Fermat의 소정리를 응용한 IDEA 암호 알고리즘의 고속 하드웨어 설계

(A High-Speed Hardware Design of IDEA Cipher Algorithm by Applying of Fermat's Theorem)

최 영 민 † 권 용 진 ††

(Young-Min Choi) (Yong-Jin Kwon)

요 약 본 논문에서는 DES 보다 암호학적 강도가 뛰어난 것으로 알려져 있는 IDEA 알고리즘에서 가장 많은 계산량이 요구되는 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산을 페르마의 소정리를 응용하여 IDEA의 처리 속도를 향상시키는 방법을 제안한다. 본 논문에서 제안하고 있는 페르마 소정리를 응용한 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산 방식은 기존의 확장 유클리드 알고리즘을 적용한 방식보다 필요한 연산 횟수를 약 50%정도 감소시킨다. 제안한 곱셈의 역원 방식을 적용하여 단일 라운드 반복 구조로 설계한 IDEA 하드웨어의 최대 동작 주파수는 20 MHz이고 게이트 수는 118,774 gate이며 처리 속도는 116 Mbits/sec이다. 동일한 단일 라운드 반복 구조로 설계된 H.Bonnenberg에 의한 기존의 연구보다 처리속도가 약 2배정도 빠르다. 이것은 본 논문에서 제안한 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산 방식이 속도면에서 효율적임을 나타내고 있다.

Abstract In this paper, we design IDEA cipher algorithm which is cryptographically superior to DES. To improve the encryption throughput, we propose an efficient design methodology for high-speed implementation of multiplicative inverse modulo $2^{16}+1$ which requires the most computing powers in IDEA. The efficient hardware architecture for the multiplicative inverse is derived from applying of Fermat's Theorem. The computing powers for multiplicative inverse in our proposal is a decrease 50% compared with the existing method based on Extended Euclid Algorithm. We implement IDEA by applying a single iterative round method and our proposal for multiplicative inverse. With a system clock frequency 20MHz, the designed hardware permits a data conversion rate of more than 116 Mbit/s. This result show that the designed device operates about 2 times than the result of the paper by H. Bonnenberg et al. From a speed point of view, our proposal for multiplicative inverse is proved to be efficient.

1. 서 론

전자상거래 및 홈뱅킹이 인터넷이나 전화망과 같은 공중망을 통하여 행해지고 있고 인터넷을 통한 전자메일 및 파일의 교환이 빈번해짐에 따라 거래 정보, 전자메일, 파일을 변조 및 유출하여 악용하려는 시도들이 끊임없이 이어지고 있다. 따라서 이러한 시도로부터 정보의 기밀성을 유지하기 위해서 암호를 통한 정보보안의 필요성이 크게 부각되고 있다. 그러나 기존의 소프트웨어적인 방식으로 구현된 암호 알고리즘은 처리 속도가 느린 단점으로 인해, 대량의 데이터를 실시간으로 처리하는 여러 응용분야에 실제적으로 적용하기란 어렵다. 이를 해결하기 위해 최근 암호 알고리즘을 하드웨어로 구현하여 처리 속도를 향상시키는 연구가 활발히 진행중이다[1][2].

본 논문에서는 이미 잘 알려진 DES보다 암호학적으로 더 안전한 IDEA를 단일 라운드 반복 구조로 하드웨

- 본 논문은 과학기술부·한국과학재단지정 「한국항공대학교 인터넷정보검색연구센터」의 연구비 지원으로 (일부) 수행되었습니다.

