필요가 있다. 분명히 승법 연산기를 많이 사용할수록 역원을 구하기 위한 연산시간은 줄어든다. 다시 말해서 한 블럭에 수행할 연산의 단위항을 크게 분할할수록 필요한 승법 연산기의 수는 증가하고, 역원연산에 소요되는 시간은 감소한다. 역원 연산기를 설계함에 있어서 승법 연산기의 gate count가 전체 회로사이즈의 대부분을 차지하므로, 역원 연산기의 gate count를 추정하기 위해서 승법 연산기의 gate count만 고려해도 무방하다. 합성에 의한 실제 실험에 의하면 승법 연산기 개수만을 고려한 gate count의 오차는 1%이내였다. 승법 연산기의 gate count를 M_{gc}라 하고, 기저체를 F_{2^m}라 가정하면, 본 논문에서 제안하는 단위항 유도 방식 따른 회로 사이즈와 역원연산을 수행하는데 걸리는 시간과의 관계는 다음과 같다.

표 1 역원 연산기의 회로 크기와 수행시간의 관계

승법 연산기 개수 / 비교항목	1	2	4	6	8	10
Gate Count	1 M_{gc}	2 M_{gc}	4 M_{gc}	6 M_{gc}	8 M_{gc}	10 M_{gc}
Reqd. Time [Clk cycle for 1 mul]	10	7	4	3	2	1

또한, 역원을 구하기 위해서 필요한 승법연산의 횟수는 기저체의 선택에 따라서 비선형적으로 변화한다. 기저체를 F_{2^m}라 할 때, 필요한 승법의 횟수는 다음과 같은 수식으로 표현될 수 있다.

$$Number\ of\ Mltiplications = \lfloor \log_2(m-1) \rfloor + \#(Set\ bits\ in\ (m-1)_2) - 1$$

그리고, 연산을 수행하는데 걸리는 시간은 다음과 같다.

$$Reqd.\ Time = \lfloor \log_2(m-1) \rfloor \times (clock\ cycle\ for\ one\ multiplication)$$

역원 연산기를 실제 회로로 설계하기 위해서는 위의 분석 결과를 토대로 승법 연산기의 지연과 제어 회로의 복잡도, 사용 가능한 device등을 고려해 응용 목적에 적합하도록 단위항을 분할하여 적용할 수 있을 것이다.

본 연구에서는 FPGA device를 사용해 속도면에서 최적화한 역원 연산기를 설계하기 위해, 비교 검토한 결과를 바

탕으로 단위항을 아래와 같은 형태로 분할한다.

$$U_x = (\textit{Feedback term}) \cdot (2^{s_x}+1) \cdot 2^{s_{x-1} \bmod 2} + (s_{x-1} \bmod 2)$$

유도한 단위항을 계산할 수 있는 연산기를 설계하고 이 연산기를 반복적으로 사용하면 역원연산을 수행할 수 있다. 기저체 F_{2^m}상에서 역원 연산기의 하드웨어 구현을 위해 유도한 식은 다음과 같다.

$$\begin{aligned} U_n &= (2^{s_n}+1) \cdot 2^{s_{n-1} \bmod 2} + (s_{n-1} \bmod 2) \\ U_{n-1} &= U_n \cdot (2^{s_{n-1}}+1) \cdot 2^{s_{n-2} \bmod 2} + (s_{n-2} \bmod 2) \\ &\vdots \\ U_0 &= U_1 \cdot 2 \end{aligned}$$

$$\textit{where, } n = \lfloor \log_2(m-1) \rfloor, s_i = \lfloor \frac{m-1}{2^i} \rfloor \quad \text{for } 0 \le i \le n.$$

위와 같이 단위항을 유도하면 역원연산의 결과는 $a^{-1} = a^{U_1}$ 가 된다. 각각의 단위항 $U_x(0 \le x \le n)$는 병렬로 수행할 연산을 의미하며 회로로 구현시 2개의 multiplication 회로, 3개의 rotation 회로, 3개의 MUX, 1개의 register로 구성된다. 각각의 비트 사이즈는 m에 의해 결정되며, 역원을 계산하는 데 걸리는 시간은 U_x항이 n개이므로 총 $n \times (\textit{clock cycle for one multiplication})$이 소요된다. 예를 들어, 기저체가 $F_{2^{155}}$인 경우 역원은 $7 \times (\textit{clock cycle for one multiplication})$에 계산될 수 있다. 유도한 식을 바탕으로 설계한 역원 연산기의 구조를 그림 2에 나타내었다.

