

ECC(Elliptic Curve Crptographics) 기반의 암호프로세서를 위한 스칼라 곱셈기의 FPGA 구현

황 정 태, *김 영 철

전남대학교 전자정보통신공학과, *전남대학교 전자컴퓨터정보통신공학부
전화 : 062-530-0369 / 핸드폰 : 011-637-9971

Design and FPGA Implementation of Scalar Multiplication for A CryptoProcessor based on ECC(Elliptic Curve Cryptographics)

Jeong-Tae Hwang, Young-Chul Kim

Dept. of Electronics and Information Communication ENG, Chonnam National University

E-mail : hwang@neuron.chonnam.ac.kr

Abstract

The ECC(Elliptic Curve Cryptogrphics), one of the representative Public Key encryption algorithms, is used in Digital Signature, Encryption, Decryption and Key exchange etc. The key operation of an Elliptic curve cryptosystem is a scalar multiplication, hence the design of a scalar multiplier is the core of this paper. Although an Integer operation is computed in infinite field, the scalar multiplication is computed in finite field through adding points on Elliptic curve.

In this paper, we implemented scalar multiplier in Elliptic curve based on the finite field $GF(2^{163})$. And we verified it on the Embedded digital system using Xilinx FPGA connected to an EISC MCU. If my design is made as a chip, the performance of scalar multiplier applied to Samsung 0.35 μm Phantom Cell Library is expected to process at the rate of 8kbps and satisfy to make up an encryption processor for the Embedded digital doorphone.

I. 서론

타원곡선 암호시스템은 다른 공개키 암호 시스템보다 적은 키길이로 고속 연산을 한다. 하지만 정보보호 기술의 발달로 높은 암호강도가 요구되어 긴 길이의 키를 필요로 하고 있다. 긴 키 길이의 타원 곡선 암호

시스템은 많은 연산량으로 인해 고속 동작이 어렵게 된다[1]. 그러므로 타원곡선 암호 시스템의 고속 동작을 위해 최적의 알고리즘 및 설계가 필요하다. 타원곡선 암호는 최상위 연산인 스칼라 곱셈과 하위 연산인 유한체 연산으로 이루어진다. 최상위 연산인 스칼라 곱셈은 m비트로부터 점 덧셈 연산과 두 배점 연산을 수행하는데 전체 연산에 미치는 영향이 크다.

본 논문에서는 두배점 연산을 이용하여 타원곡선 연산을 수행하도록 최소한의 내부 연산을 하기 위해 MSR방식의 유한체 곱셈기를 적용하였다. 최적의 유한체연산을 위해 MSR(modified shift register)을 이용한 곱셈 연산과 한번의 연산으로 역원을 구할 수 있는 확장 Euclidean 알고리즘으로 역원 연산기를 설계하였고, 유한체 직렬 곱셈기의 레지스터 구조를 이용한 제공기를 설계하였다[2][3].

II. 타원곡선 암호 알고리즘

타원곡선 암호 알고리즘은 타원곡선 이산 로그 문제(ECDLP: Elliptic Curve Discrete Logarithm Problem)에 근간을 두고 있다[4]. ECDLP는 타원곡선상의 임의의 한 점 P에 정수 K를 곱한 값이 $Q = KP$ 일 때, 점 Q와 P를 알고 있더라도 정수K를 계산하기 어려움을 나타낸다. 따라서 타원곡선 암호시스템을 구현하는데 필요한 핵심연산은 스칼라 곱셈, 즉 $Q = KP$ 를 구하는 것이다. 소수체(Prime Field)인 $GF(p)$ 와 유한체인 $GF(2^m)$ 상에서 모두 스칼라 곱셈이 정의되나, 유한체 덧셈연산에서는 캐리가 발생하지 않아 소수체보다 하

드웨어 구현이 용이하다는 장점 때문에 대부분 GF(2^m)상에서의 연산을 사용하며, 본 논문에서도 GF(2^m)상에서 스칼라 곱셈을 계산할 수 있는 연산기를 하드웨어로 구현하였다.

스칼라 곱셈이 이루어지는 타원곡선은 GF(2^m)상에서 (x,y)인 점들로 구성되어지고, 타원곡선은 y² + xy = x³ + ax² + b 의 형태를 가진다. 여기서 a,b ∈ GF(2^m), b≠0이다. 타원 곡선 위의 점들은 점 덧셈 연산에 대해서 군을 이루고, 무한 원점 O는 이 덧셈에 대한 가환군의 항등원이 된다. GF(2^m) 상에서 정의된 타원 곡선 군은 유한 개의 원소를 가지게 되고 반올림에 따른 오차가 없기 때문에 이진 컴퓨터 연산에 많이 쓰이게 된다. P와 Q를 타원 곡선 E 위의 두 점이라 하면 아래 같은 연산 공식들이 성립한다 [5].