† 비 회 원 : 한국항공대학교 통신정보공학과
choiym@mail.hangkong.ac.kr
†† 학생회원 : 한국항공대학교 통신정보공학과 교수
yjkwon@tikwon.hangkong.ac.kr

논문접수 2001년 1월 5일
심사완료 : 2001년 9월 14일

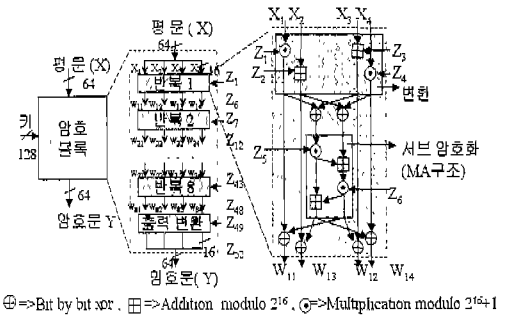
어 설계한다. IDEA의 처리속도를 증가시키기 위해서, 계산량적으로 가장 많은 연산량이 요구되는 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산에 대해 페르마 소정리를 응용한 새로운 방식을 제안한다. 이 역원 연산 방식을 적용하여 IDEA 알고리즘을 하드웨어로 구현하며 기존 연구와 처리 속도면에서 성능을 비교하여 본 연구의 타당성을 검증한다.

본 논문에서 설계하는 IDEA 암호 알고리즘은 VHDL로 기술하고 Synopsys의 Design Compiler를 이용하여 SAMSUNG 0.5 μm CMOS library로 합성되며 SAUMSUNG의 Design Kit인 SADAS와 Cadencde의 Verilog-XL 시뮬레이터를 사용하여 설계의 정확성을 검증한다.

본 논문의 구성은 다음과 같다. 2장에서는 IDEA의 동작 구조에 대해 소개하고 3장에서는 IDEA의 내부 연산과 암·복호 알고리즘에 대한 설계 방식을 기술하고 4장에서는 IDEA의 구현과 성능 비교를 설명하며 5장에서 결론을 맺는다.

2. IDEA의 동작 구조

IDEA는 64 비트의 데이터 블록을 암·복호화하기 위해 128 비트의 키 데이터를 사용하는 블록 암호 방식으로 Xuejia Lai와 James Massey에 의해 제안되었다[3]. 암·복호 블록은 8개의 반복과 출력 변환으로 이루어져 있고 각각의 반복은 치환과 곱셈·덧셈 구조로 이루어져 있으며 실제로 그 내부는 암호학적 강도를 증대시키기 위해 대수적 구조가 서로 다른 비트 대 비트 XOR 연산과 모듈러 2^{16}에 대한 덧셈과 모듈러 $2^{16}+1$에 대한 곱셈 연산의 조합으로 구성된다. 즉 반복은 평문과 서브키를 입력받아 비트 대 비트 xor 연산과 모듈러 2^{16}에 대한 덧셈과 모듈러 $2^{16}+1$에 대한 곱셈 연산의 조합으로 출력문을 생성하고 이 출력 문은 다시 다음 반복의 입력으로 사용된다. 키 데이터에 따라 암호문이 출력되기도 하고, 복호문이 출력되기도 한다. 이 동일한 반복 구조를 8회 수행하여 암·복호문이 생성되게 된다. IDEA의 암호 과정과 복호 과정에서 사용되는 동작 구조는 그림 1에 나타난다.



⊕=>Bit by bit xor , ⊞=>Addition modulo 2^{16} , ⊙=>Multiplication modulo $2^{16}+1$

그림 1 IDEA의 동작 구조

IDEA는 128비트의 키 데이터를 입력받아 52개의 16비트 서브키를 생성한다. 서브키에는 암호화에 사용되는 암호 서브키와 복호화에 사용되는 복호 서브키가 있으며 키 생성 방식은 그림 2에 나타나 있다. 암호 키 생성 방식은 128비트 키 입력을 16비트씩 분할하여 8개의 서브키를 만들고 이 과정이 끝나면 25비트씩 왼쪽으로 자리 이동하여 나머지 서브키를 만든다. 이러한 과정을 52개의 서브키가 만들어질 때까지 반복 수행하면 된다. 복호 키 생성 방식은 암호 키 생성 방식에 좀 더 복잡한 연산이 첨가되어 있다. 즉 복호 라운드 i에서 사용되는 처음 4개의 복호 서브키는 $10-i$ 번째 암호 라운드에서의 처음 4개의 서브키로부터 유도되고 마지막 두 개의 서브키는 $9-i$ 번째 암호 키에서 유도된다. 여기서 처음과 네 번째의 복호 서브키는 처음과 네 번째의 암호 서브키의 모듈러 $2^{16}+1$에 대한 곱셈의 역원이고 두 번째와 세 번째의 복호 서브키는 두 번째와 세 번째의 암호 서브키의 모듈러 2^{16}에 대한 덧셈의 역원이 된다[2][3].

