다정도 CSA를 이용한 Dual-Field상의 확장성 있는 Montgomery 곱셈기

준회원 김 태 호*, 정회원 김 창 훈**°, 홍 춘 표*

Scalable Dual-Field Montgomery Multiplier Using Multi-Precision Carry Save Adder

Tae Ho Kim* *Associate Member,* Chang Hoon Kim** Chun Pyo Hong* *Regular Members*

요 약

본 논문에서는 새로운 다정도 캐리 세이브 가산기를 이용한 dual-field상의 확장성 있는 Montgomery 곱셈기를 제안한다. 제안한 구조는 유한체 $GF(p)$와 $GF(2^m)$상의 곱셈 연산을 수행한다. 제안한 다정도 캐리 세이브 가산기는 두 개의 캐리 세이브 가산기로 구성되며, w-비트의 워드를 처리하기 위한 하나의 캐리 세이브 가산기는 $n=\lceil w/b \rceil$ 개의 캐리 전파 가산기로 이루어진다. 여기서 b는 하나의 캐리 전파 가산기가 포함하는 dual-filed 가산기의 개수이다. 제안된 Montgomery 곱셈기는 기존의 연구결과에 비해 거의 동일한 시간 복잡도를 가지지만 낮은 하드웨어 복잡도를 가진다. 뿐만 아니라 제안한 연산기는 기존의 연구와 달리 연산의 종료 시 정확한 모듈러 곱셈의 결과를 출력한다. 더욱이 제안한 회로는 m과 w에 대해 높은 확장성을 가진다. 따라서 본 논문에서 제안한 구조는 암호응용을 위한 $GF(p)$와 $GF(2^m)$상의 곱셈기로서 매우 적합하다 할 수 있다.

Key Words : Montgomery Multiplication, Multi-Precision CSA, Scalable Multiplier, Finite Field, VLSI

ABSTRACT

This paper presents a scalable dual-field Montgomery multiplier based on a new multi-precision carry save adder (MP-CSA), which operates in both types of finite fields $GF(p)$ and $GF(2^m)$. The new MP-CSA consists of two carry save adders (CSA). Each CSA is composed of $n=\lceil w/b \rceil$ carry propagation adders (CPA) for a modular multiplication with w-bit words, where b is the number of dual field adders (DFA) in a CPA. The proposed Montgomery multiplier has roughly the same timing complexity compared with the previous result, however, it has the advantage of reduced chip area requirements. In addition, the proposed circuit produces the exact modular multiplication result at the end of operation unlike the previous architecture. Furthermore, the proposed Montgomery multiplier has a high scalability in terms of w and m. Therefore, it can be used to multiplier over $GF(p)$ and $GF(2^m)$ for cryptographic applications.

Ⅰ. 서 론

RSA[1], Diffie-Hellman 키교환 알고리즘[2], 타원곡선 암호시스템(Elliptic Curve Cryptosystems: ECC)[3]과 같은 암호 응용에서 모듈러 곱셈 및 지수승은 중요한 연산이다. 특히 모듈러 곱셈은 모듈러

※ 본 연구는 2007학년도 대구대학교 학술연구비에 의하여 수행되었음
* 대구대학교 정보통신공학과(thkim@dsp.daegu.ac.kr, cphong@daegu.ac.kr)
** 대구대학교 컴퓨터 · IT공학부(kimch@daegu.ac.kr) (°:교신저자)
논문번호 : KICS2007-08-354, 접수일자 : 2007년 8월 7일, 최종게재논문통보일자 : 2007년 12월 5일

지수 및 역원의 기본 연산으로 현재까지 많은 연구 결과가 발표되었다. 모듈러 곱셈을 위해 고전적인 방법, Montgomery 알고리즘[4], Barret 알고리즘[5] 등이 사용 되었다. 모듈러 곱셈 알고리즘 중, Montgomery 곱셈 방법은 내부 연산이 규칙적일뿐 만 아니라 나눗셈 연산은 쉬프트 연산으로 대체되기 때문에 하드웨어 구현에 매우 적합하다. 이러한 장점 때문에 다양한 형태의 변형된 Montgomery 알고리즘과 그에 따른 하드웨어 구현 방법이 연구되어 왔다[6-10].

확장성 있는 Montgomery 곱셈기 설계 방법과 하드웨어 구현 방법은[11-13]에서 소개되었다. 확장성 있는 구조는 입력값의 크기에 제한을 받지 않고 고정된 크기의 회로를 이용하여 연산을 수행한다. 데이터를 일정한 크기의 워드 단위로 나눈 후 워드 단위로 처리 및 전송한다. 데이터 크기가 m-비트이고 워드의 크기가 w-비트이면 워드의 개수는 $e=\lceil m/w \rceil$ 개 이다. 확장성 있는 구조는 워드의 크기가 커질수록 연산 시간을 단축할 수 있으나 하드웨어 복잡도가 증가한다. 그러나 면적 및 속도를 만족시키는 가장 적합한 워드 크기를 찾는다면 연산시간 및 하드웨어 복잡도에 있어 상충 관계를 개선할 수 있다. 확장성 있는 Montgomery 곱셈기의 연산시간 및 하드웨어 복잡도의 상충관계를 [13]에서 분석하였다.

