JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지 Vol. 25, No. 12: 1882~1889, Dec. 2021

3-way Toom-Cook 곱셈과 고속 축약 알고리듬을 이용한 521-비트 고성능 모듈러 곱셈기

양현준[1] · 신경욱[2*]

A 521-bit high-performance modular multiplier using 3-way Toom-Cook multiplication and fast reduction algorithm

Hyeon-Jun Yang[1] · Kyung-Wook Shin[2*]

[1]Graduate Student, Department of Electronic Engineering, Kumoh National Institute of Technology, Gumi, 39177 Korea
[2*]Professor, School of Electronic Engineering, Kumoh National Institute of Technology, Gumi, 39177 Korea

요 약

본 논문은 타원곡선 암호에 핵심 연산으로 사용되는 모듈러 곱셈의 고성능 하드웨어 구현에 대해 기술한다. NIST P-521 곡선에 적합한 521-비트 고성능 모듈러 곱셈기를 3-way Toom-Cook 정수 곱셈과 고속 축약 알고리듬을 적용하여 설계하였다. 정수곱셈 결과에 3이 곱해져 출력되는 3-way Toom-Cook 알고리듬의 속성을 고려하여, 피연산자에 1/3을 곱한 Toom-Cook 도메인 상에서 모듈러 곱셈이 연산되도록 구현하였다. 모듈러 곱셈기를 xczu7ev FPGA 디바이스에 구현하여 하드웨어 동작을 검증하였으며, 69,958개의 LUT와 4,991개의 플립플롭 그리고 101개의 DSP 블록의 하드웨어 자원이 사용되었다. Zynq7 FPGA 디바이스에서 최대 동작주파수는 50 MHz으로 예측되었으며, 초당 약 416만 번의 모듈러 곱셈을 연산할 수 있는 것으로 평가되었다.

ABSTRACT

This paper describes a high-performance hardware implementation of modular multiplication used as a core operation in elliptic curve cryptography. A 521-bit high-performance modular multiplier for NIST P-521 curve was designed by adopting 3-way Toom-Cook integer multiplication and fast reduction algorithm. Considering the property of the 3-way Toom-Cook algorithm in which the result of integer multiplication is multiplied by 1/3, modular multiplication was implemented on the Toom-Cook domain where the operands were multiplied by 3. The modular multiplier was implemented in the xczu7ev FPGA device to verify its hardware operation, and hardware resources of 69,958 LUTs, 4,991 flip-flops, and 101 DSP blocks were used. The maximum operating frequency on the Zynq7 FPGA device was 50 MHz, and it was estimated that about 4.16 million modular multiplications per second could be achieved.

키워드 : 모듈러 곱셈, 타원곡선 암호, 3-way 툼-쿡 곱셈, 모듈러 축약

Keywords : Modular multiplication, Elliptic curve cryptography, 3-way Toom-Cook multiplication, Modular reduction

Received 13 October 2021, **Revised** 25 October 2021, **Accepted** 10 November 2021

*** Corresponding Author** Kyung-Wook Shin (kwshin@kumoh.ac.kr, Tel:+82-54-478-7427)
Professor, School of Electronic Engineering, Kumoh National Institute of Technology, Gumi, Kyungbuk, 39177 Korea

Open Access http://doi.org/10.6109/jkiice.2021.25.12.1882 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. 서 론

공개키 암호 (public-key cryptography)는 대칭키 암호에 비해 키 관리가 용이하고, 인증 (authentication), 권한 부여 등 다양한 보안기능 구현에 폭넓게 응용되고 있다. 대표적인 공개키 암호 방식인 타원곡선 암호 (elliptic curve cryptography; ECC)는 블록체인과 가상화폐 시스템의 트랜잭션 검증을 위한 타원곡선 디지털 서명 알고리듬 (EC-DSA)의 핵심 기술로 사용되고 있다. 자율주행 자동차의 차량간 (V2V) 및 차량-인프라간 (V2I) 통신을 위한 국제 보안 규격 IEEE 1609.2, CAPM VSC3 등과 무인비행체 (드론)의 식별 인증, 상호 인증 등에서도 EC-DSA가 사용되고 있다. 블록체인 플랫폼, 자율주행 이동체 보안을 위해서는 초당 수 백 ~ 수 천 회의 디지털 서명과 검증이 요구되므로, 고성능 ECC의 구현이 핵심 요소가 된다 [1, 2].

ECC는 타원곡선 상의 점 덧셈 (point addition)과 점 두배 (point doubling) 연산으로 구현되며, 점 덧셈과 점 두배 연산은 유한체 연산인 모듈러 (modular) 곱셈, 나눗셈, 덧셈 등의 연산으로 구현된다. 따라서 타원곡선 암호의 고성능 구현을 위해서는 유한체 상의 모듈러 곱셈의 고속 하드웨어 구현이 필수적이다.

