ISSN : 1226-7244 (Print)
ISSN : 2288-243X (Online)
논문번호 19-04-30
206
j.inst.Korean.electr.electron.eng.Vol.23,No.4,1321～1327,December 2019
http://dx.doi.org/10.7471/ikeee.2019.23.4.1321

GF(p) 상의 제곱근 연산의 효율적인 하드웨어 구현
An Efficient Hardware Implementation of Square Root Computation over GF(p)

최 준 영*, 신 경 욱*★

Jun-Yeong Choe*, Kyung-Wook Shin*★

Abstract

This paper describes an efficient hardware implementation of modular square root (MSQR) computation over $GF(p)$, which is the operation needed to map plaintext messages to points on elliptic curves for elliptic curve (EC)-ElGamal public-key encryption. Our method supports five sizes of elliptic curves over $GF(p)$ defined by the National Institute of Standards and Technology (NIST) standard. For the Koblitz curves and the pseudorandom curves with 192-bit, 256-bit, 384-bit and 521-bit, the Euler's Criterion based on the characteristic of the modulo values was applied. For the elliptic curves with 224-bit, the Tonelli-Shanks algorithm was simplified and applied to compute MSQR. The proposed method was implemented using the finite field arithmetic circuit with 32-bit datapath and memory block of elliptic curve cryptography (ECC) processor, and its hardware operation was verified by implementing it on the Virtex-5 field programmable gate array (FPGA) device. When the implemented circuit operates with a 50 MHz clock, the computation of MSQR takes about 18 ms for 224-bit pseudorandom curves and about 4 ms for 256-bit Koblitz curves.

요 약

본 논문에서는 $GF(p)$ 상에서 모듈러 제곱근 (MSQR) 연산의 효율적인 하드웨어 구현에 대해 기술한다. MSQR 연산은 타원곡선 기반의 EC-ElGamal 공개키 암호를 위해 평문 메시지를 타원곡선 상의 점으로 매핑하기 위해 필요하다. 본 논문의 방법은 NIST 표준으로 규정된 5가지 크기의 $GF(p)$ 타원곡선을 지원하며, 192-비트, 256-비트, 384-비트 그리고 521-비트 크기의 Kobliz 곡선과 슈도 랜덤 곡선들은 모듈러 값의 특성을 기반으로 오일러 판정법을 적용하고, 224-비트 크기의 경우에는 Tonelli-Shanks 알고리듬을 간략화시켜 적용하였다. 제안된 방법을 ECC 프로세서의 32-비트 데이터 패스를 갖는 유한체 연산회로와 메모리 블록을 이용하여 구현하였으며, FPGA 디바이스에 구현하여 하드웨어 동작을 검증하였다. 구현된 회로가 50 MHz 클록으로 동작하는 경우에, 224-비트 슈도 랜덤 곡선의 경우에는 MSQR 계산에 약 18 ms가 소요되고, 256-비트 Kobliz 곡선의 경우에는 약 4 ms가 소요된다.

Key words : ECC, EC-ElGamal, Modular square root, Euler's Criterion, Tonelli-Shanks algorithm

---

* School of Electronic Engineering, Kumoh National Institute of Technology
★ Corresponding author
E-mail : kwshin@kumoh.ac.kr, Tel : +82-54-478-7427
※ Acknowledgment
- This research was supported by the KIAT(Korea Institute for Advancement of Technology) grant funded by the Korea Government(MOTIE : Ministry of Trade Industry and Energy). (No. N0001883, HRD Program for Intelligent semiconductor Industry)
- This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. 2017R1D1A3B03031677)
- Authors are thankful to IDEC for supporting EDA software.

Manuscript received Dec. 10, 2019; revised Dec. 26, 2019; accepted Dec. 27, 2019.

Ⅰ. 서론

타원곡선 암호(elliptic curve cryptography: ECC) [1]는 RSA, Diffie-Hellman (DH)과 같은 기존의 공개키 암호 방식에 비해 짧은 키 길이를 사용하면서도 유사한 보안 강도를 얻을 수 있고, 빠른 연산 속도, 적은 메모리 요구량 등의 장점들을 가져 대표적인 공개키 암호 방식으로 자리 잡아 가고 있다. ECC 기반의 공개키 암호 체계는 드론, 자율주행 자동차 등의 통신 네트워크 보안과 블록체인의 트랜잭션 검증을 위한 전자서명/검증 등에 필수적으로 사용되고 있다.