최초에 Montgomery 곱셈 알고리즘은 홀수 모듈러스와 함께 모듈러 곱셈 알고리즘을 수행하는 효율적인 방법으로 제안되었다. 만약 모듈러스가 소수이면 Montgomery 곱셈 알고리즘은 유한체 $GF(p)$ 상에서 매우 효율적인 곱셈 연산을 수행할 수 있으며, 다항식기저를 사용하고 기약다항식이 선택되면 유한체 $GF(2^m)$상에서도 곱셈 연산을 수행할 수 있다[10-11]. 유한체 $GF(p)$상에서 Montgomery 곱셈 연산을 위해 캐리 전파 가산기(Carry Propagation Adder: CPA)를 이용하면 최대 처리 지연시간이 증가하고 비트수가 커지면 캐리 전파 문제가 발생한다. 이러한 문제를 해결하기 위해 기존에 제안된 Montgomery 곱셈기는 캐리 세이브 가산기(Carry Save Adder: CSA)를 이용한다. 그러나 CSA는 곱셈 결과가 합과 캐리로 구분되는 캐리 세이브(Carry Save: CS) 형태이기 때문에 정확한 곱셈결과를 얻기 위해서는 추가적인 m-비트의 가산기 회로 또는 m 클럭 사이클이 필요하다[14]. 최근 김 등[15]은 다정도(Multi-Precision: MP) CSA에 기반한 $GF(p)$상의 효율적인 곱셈기를 제안하였다. 다정도 CSA는 CSA와 CPA를 결합한 형태로 캐리 전파 문제를 해결하는 동시에 결과값을 보정하기 위한 클럭 사이클 수를 감소시킨다. 하지만 김 등의 구조는 m과 w에 대해 확장성을 제공하지 못한다.

본 논문에서는 새로운 다정도 CSA를 이용한 dual-field상의 확장성 있는 Montgomery 곱셈기를 제안한다. 제안한 구조는 유한체 $GF(p)$와 $GF(2^m)$ 상의 곱셈 연산을 수행하며 기존에 제안된 유사한 구조[11]에 비해 비교적 적은 플립플롭(Flip Flop: FF)을 사용한다. 또한 제안한 회로는 Savas 등[11]이 제안한 구조와 달리 결과값을 보정하기 위한 추가 회로를 필요로 하지 않고 보정에 필요한 클럭 사이클 수를 감소시킨다. 더욱이 제안한 곱셈기 회로는 덧셈을 위해 재사용될 수 있고 m과 w에 대해 높은 확장성을 가진다. 따라서 본 논문에서 제안한 구조는 암호응용을 위한 $GF(p)$와 $GF(2^m)$상의 곱셈기로서 매우 적합하다 할 수 있다.

Ⅱ. Montgomery 곱셈 알고리즘

정수 A, B와 모듈러스 p가 주어졌을 때, Montgomery 곱셈 알고리즘은

$$C=A \cdot B \cdot R^{-1} \bmod p, \tag{1}$$

를 계산한다. 여기서 $R=2^m$, $A, B<p<R$이고 p는 $m=\lceil \log_2 p \rceil$-비트이다. 본 논문에서는 p가 소수라고 가정한다. 식 (1)을 바탕으로 [알고리즘 1]과 같은 $GF(p)$상의 Montgomery 곱셈 알고리즘을 얻을 수 있다.

[알고리즘 1] $GF(p)$상의 Montgomery 곱셈 알고리즘[4]

Input : A, $B \in GF(p)$ and p
Output : $C \in GF(p)$
1. $C=0$
2. for $i=0$ to $m-1$
3. $\quad C=C+a_i \cdot B$
4. $\quad C=C+c_0 \cdot p$
5. $\quad C=C/2$
6. if $C \geq p$ then $C=C-p$
7. return C

만약 최종 결과값이 p보다 크면 단계 6과 같이

뺄셈 연산이 수행되어야 한다.

바이너리 확장 필드 $GF(2^m)$은 $GF(2)$상의 차수 $(m-1)$인 다항식으로 필드 원소가 표현된다. 두 다항식 $A(x)$, $B(x)$가 주어졌을 때, Montgo- mery 곱셈 연산은 아래와 같이 정의된다.

$$C(x) = A(x) \cdot B(x) \cdot x^{-m} \bmod p(x), \quad (2)$$

여기서 $C(x) \in GF(2^m)$이고, $p(x)$는 기약다항식이다. [알고리즘 1]의 $R = 2^m$은 x^m으로 대체된다. 식 (2)를 바탕으로 [알고리즘 2]와 같은 $GF(2^m)$상의 Montgomery 곱셈 알고리즘을 얻을 수 있다.

[알고리즘 2] $GF(2^m)$상의 Montgomery 곱셈 알고리즘[10]

Input : $A(x)$, $B(x) \in GF(2^m)$ and $p(x)$
Output : $C(x) \in GF(2^m)$
1. $C(x) = 0^{m+1}$
2. for $i = 0$ to $m-1$
3. $\quad C(x) = C(x) \oplus a_i \cdot B(x)$
4. $\quad C(x) = C(x) \oplus c_0 \cdot p(x)$
5. $\quad C(x) = C(x)/2$
6. return $C(x)$

[알고리즘 2]에서 0^m은 m-비트가 모두 0인 상태를 나타낸다. [알고리즘 1]의 단계 6에서 수행하는 추가적인 뺄셈 연산은 $GF(2^m)$상의 Montgomery 곱셈 알고리즘에서는 생략한다[10]. 또한 $GF(2^m)$상의 연산은 캐리가 발생하지 않기 때문에 덧셈 연산은 단순한 비트별 XOR 연산으로 대체할 수 있으며 $\oplus$ 기호로 나타낸다.