[GF(2^m) 상의 점 덧셈 연산 알고리즘]

- ⊙ P=(x₁, y₁)와 Q=(x₂, y₂)는 타원 곡선 위의 두 점
- ⊙ 둘 중 하나가 무한 원점이면, 덧셈 결과는 나머지 한 점이다.
- ⊙ 만일 P = Q이면, 두배점 연산을 사용한다.
- ⊙ P≠Q이면, P+Q = R(x₃, y₃)이고, x₃, y₃의 값은 다음과 같다.

$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a,$$

$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1,$$

$$\lambda = (y_1 + y_2) / (x_1 + x_2)$$

[GF(2^m) 상의 두배점 연산 알고리즘]

- ⊙ P(x₁, y₁) = Q(x₁, y₁)는 타원 곡선 위의 같은 한 점
- ⊙ 만일 x₁ = 0이면, 결과 값 2P는 무한 원점 O이다.
- ⊙ 만일 x₁ ≠ 0이면, 결과 값은 2P(x₁, y₁) = R(x₃, y₃) 이고, x₃, y₃의 값은 다음과 같다.

$$x_3 = \lambda^2 + \lambda + a, y_3 = x_2$$

$$1 + (\lambda + 1)x_3, \lambda = (x_1 + y_1 / x_1)$$

[GF(2^m)상의 점 역원 연산 알고리즘]

- ⊙ P(x₁, y₁)는 타원 곡선 위의 한 점
- ⊙ - P(x₁, y₁) = R(x₃, y₃) 이고, x₃, y₃의 값은 다음과 같다.

$$(x_3, y_3) = -(x_1, y_1) = (x_1, x_1+y_1)$$

위 공식들을 살펴보면 연산 알고리즘을 처리하는 데 필요한 유한체 연산이 얼마나 요구되는지 계산할 수 있다[6]. 점 덧셈 연산에서는 8번의 유한체 덧셈, 1번의 유한체 곱셈, 1번의 유한체 역원 연산, 1번의 유한체 제곱 연산이 필요하다는 것을 알 수 있고, 두배점 연산에서는 4번의 유한체 덧셈, 1번의 유한체 곱셈, 2번의 유한체 제곱 연산, 그리고 1번의 유한체 역원 연산이 필요하다. 이를 종합하여 타원곡선 암호시스템의 연산과정을 크게 네부분으로 나누어 도식하면 아래 그림 1 과 같다.

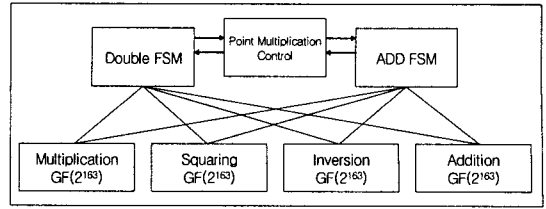


그림 1. 스칼라 곱셈기의 연산 구조

III. 타원곡선 암호시스템의 스칼라 곱셈

스칼라 곱셈을 한번 수행할때는 여러번의 덧셈과 두배점 연산이 필요하다. 여기에서는 일반적인 Double and Add 방식과 Binary NAF 방식 두 가지를 비교한다.

3.1 Double and Add 방식

m비트의 최상위비트부터 차례로 읽으면서 두배점 연산과 점 덧셈 연산을 수행하는 방식으로 최소 m-1번의 두배점 연산에 더하여 k를 이진수로 표현했을때의 hamming weight만큼의 점 덧셈연산이 필요하다[7].

- ⊙ 점 덧셈 연산 : add(), 두배점 연산 : double()

```

kP : k = ∑ bi2i (bi ∈ {0,1})
      P: =P(x1, y1)      Q: =P
      for i from m-1 downto 0 do
          Q: =double(Q)
          if bi = 1 then
              Q: =add(P, Q)
          end (Q = kP)
    
```

그림 2. Double and Add 방식

3.2 Binary NAF(Non Adjacent Format) 방식

k의 hamming weight가 필요한 연산량을 결정하므로 k의 1의 개수를 줄여서 연산을 수행하는 방법이다. 하지만 k를 NAF형태로 미리 바꿔야 하는 단점을 가진다[8].

```

NAF(k) = ∑ ki · 2i
kP : Q: =0
      for i from t-1 downto 0 do
          Q: =2Q
          if ki = 1 then
              Q: =Q + P
          if ki = -1 then
              Q: =Q - P
          end (Q = kP)
    
```

그림 3. Binary NAF 방식

본 논문에서는 Double and ADD 방식을 적용하여 점 덧셈연산을 수행하도록 하였다.