ECC를 기반으로 하는 공개키 암호 프로토콜들이 정보보안 표준으로 채택되고 있으며, 전자서명/검증을 위한 EC-DSA(Elliptic Curve Digital Signature Algorithm) [2], 키교환을 위한 EC-DH(Elliptic Curve Diffie Hellman) [3], 대칭키 암호와 ECC를 결합한 ECIES (Elliptic Curve Integrated Encryption Scheme) [4], 메시지 암호/복호와 전자서명을 위한 EC-ElGamal [5] 등이 다양한 분야에 사용되고 있다. EC-ElGamal은 엘가말 (ElGamal) 공개키 암호 [6]를 타원곡선 암호 기반으로 구현한 방식이다. 엘가말 암호는 이산대수 문제(discrete logarithm problem: DLP)를 안전성의 근거로 하는 최초의 공개키 암호방식이며, 전자서명과 메시지 암호/복호를 위해 사용된다.

EC-ElGamal에 의한 공개키 암호를 위해서는 암호화될 평문 메시지를 EC 상의 한 점으로 매핑 해야 한다. 평문 메시지 m을 EC 상의 한 점 $P_m(x,y)$로 매핑하기 위해서는 m을 타원곡선 방정식의 x에 대입하여 y^2를 계산하고, 모듈러 제곱근(modular square root: MSQR) 연산을 통해 EC 상의 y 좌표값을 계산해야 한다. 실수계(real number) 연산과는 다르게, 정수계(integer number)에서는 제곱수가 존재하는 이차잉여(quadratic residue: QR)인 경우에만 제곱근을 구할 수 있다. 평문 메시지 m을 EC 상의 한 점으로 매핑하는 방법과 MSQR 계산을 위한 방법들이 제안되고 있으며[7-9], 소프트웨어로 구현한 사례들은 발표되고 있으나 하드웨어 구현 사례는 없는 것으로 조사되었다.

EC-ElGamal 공개키 암호의 하드웨어 구현을 위해서는 MSQR 연산의 효율적인 하드웨어 구현이 필요하며, 본 논문에서는 $GF(p)$ 상의 MSQR 연산의 효율적인 하드웨어 구현 방법을 제안하고, 회로 구현과 검증 결과에 대해 기술한다. Ⅱ장에서는 $GF(p)$ 상의 MSQR 계산을 위한 수학적인 방법에 대해 간략히 소개하고, Ⅲ장에서는 오일러 판정법(Euler's criterion)과 Tonelli-Shanks 알고리듬을 적용하여 MSQR 계산을 하드웨어로 구현하는 방법을 설명한다. Ⅳ장에서는 본 논문에서 제안된 방법을 FPGA 디바이스에 구현하여 하드웨어 동작을 확인한 검증 결과에 대해 기술하고, Ⅴ장에서 결론을 맺는다.

Ⅱ. $GF(p)$ 상의 MSQR 연산

ECC는 소수체 (prime field) $GF(p)$ 또는 이진체 (binary field) $GF(2^m)$ 상의 타원곡선 군 (group)을 기반으로 하며, $GF(p)$ 상의 타원곡선 EC는 식 (1)과 같이 정의된다. 타원곡선에 따라 생성점, 생성점의 위수 (order), 타원곡선 계수 (a,b), 모듈러 값 등의 도메인 파라미터가 표준으로 정의되어 있다.

$$EC: y^2 = x^3 + ax + b,\ (4a^3 + 27b^2 \neq 0) \tag{1}$$

메시지 m을 EC 상의 점 $P_m(x,y)$로 매핑하기 위해서는 $x=m$을 식 (1)의 타원곡선 방정식에 대입하여 y^2을 구한 후, y^2의 모듈러 제곱근 y를 구해야 한다. 먼저, 식 (2)를 만족하는 y를 찾는 문제를 생각한다.

$$y^2 \equiv n \pmod{p} \tag{2}$$

홀수인 소수 p와 정수 n에 대해, $\left(\frac{n}{p}\right)=1$인 경우에 법 (modulo) p에 대한 합동식 식 (2)의 해가 존재하며, 이 때의 n을 법 p에 대한 QR이라고 한다. 여기서 $\left(\frac{n}{p}\right)$은 QR과 이차비잉여(quadratic nonresidue: QNR)를 나타내는 Legendre 기호이며, 어떤 수가 QR이면 제곱근이 존재함을 의미한다. 식 (2)의 해를 구하는 방법은 법 p의 특성에 따라 달라진다.