모듈러 곱셈은 유한체 GF(p) 상의 곱셈을 의미하며, 피승수 A, 승수 B 그리고 모듈러스 p가 존재할 때 모듈러 곱셈 연산은 $C = A \times B \bmod p$로 정의된다. 모듈러 곱셈을 구현하는 방법에는 몽고메리 모듈러 곱셈 알고리듬 [3]과 같이 정수 곱셈과 축약 연산을 하나의 알고리듬으로 구현 하는 방법과 정수 곱셈 후 축약 연산을 통해 구현 하는 방법 등이 있다. 정수 곱셈 알고리듬에는 카라츄바-오프만 (Karatsuba-Ofman) 알고리듬 [4], Toom-Cook 알고리듬 [5,6] 등이 있으며, 축약 알고리듬에는 Lazy 축약 (Lazy reduction) 알고리듬 [7]과 바렛 축약 (Barrett reduction) 알고리듬 [8], NIST prime에 대한 고속 축약 (fast reduction) 알고리듬 [9] 등이 있다. 본 논문에서는 3-way Toom-Cook 알고리듬과 고속 축약 알고리듬을 기반으로 고성능 모듈러 곱셈기를 설계하였다. 2장에서는 3-way Toom-Cook 곱셈과 고속 축약 연산 알고리듬을 설명하며, 3장은 모듈러 곱셈기의 고성능 하드웨어 구현에 대해 서술한다. 4장은 구현된 모듈러 곱셈기의 검증 및 성능 평가를 서술하고 5장에서 결론을 맺는다.

Ⅱ. m3WTC와 mFRed 알고리듬

2.1. Toom-Cook 곱셈 알고리듬

Toom-Cook 알고리듬은 안드레이 툼 (Andrei Toom)이 제안하고, 스테픈 쿡 (Stephen Cook)이 수정한 알고리듬이며 [5,6], 매우 큰 정수의 곱셈을 고속으로 계산하기 위해 고안되었다. 큰 정수의 곱셈은 타원곡선 암호, RSA 등의 공개키 암호와 동형암호 (homomorphic encryption)에 핵심 연산으로 사용된다. Toom-Cook 알고리듬은 피승수 A와 승수 B를 각각 d개 (단, $d \geq 1$)로 분할하여 연산하며, 이를 d-Way Toom-Cook이라고 한다. 피연산자를 분할한 개수 d가 클수록 내부 연산량이 증가하지만, 연산시간 복잡도는 $O(N^{\log_d(2d-1)})$로 감소하는 특징을 가지며, 여기서 N은 피승수와 승수의 비트 수를 나타낸다 [10]. $d = 1$인 경우의 연산시간 복잡도는 $O(N^2)$이며, 카라츄바-오프만 곱셈과 등가인 2-Way Toom-Cook 곱셈은 $O(N^{1.585})$의 연산시간 복잡도를 가지며, 3-Way Toom-Cook 곱셈의 연산시간 복잡도는 $O(N^{1.464})$ 이다.

본 논문에서는 2-Way Toom-Cook 곱셈보다 연산시

Input: $A = \{p_{3m-1}, p_{3m-2}, \cdots, p_1, p_0\}, 0 < A, B < 2^{521}$
$B = \{q_{3m-1}, q_{3m-2}, \cdots, q_1, q_0\}, p_i, q_i, c_i \in \{0,1\}, m = 174$
Output: $C = A \times B$

Splitting:
$U_2 := \{p_{3m-1}, p_{3m-2}, \cdots, p_{2m}\}$; $V_2 := \{q_{3m-1}, q_{3m-2}, \cdots, q_{2m}\}$;
$U_1 := \{p_{2m-1}, p_{2m-2}, \cdots, p_m\}$; $V_1 := \{q_{2m-1}, q_{2m-2}, \cdots, q_m\}$;
$U_0 := \{p_{m-1}, p_{m-2}, \cdots, p_0\}$; $V_0 := \{q_{m-1}, q_{m-2}, \cdots, q_0\}$;
Evaluation:
$P(0) := U_0$; $Q(0) := V_0$;
$P(1) := U_2 + U_1 + U_0$; $Q(1) := V_2 + V_1 + V_0$;
$P(-1) := U_2 - U_1 + U_0$; $Q(-1) := V_2 - V_1 + V_0$;
$P(2) := 4U_2 + 2U_1 + U_0$; $Q(2) := 4V_2 + 2V_1 + V_0$;
$P(\infty) := U_2$; $Q(\infty) := V_2$;
Multiplication:
$R(0) := P(0) \times Q(0)$;
$R(1) := P(1) \times Q(1)$;
$R(-1) := P(-1) \times Q(-1)$;
$R(2) := P(2) \times Q(2)$;
$R(\infty) := P(\infty) \times Q(\infty)$;
Interpolation:
$W_0 := R(0)$;
$W_1 := -\frac{1}{2}R(0) + R(1) - \frac{1}{3}R(-1) - \frac{1}{6}R(2) + 2R(\infty)$;
$W_2 := -R(0) + \frac{1}{2}R(1) + \frac{1}{2}R(-1) - R(\infty)$;
$W_3 := \frac{1}{2}R(0) - \frac{1}{2}R(1) - \frac{1}{6}R(-1) + \frac{1}{6}R(2) - 2R(\infty)$;
$W_4 := R(\infty)$;
Recomposition:
$C := R(2^m)$;
Return C

Fig. 1 3-way Toom-Cook algorithm [11]

간 복잡도가 낮고, 내부 곱셈기의 크기가 작은 3-Way Toom-Cook 곱셈 알고리듬을 사용하여 정수 곱셈을 구현하였다. 그림 1은 3-Way Toom-Cook 알고리듬의 슈도코드이며, 분할 (splitting), 평가 (evaluation), 재귀적 곱셈 (recursive multiplication), 보간 (interpolation), 재구성 (recomposition)의 5단계로 구성되며, 각 단계는 나누어진 개수 d에 따라 상세 연산이 달라진다 [11].