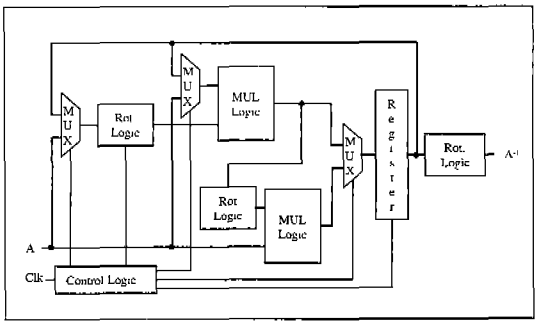

그림 2 F_{2^m}에 대한 역원 연산기의 구조

본 논문에서 제안한 방식으로 설계한 역원 연산기의 속도를 레퍼런스 [7]과 비교하면 다음과 같다.

- Reference [7]

Assume $m-1 = gh$.

$$a^{-1} = a^{2^m-2} = a^{2(2^{m-1}-1)} = \gamma^{(2^{g-1})(\sum_{i=0}^{h-1} 2^{gi})}$$

$$\textit{where } \gamma = a^2$$

If $F_{2^{155}}$,

Inversion: 23 multiplications are required.

Reqd. time: 23×(clock cycle for one multiplication)

- 본 논문

If $F_{2^{155}}$, then $n=7$.

U_7 : 1 *multiplication*
U_6 : 1 *multiplication*
U_5 : 2 *multiplications*
U_4 : 2 *multiplications*
U_3 : 1 *multiplication*
U_2 : 2 *multiplications*
U_1 : 1 *multiplication*
U_0 : 0 *multiplication*

Hence,

Inversion: 10 multiplications are required.

Reqd. time: 7×(clock cycle for one multiplication)

위의 비교 결과는 역원연산을 위한 승법연산 횟수와 연산 수행 시간만을 고려한 것인데, 승법연산 수행시간은 단순한 구현 방식의 차이이므로 반영하지 않았고 상위의 수식 단계에서 비교한 결과이다. 분석해 보면, 10번의 승법연산이 필요하며, 하드웨어 구현을 위한 적절한 단위항 분할의 결과로서, 소요되는 시간은 승법연산의 횟수보다 작은 7단위 시간 이다. 사실 승법 연산기를 병렬적으로 구현해서 발생하는 수행시간의 이득은 당연한 결과이나, 본 논문에서 제안한 방식에 따라 단위항을 유도하여 역원 연산기를 구현하면 승법 연산기의 속도에 관계없이 역원연산의 수행시간은 $\lfloor \log_2(m-1) \rfloor \times (\textit{clock cycle for one multiplication})$ 이 된다.

이러한 결과는 약 15,000 gate 정도의 FPGA device를 대상으로 최적화 한 것이므로, 본 논문에서 제안한 방식을 사용하여 다른 device와 응용에 적합하도록 단위항을 유도한다면 보다 나은 결과를 얻을 수 있다.

6. FPGA Implementation

제안한 연산기 구조의 성능 검증을 위해 기저체를 $F_{2^{155}}$로 하여 연산기를 디자인하고 이를 VHDL로 기술한 뒤, Xilinx Foundation Series 2.1i 환경에서 VIRTEX V1000FG680에 targeting 하여 합성한다. 연산기의 구조는 그림 4에 나타나 있는 코어부분과 같으며, 블록별로 설계하고 합성을 통해 검증한 뒤 최종적으로 전체 코어를 하나의 entity로 결합한다. 타이밍 시뮬레이션 결과를 그림 3에 나타낸다.