1. 오일러 판정법 (Euler's Criterion) 적용

식 (2)의 해가 존재하여 $\left(\frac{n}{p}\right)=1$인 경우를 생각한다. $\gcd(n,p)=1$이므로 $\gcd(y,p)=1$이다. 페르마 소정리에 의해 $y^{p-1} \equiv 1 \pmod{p}$이므로, $n^{(p-1)/2} \equiv y^{p-1} \equiv 1 \pmod{p}$

가 되어 식 (3)의 관계가 성립한다.

$$\left(\frac{n}{p}\right) = 1 \equiv n^{(p-1)/2}\,(\bmod p) \qquad (3)$$

따라서 $p=4k+3$를 만족하는 경우에 오일러 판정법에 의해 식 (2)의 해는 $y \equiv n^{k+1}(\bmod p)$가 되므로, 식 (4)로 쉽게 구할 수 있다. [10, 11]

$$y \equiv n^{(p+1)/4}\,(\bmod p) \qquad (4)$$

NIST 표준 [2]에 규정된 소수체 타원곡선 중, 192-비트, 256-비트, 384-비트, 521-비트의 Kobliz 곡선과 슈도랜덤 곡선들은 $p=4k+3$을 만족하는 법 p를 가지므로, 식 (4)에 의해 식 (2)의 해인 모듈러 제곱근 y를 쉽게 구할 수 있다.

2. Tonelli-Shanks 알고리듬

NIST 표준에 규정된 224-비트 소수체 타원곡선은 법 p가 $p \equiv 1 \bmod 4$이므로, 식 (4)에 의해 MSQR을 계산할 수 없다. 식 (2)의 해를 구하기 위한 일반 해법으로 Tonelli-Shanks 알고리듬 [12]를 비롯한 다양한 방법들이 제안되고 있다[13, 14]. Tonelli-Shanks 알고리듬은 $GF(p)$에서 모듈러 지수승(exponentiation) 연산을 기반으로 해를 구한다.

본 논문에서는 NIST P-224 Kobliz 곡선과 슈도랜덤 곡선의 법 p에 대한 MSQR을 효율적으로 계산할 수 있도록 Tonelli-Shanks 알고리듬을 간략화하여 회로설계에 적용하였으며, 그림 1은 간략화된 Tonelli-Shanks 알고리듬의 슈도코드이다. 개략적인 연산과정은 다음과 같다. 단계-1에서는 $GF(p)$의 원소 x와 타원곡선의 계수들을 이용하여 z를 계산한다. 단계 2는 NIST P-244 Kobliz 곡선과 슈도랜덤 곡선인 경우 고정되는 값들을 입력하는 단계이며, 법 p가 일정한 값이기 때문에 이차 비잉여인 v와 $2^S \times Q = p-1$을 만족하는 정수인 S와 홀수인 Q는 특정한 값으로 고정되고 c와E 또한 $c = v^Q \bmod p$이고, $E=S$이므로 고정된 값이 된다. 단계-4부터 단계-10까지의 반복문은 $h = z^{(p-1)/2}$과 $t = z^Q$를 계산하는 단계이다. 이때 t는 단계-8의 조건을 만족할 경우 h를 계산하는 과정의 중간 결과값이 된다. $h \neq 1$이면 MSQR이 존재하지 않는 경우이며, 연산을 종료하고 새로운 x값을 기다린다. $h=1$이면 MSQR이 존재하는 경우이며, 단계-16으로 이동하여 이후의 연산과정을 수행하며, 최종 R 값이 MSQR이다.

```
    Input:   x, a, b, p
    Output:  R

 1:  z ← x^3 + ax + b
 2:  S, Q, c, E ← given values;
 3:  R_0 ← 1; R_1 ← z;
 4:  for i = m − 1 down to 1 do
 5:     R_0 ← R_0 × R_0 mod p;
 6:     if p_i = 1 then
 7:        R_0 ← R_0 × R_1 mod p;
 8:     else if i = S then
 9:        t ← R_0;
10:  end for
11:  h ← R_0;
12:  if h = 1 then
13:     go to step 16
14:  else
15:    Return 0
16:  R ← z^((Q+1)/2) mod p;
17:  if t = 1 then
18:     go to step 35
19:  else
20:     u ← t;
21:     for i = 1 up to E − 1 do
22:        u ← u × u mod p;
23:        if u = 1 then
24:           go to step 25
25:     d ← c^(2^(E−i−1)) mod p;
26:     R ← R × d mod p;
27:     if i = 1 then
28:        go to step 35
29:     else
30:        go to step 31
31:     c ← d^2 mod p;
32:     t ← d^2 × u mod p;
33:     E ← i;
34:     go to step 17
35:  Return R
```