(1) 분할 단계: $3m$-비트의 피연산자 A, B를 각각 U_2, U_1, U_0,와 V_2, V_1, V_0로 분할하며, 분할된 피연산자 블록은 m-비트이다. 피승수 A와 승수 B는 각각 식 (1), 식 (2)와 같이 U_2, U_1, U_0와 V_2, V_1, V_0를 계수로 하는 다항식으로 표현될 수 있으며, 정수 곱셈의 결과는 식 (3)으로 표현된다.

(2) 평가 단계: 다항식 $P(x), Q(x)$의 x에 임의의 값을 대입하는 단계이다. 결과값 $C = R(x)$의 계수가 5개이므로 $0, 1, -1, 2, \infty$의 5개 값을 사용한다. ∞값은 각 다항식의 최고차항 계수를 의미한다 [12].

(3) 재귀적 곱셈 단계: 평가 단계의 결과를 이용하여 다항식 $R(x) = P(x)\,Q(x)$의 평가값을 얻는다. Toom- Cook 곱셈은 분할된 부분에도 재귀적으로 Toom-Cook 알고리듬 적용이 가능하지만, 본 논문에서는 적용하지 않았다.

(4) 보간 단계: 단계-3의 재귀적 곱셈 결과를 이용해 $R(x)$ 다항식의 계수 값 W_4, W_3, W_2, W_1, W_0을 구한다. 이 때 연산 과정에서 역행렬 연산이 사용된다.

(5) 재구성 단계: 보간 단계에서 구한 계수 값으로 결과값을 재구성하며, $R(2^m)$ 연산을 통해 $C = A \times B$를 얻는다.

$$A = P(x) = U_2 x^2 + U_1 x + U_0 \tag{1}$$

$$B = Q(x) = V_2 x^2 + V_1 x + V_0 \tag{2}$$

$$C = W_4 x^4 + W_3 x^3 + W_2 x^2 + W_1 x + W_0 \tag{3}$$

3-way Toom-Cook 곱셈은 부분 곱의 개수를 줄여 고속 연산이 가능한 장점이 있지만, 보간 단계의 연산에 1/3, 1/6 계산이 포함되어 있어 하드웨어 구현에 비효율성이 존재한다는 단점이 있다. 본 논문에서는 문헌 [11]에 소개된 $d_3 = 3$을 곱하는 방법과 Toom-Cook 도메인을 적용하여 이런 단점을 제거하였다. $d_3 = 3$은 그림 1의 보간 단계에 존재하는 1/3, 1/6을 제거하기 위한 값이며, 나누기 2 연산은 오른쪽 시프트 연산으로 대체할 수 있다. 이 경우 정수 곱셈 연산 결과는 $C = 3 \times A \times B$으로 3이 곱해진 값이 얻어진다. 모듈러 곱셈은 타원곡선 암호의 점 스칼라 곱셈 (point scalar multiplication)에서 가장 많이 사용되는 연산이므로, 정수 곱셈 연산 결과를 매번 3으로 나누는 것은 클럭 소요 사이클 측면에서 매우 비효율적이다. 본 논문에서는 Toom-Cook 도메인을 적용하여 정수 곱셈 연산을 계산함으로써 이와 같은 단점을 개선하였다. Toom-Cook 도메인의 데이터는 피승수와 승수를 d_3^2의 모듈러 역원값과 모듈러 곱셈한 결과값을 의미하며, 피승수와 승수가 모두 Toom-Cook 도메인 데이터인 경우, 결과값 또한 Toom-Cook 도메인에 존재한다. 타원곡선 암호 연산의 경우, 입력 데이터를 Toom-Cook 도메인에 매핑한 뒤 최종 출력값만 도메인 리매핑하면 되므로 소요 사이클 측면에서 효율적이다. 그림 2는 Toom-Cook 도메인이 적용되어 1/3, 1/6 연산을 제거한 수정형 3-Way Toom-Cook (m3WTC) 알고리듬의 슈도코드이며, 단계별 연산은 다음과 같다.

Input: $A^* = \{p_{3m-1}, p_{3m-2}, \cdots, p_1, p_0\}, (0 < A^*, B^* < 2^{521})$,
$B^* = \{q_{3m-1}, q_{3m-2}, \cdots, q_1, q_0\}, m = 174,\ p_i, q_i, c_i \in \{0,1\}$
$*$ *denotes data mapped into Toom − Cook domain*
$A^* = A \times d_3^{-1}\ mod\ p_{521r}$
$B^* = B \times d_3^{-1}\ mod\ p_{521r}$
Output: $C^* = TCM(A^*, B^*) = 3 \times A^* \times B^*$
$= \{c_{6m-1}, c_{6m-2}, \cdots, c_1, c_0\}$