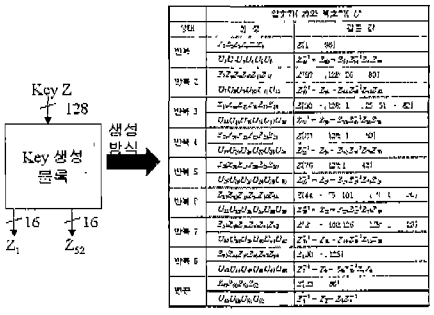


그림 2 서브키 생성 방식

3. IDEA의 효율적 설계 방식

3.1 내부 연산 회로 설계 방식

IDEA 알고리즘을 효율적으로 설계하기 위해서 IDEA에서 정의된 연산들의 복잡도를 알아본다. 다음은 계산량 측면에서 IDEA 알고리즘에서 연산들을 복잡도를 나타낸다.

① 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산

② 모듈러 $2^{16}+1$에 대한 곱셈 연산
③ 모듈러 2^{16}의 덧셈의 역원 연산
④ 모듈러 2^{16}의 덧셈 연산
⑤ 비트 대 비트로 xor 연산

②, ④, ⑤ 연산은 암·복호화 과정에서 라운드 처리에 사용되는 연산이며 ①, ③ 연산은 복호화 과정에서 서브 키 생성에 사용되는 연산이다. ①, ②, ③, ④, ⑤ 연산의 입·출력은 16비트 정수로 되어 있다[2].

일반적으로 ③, ④, ⑤ 연산은 시스템 설계시 자주 사용되는 연산으로써 하드웨어 설계에 있어서 라이브러리를 사용하면 되므로 설계가 용이하다. 그러나 ①과 ② 연산과 같은 모듈러 $2^{16}+1$와 관련된 연산은 residue number system, 신호 처리, IDEA와 같은 암호 분야에서만 사용되는 특별한 연산으로써 라이브러리가 지원하지 않는다[9]. 뿐만 아니라 이 연산은 복잡도가 높아서 시스템의 성능에 큰 영향을 주므로 이 연산을 어떻게 설계하느냐에 따라 설계한 하드웨어의 면적과 동작 속도가 크게 좌우된다. 따라서 IDEA가 고성능으로 동작하기 위해서는 ①과 ② 연산이 최적으로 동작할 수 있도록 내부 구조를 설계해야 한다.

본 논문에서는 두 번째로 복잡한 ② 연산을 기존의 연구 중 가장 우수하다고 평가되는 Low-high 알고리즘[7]~[9]을 사용하여 하드웨어 설계를 하고 있다. 그림 3은 모듈러 $2^{16}+1$에 대한 곱셈에 Low-high 알고리즘의 적용을 보여준다.

$Z = X \times Y \mod F_4 \quad [F_4 = 2^{16}+1]$

1. $D = X \times Y$
   $D.lower = D(15\ downto\ 0)$
   $D.higher = D(31\ downto\ 16)$
2. *If* $(D.lower \geq D.higher)$ *then*
   $Z = D.lower - D.higher$
   else
   $Z = D.higher - D.lower + F_4$

그림 3 $X \cdot Y \bmod F_4$에 대한 Low-high 알고리즘

한편 본 논문에서는 IDEA의 연산 속도를 증가시키기 위하여 복잡도가 가장 높은 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산에 대해 페르마의 소정리를 응용한 새로운 설계 방식을 제안한다. 먼저 기존의 모듈러 p에 관한 곱셈의 역원 설계 방법인 확장 유클리드 알고리즘에 대해서 간단히 기술한다.