Fig. 1. Pseudo code of simplified Tonelli-Shanks algorithm for NIST P-224 elliptic curves.
그림 1. NIST P-224 타원곡선을 위한 간략화된 Tonelli-Shanks 알고리듬의 슈도코드

III. $GF(p)$ MSQR 연산의 하드웨어 구현

본 논문에서는 그림 2의 ECC 프로세서 [15] 내부의 32-비트 데이터패스를 갖는 유한체 연산회로와 메모리를 이용하여 $GF(p)$ 상의 MSQR 연산을 하드웨어로 구현했다. DF_ALU은 32-비트 데이터패스로 유한체 연산을 수행하는 블록이며, Data_MEM은 32-비트×342 단일포트 RAM을 사용하며 유한체 연산의 중간결과 값과 유한체 곱셈에 사용되는 파라미터 값들을 저장한다. REGs 블록 내부의 ALU_Data에는 유한체 연산과정에 사용되는 데이

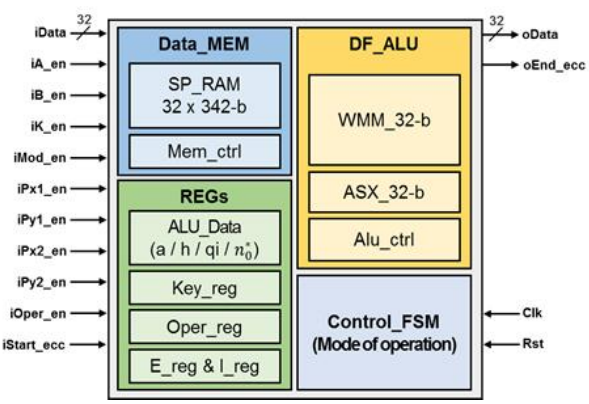
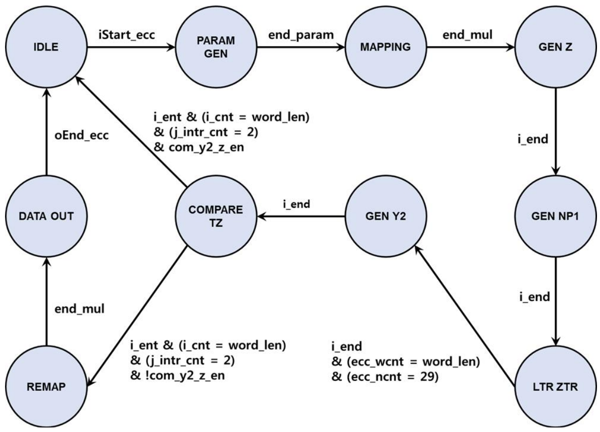


Fig. 2. Architecture of DF-ECC processor.
그림 2. DF-ECC 프로세서의 구조

Input : x, a, b, p
Output : R

1 : $z \leftarrow x^3 + ax + b$
2 : $\mathrm{R} \leftarrow z^{\frac{p+1}{4}} \bmod p;$
3 : $u \leftarrow R \times R \bmod p;$
4 : **if** $\mathrm{u} = \mathrm{z}$ **then**
5 :   **go to** step 8
6 : **else**
7 :   **go to** step 1
8 : **Return** R

(a) pseudo code



(b) stage transition diagram

Fig. 3. Modular square root computation in the case of $p \equiv 3 \bmod 4$.
그림 3. $p \equiv 3 \bmod 4$인 경우의 모듈러 제곱근 계산

터가 저장되고, Oper_reg는 타원곡선과 연산모드 관련 정보가 저장되며, E_reg와 I_reg에는 그림 1의 간략화된 Tonelli-Shanks 알고리듬 슈도코드의 E와 i 값이 저장된다.