IV. 타원곡선 암호시스템에서의 유한체 연산

4.1 유한체 곱셈기

유한체 곱셈기는 MSR(Modified Shift Register)구조를 갖고 그림 4와 같은 회로로 구성된다. Z레지스터는 $M \times M$ 의 배열구조를 갖고 A로 초기에 로드되어지는 feedback 다항식 $P(x)$ 와 함께 Galois 형태의 연속된 상태의 m 를 나타낸다. 곱은 결과인 A와 계산한 b_0Z_{-0} 을 MSR에 첫 번째 로딩에서 포함시키고 m 번째 레지스터의 결과를 저장한다. 다음 클럭에 MSR은 b_1Z_{-0} 에 더하고 결과의 곱을 저장한다. m 클럭 사이클 뒤에 결과는 더 낮은 레지스터에서 사용할 수 있다.

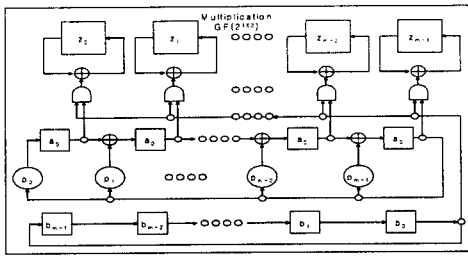


그림 4. $GF(2^m)$ 상에서 유한체 곱셈기 구조

MSR의 구조는 규칙적이고 단순한 형태의 회로로 구현되는 장점이 있으며 성능측면에서는 기존 직렬 곱셈기보다 개선된 결과를 보인다[9].

4.2 유한체 역원기

유한체 역원기는 다항식의 계수를 이진 수열로 받아 모듈라 곱셈에 대한 역원을 계산하여 출력하는 기능을 갖는다. 역원을 구하는 알고리즘 가운데 유클리드 알고리즘을 비롯하여 여러 가지가 사용되고 있다 유클리드 알고리즘 가운데 특히 확장 유클리드 알고리즘은 회로의 면적과 처리속도에서 뛰어난 성능을 보인다[10].

Repeat the following steps while $\deg(f) \neq 0$:
 If $\deg(F) < \deg(G)$, then
 exchange F, B with G, C, respectively.
 Update F and B as follows(let $j = \deg(F) - \deg(G)$)
 $a \leftarrow F_{\deg(F)}G^{-1}\deg(G)$,
 $F \leftarrow F - axjG$,
 $B \leftarrow B - axjC$

그림 5. 확장 유클리드 알고리즘

V. 스칼라 곱셈기 구현과 FPGA 검증

스칼라 곱셈기의 전체 구조는 그림 6과 같이 레지스

터, 유한체 곱셈기, 역원기, 점 덧셈연산 FSM, 두배점 연산 FSM, 컨트롤 블록으로 구성된다.

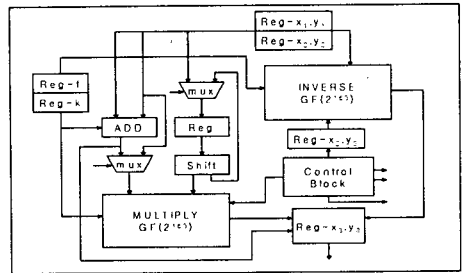


그림 6. 스칼라곱셈기의 FPGA 블록도

VHDL로 구현한 스칼라 곱셈기를 synopsys VCS에서 검증하였다[11]. 그림 7은 $m=163$ 일때 구현값으로 m 클럭후에 연산된 값이 출력됨을 알 수 있다.

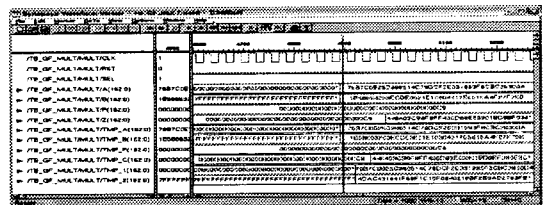


그림 7. 스칼라 곱셈기의 시뮬레이션 결과 값

본 논문에서는 구현한 하드웨어 모듈을 이용한 실제 시스템상에서의 SoC 검증을 위하여, EISC MCU 기반의 임베디드 개발보드인 IFC-ETK100보드와 연동가능한 Xilinx 100만게이트급 FPGA보드를 구성 및 제작하였다[12]. 그림 8은 IFC-ETK100보드와 연동하기 위한 Xilinx FPGA보드의 구성도를 보여준다. Xilinx FPGA보드는 임베디드 개발보드에 에드온하는 형식으로 탑재되어 동작하게 되고 개발보드와 임베디드 보드가 서로 연동하여 동작하는 것을 그림 9와 같이 직렬 통신을 이용하여 하이퍼 터미널상에서 확인하였다[13].