00: $U_0 \leftarrow \{p_{m-1}, p_{m-2}, \cdots, p_0\}$; $V_0 \leftarrow \{q_{m-1}, q_{m-2}, \cdots, q_0\}$;
$U_1 \leftarrow \{p_{2m-1}, p_{2m-2}, \cdots, p_m\}$; $V_1 \leftarrow \{q_{2m-1}, q_{2m-2}, \cdots, q_m\}$;
$U_2 \leftarrow \{p_{3m-1}, p_{3m-2}, \cdots, p_{2m}\}$; $V_2 \leftarrow \{q_{3m-1}, q_{3m-2}, \cdots, q_{2m}\}$;
01: $R(1)_H \leftarrow U_2 + U_1 + U_0$; $R(1)_L \leftarrow V_2 + V_1 + V_0$;
$R(-1)_H \leftarrow U_2 - U_1 + U_0$; $R(0) \leftarrow U_0 \times V_0$;
02: $R(-1)_L \leftarrow V_2 - V_1 + V_0$; $R(2)_H \leftarrow 4U_2 + 2U_1 + U_0$;
$R(2)_L \leftarrow 4V_2 + 2V_1 + V_0$; $R(\infty) \leftarrow U_2 \times V_2$;
03: $S_0 \leftarrow R(0) + 2R(0)$; $S_2 \leftarrow R(\infty) + 2R(\infty)$;
$R(1) \leftarrow R(1)_H \times R(1)_L$;
04: $T_1 \leftarrow -\frac{1}{2}S_0 + 2S_2$; $T_3 \leftarrow \frac{1}{2}S_0 - 2S_2$;
$S_1 \leftarrow R(1) + 2R(1)$; $R(-1) \leftarrow R(-1)_H \times R(-1)_L$;
05: $T_1 \leftarrow T_1 - R(-1) + S_1$; $T_2 \leftarrow -S_0 + \frac{1}{2}S_1 + \frac{1}{2}R(-1)$;
$T_3 \leftarrow T_3 - \frac{1}{2}S_1$; $R(2) \leftarrow R(2)_H \times R(2)_L$;
06: $T_1 \leftarrow T_1 - \frac{1}{2}R(2)$; $T_2 \leftarrow T_2 + R(-1) - S_2$;
$T_3 \leftarrow T_3 + \frac{1}{2}R(2) - \frac{1}{2}R(-1)$;
07: $\{T_{2L}, S_{0H}\} \leftarrow \{T_{2L}, S_{0H}\} + T_1 + Cr1$; $S_{2H} \leftarrow S_{2H} + Cr3$
$\{S_{2L}, T_{2H}\} \leftarrow \{S_{2L}, T_{2H}\} + T_3 + Cr2$;
08: $T_2 \leftarrow T_2 + Cr4$; $S_2 \leftarrow S_2 + Cr5 + Cr6$;
09: $C^* \leftarrow \{S_2, T_2, S_0\}$;
Return C^*

Fig. 2 Modified 3-Way Toom-Cook (m3WTC) algorithm

(1) 단계-0: 피승수와 승수를 U_2, U_1, U_0와 V_2, V_1, V_0로 분할하는 단계이며, 데이터 입력과 동시에 수행되므로 클럭 사이클을 소모하지 않는다.
(2) 단계-1~단계-2: 평가 단계와 재귀적 곱셈 단계가 동시에 연산되는 단계이며, 단계-2가 끝나면 평가 단계의 연산이 완료된다.
(3) 단계-3~단계-6: 재귀적 곱셈과 보간 단계에 해당하며, $S_0, S_1, S_2, T_1, T_2, T_3$가 생성된다. 단계-3~단계-5까지는 재귀적 곱셈 단계와 보간 단계가 동시에 진행되며, 단계-6이 완료되면 보간 단계가 완료된다.
(4) 단계-7~단계-8: 보간 단계에서 생성된 S_0, T_1, T_2, T_3, S_2 값들을 재구성하는 단계이며, 캐리 값들을 함께 처리해준다. 재구성은 자리수에 맞춰 가산하는 것을 의미한다. 단계-8의 $T_2 \leftarrow T_2 + Cr4$에서 캐리 값 $Cr6$이 생성되며, $Cr6$의 연산을 다음 사이클에서 처리할 경우 하나의 사이클을 더 소모하므로 S_2를 처리하는 연산에서 $Cr6$이 더해지는 경우와 더해지지 않는 경우 두 가지를 단계-8에서 미리 계산한다.
(5) 단계-9: $Cr6 = 1$인 경우, 단계-8에서 S_2 연산에 $Cr6$이 더해진 값을 사용하고, $Cr6 = 0$인 경우에는 S_2 연산에 $Cr6$이 더해지지 않은 값을 사용하여 결과값을 출력한다.

2.2. 고속 축약 알고리듬

NIST prime에 대한 고속 축약 알고리듬 (fast reduction algorithm; FRed) [9]은 모듈러 합동 특성을 이용하며, NIST의 소수체 곡선 상에서 피연산자를 고속으로 축약하기 위해 제안되었다. 축약 연산은 피연산자 C에 대해 $D = C \bmod p$ 연산을 의미한다. 두 정수 E, F와 모듈러스 p에 대해 $E \bmod p = F \bmod p$와 같이 모듈러 연산의 결과값이 동일한 경우, 두 정수 E, F는 모듈러 합동 관계이며, $E \equiv F \pmod p$로 표현된다.