NIST 표준에 규정된 소수체 타원곡선 중, 법 p가 $p \equiv 3 \bmod 4$인 타원곡선 크기 192-비트, 256-비트, 384-비트, 그리고 521-비트의 경우에는 식 (4)를 적용하여 모듈러 제곱근을 연산하고, $p \equiv 1 \bmod 4$인 키 길이 224-비트의 경우에는 그림 1의 간략화된 Tonelli-Shanks 알고리듬의 슈도코드를 적용하여 연산이 이루어지도록 설계하였다.

그림 3은 식 (4)에 의한 모듈러 제곱근 연산을 하드웨어로 구현하기 위한 슈도코드와 세부 연산과정의 동작 상태천이도이다. PARAM_GEN 상태에서는 합동(congruence) 연산에 사용되는 값 $-p_0^{-1} \bmod 2^{32}$ (단, p_0는 법 p의 최하위 워드)와 몽고메리 도메인으로 변환에 사용되는 R^2(단, $R=2^{fs}$, fs는 필드 크기를 나타내는 비트 수) 값을 생성한다. MAPPING 상태에서는 타원곡선 상의 점에 R^2을 곱해서 몽고메리 도메인으로 변환이 이루어지며, GEN_Z 상태에서는 슈도코드의 단계-1에 해당하는 연산이 수행된다. GEN_NP1 상태에서는 슈도코드의 단계-2에서 사용될 $p+1$을 계산하며, LTR_ZTR 상태에서는 슈도코드 단계-2의 멱승 연산이 진행된다. 멱승 연산은 left-to-right (LR)이진 알고리듬을 이용하여 계산된다. GEN_Y2 상태에서는 단계-3 연산이 수행되고, 이 값을 COMPARE_TZ 상태에서 기존의 z값과 비교한다. 두 값이 일치하면 REMAP 상태로 이동하여 몽고메리 도메인 결과를 일반 도메인으로 역변환하고, 그렇지 않은 경우에는 IDLE 상태로 돌아가 새로운 x값을 입력받을 때까지 대기한다. 마지막으로, DATA_OUT 상태에서는 역변환 과정을 거친 최종결과 값을 외부로 출력하며, 출력이 완료된 후에는 IDLE 상태로 돌아간다.

그림 4는 그림 1의 간략화된 Tonelli-Shanks 알고리듬을 하드웨어로 구현하기 위한 세부 연산과정의 동작 상태천이도이다. IDLE 상태에서부터 GEN_Z 상태까지는 그림 3-(b)와 동일하게 진행이 된다. LTR_ZTR 상태에서는 그림 1의 슈도코드의 h, t, R, d 값들을 LR 이진 알고리듬을 이용하여 계산한다. COMPARE_TZ 상태는 연산된 h 값과 t 값이 1인지 판단을 하는 단계이며, GEN_EMI 상태는 그림 1의 슈도코드의 단계-25에서 사용될 $E-i-1$를 계산한다. LTR_B상태에서는 슈도코드의 단계-25의 연산을 수행하고, GEN_R 상태에서는 단계-26의 연산을 통해 R 값을 계산한다. GEN_CT 상태에서는 단계-31과 단계-32의 c 값과 t 값을 계산하며, 이 상태가 끝나면 $t=1$인지 판단하는 COMPARE_TZ

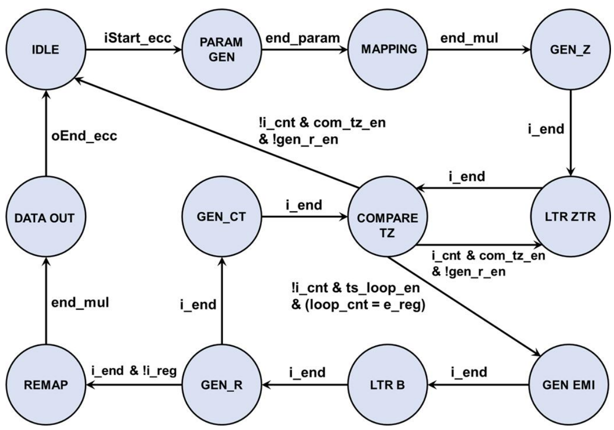


Fig. 4. Modular square root computation based on Tonelli-Shanks algorithm (in the case of $p \equiv 1 \bmod 4$).
그림 4. 간략화된 Tonelli- Shanks 알고리듬에 의한 모듈러 제곱근 계산 ($p \equiv 1 \bmod 4$인 경우)

상태로 돌아간다. REMAP 상태는 몽고메리 도메인으로 맵핑된 최종결과 값을 일반 도메인으로 역변환하는 단계이고, DATA_OUT 상태에서 일반 도메인으로 변환된 최종결과 값이 외부로 출력된다.