660ns 지점에서 multiplication 결과가 출력됨을 확인할 수 있고, Inversion 수행 명령이 입력된 1.28us 에서

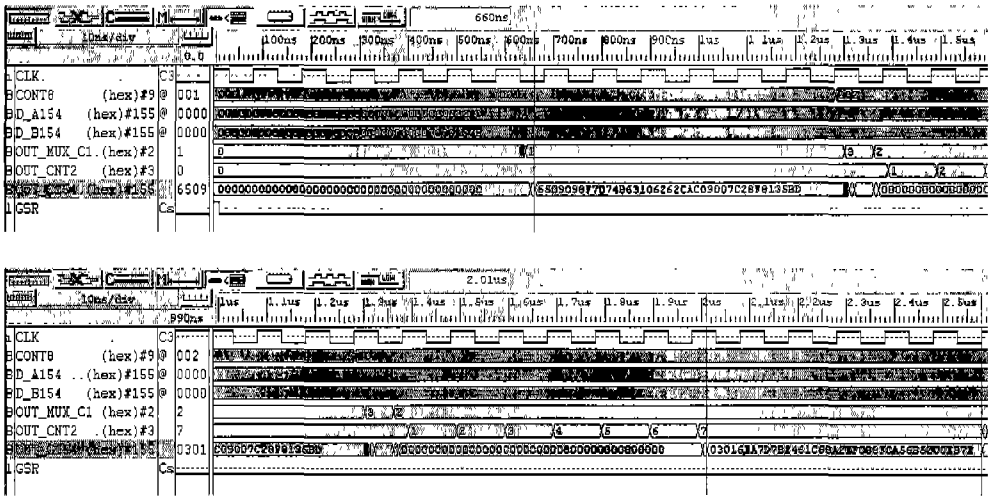

그림 3 FPGA 타이밍 시뮬레이션

부터 8클럭 후에 그 결과가 출력되고 있다. 역원 연산을 수행하는데 1클럭이 더 소요된 이유는 역원값이 출력을 위한 레지스터를 거치기 때문이다.

시뮬레이션 결과, [5] 및 [6]에서 구현된 소프트웨어에 근거하여 올바른 연산결과가 출력됨을 확인할 수 있다. 표 2는 레퍼런스 [7]과 연산기의 성능을 비교한 결과이다. 단, 동작 주파수는 10MHz로 동일하다.

표 2 타 논문과의 성능비교

	Target Device	Gate Count	Multiplication Speed	Inversion Speed
Ref.[7]	XC4020XL	17,000	157clk	3887clk
본 논문	V1000FG680	150,000	1clk	8clk

비교 결과 gate count는 약 8.8배가 증가했으나, 승법연산 속도는 약 150배, 역원연산 속도는 약 480배로 빨라졌다. 이것은 병렬성을 적용함으로서 당연히 얻어지는 trade-off를 능가하는 성능으로, 본 논문에서 제안한 구조의 효율성을 입증하는 결과이다.

7. VLSI Implementation

디자인한 연산기를 삼성 0.5um SOG 공정을 사용해 칩으로 제작하기 위해서 SADAS 라이브러리로 합성하였다. 최상위 단계의 블록 다이어그램은 그림 4와 같다.

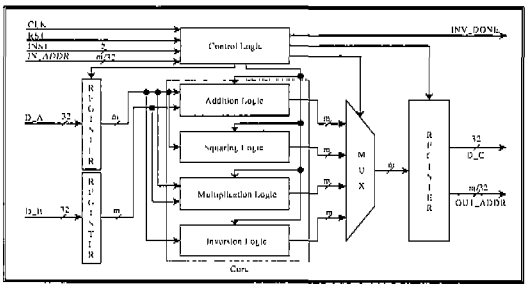

그림 4 최상위 단계의 블록 다이어그램

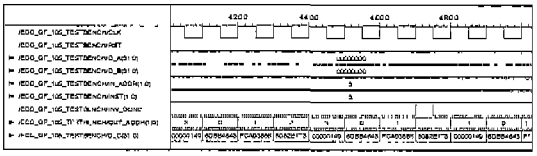

그림 5 기능 시뮬레이션

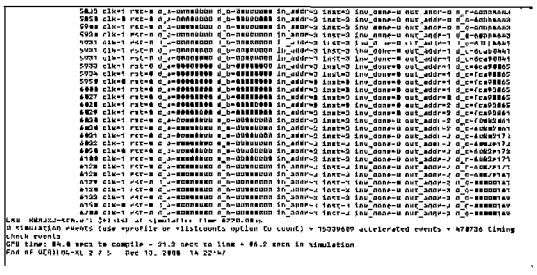

그림 6 레이아웃 전단계의 시뮬레이션

VERILOG-XL을 사용하였으며, 시뮬레이션 동작 주파수는 20MHz이다. [5] 및 [6]에서 구현된 소프트웨어에 근거하여 올바른 연산결과가 32bit로 분할되어 출력됨을 확인하였다.