본 논문은 NIST의 SP 800-186 문서 [13]에 정의된 P-521 곡선의 521 비트 모듈러스 값 $p_{521r} = 2^{521} - 1$을 사용하였으며, P-521 곡선의 경우 정수 곱셈의 결과값 C는 최대 1042 비트이므로, 이를 521 비트의 결과값 $D = C \bmod p_{521r}$로 축약해야 한다. 본 논문에서 적용한 모듈러스가 p_{521r}인 경우에 대한 고속 축약 연산 알고리듬의 슈도코드는 그림 3과 같다. 피연산자인 곱셈의 결과값 C는 최대 1042-비트이며, 식 (4)와 같이 표현된다.

Input: $C = \{c_{1041}, c_{1040}, \cdots, c_1, c_0\}, (0 < C \le A \times B)$
Output: $D = MR(C, p_{521r}) = C \bmod p_{521r}$

$Re - grouping$:
  $H_0 \leftarrow (c_{1041}, c_{1040}, \cdots, c_{522}, c_{521})$;
  $H_1 \leftarrow (c_{520}, c_{519}, \cdots, c_1, c_0)$;
01: $d_1 \leftarrow H_0 + H_1$; $d_2 \leftarrow H_0 + H_1 - p_{521r}$; $d_3 \leftarrow H_0 + H_1 - 2p_{521r}$;
02: $D \leftarrow selection\ of\ d_i\ depending\ on\ the\ sign\ of\ d_i$
Return D

Fig. 3 Fast reduction algorithm

$c_{1041}, c_{1040}, \sim, c_{522}, c_{521}$를 계수로 갖는 값은 p_{521r} 보다 큰 값을 가지므로, 식 (5)의 모듈러 합동 특성을 이용해 p_{521r} 보다 작은 값으로 표현될 수 있다. 그림 3의 슈도코드에서 H_0는 식 (5)에 의해 $H_0 \times 2^{521}$는 H_0와 모듈러 합동이다. 즉, 그림 3의 Re-grouping 단계는 식 (5)를 이용하여 C를 521 비트의 $H_n (n = 0,1)$로 표현한 것이다. 단계-1에서 식 (6)의 연산과 추가 축약 (extra reduction) 연산이 수행된다. 추가 축약은 $d_1 = H_0 + H_1$의 값이 $0 \le d_1 \le 2p_{521r}$의 범위에 존재하기 때문에 d_1의 범위를 $0 \le d_1 < p_{521r}$으로 줄이기 위해 필요한 연산이다. 단계-1의 연산이 종료되면 $d_i (1 \le i \le 3)$의 부호에 따라 하나의 값이 선택되어 521 비트의 축약 결과가 얻어진다.

$$C = \sum_{k=0}^{1041} c_k \cdot 2^k \tag{4}$$

$$2^{521} \equiv 1 \pmod{p_{521r}} \tag{5}$$

$$D = (H_0 + H_1) \bmod p_{521r} \tag{6}$$

본 논문의 m3WTC 곱셈은 정수 곱셈 결과값에 3이 곱해진 값이 출력되므로 실제로는 최대 1044-비트의 C^*이 얻어진다. 그림 4는 본 논문에서 제안하는 수정된 고속 축약 연산 (modified fast reduction; mFRed) 알고리듬의 슈도코드이며, 그림 3의 알고리듬과 달리 c_{1043},

Input: $C = \{c_{1043}, c_{1042}, \cdots, c_1, c_0\}, (0 < C \le 3 \times A \times B)$
Output: $D = MR(C, p_{521r}) = C \bmod p_{521r}$

$Re - grouping$:
  $H_0 \leftarrow (0,0, \cdots, 0,0, c_{1043}, c_{1042})$;
  $H_1 \leftarrow (c_{1041}, c_{1040}, \cdots, c_{522}, c_{521})$;
  $H_2 \leftarrow (c_{520}, c_{519}, \cdots, c_1, c_0)$;
01: $d_1 \leftarrow H_0 + H_1 + H_2$;
02: $d_2 \leftarrow d_1 - p_{521r}$; $d_3 \leftarrow d_1 - 2p_{521r}$;
03: $D \leftarrow selection\ of\ d_i\ depending\ on\ the\ sign\ of\ d_i$
Return D

Fig. 4 Modified fast reduction algorithm

c_{1042} 계수 값과 H_2가 추가되었다. c_{1043}, c_{1042}계수 값에 의해 생성되는 2^{1042} 값의 모듈러 합동 값은 식 (7)과 같다. 식 (8)은 식 (6)에 H_2항을 추가한 것으로 기본 원리는 동일하다. Re-grouping 단계는 3개의 521 비트 $H_n(0 \le n \le 2)$을 생성하며, 단계-1에서 식 (8)의 가산을 수행한다. 단계-2는 추가 축약 연산이다. 계수 값 c_{1043}, c_{1042}이 추가되어 d_1의 범위가 $0 \le d_1 \le 2p_{521r} + \alpha$로 $\alpha = \{c_{1043}, c_{1042}\}$만큼 가산된다. 단계-2가 종료되면 $d_i(1 \le i \le 3)$의 부호에 따라 축약 연산의 결과 값이 얻어진다.