그림 1의 간략화된 슈도코드는 타원곡선 크기가 224-비트인 경우에만 적용되어 법 p가 고정되므로, 일부 파라미터는 단계-2와 같이 고정된 값으로 지정된다. 본 논문의 설계에서는 이들 파라미터 값을 레지스터에 저장하여 사용함으로써 이 값들의 연산에 소요되는 과정을 생략하였다. t 값은 h 값을 구하는 과정의 중간결과 값이므로, 단계-8의 조건을 만족하는 경우 그 값을 t로 취하여 t 값 계산에 소요되는 사이클을 줄였다. 단계-27과 같이 $i=1$이 되는 경우에는 앞서 연산된 R 값이 바로 출력되도록 하여 이후의 c와 t 값의 연산과정이 생략되도록 최적화 하였다.

Ⅳ. FPGA 검증

설계된 모듈러 제곱근 연산회로를 그림 5의 FPGA 검증 플랫폼에 구현하여 하드웨어 동작을 검증하였다. Virtex5 XC5VSX95T 디바이스가 사용되었으며, FPGA 보드는 UART 통신을 통해 PC와 데이터를 송·수신한다. 입력된 데이터에 대해 제곱근이 존재하지 않는 것으로 판정되면 새로운 데이터를 입력받아 제곱근 연산이 다시 수행되도록 검증시스템을 구성하였다.

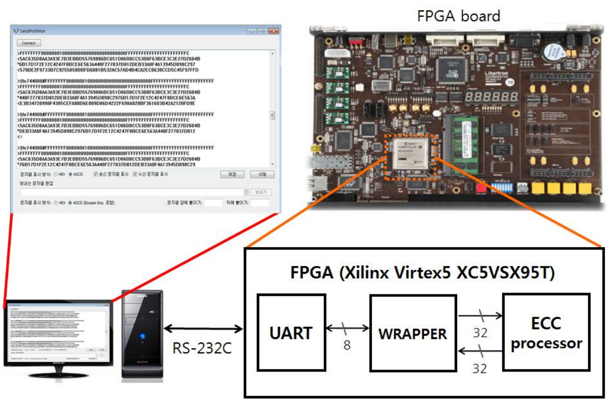
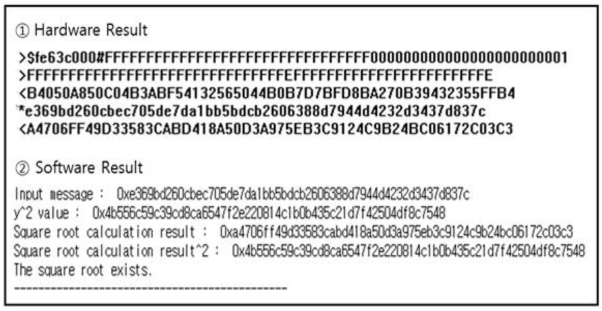


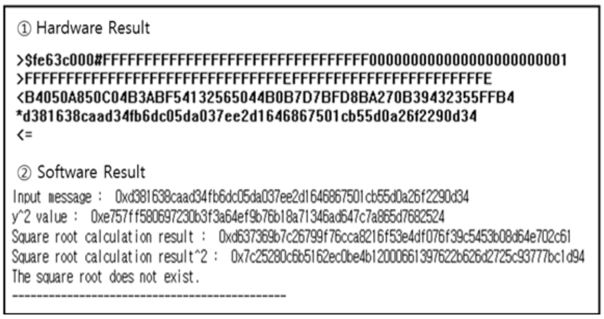
Fig. 5. FPGA verification system.
그림 5. FPGA 검증시스템