Ⅲ. 다정도 CSA에 기반한 워드-레벨 Montgomery 곱셈 알고리즘

워드-레벨 구조는 데이터를 일정한 크기의 워드 단위로 나눈 후, 워드 단위로 처리 및 전송한다. 데이터 크기가 m-비트이고 워드 크기가 w-비트이면 워드-레벨 Montgomery 곱셈 연산은 $e = \lceil (m+1)/w \rceil$개의 워드로 나누어진다.

3.1 다정도 CSA에 기반한 Montgomery 곱셈기

CPA에 기반한 Montgomery 곱셈기는 non-redundant 형식의 결과값을 바로 얻어올 수 있다는 장점이 있지만 비트수가 커지면 캐리 전파 문제가 발생한다. 또한, 공개키 암호 응용은 2048-비트 이상의 키를 요구하기 때문에 캐리 전파 지연은 심각한 문제가 될 수 있다. 반면에 CSA는 1-비트 전가산기(Full Adder: FA)와 동일한 캐리 전파 지연을 가지기 때문에 캐리 전파 문제가 발생하지 않고 고속의 연산이 가능하다. 그러나 내부 곱셈 결과가 CS 형태로 저장되기 때문에 non-redundant 형태의 결과값을 얻기 위한 추가적인 연산이 요구된다. 일반적으로 이러한 문제를 해결하기 위해 추가적인 회로 또는 클럭 사이클이 사용된다[14]. 최근 CSA와 CPA를 결합한 방식을 사용하는 하이브리드 형태의 다정도 CSA 구조가 김 등에 의해 제안되었다. 제안된 구조는 두 개의 CSA로 구성되며 데이터 크기가 m-비트이면 하나의 CSA는 $n = \lceil m/b \rceil$개의 CPA로 나누어진다. 여기서 b는 하나의 CPA가 포함하는 FA의 개수이다. 김 등에 의해 제안된 Montgomery 곱셈기는 n 클럭 사이클의 추가로 결과값을 보정한다.

3.2 $GF(p)$상의 워드-레벨 Montgomery 곱셈 알고리즘

본 논문에서는 확장성 있는 Montgomery 곱셈기를 설계하기 위해 새로운 $GF(p)$상의 워드-레벨 Montgomery 곱셈 알고리즘을 제안한다. 제안한 곱셈 알고리즘은 [알고리즘 3]과 같다. 두 개의 오퍼랜드 B(피승수), A(승수)와 모듈러스 p가 주어졌을 때, 워드-레벨 Montgomery 곱셈 알고리즘은 $ABR^{-1} \bmod p$를 계산한다. 여기서 B는 워드단위로, A는 비트단위로 읽어온다. [알고리즘 3]에서 w-비트인 B와 p는 b-비트의 CPA로 나누어지며 곱셈 연산에서 사용되는 벡터 A, B, p는 다음과 같이 나타낸다.

$$\begin{aligned} A &= (a_{m-1}, \cdots, a_1, a_0), \\ B &= (B^{(e-1)(b-1)}, \cdots, B^{(e-1)(1)}, B^{(e-1)(0)}, \cdots, \\ &\quad B^{(0)(b-1)}, \cdots, B^{(0)(1)}, B^{(0)(0)}), \\ p &= (p^{(e-1)(b-1)}, \cdots, p^{(e-1)(1)}, p^{(e-1)(0)}, \cdots, \\ &\quad p^{(0)(b-1)}, \cdots, p^{(0)(1)}, p^{(0)(0)}), \end{aligned} \quad (3)$$

여기서 워드와 CPA 인덱스는 위첨자로 표시되고 비트는 아래 첨자로 표시한다. 예를 들어, 벡터 B에서 j워드의 k번째 CPA의 i번째 비트는 $B_i^{(j)(k)}$로 나타낼 수 있다. 벡터 B의 i에서 j까지의 비트는

$B_{j\cdots i}$로 나타낸다($j>i$). 그리고 $(x|y)$는 두 비트시퀀스의 결합을 나타낸다.

제안된 곱셈 알고리즘은 A의 각 비트에 대해 TS, B, p의 부분합을 계산한다. B가 완전히 읽혀졌을 때, A의 다음 비트를 읽어온 후 계산을 반복한다. [알고리즘 3]은 내부 결과를 저장하기 위해서 CS 형태를 사용한다. 덧셈 결과는 m-비트의 $TS^{(j)}$와 n-비트의 $TC^{(j)}$에 저장된다. m번의 반복을 수행한 후 CS 형태의 곱셈 결과를 얻을 수 있으며 non-redundant 형태의 결과를 얻기 위한 n번의 추가적인 덧셈 연산을 수행한다(단계 3). 여기서 입력값 a_i와 B는 모두 0으로 인가한다. 경우에 따라서 m번의 반복이 끝난 후 출력되는 결과값이 모듈러스 p보다 클 수 있으며 2번의 반복수행을 통해 최종결과가 p미만이 되도록 조정할 수 있다[16]. 제안된 곱셈 알고리즘은 n번의 추가 클럭을 수행하기 때문에 n이 2보다 클 경우 뺄셈 연산을 제거할 수 있다.