$$2^{1042} \equiv 1 \pmod{p_{521r}} \tag{7}$$

$$D = (H_0 + H_1 + H_2) \bmod p_{521r} \tag{8}$$

2.3. m3WTC와 mFRed 기반의 모듈러 곱셈

본 논문은 m3WTC 정수 곱셈 알고리듬과 mFRed 축약 연산을 기반으로 모듈러 곱셈을 구현하였다. 그림 5는 m3WTC와 mFRed 기반 모듈러 곱셈 알고리듬의 슈도코드이다. 모듈러 곱셈은 Toom-Cook 도메인 매핑, 정수 곱셈, 축약 연산, 도메인 리매핑까지 총 4 단계의 과정으로 수행되며, 모듈러 곱셈 연산을 반복할 경우 단계-1의 도메인 매핑과 단계-4의 도메인 리매핑은 전체 연산의 처음과 끝에서 각각 한 번씩만 수행된다. 단계-2와 단계-3의 연산은 각각 m3WTC 정수 곱셈 연산과 mFRed 축약 연산을 의미한다. 식 (9)는 피승수와 승수의 도메인 매핑 연산이며, d_3^2의 모듈러 역원 값이 필요하다. 리매핑은 식 (10)과 같이 Toom-Cook 도메인에 있는 모듈러 곱셈의 결과값에 1을 곱해주는 과정이며, 리매핑 후 정상적인 모듈러 곱셈의 결과가 얻어진다.

Input: $A = \{p_{3m-1}, p_{3m-2}, \cdots, p_1, p_0\}, (0 < A, B < 2^{521})$,
$B = \{q_{3m-1}, q_{3m-2}, \cdots, q_1, q_0\}, m = 174,\ p_i, q_i, r_i \in \{0,1\}$,
p_{521r}
Output: $D = A \times B \ mod p_{521r}$

01: $A^* \leftarrow$ ***Domain mapping*** (A, d_3^{-2}, p_{521r});
$B^* \leftarrow$ ***Domain mapping*** (B, d_3^{-2}, p_{521r});
02: $C^* \leftarrow$ ***TCM*** (A^*, B^*);
03: $D^* \leftarrow$ ***MR*** (C^*, p_{521r});
04: $D \leftarrow$ ***Domain remapping*** $(D^*, 1, p_{521r})$;
Return D

Fig. 5 Modular multiplication based on m3WTC and mFRed algorithms

$$A^* = MR(TCM(A, d_3^{-2}), p_{521r}) = A \times d_3^{-1} \bmod p_{521r} \tag{9-a}$$

$$B^* = MR(TCM(B, d_3^{-2}), p_{521r}) = B \times d_3^{-1} \bmod p_{521r} \tag{9-b}$$

$$MR(TCM(D^*, 1), p_{521r}) = D \bmod p_{521r} \tag{10}$$

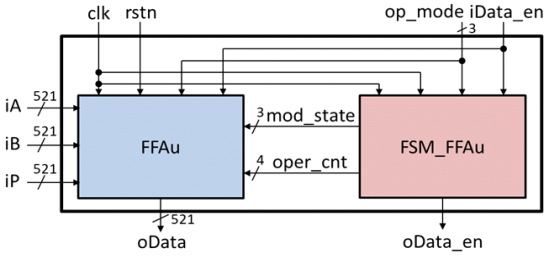

Fig. 6 Block diagram of modular multiplier

Ⅲ. 모듈러 곱셈기 구현

II장에서 설명된 알고리듬들을 적용하여 설계된 모듈러 곱셈기는 그림 6의 구조를 가지며, 산술 연산기 블록인 FFAu (finite field arithmetic unit)와 이들의 동작을 제어하는 FSM_FFAu로 구성된다. FFAu는 그림 7과 같이 3개의 525-비트 가산기와 1개의 178-비트 곱셈기, 5개의 레지스터 파일 (U,V,S,T,R)과 데이터 선택기 및 캐리 저장 레지스터 그리고 MUX로 구성된다. 525-비트 가산기는 3:2 압축기이며, 1개의 캐리저장 가산기 (carry-save adder)와 1개의 캐리선택 가산기 (carry-select adder)로 구성된다. 캐리저장 가산기는 피연산자 3개를 동시에 가산할 수 있는 장점이 있으며, 결과값으로 합과 캐리를 출력한다. 출력된 합과 캐리는 캐리선택 가산기에서 최종 가산되며, 이때 캐리는 합보다 1-비트 상위에 있

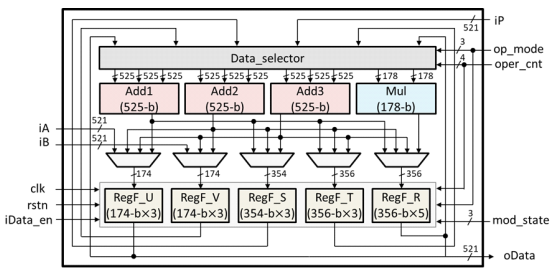

Fig. 7 FFAu block

는 것으로 처리된다. 캐리선택 가산기는 캐리전파 지연을 줄일 수 있는 장점이 있다.