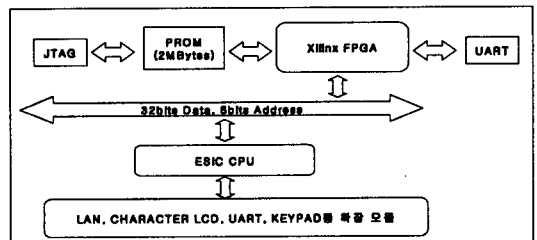


그림 8. FPGA 검증을 위한 시스템 구성도



그림 9. 하이퍼 터미널상에서의 FPGA 검증

VI. 결론

스마트카드나 임베디드 디지털 도어폰처럼 저가의 Embedded-system을 사용하는 휴대용 보안기기에는 하드웨어 리소스가 작은 암호프로세서가 필수적이다. 따라서, 본 논문에서는 타원곡선 암호알고리즘을 이용한 암호프로세서를 구현하는데 근간인 스칼라 곱셈기를 구현하였다. Xilinx Virtex-1000E-HQ240-6c FPGA에서는 동작속도는 약 80MHz로 동작하였으며, 이는 임베디드 디지털 도어폰시스템에서 요구하는 55MHz 이상의 동작속도를 보장하고 있다.

Synopsys상에서의 합성결과에 따르면 클럭 속도를 결정하는 최장 지연 경로(Critical Path)는 MUX와 AND 게이트, XOR 게이트 그리고 레지스터로 이루어지며, 이는 삼성 0.35um Phantom Cell Library를 사용했을 때 Data arrival time이 약 1.25ns가 걸린다. 이를 근거로 GF(2¹⁶³)에서 163비트 곱셈을 계산하는데 약 8kbps의 처리 속도가 예상된다.

본 논문에서 구현한 스칼라 곱셈기는 작은 면적을 요구하는 smart card, 휴대 단말기, 정보가전, 임베디드 시스템등에서 요구되는 타원곡선 암호프로세서의 내부 연산기로서 효과적으로 사용될 것으로 기대된다. 또한 구현한 곱셈기를 기반으로 ECC암호프로세서가 보안 시스템 칩으로 구현된다면 현재의 여러 시스템에서 유용하게 사용될것으로 기대된다.

참고문헌(또는 Reference)

[1] Don Johnson and Alfred Menezes, "The Elliptic Curve Digital Signature Algorithm (ECDSA)", Technical Report CORR 99-31, Dept. of C&O, Univ. of Waterloo, Canada, August 1999.
 [2] Lijun Gao, Sarvesh Shrivastava, and Gerald E.Sobelman, "Elliptic Curve Scalar Multiplier Design Using FPGAs", Workshop on Cryptographic Hardware and Embedded Systems(CHES), August 1999.

[3] Yongjin Jeong and Wayne Burleson, "VLSI Synthesis of Finite Field Arithmetic", TR-91-CSE-22, Dept. of ECE. Univ. of Massachusetts. 1991.
 [4] IEEE P1363a / D5(Draft Version 5), Standard Specifications for Public key Cryptography: Additional Techniques, August 16 2000.
 [5] ECC Tutorial, http://www.certicom.com/resources/ecc_tutorial/ecc_tutorial.html, 2001.
 [6] William Stallings, Cryptography and Network Security, Prentice Hall, 1999.
 [7] D. Hankerson, J. L. Hernandez, and A. Menezes, "Software Implementation of Elliptic Curve Cryptography over Binary Fields,"Crypto95.
 [8] 문상국, "타원 곡선 암호용 프로세서를 위한 고속 VLSI 알고리즘의 연구와 구현," 연세대학교 전기전자공학과 박사학위 논문, 2001, 12.
 [9] E.R.Berlekamp, "Algebraic Coding Theory" New York: McGraw-Hill, 1998
 [10] Lee Kwang Yub, "Development of Crypto processor IP using Elliptic logarithm" SeoKyung University, 2002
 [11] Synopsys Manual, "<http://solvnetsynopsys.com>"
 [12] Xilinx FPGA Manual, "<http://www.xilinx.com>"
 [13] INTER-FC, "IFC-ETK100 manual", 2003

※ 본 논문은 "정보통신부 정보통신연구진흥원에서 지원하고 있는 정보통신기초기술연구지원사업"와 "IDEC의 CAD툴 지원사업"의 연구결과임