[알고리즘 3] 다정도 CSA에 기반한 $GF(p)$상의 워드-레벨 Montgomery 곱셈 알고리즘

Input : A, $B \in GF(p)$ and p
Output : $C \in GF(p)$
1. $(TS, TC) = (0^m, 0^{n-1})$
2. $(CT, CB) = (0^m, 0^n)$
3. for $i=0$ to $(m-1)+n$
4. if $i>m-1$ then
5. $(a_i, B) = (0, 0^m)$
6. $(CT_0, TS^{(0)(0)}) = TS^{(0)(0)} + a_i \cdot B^{(0)(0)} + CT_{n-1}$
7. $parity = TS_0^{(0)(0)}$
8. $(CB_0, TS^{(0)(0)}) = TS^{(0)(0)} + odd \cdot p^{(0)(0)} + CB_{n-1}$
9. for $k=1$ to $n-1$
10. $(CT_k, TS^{(0)(k)}) = TS^{(0)(k)} + a_i \cdot B^{(0)(k)} + TC_{k-1}^{(0)}$
11. $(CB_k, TS^{(0)(k)}) = TS^{(0)(k)} + parity \cdot p^{(0)(k)} + CT_{k-1}$
12. $TC_{k-1}^{(0)} = TS_0^{(0)(k)} \cdot CB_{k-1}$
13. $TS_0^{(0)(k)} = TS_0^{(0)(k)} \oplus CB_{k-1}$
14. for $j=1$ to $e-1$
15. $(CT_0, TS^{(j)(0)}) = TS^{(j)(0)} + a_i \cdot B^{(j)(0)} + CT_{n-1}$
16. $(CB_0, TS^{(j)(0)}) = TS^{(j)(0)} + parity \cdot p^{(j)(0)} + CB_{n-1}$
17. for $k=1$ to $n-1$
18. $(CT_k, TS^{(j)(k)}) = TS^{(j)(k)} + a_i \cdot B^{(j)(k)} + TC_{k-1}^{(j)}$
19. $(CB_k, TS^{(j)(k)}) = TS^{(j)(k)} + parity \cdot p^{(j)(k)} + CT_{k-1}$
20. $TC_{k-1}^{(j)} = TS_0^{(j)(k)} \cdot CB_{k-1}$
21. $TS_0^{(j)(k)} = TS_0^{(j)(k)} \oplus CB_{k-1}$
22. $TS^{(j-1)} = (TS_0^{(j)} | TS_{w-1\cdots 1}^{(j-1)})$
23. $TS_{w-1}^{(e-1)} = 0$
24. $C = TS$
25. return C

3.3 다정도 CSA에 기반한 $GF(2^m)$상의 워드-레벨 Montgomery 곱셈 알고리즘

[알고리즘 4]는 $GF(2^m)$상의 워드레벨 곱셈 알고리즘을 나타낸다. $GF(2^m)$상의 연산은 캐리가 발생하지 않기 때문에 내부 덧셈 연산은 비트별 XOR 연산으로 대체할 수 있다. $GF(2^m)$상의 기약다항식은 $(m+1)$-비트이기 때문에 인덱스 i는 0에서 m까지 수행한다.

Ⅳ. Montgomery 곱셈 알고리즘의 병행성

[알고리즘 4] 다정도 CSA에 기반한 $GF(2^m)$상의 워드-레벨 Montgomery 곱셈 알고리즘[11]

Input : $A(x)$, $B(x) \in GF(2^m)$ and $p(x)$
Output : $C(x) \in GF(2^m)$
1. $TS = 0^{m+1}$
2. for $i=0$ to m
3. $TS^{(0)} = a_i \cdot B^{(0)} \oplus TS^{(0)}$
4. $parity = TS_0^{(0)}$
5. $TS^{(0)} = parity \cdot p^{(0)} \oplus TS^{(0)}$
6. for $j=1$ to $e-1$
7. $TS^{(j)} = a_i \cdot B^{(j)} \oplus TS^{(j)}$
8. $TS^{(j)} = parity \cdot p^{(j)} \oplus TS^{(j)}$
9. $TS^{(j)} = (TS_0^{(j)} | TS_{w-1\cdots 1}^{(j-1)})$
10. $TS_{w-1}^{(e-1)} = 0$
11. $C = TS$
12. return $C(x)$

4.1 다정도 CSA에 기반한 워드-레벨 Montgomery 곱셈기

워드-레벨 Montgomery 곱셈 알고리즘에서 i번째 반복의 $j=0$, $j=1$의 반복이 끝나면 $(i+1)$번째 반복이 즉시 수행된다. 이와 같은 계산을 나타내는 자료의존 그래프는 그림 1(a)와 같다. 태스크 A, B는 기본적으로 다음과 같은 동작을 수행한다. 1) TS, $a_i \cdot B$, p의 각 워드에 대한 덧셈 연산(p의 가산 여부는 TS의 최하위 비트에 의해 결정됨), 2) TS 워드의 1-비트 우측 쉬프트 연산. 쉬프트된 $TS^{(j-1)}$의 생성은 $TS^{(j)}$의 최하위 비트가 계산된 후에 가능하다[12]. 태스크 A는 2가지 동작에 추가적으로 $a_i \cdot B^{(0)} + TS^{(0)}$의 덧셈 결과로부터 TS의 최하위 비트를 저장한다([알고리즘 3]의 단계 7). 저장된 비트는 동일한 오퍼랜드의 다음 워드에 대해 p의 가산 여부를 결정하는데 사용된다. 자료의존 그래프는 한 개의 열에 대해 $(e+1)$개의 태스크를 가진다. 각 열의 태스크는 다른 처리기(Processing Element: PE)에서 계산되며 하나의 PE에서 생성된 데이터는 파이프라인 형태로 구성된 다음 PE로 전달한다. 각 태스크는 한 클럭 사이클에 계산된다. 제안된 구조는 $GF(p)$상의 Montgomery 곱셈을 위해 $t = \lceil (m+n)/k \rceil$ 커널 사이클이 필요하다. 여기서 k는 파이프라인상의 PE 개수이다. 반면에 $GF(2^m)$상의 Montgomery 곱셈은 $t = \lceil m/k \rceil$ 커널 사이클이 필요하다.