본 논문의 모듈러 곱셈기는 여러 피연산자를 동시에 처리하도록 설계되었으므로, 중간 연산 결과값을 RAM에 저장하고 읽어내는 방식은 비효율적이다. 따라서 동시에 여러 데이터를 저장 또는 출력할 수 있는 레지스터 파일을 사용하여 설계하였으며, 레지스터 파일 U,V,S,T는 각각 3개의 레지스터로 구성되며, R은 5개의 레지스터로 구성된다. 데이터 선택기와 MUX는 FSM_FFAu에서 생성되는 mod_state, oper_cnt 신호에 따라 연산기의 입력과 레지스터에 저장될 값을 선택한다. FSM_FFAu는 mod_state 신호와 연산 계수값 oper_cnt 그리고 연산 완료 신호 oData_en을 생성한다. mod_state는 전체 모듈러 연산을 제어하는 상태 머신이며, 데이터 입력 신호 iData_en와 oper_cnt 값에 따라 동작된다. 모듈러 곱셈이 완료되면 FSM_FFAu 블록에서 연산완료 신호가 출력되고, FFAu 블록에서는 연산 결과값이 출력된다.

Ⅳ. 검증 및 성능 평가

설계된 모듈러 곱셈기는 xczu7ev FPGA 디바이스에 구현하여 하드웨어 동작을 검증하였다. 검증 플랫폼은 그림 8과 같으며, FPGA에 회로를 구현한 뒤, PC와 FPGA간의 UART 통신을 통해 입력값과 출력값을 전송한다. 그림 9는 FPGA 검증 결과가 표시되는 GUI 화면

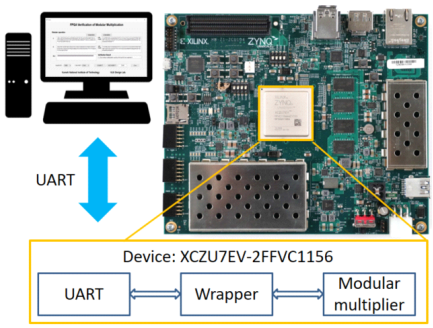

Fig. 8 FPGA verification platform

의 캡처를 보인 것이며, 피승수 A와 승수 B의 모듈러 곱셈 결과가 TC 도메인 상의 값 $3\times A\times B$로 표시되며, FPGA에 구현된 모듈러 곱셈기에서 계산된 값과 소프트웨어로 계산된 값이 동일함을 확인하여 설계된 모듈러 곱셈기가 올바로 동작함을 확인하였다.

설계된 모듈러 곱셈기는 xczu7ev FPGA 디바이스의 LUT 69,958개 플립플롭 4,991개 그리고 101개의 DSP 블록을 사용하여 합성되었으며, 최대 동작주파수는 50 MHz로 평가되었다. P-521 곡선 상의 모듈러 곱셈 연산에 12 클럭 사이클이 소요되며, 50 MHz 주파수에서 초당 약 416만 번의 모듈러 곱셈이 연산된다.

표 1은 521-비트 모듈러 곱셈기를 비교한 것이다. 본 논문의 모듈러 곱셈기는 문헌 [14]와 [15]에 비해서 초당 연산 속도가 각각 약 2배, 50배 빠르며, 문헌 [16]에 비해 초당 연산 속도가 20% 느리지만 DSP 사용이 약 60% 감소된 장점이 있다.

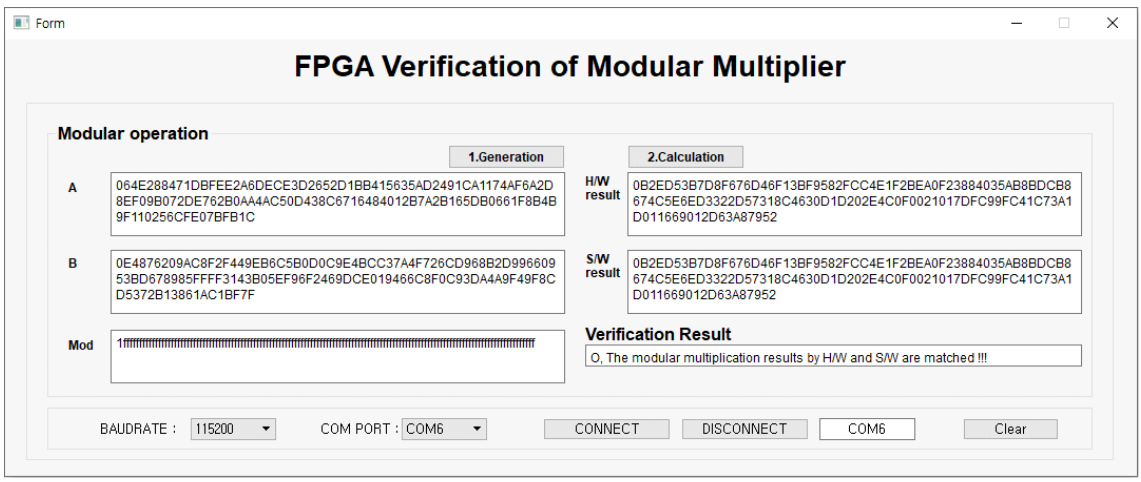

Fig. 9 Screenshot of FPGA verification result of modular multiplier

Table. 1 Comparison of modular multipliers

	This paper	[14]	[15]	[16]
Device	xczu7ev	Virtex-6	Virtex-4	Virtex-6
Field size	521-bit			
Hardware Resources	69,958 LUTs + 4,491 FF + 101 DSPs	46,382 LUTs + 5,710 FF + 208 DSPs	5,368 LUTs	8,400 slices + 259 DSPs
Frequency (MHz)	50	32	42.7	100
Time (μs)	0.24	0.53	12.22	0.20