8. Conclusion

본 논문에서는 타원곡선 암호시스템을 구현하기 위한 일환으로 타원곡선 상의 점을 고속으로 연산할 수 있는 전용의 기저체 연산기의 구조를 제안하고 실제 구현을 통해 그 기능을 검증하였다. 또한, 기저체 연산의 면밀한 분석을 통해 역원 연산기의 하드웨어 구현을 위한 효율적인 방법론을 제시하였으며, 이를 바탕으로 현실적인 제한 조건하에서 구현 가능한 수준의 게이트 수를 가지는 고속의 역원 연산기 구조를 제안하였다.

그 결과 기존 논문에 비해 게이트 수는 약 8.8배가 증가했으나, 승법연산 속도는 약 150배, 역원연산 속도는 약 480배로 빨라졌다. 이것은 병렬성을 적용함으로서 당연히 얻어지는 속도면에서의 이득을 능가하는 성능으로, 본 논문에서 제안한 구조의 우수성을 입증하는 결과이다. 실제로, 승법 연산기의 속도에 관계없이 역원연산의 수행시간은 $\lfloor \log_2(m-1) \rfloor \times (clock\ cycle\ for\ one\ multiplication)$으로 최적화가 되며, 제안한 구조는 임의의 유한체 F_{2^m}에 적용 가능하므로, 유한체를 사용하는 다양한 응용분야로 확장할 수 있다.

설계한 기저체 연산기는 IDEC에서 주관한 제11회 MPW 설계 공모전에 선정되어 삼성 0.5um SOG 공정으로 칩 제작 중에 있으며, 현재 레이아웃 과정을 진행하고 있다.

제안한 전용의 연산기는 암호 프로세서 설계의 기초자료로 활용되거나, 타원곡선 암호 시스템 구현시 직접 co-processor 형식으로 임베드 되어 사용할 수 있을 것으로 사료된다.

향후 연구과제는 연산기의 게이트 수를 최적화하고 연산속도와의 trade-off를 보다 정밀하게 해석하여 보다 효율적인 기저체 연산기를 설계하는 것과, 구현한 연산기를 암호 시스템에 실제로 응용하는 것이다. 그리고 타원곡선 암호 시스템의 다른 응용분야에 관해서도 연구가 필요하다고 사료된다.

참 고 문 헌

[1] Working Draft IEEE P1363, Part 6.

[2] http://www.kordic.re.kr/

[3] http://www.ctimesi.com/

[4] Neal Koblitz, "A Course in Number Theory and Cryptography." Springer-Verlag pub., 1994.

[5] Michael Rosing, "Implementing Elliptic Curve Cryptography." Manning pub., 1999.

[6] Certicom ECC Whitepaper, 1999.

[7] Sarwono Sutikno, Ronny Effendi, Andy Surya, "Design and implementation of arithmetic processor $F_{2^{155}}$ for elliptic curve cryptosystems." IEEE APCCAS, 1998, Pages: 647-650.

[8] Robert J., McEliece, "Finite Fields for Computer Scientists and Engineers." Kluwer Academic pub., 1989.

[9] Sutikno, S., Surya, A., Effendi, R., "An implementation of ElGamal elliptic curves cryptosystems." Circuits and Systems, 1998. IEEE APCCAS 1998. The 1998 IEEE Asia-Pacific Conference on, 1998, Pages: 483-486.

[10] 조성제, 권용진, "타원곡선 암호시스템을 위한 기저체 연산기의 FPGA 구현", 대한전자공학회 추계종합학술발표대회 논문집 II, 2000. 11, Pages: 148-151.

조 성 제

1976년 11월 26일생. 1999년 2월 한국항공대학교 항공통신정보공학과 졸업. 1999년 3월 ~ 현재 한국항공대학교 항공통신정보공학과 대학원 석사과정 재학중. 관심분야는 논리회로 설계 및 합성, 정보보호, 암호이론

권 용 진

1964년 6월 7일생. 1986년 2월 한국항공대학교 항공전자공학과 졸업. 1990년 3월 일본 쿄도대학 대학원 정보공학과 졸업(공학석사). 1994년 3월 일본 쿄도대학 대학원 정보공학과 졸업(공학박사). 1994년 3월 ~ 현재 한국항공대학교 전자정보통신컴퓨터공학부 부교수. 관심분야는 정보 보호, 논리회로 설계 및 합성, 알고리즘 개발