그림 1(b)는 두 개의 PE를 사용하는 4-비트 곱셈 연산을 나타낸다. 여기서 $w=1$이고, $n=2$이다. 이 경우에 Montgomery 곱셈 연산은 3 커널 사이클을 요구한다. 회색 상자는 non-redundant 형태의 결과값을 출력하기 위한 추가적인 반복을 나타낸다. 그림 1(c)는 그림 1(b)와 동일한 계산에서 3개의 PE를 사용한 경우를 나타낸다.

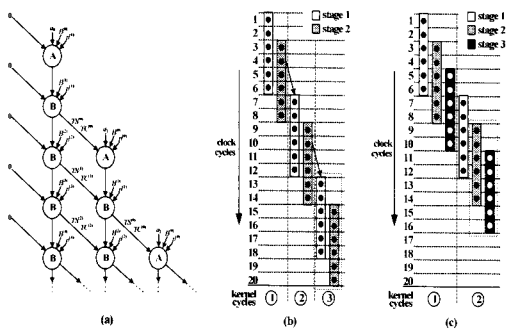


그림 1. (a) 확장성 있는 Montgomery 곱셈 알고리즘을 위한 자료의존 그래프, (b) 파이프라인된 2개의 PE를 이용한 4-비트 곱셈 연산($w=1$, $n=2$), (c) 3개의 PE를 이용한 (b)와 동일한 계산

이 경우, 추가적인 n번의 반복에 관계없이 2 커널 사이클에 동작한다.

제안된 Montgomery 곱셈 연산의 전체 수행 시간(Clock Cycle: CC)은 아래 식과 같다[11].

$$CC = \begin{cases} t(2k+1)+e-2 & \text{if } (e+1) \le 2k, \\ t(e+1)+2(k-1) & \text{otherwise,} \end{cases} \quad (4)$$

첫 번째 계산식은 주어진 워드의 개수보다 PE의 개수가 더 많은 경우로 1 커널 사이클에 결과값을 출력하고, 두 번째 계산식은 파이프라인내의 PE 개수가 워드의 개수보다 적은 경우로 결과값을 출력하기 위해 2 커널 사이클 이상을 요구한다.

4.2 다정도 CSA에 기반한 워드-레벨 가산기

제안된 구조는 곱셈기 회로를 재사용해서 워드-레벨 덧셈 연산을 수행한다. $GF(p)$상의 덧셈 연산은 CS 형태의 결과를 출력하기 때문에 non-redun- dant 형태로 보정하기 위한 추가연산이 필요하다. 그림 2(a)는 워드-레벨 덧셈 연산을 위한 자료의존 그래프를 나타낸다. 각 열은 $e = \lceil m/w \rceil$ 개의 태스크를 가지며 구분된 PE에 의해 계산을 수행한다. 첫 번째 PE는 오퍼랜드 A, B를 입력 받아서 CS 형태의 덧셈 결과를 출력한다. 다음 PE는 non-redundant 형태의 결과를 위해 추가적인 덧셈 연산을 수행한다. 여기서 a_i, B, p는 모두 0이 인가된다.

그림 2(b)는 4개의 PE를 사용한 6-비트 덧셈 연산을 나타낸다. 여기서 $w=1$, $n=2$이다. 회색 상자는 덧셈 연산을 위한 태스크를 나타내며, 두 번째 PE로부터 non-redundant 형태의 덧셈 연산 결과를 얻을 수 있다. 제안한 워드-레벨 가산기는 $GF(p)$와 $GF(2^m)$상의 덧셈 연산 결과를 $(e+n-1)$ 클럭 사이클 후에 출력한다.

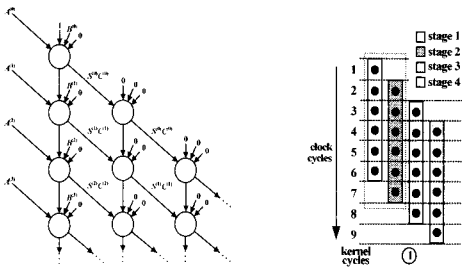


그림 2. (a) 워드-레벨 덧셈 연산을 위한 자료의존 그래프, (b) 파이프라인된 4개의 PE를 이용한 6-비트 덧셈 연산($w=1$, $n=2$)

V. 확장성 있는 Montgomery 곱셈기

그림 3은 파이프라인으로 구성된 확장성 있는 구조를 나타낸다. 파이프라인 구조는 커널로 불리어지며 k개의 PE로 구성된다. 파이프라인의 각 PE는 전달 받은 워드를 다음 PE로 전달한다. 1-비트 입력 a_i와 n-비트 내부 캐리 $C^{(j)}$를 제외한 모든 경로는 w-비트로 구성된다. 또한 데이터 B, p, S는 커널에 의해 워드단위로 처리한다. 회색 상자는 레지스터를 나타낸다.