Ⅴ. 결 론

본 논문의 모듈러 곱셈기는 3-way Toom-Cook 알고리듬과 고속 축약 알고리듬을 사용하여 P-521곡선에 적합하게 설계되었다. xczu7ev 디바이스에 69,958개의 LUT와 4,991개의 플립플롭 그리고 101개의 DSP 블록을 사용하여 합성되었으며, 최대 동작 주파수는 50 MHz이다. 모듈러 곱셈에 12 클럭 사이클을 소요하므로 초당 연산은 약 416만 번 가능하다. 설계된 모듈러 곱셈기는 타원곡선 암호 프로세서에 사용될 예정이다.

ACKNOWLEDGEMENT

- This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. 2020R1I1A3A04038083)
- This paper was supported by Korea Institute for Advancement of Technology (KIAT) grant funded by the Korea Government (MOTIE) (P0017011, HRD Program for Industrial Innovation)
- The authors are thankful to IDEC for EDA tool support.

REFERENCES

[1] S. Sugiyama, H. Awano, and M. Ikeda, "Low Latency 256-bit Fp ECDSA Signature Generation Crypto Processor," *IEICE Transaction on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E101-A, no. 12, pp. 2290-2296, Dec. 2018. DOI: 10.1587/transfun.E101.A.2290.

[2] M. Kneževic, V. Nikov, and P. Rombouts, "Low-latency ECDSA signature verification-A road toward safer traffic," *IEEE Transaction on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 11, pp. 3257-3267, 2016.

[3] P. L. Montgomery, "Modular multiplication without trial division," *Mathematics of Computation*, vol. 44, no. 170, pp. 519-521, May. 1985.

[4] A. Karatsuba and Y. Ofman, "Multiplication of many-digital numbers by automatic computers," *Proceedings of the USSR Academy of Sciences*, vol. 145, no. 2, pp. 293-294, 1962.

[5] A. L. Toom, "The complexity of a scheme of functional elements realizing the multiplication of integers," *Soviet Math. Doklady*, vol. 3, no. 4, pp. 714-716, 1963.

[6] S. A. Cook and S. O. Aanderaa, "On the minimum computation time of functions," *Transaction of the American Mathematical Society*, vol. 142, pp. 291-314, Aug. 1969.

[7] S. Li and Z. Gu, "Lazy Reduction and Multi-Precision Division Based on Modular Reductions," *2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, Chengdu, pp. 407-410, 2018.

[8] P. Barrett, "Implementing the Rivest Shamirand Adleman public key encryption algorithm on a standard digital signal processor," *Conference on the Theory and Application of Cryptographic Techniques*, Springer, vol. 263, pp. 311-323, Aug. 1986.

[9] D. Hankerson, A. Menezes, and S. Vanstone, *Guide to elliptic curve cryptography*, Springer Science & Business Media, 2006.

[10] J. M. B. Mera, A. Karmakar, and I. Verbauwhed, "Time-memory trade-off in Toom-Cook multiplication: an application to module-lattice based cryptography," *International Association for Cryptologic Research (IACR) Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, no. 2, pp. 222-244, Mar. 2020. DOI: 10.13154/tches.v2020.i2.222-244.

[11] J. Ding, S. Li, and Z. Gu, "High-speed ECC processor over NIST prime fields applied with Toom–Cook multiplication," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 3, pp. 1003-1016, Mar. 2019.

[12] M. Bodrato and A. Zanoni, "Integer and polynomial multiplication: Towards optimal black Toom-Cook matrices," *Proceedings of the 2007 international symposium on*

Symbolic and algebraic computation, pp. 17-24, Jul./Aug. 2007.

[13] L. Chen, D. Moody, A. Regenscheid, and K. Randall, "Recommendations for discrete logarithm-based cryptography: elliptic curve domain parameters," *Computer Security Recource Center, SP 800-186 (draft)*, Oct. 2019.

[14] J. Y. Choi and K. Y. Shin, "A High Performance Modular Multiplier for ECC," *Journal of Institute of Korean Electrical and Electronics Engineers*, vol. 24, no. 4, pp. 961-968, Dec. 2020.

[15] M. Islam, S. Hossain, Shahjalal, K. Hasan, and Y. M. Jang, "Area-Time Efficient Hardware Implementation of Modular Multiplication for Elliptic Curve Cryptography," *IEEE Access*, vol. 8, pp. 73898-73906, Apr. 2020.

[16] H. Alrimeih and D. Rakhmatov, "Pipelined modular multiplier supporting multiple standard prime fields," *2014 IEEE 25th International Conference on Application- Specific Systems, Architectures and Processors*, Jun. 2014.

Hyeon-Jun Yang

2020 : BS degree in Electronic Engineering, Kumoh National Institute of Technology.
2020~ : Graduate student, Kumoh National Institute of Technology

Kyung-Wook Shin

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, 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)