FPGA에 구현된 MSQR 연산회로의 연산결과와 소프트웨어로 계산된 결과의 비교를 통해 하드웨어 동작을 검증하였다. 타원곡선 크기와 생성점 값, 타원곡선 계수 값, 모듈러 값 등의 도메인 파라미터 정보가 담긴 OP 데이터와 타원곡선 상의 점으로 매핑될 메시지 x가 입력되면 MSQR 연산이 시작된다. 그림 6-(a)는 입력된 메시지 x에 대해 MSQR이 존재하는 경우의 동작 결과를 보이고 있다. 입력 메시지 x="e369_bd26_0cbe_c705_de7d_a1bb_5bdc_b260_6388_d794_4d42_32d3_437d_837c"에 대한 MSQR 연산 결과로 y="a470_6ff4_9d33_583c_abd4_18a5_0d3a_975e_b3c9_124c_9b24_ bc06_172c_03c3"이 얻어져 입력 메시지 x가 타원곡선 상의 점 $P_x(x,y)$로 매핑되며, 소프트웨어 계산 결과와 일치하여 설계된 회로가 올바로 동작함을 확인하였다. 그림 6-(b)는 입력 메시지 x="d381_638c_aad3_4fb6_dc05_da03_7ee2_d164_6867_501c_b55d_0a26_f229_0d34"에 대해 MSQR이 존재하지 않는 경우의 동작결과이다.

$GF(p)$ 상의 제곱근 연산기능이 구현된 ECC 프로세서를 Virtex 5 XC5VSX95T 디바이스로 합성한 결과 1,128 슬라이스가 사용되었으며, 50 MHz의 클록 주파수로 동작할 수 있음이 확인되었다. 타원곡선 크기에 따라 MSQR 연산에 소요되는 클록 사이클은 표 1과 같다. Param_GEN은 유한체 연산회로의 워드기반 Montgomery 곱셈을 위한 파



(a) in the case when modular square root exists



(b) in the case when modular square root does not exist
Fig. 6. FPGA verification results.
그림 6. FPGA 검증 결과

라미터 생성 및 Montgomery 도메인으로 변환에 소요되는 클록 사이클 수를 나타내고, COM_Y2Z는 계산된 제곱근의 유효성 확인에 소요되는 클록 사이클 수를 나타낸다. 그림 1의 간략화된 Tonelli-Shanks 알고리듬을 적용하는 경우에는 입력되는 메시지 x에 따라 MSQR 연산에 소요되는 클록 사이클 수가 달라지며, 표 1에는 임의의 메시지 데이터에 대한 반복 연산의 평균값을 제시하였다. 구현된 회로가 50 MHz 클록 주파수로 동작하는 경우에, 224-비트 슈도 랜덤 곡선의 경우에는 MSQR 연산에 약 18 ms가 소요되며, 256-비트 Koblitz 곡선의 경우에는 약 4 ms가 소요되는 것으로 평가되었다.

Table 1. Number of clock cycles used to compute MSQR depending on elliptic curve size.
표 1. 타원곡선 크기에 따른 MSQR 계산에 소요되는 클록 수

Field Size	Data Input	Param GEN	ECPMAP	COM Y2Z	Dout	Total
P192K	26	17,005	77,442	222	210	94,905
P192R	26	17,005	66,018	222	210	83,481
P224K	30	23,343	352,179	0	273	375,825
P224R	30	23,343	874,354*	0	273	898,000*
P256K	34	30,077	170,184	360	334	200,989
P256R	34	30,077	98,616	360	334	129,421
P384R	50	67,357	470,052	732	708	538,899
P521R	70	134,837	727,005	1,377	1,343	864,632

* data dependent(average values)

V. 결론

본 논문에서는 EC-ElGamal 공개키 암호를 위해 필요한 모듈러 제곱근 연산의 하드웨어 구현 방법을 제안하고, ECC 프로세서의 유한체 연산회로와 메모리를 이용하여 $GF(p)$ 상에서의 제곱근 연산을 하드웨어로 구현하였다. 설계된 MSQR 연산회로를 Virtex-5 FPGA 디바이스에 구현하여 하드웨어 동작을 검증하였으며, 소프트웨어 연산 결과와 일치하여 올바로 동작하는 것을 확인하였다. $GF(p)$ 상에서의 제곱근 연산이 구현된 ECC 프로세서를 Virtex 5 XC5VSX95T 디바이스로 합성한 결과 1,128 슬라이스로 구현되었으며, 약 50 MHz의 클록 주파수로 동작 가능함을 확인하였다. 본 논문에서는 제안된 MSQR 연산방법을 32-비트 데이터패스의 유한체 연산회로와 메모리를 사용하여 구현하였으나, 데이터패스 비트 수가 큰 유한체 연산회로에 구현하면 소요되는 클록 사이클 수가 작아져 고속연산이 가능하다.