5.1 워드-레벨 곱셈 및 덧셈 연산

제안된 구조는 워드-레벨 Montgomery 곱셈 연산을 수행하며 추가적인 가산기 없이 워드-레벨 덧셈 연산을 수행한다. 이와 같은 연산을 위해 멀티플렉서와 컨트롤 신호를 추가한다. 곱셈 또는 덧셈 연산은 컨트롤 신호(Add/Mult SEL)에 의해 수행된다. Add/Mult SEL이 0이면 덧셈 연산을 수행하고 n번째 PE에서 결과값을 얻을 수 있다. 반면에 Add/Mult SEL이 1이면 큐로부터 곱셈결과를 얻을 수 있다. 각 PE는 m-비트의 k-비트 쉬프트 레지스터로부터 오퍼랜드 A의 각 비트($a_i,\cdots,a_{i+k-1}$)를 가져온다. 시스템을 유연하게 하기 위해서 S와 C를 저장하기 위한 큐를 사용한다. 큐의 최대 길이는 메모리에 저장되는 워드의 최대 개수와 파이프라인 단계 수에 의존되며 아래와 같이 계산한다[11].

$$Q_{\max}=\begin{cases} e-2\cdot(k-1) & \text{if } (e+1)>2k, \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

5.2 처리기 구조

그림 4는 PE의 블록 다이어그램을 나타낸다. 데이터 패스는 파이프라인의 이전 단계로부터 워드 $S^{(j)}$, $C^{(j)}$, $B^{(j)}$, $p^{(j)}$를 전달 받아서 새로운 워드 $S^{(j-1)}$, $C^{(j-1)}$을 계산한다. 입력 B와 p를 지연시키는 것은 출력으로 $B^{(j-1)}$, $p^{(j-1)}$, $S^{(j-1)}$, $C^{(j-1)}$을 출력하기 위해서이다. 즉, 하나의 PE가 j번째 워드로 계산하면 다음 PE는 $(j-2)$번째 워드로 계산한다. 데이터 패스는 덧셈 또는 곱셈 연산 결과를 출력한다(덧셈: A, 곱셈: B). 제안된 구조는 곱셈 및 덧셈 연산을 모두 수행하기 위해 n개의 PE에 $(w+n)$-비트 멀티플렉서, w-비트 멀티플렉서, 컨트롤 신호를 추가한다.

데이터 패스는 두 개의 다정도 CSA로 구성되며 워드 크기가 w-비트이면 하나의 다정도 CSA는 $n=\lceil w/b \rceil$ 개의 CPA로 이루어진다. 여기서 b는 하나의 CPA를 구성하는 1-비트 DFA의 개수이다. 김 등이 제안한 다정도 CSA는 확장성 있는 구조를 지원하지 못하지만 본 논문에서 제안된 새로운 다정도 CSA는 확장성 있는 구조를 지원하기 위해 캐리를 최하위 비트로 피드백(feedback)하는 형태로 설계하였다. 그림 5는 $w=4$이고 $b=2$인 데이터 패스를 나타낸다. 각 4-비트 다정도 CSA는 두 개의 2-비트 CPA로 나누어지며 하나의 CPA는 두 개의 1-비트 DFA로 구성된다. DFA는 캐리를 가지는 $GF(p)$상의 덧셈 연산과 캐리를 가지지 않는 $GF(2^m)$상의 덧셈 연산을 모두 수행한다[11]. FSEL은 유한체 $GF(p)$와 $GF(2^m)$을 선택한다. FSEL이 1이면 DFA는 $GF(p)$상의 덧셈 연산을 FSEL이 0이면 $GF(2^m)$상의 덧셈 연산을 수행한다[11]. 데이터 패스는 $S^{(0)}+a_i\cdot B^{(0)}$의 최하위 비트를 로컬 컨트

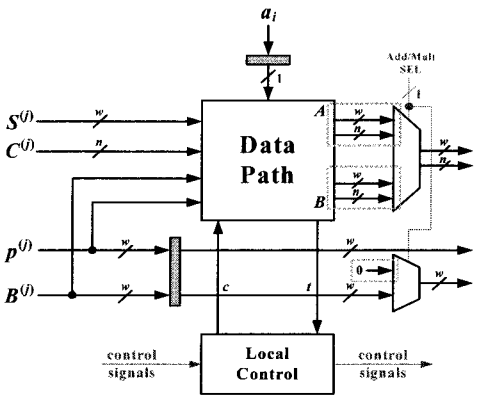


그림 4. PE 블록 다이어그램

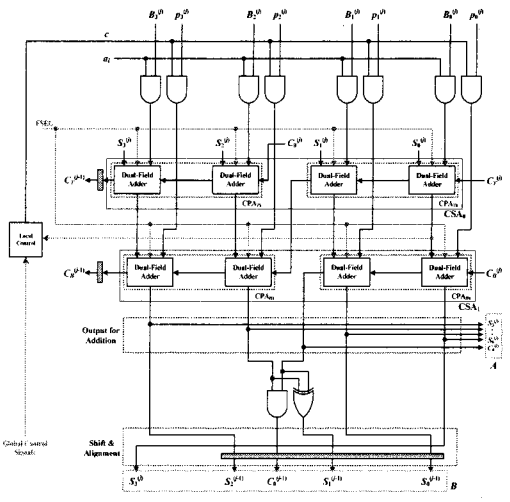


그림 5. PE 데이터 패스($w=4$, $b=2$, $n=2$)