확장 유클리드 알고리즘은 $\gcd(d, f)=1$이면 d는 모듈러 f에 대한 곱셈의 역원을 가짐을 이용하여 d의 곱셈의 역원을 찾는 방법[6]으로 그림 4에 확장 유클리드 알고리즘이 나타나 있다. 여기서 Y[1]은 $2^{16}+1$로 X[1]은 0에서 2^{16}사이의 입력 데이터로 설정하여 유클리드 알고리즘을 X[1]과 Y[1]의 최대 공약수가 1이 될 때까지 반복한다. X[1]의 모듈러 $2^{16}+1$에 대한 곱셈의 역원을 구하기 위해 $Q_1[i], Q_2[i], T_1[i], T_2[i]$을 계산해야 한다. 최종적으로 구해진 $T_2[9]$가 모듈러 $2^{16}+1$에 대해서 X[1]의 곱셈의 역원이 된다. 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산에 대한 기존의 확장 유클리드 알고리즘과 본 논문에서 제안한 방식을 비교하기 위해서 확장 유클리드 알고리즘에서 사용되는 곱셈 연산과 나눗셈 연산과 잉여 연산을 동일 계산량으로 평가한다. 확장 유클리드 알고리즘은 두 개의 잉여 연산과 두 개의 나눗셈 연산과 두 개의 곱셈연산을 9회 반복으로 수행되므로 총 54회의 곱셈 횟수가 필요하다.

$T_1[1]$	$Q_1[1]$	$Y[1]$	$X[1]$	$Q_2[1]$	$T_2[1]$
$T_1[9]$	$Q_1[9]$	$Y[9]$	1	$Q_2[9]$	$T_2[9]$

여기서, $Y[1] = 2^{16}+1$, $X[1]$ $0 \sim 2^{16}$ 임의의 수.
$T_2[9]$: $X[1]$의 법 $2^{16}+1$에 대한 곱셈에 대한 역원
$Y[i] = Y[i-1]\%X[i-1]$, $Q_1[i] = Y[i]/X[i]$, $T_1[i] = T_1[i-1] + T_2[i] \times Q_2[i]$,
$X[i] = X[i-1]\%Y[i]$, $Q_2[i] = X[i-1]/Y[i]$, $T_2[i] = T_2[i-1] + T_1[i] \times Q_1[i]$,

그림 4 모듈러 $2^{16}+1$에 대한 곱셈의 역원의 확장 유클리드 알고리즘

본 논문에서는 $2^{16}+1$가 페르마 소수라는 점에 아이디어를 얻어서 페르마 소정리를 응용하여 모듈러 $2^{16}+1$에 대한 곱셈의 역원를 구하는 문제를 해결하고 있다. 페르마의 소정리는 p가 소수일 때 p와 a의 최대 공약수가 1이면 a의 $p-1$ 멱승은 모듈러 p에 관해 1과 합동임을 뜻한다[6]. 여기서 a^{p-1}을 a와 a^{p-2}와 곱으로 표현하면 a와 a^{p-2}의 곱셈은 모듈러 p에 대해 1과 합동이 되므로 a와 a^{p-2}는 모듈러 p에 대해 서로 곱셈의 역원관계이다. p는 $2^{16}+1$이므로 a^{p-2}은 $a^{2^{16}-1}$가 되고 $2^{16}-1$를 하드웨어로 설계하기 위해 이진법으로 나타내면 최종적으로 $a^{1\cdots1}$ 형태로 변환된다. 이것은 실제로 모듈러 $2^{16}+1$에 대한 a와 a^2의 규칙적인 곱셈 형태로 변환 가능하며 이러한 곱셈 형태가 15회 반복된다. 그래서 본 논문에서는 모듈러 $2^{16}+1$에 대한 곱셈 역원 연산을 두 개의 모듈러 $2^{16}+1$에 대한 곱셈기로 구성하고 두 개의 곱셈기의 곱을 15회 반복 적용하는 하드웨어 구조로 구현한다. 계산량적으로 필요한 곱셈 횟수는 총 30회이다. 다음은 이러한 수식 변환 과정을 나타내고 있다.