References

[1] N. Koblitz, “Elliptic curve cryptosystems,” *Mathematics of Computation,* vol.48, no.177, pp. 203-209, 1987.
DOI: 10.1090/S0025-5718-1987-0866109-5

[2] National Institute of Standards and Technology (NIST), *Digital Signature Standard (DSS)*, FIPS 186-3, 2009.

[3] SECG SEC1, *Elliptic Curve Cryptography*, Standards for Efficient Cryptography Group, ver.2, 2009, http://www.secg.org/download/aid-780/sec1-v2.pdf.

[4] V. Gayoso Martínez, F. Hernández Á lvarez, L. Hernández Encinas and C. Sánchez Á vila, “A comparison of the standardized versions of ECIES,” *2010 Sixth International Conference on Information Assurance and Security*, Atlanta, pp.1-4, 2010. DOI:10.1109/ISIAS.2010.5604194

[5] K. Rabah, “Elliptic Curve ElGamal Encryption and Signature Schemes,” *Information Technology Journal*, vol.4, no.3, pp.299-306, 2005.
DOI: 10.3923/itj.2005.299.306

[6] T. ElGamal, “A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms,” *IEEE Transactions on Information Theory,* vol.31, no.4, pp.469-472, 1985.
DOI: 10.1109/TIT.1985.1057074

[7] W. Trappe, and L. C. Washington. *Introduction to Cryptography with Coding Theory. 2nd Edition, Prentice Hall, 2006.*

[8] Z. E. Dawahdeh, S. N. Yaakob and A. M. Sagheer, “Modified ElGamal Elliptic Curve Cryptosystem using Hexadecimal Representation,” *Indian Journal of Science and Technology*, vol.8, No.15, pp.1-7, 2015.
DOI: 10.17485/ijst/2015/v8i15/64749

[9] B. King, "Mapping an Arbitrary Message to an Elliptic Curve when Defined over GF(2^n)," *International Journal of Network Security*, vol.8, no.2, pp.169-176, 2009.
DOI: 10.1016/j.jnca.2015.11.011

[10] E. Bach and K. Huber, "Note on Taking Square-Roots Modulo N," *IEEE Transactions on Information Theory*, vol.45, no.2, pp.807-809, 1999. DOI: 10.1109/18.749034

[11] "Computing square roots mod p,"http://course1.winona.edu/eerrthum/13Spring/SquareRoots.pdf

[12] D. Shanks, "Five Number-Theoretic Algorithms," *Proceedings of the Second Manitoba Conference on Numerical Mathematics*, Congressus Numerantium, no.VII, pp.51-70, 1973.

[13] G. Tornaría, "Square roots modulo p," in LATIN 2002: *Theoretical Informatics*, S. Rajsbaum, Ed. Berlin, Germany: Springer, pp.430-434, 2002.
DOI: 10.1007/3-540-45995-2_38

[14] G. Adj and F. Rodríguez-Henríquez, "Square Root Computation over Even Extension Fields," *IEEE Transactions on Computers*, vol.63, no.11, pp.2829-2841, 2014. DOI: 10.1109/TC.2013.145

[15] S. H. Lee and K. W. Shin, "An Area-efficient Design of ECC Processor Supporting Multiple Elliptic Curves over $GF(p)$ and $GF(2^m)$," *Proceedings of Conference on Korea Information and Communication Engineering*, vol.23, no.1, pp.254-256, 2019.
DOI: 10.1109/ACCESS.2019.2958491

BIOGRAPHY

Jun-Yeong Choe (Student Member)

2019 : BS degree in Electronic Engineering, Kumoh National Institute of Technology.
2019~ : Graduate student, Kumoh National Institute of Technology

Kyung-Wook Shin (Member)

1984 : BS degree in Electronic Engineering, Korea Aerospace University
1986 : MS degree in Electronic Engineering, Yonsei University
1990 : Ph.D. degree in Electronic Engineering, Yonsei University
1990~1991 : Senior Researcher, Semiconductor Research Center, Electronics and Telecommunications Research Institute (ETRI)
1991~ : Professor in School of Electronic Engineering, Kumoh National Institute of Technology
1995~1996 : University of Illinois at Urbana- Champaign (Visiting Professor)
2003~2004 : University of California at San Diego (Visiting Professor)
2013~2014 : Georgia Institute of Technology (Visiting Professor)