표 1. Dual-field상의 확장성 있는 Montgomery 곱셈기 성능 비교

	Savas 등[11]	그림 3
Latency (kernel cycles)	$\lceil m/k \rceil$	$\lceil (m+n)/k \rceil$
Circuit Requirement (datapath)	DFA : $2w$ AND2 : $2w$ XOR2 : 0 FF : $2w$	DFA : $2w$ AND2 : $2w+(n-1)$ XOR2 : $n-1$ FF : $w+n$
Circuit Requirement (kernel)	FF : $8wk-4w$	FF : $6wk+2nk-3w-n$
Circuit Requirement (adder)	N/A	MUX2 : $2n+2$
Results Form	Carry Save	non-redundant
Operations	Multiplication	Multiplication and Addition

AND2 : 2-to-1 AND gate
XOR2 : 2-to-1 XOR gate
MUX2 : 2-to-1 MUX

롤로 사용하며, 이 비트는 p의 가산여부를 결정하는 컨트롤 신호를 생성하는데 사용된다.

데이터 패스는 m-비트 덧셈값과 n-비트 캐리를 저장하기 위해 $(m+n)$-비트 레지스터를 가진다. 또한 덧셈 연산을 수행하기 위한 CSA_1의 출력(A) 또는 곱셈 연산을 위한 결과값(B)을 출력한다.

Ⅵ. 결 론

본 논문에서는 워드-레벨 Montgomery 곱셈 알고리즘을 유도하고 이를 바탕으로 새로운 다정도 CSA를 이용한 dual-field상의 확장성 있는 Montgomery 곱셈기를 제안하였다. 제안된 구조는 Montgomery 곱셈 연산을 위해 t 커널 사이클 후에 완전한 결과를 출력한다. 표 1에 본 논문에서 제안된 구조와 기존에 제안된 구조의 성능을 비교하였다. [11]의 데이터 패스는 내부 결과를 저장하기 위해 $2w$-비트 FF을 가지는 반면에 본 논문에서 제안된 구조는 $(w+n)$-비트 FF을 가진다. 일반적으로 n은 w보다 작은 값을 사용하기 때문에 기존에 제안된 구조에 비해 비교적 적은 FF 개수를 가진다. 또한 제안된 구조는 CS 형태의 결과를 non-redundant 형태로 변환하기 위한 추가적인 회로를 요구하지 않는다. 또한 곱셈 및 덧셈 연산을 모두 수행한다. 표 2에 [11]과 제안된 구조에 대해 w, k, m의 다양한 선택에 따른 커널 사이클, 클럭 사이클, FF 개수를 비교

표 2. $GF(p)$상의 곱셈기 성능 비교

w	k	m	e	커널 사이클		클럭 사이클		FF 개수	
				[11]	그림 3	[11]	그림 3	[11]	그림 3
16	16	192	12	12	13	406	439	1984	1612
		224	14	14	15	474	507	1984	1612
		266	17	17	17	576	576	1984	1612
		384	24	24	25	814	847	1984	1612
		521	33	33	33	1152	1152	1984	1612
16	24	192	12	8	9	402	451	3008	2444
		224	14	10	10	502	502	3008	2444
		266	17	12	12	603	603	3008	2444
		384	24	16	17	806	855	3008	2444
		521	33	22	22	1109	1109	3008	2444
32	16	192	6	12	13	400	433	3968	3100
		224	7	14	15	467	500	3968	3100
		266	9	17	17	568	568	3968	3100
		384	12	24	25	802	835	3968	3100
		521	17	33	33	1104	1104	3968	3100

하였다. 표 2에서 $n=4$로 가정하며 ECC를 위해 NIST[17]에서 권고하는 다섯 가지 $GF(p)$, $m \in \{192, 224, 266, 384, 521\}$, 표준 필드 크기를 선택하였다. 표 2에 나타나듯이 두 개의 구조가 동일한 커널 사이클을 가지면 각 곱셈 연산은 동일한 클럭 사이클에 수행된다. 또한 w 또는 k가 증가할수록 [11]과 제안된 곱셈기의 FF 개수 차이는 커진다. $GF(p)$상의 곱셈 연산과 달리 $GF(2^m)$상의 곱셈 연산은 두 개의 구조가 동일한 시간 및 하드웨어 복잡도를 가진다.

본 논문에서 제안된 구조는 크게 다음과 같은 세 가지 장점을 가진다. 1) 기존에 제안된 구조에 비해 비교적 적은 FF 개수를 가진다. 2) non-redundant 형태의 결과값을 출력하기 위한 추가 회로가 필요하지 않고, 보정에 필요한 클럭 사이클 수를 감소시켰다. 3) 제안된 곱셈기 회로를 재사용해서 덧셈 연산을 수행한다. 또한 본 논문에서 제안된 다정도 CSA는 기존에 제안된 다정도 CSA와 달리, 확장성 있는 구조로 설계되었다. 따라서 제안된 다정도 CSA에 기반한 dual-field상의 확장성 있는 Montgomery 곱셈기는 암호 디바이스의 곱셈 및 덧셈 연산을 위해 적합하며, 특히 스마트카드나 휴대용 장치와 같은 저면적 응용에 매우 적합하다.