$$
\begin{aligned}
a^{-1} &= a^{p-2} && \bmod p \\
a^{-1} &= a^{2^{16}-1} && \bmod (2^{16}+1) \\
&= a^{11\cdots 11} && \bmod (2^{16}+1) \\
&= a^{1+2+2^2+\cdots+2^{15}} && \bmod (2^{16}+1) \\
&= a(a\cdots(a(a^2))^2\cdots)^2 && \bmod (2^{16}+1)
\end{aligned}
$$

표 1은 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산에 대하여 기존의 확장 유클리드 알고리즘 방식과 본 논문에서 제안하는 방식과의 비교한 표로써 본 논문의 방식이 확장 유클리드 알고리즘을 사용하는 방법보다 약 1/2배 작은 계산량이 요구됨을 나타내고 설계에 있어서도 용이함을 나타내고 있다. 왜냐하면 유클리드 알고리즘은 18개의 잉여 연산과 18회의 나눗셈 연산 그리고 18회의 곱셈 연산이 필요했지만, 본 논문에서 제안하는 설계 방식을 사용하면 30회의 곱셈 연산만으로 해결되기 때문이다.

표 1 모듈러 $2^{16}+1$에 관한 곱셈의 역원 연산에 대하여 확장 유클리드 알고리즘과 본 논문의 방식과의 비교

설계 방법	계산량 (필요한 곱셈 횟수)	필요한 연산
확장유클리드 알고리즘	54개	곱셈, 나눗셈, 잉여 연산
본 논문	30개	곱셈 연산

3.2 IDEA 알고리즘 설계 방안

일반적으로 블록 암호 알고리즘의 암·복호 처리 부분은 동일한 동작을 수행하는 여러 개의 라운드로 구성되어 있다. 이러한 암·복호 처리 부분을 하드웨어 설계할 때의 설계 방식으로는 라운드 전체를 직렬로 연결하는 전 라운드 구현형 방식과 하나의 단일 라운드로 여러 번 반복 수행하는 단일 라운드 반복형 방식이 있다[12]. 본 논문에서는 8회의 라운드 동작으로 구성되는 IDEA의 동작 구조를 하나의 단일 라운드 프로세서로 구현하고 이를 8회 반복 수행하도록 하는 단일 라운드 반복 구조로 하드웨어 설계한다. 이 방법은 전 라운드 구현형보다 회로의 크기를 줄 일수 있는 장점이 있다.

4. 하드웨어 구현 및 성능 평가

4.1 하드웨어 구현 절차

IDEA 암호 알고리즘을 논리적으로 분석하고 설계 사양을 결정 한 후 그 기능에 적합한 전체 구조를 독자적으로 블록 설계하고 있다. 실제로 그림 6에 설계한 IDEA의 구조가 나타나고 있다. 그리고 각 블록의 내부 구조를 RTL 레벨에서 VHDL로 모델링하고 이를 기능 검증 수행한 후 Synopsys의 Design Compiler에 면적과 타이밍에 대한 제약 조건을 가하여 논리 설계된 블록을 SAMSUNG 0.5 μm CMOS library로 합성한다. SAMSUNG Design Kit인 SADAS을 구동하여 게이트 딜레이를 추출하여 Cadence의 Verilog-XL 시뮬레이터에서 타이밍 시뮬레이션을 수행한다.

4.2 하드웨어 내부 구조 설계

4.2.1 암·복호 처리 부분

IDEA를 단일 라운드 반복 구조로 설계하기 위해서는 라운드, 출력변환, 멀티플렉서 및 레지스터가 필요하다. 레지스터는 매 라운드의 결과 값과 최종 라운드 결과 값을 저장하기 위해 필요하며 데이터 길이는 64비트이다. 멀티플렉서는 입력된 64 비트의 평문과 각 라운드의 출력 값을 택일하기 위해 필요하다. 라운드는 대수학적으로 서로 다른 세 가지의 연산으로 구성된다. 즉 라운드는 6개의 16비트 대 비트 xor 연산과 4개의 모듈러 2^{16}에 대한 덧셈와 4개의 모듈러 $2^{16}+1$ 대한 곱셈으로 이루어진 조합 회로이다(그림 1). 비트 대 비트 xor 연산과 덧셈 연산은 설계 라이브러리를 이용하여 간단하게 설계하고 곱셈회로는 기존의 Low-high 알고리즘을 채용하여 설계한다.