참 고 문 헌

[1] J.-J. Quisquater and C. Couvreur, "Fast Decipherment Algorithm for RSA Public-key Cryptosystem," IEE Electronics Letters, Vol. 18, No. 21, pp. 905-907, 1982.

[2] W. Diffie and M.E. Helman, "New Directio- ns in Cryptography," IEEE Transactions on Information Theory, Vol. 22, pp. 644-654, 1976.

[3] N. Koblits, "Elliptic Curve Cryptosystems," Mathematics of Computation, Vol. 48, No. 177, pp. 203-209, 1987.

[4] P.L. Montgomery, "Modular Multiplication without Trial Division," Math. Computation, Vol. 44, pp. 519-521, 1985.

[5] P. Barrett, "Implementing the Rivest Shamir and Adleman Public Key Encryption Algori- thm on a Standard Digital Signal Processor," Lecture Notes in Computer Science, Vol. 263, pp. 311-323, 1987.

[6] C.D. Walter, "Systolic Modular Multiplication," IEEE Trans. on Computers, Vol. 42, pp. 376-378, 1993.

[7] S.E. Eldridge and C.D. Walter, "Hardware Implementation of Montgomery's Modular Multiplication Algorithm," IEEE Trans. on Computers, Vol. 42, No. 6, pp. 693-699. 1993.

[8] S.E. Eldridge, "A Faster Modular Multiplication Algorithm," Intern. J. Computer Math, Vol. 40, pp. 63-68, 1991.

[9] C.Y. Su, S.A. Hwang, P. S. Chen, and C. W. Wu, "An Improved Montgomery's Algo- rithm for High-speed RSA Public-key Cryptosystem," IEEE Trans. on Very Large Scale Integration (VLSI) Systems, Vol. 7, No. 2, 1999.

[10] Ç.K. Koç and T. Acar, "Montgomery Multiplication in $GF(2^k)$," Designs, Codes and Cryptography, Vol. 14, pp. 57-69, 1998.

[11] E. Savas, A.F. Tenca, and Ç.K. Koç, "A Scalable and Unified Multiplier Architecture for Finite Fields $GF(p)$ and $GF(2^m)$," Lecture Notes in Computer Science, Vol. 1965, pp. 277-292, 2000.

[12] A. Tenca and Ç.K. Koç, "A Scalable Architecture for Modular Multiplication Based on Montgomery's Algorithm," IEEE Trans. on Computers, Vol. 52, No. 9, pp. 1215-1221, 2003.

[13] A.F. Tenca and Ç.K. Koç, "A Scalable Architecture for Montgomery Multiplication," Lecture Notes in Computer Science, Vol. 1717, pp. 94-108, 1999.

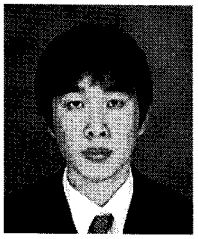
[14] J.C. Ha and S.J. Moon, "A Design of Modular Multiplier Based on Multi-Precision Carry Save Adder," Joint Workshop on Information Security and Cryptology (JWISC' 2000), pp. 45-51, 2000.

[15] 김대영, 이준용, "개선된 다정도 CSA에 기반한 모듈라 곱셈기 설계," 정보과학회논문지 : 시스템 및 이론, 제33권, 제3·4호, pp. 223- 230, 2006.

[16] T. Blum and C. Paar, "Montgomery modular exponentiation on reconfigurable hardware," in Proc. 14th IEEE Symp. on Computer Arithmetic, pp. 70-77, 1999.

[17] NIST, Recommended elliptic curves for fede- ral government use, May 1999. http://csrc.nist.gov/encryption.

김 태 호 (Tae Ho Kim) 준회원

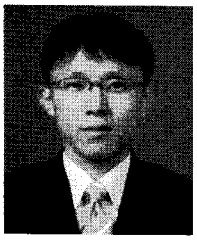


2006년 2월 대구대학교 컴퓨터·IT공학부 (학사)

2006년 3월~현재 대구대학교 정보통신공학과 석사과정

<관심분야> 암호 시스템, Embedded System, RFID/USN 보안

김 창 훈 (Chang Hoon Kim) 정회원



2001년 2월 대구대학교 컴퓨터정보공학부 (학사)

2003년 2월 대구대학교 컴퓨터정보공학과 (석사)

2006년 8월 대구대학교 컴퓨터정보공학과 (박사)

2006년 9월 대구대학교 정보통신공학부 BK21, 연구교수

2007년 9월~현재 대구대학교 컴퓨터IT공학부, 전임강사

<관심분야> 암호 시스템, Embedded System, RFID/USN 보안

홍 춘 표 (Chun Pyo Hong) 정회원

1978년 2월 경북대학교 전자 공학과 (학사)

1986년 12월 Georgia Institute of Technology ECE (석사)

1991년 12월 Georgia Institute of Technology ECE (박사)

1994년 9월~현재 대구대학교 정보통신공학부 교수

<관심분야> DSP 하드웨어 및 소프트웨어, 컴퓨터 구조, VLSI 신호처리, Embedded System