4.2.2 키 생성 부분

키 생성부를 하드웨어 설계하기 위해서는 시프트 레지스터, 역원 모듈, 메모리 및 멀티플렉서가 필요하다. 메모리는 암·복호 처리부분의 각 라운드에서 필요한 서브키를 저장하기 위해서 필요하며 멀티플렉서는 출력될 서브키가 암호 키인지 복호 키인지를 선택하기 위해서 필요하다. 그리고 역원 모듈은 모듈러 2^{16}에 대한 덧셈의 역원과 모듈러 $2^{16}+1$에 대한 곱셈의 역원으로 구성된다. 2^{16}에 대한 덧셈에 대한 역원은 뺄셈 회로를 이용하여 간단히 설계한다. 모듈러 $2^{16}+1$에 대한 곱셈 역

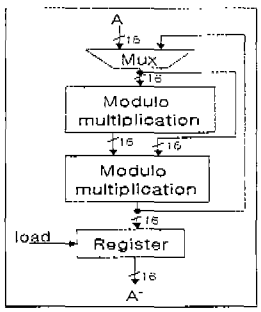


그림 5 모듈러 $2^{16}+1$에 대한 곱셈 역원기의 하드웨어 구조

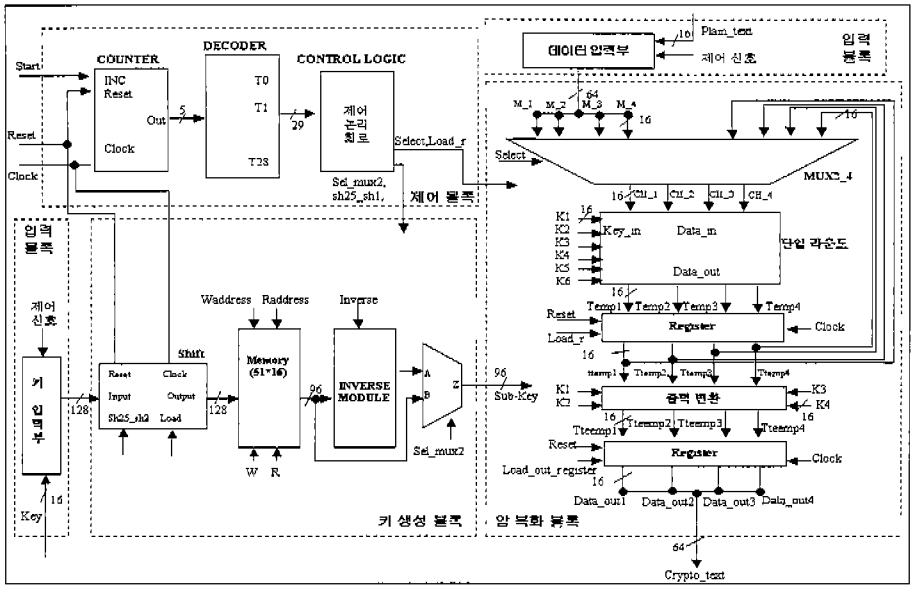


그림 6 제안한 IDEA의 전체 하드웨어 구조

원 연산은 두 개의 모듈러 $2^{16}+1$에 대한 곱셈기로 구성하고 실제로 두 개의 곱셈기의 곱을 15회 반복 적용하는 하드웨어 구조로 설계한다. 그림 5는 모듈러 $2^{16}+1$에 대한 곱셈 역원연산의 블록 다이어그램이다.

4.2.3 제어 부분

제어부분은 IDEA의 암·복호 수행 시간이 최소가 되도록 암·복호화 처리 부분과 키 생성 부분에 필요한 제어신호를 발생하도록 설계한다. 그림 6에 제어 부분의 내부 블록도을 나타낸다. 실제로 클럭 입력에 따라 암·복호 알고리즘의 동작을 마이크로 오퍼레이션 레벨으로 기술하여 내부 신호를 발생하도록 설계한다[12].

4.2.4 입·출력 부분

설계한 암호 칩이 PC나 16비트 마이크로컨트롤러와 데이터 송·수신이 가능하도록 16비트 단위로 입력받도록 설계한다. 64비트의 평문과 128비트의 키를 16비트로 나누어 입력 받기 위해 데이터용 어드레스 입력 포트의 정의가 필요하다. 64비트의 평문을 16비트 단위로 입력받는 것을 예를 들어 설명한다. plain_address가 "00"이면 입력된 16비트 평문은 0번째에서 16번째 비트까지의 평문 입력을 나타내고 plain_address가 "01"이면 17번째에서 31번째 비트까지의 평문 입력을 나타내며 plain_address가 "10"이면 입력된 16비트 평문은 32번째에서 47번째 비트까지의 평문 입력을 나타내고 plain_address가 "11"이면 48번째에서 63번째 비트까지의 평문 입력을 나타낸다.

4.3 동작 검증

가변 데이터 known answer test와 가변 키 known answer test와 table known answer test를 이용하여 설계한 IDEA 암호 알고리즘의 동작을 ECB 모드에서 검증하고 있다. 가변 데이터 known answer test에서는 IDEA에 입력되는 키 값을 고정하고 데이터 값을 변화해가면서 설계한 회로가 올바르게 동작하는지를 검증한다. 가변 키 known answer test에서는 IDEA에 입력되는 데이터 값을 고정하고 키 값을 변화해가면서 설계한 회로가 올바르게 동작하는지를 검증한다. Table known answer test에서는 IDEA에 입력되는 데이터 값과 키 값을 변화해가면서 설계한 회로가 올바르게 동작하는지를 검증한다. 각각의 test에서 사용되는 비교 데이터는 PGP에 구현된 IDEA[11] 프로그램에 의해 제공되었고 이것은 C언어로 작성되어있다. 표 2에는 하나의 예로써 데이터를 $(3736353433323130)_{16}$로 키를$(707E6E486C6B6A6968676665646362 6160)_{16}$로 입력하였을때 암호문 $(509A8E0AE6E5EC9C)_{16}$가 출력됨을 나타낸다. 그림 7과 그림 8은 설계한 IDEA 암호 알고리즘의 타이밍 시뮬레이션이다. 그림 7은 평문을 암호문으로 변환하는 암호 과정을 나타내고 그림 8는 암호문을 평문으로 변환하는 복호 과정을 나타낸다. 표 2와 그림 7과 그림 8로부터 시뮬레이션 레벨에서 설계의 정확성을 검증된다. 또한

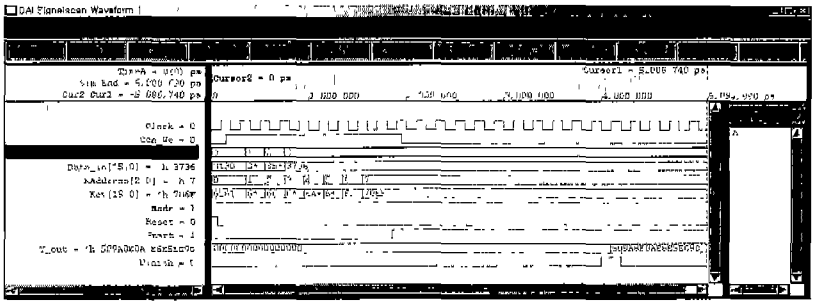


그림 7 암호화 과정의 타이밍 시뮬레이션

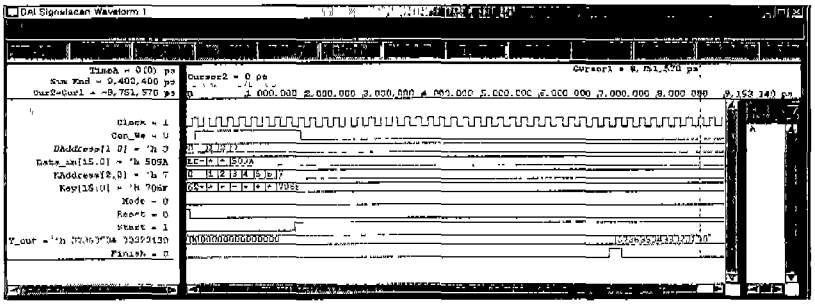


그림 8 복호화 과정의 타이밍 시뮬레이션

IDEA의 각각의 블록들을 Xilinx FPGA XC4062XL에 프로그램하여 블록별 칩 테스트를 수행하여 하드웨어 검증을 완료한다[12].

표 2 IDEA의 소프트웨어 결과 값

```
키: "ponmlkjihgfedcba" ASCII
      = 70 6F 6E 6D 6C 6B 6A 69 68 67 66 65 64 63
        62 61 60 Hex
평 문: "7 6 5 4 3 2 1 0" ASCII
      = 37 36 35 34 33 32 31 30 Hex
암호문: "ï  "   J   ~Ã" ASCII
      = 50 9A 8E 0A E6 E5 EC 9C Hex
```

4.4 성능 비교

표 3은 본 논문과 기존 연구와의 동작 주파수와 처리 속도 관점에서 성능 비교를 나타내고 있다. 참고 문헌 [5]는 단일 라운드 반복 구조로 설계한 사례이고 참고 문헌 [10]은 전 라운드 구조를 기반으로 파이프라인 처리가 되도록 설계한 사례이며 참고 문헌 [13]은 설계한 코어를 상용화한 사례이다.

본 논문의 처리 속도는 동일한 설계 구조를 사용한 [5]보다 약 2배 빠르고 [10]보다 약 7배 느리며 [13]보다 약 6.2배 느리다. 본 논문이 [5]보다 2배 빠른 이유는 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산을 확장 유클리드 알고리즘 대신 페르마의 소정리를 응용하여 필요한 계산량을 1/2배 줄였기 때문이다. 그리고 [10]과 [13]보다 느린 이유는 설계 구조에 있어서의 차이 때문이다. 즉 본 논문에서 채택한 단일 라운드 반복 구조는 하나의 라운드로 구성되어 8회의 클럭 사이클이 경과하여야만 암·복호문이 생성된다. 전 라운드 구조는 8개의 라운드로 구성되고 클럭 사이클당 서로 다른 데이터를 암·복호화하는 파이프라인 구조로 수행되므로 처리속도가 뛰어난다.

표 3 기존 논문과의 성능 비교

	Frequency (MHz)	throughtput (Mbits/sec)	설계 구조
본 논문	20 MHz	116 Mbits/s	단일 라운드
참고 문헌 [5]	33 MHz	55 Mbits/s	단일 라운드
참고 문헌 [10]	53 MHz	809 Mbits/s	전 라운드
참고 문헌 [13]	100 MHz	720 Mbits/s	전 라운드

5. 결 론

본 논문에서는 페르마의 소정리를 응용해서 IDEA 알고리즘에서 계산하기 어렵고 복잡도가 가장 높은 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산에 대해 새로운 방식을 제안하였다. 페르마의 소정리를 응용한 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산 방식은 기존의 확장 유클리드 알고리즘을 적용한 방식보다 필요한 연산 횟수가 약 50% 정도 감소된다. 본 논문에서 제안한 곱셈의 역원 연산 방식과 단일 라운드 반복 방식을 바탕으로 설계한 IDEA 하드웨어의 회로의 최대 동작 주파수는 20 MHz이고 게이트 수는 118,774 gate이며 처리 속도는 116 Mbits/sec이다. 동일한 단일 라운드 설계 방식을 적용한 기존 연구 [5]보다 처리 속도가 약 2배 정도 향상됨을 확인하였다. 이는 본 논문에서 제안한 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산 방식이 고속임을 나타낸다.

그리고 본 연구를 통해 IDEA 암호 알고리즘의 하드웨어 코어에 대한 IP를 구축했으며 16비트 단위로 데이터를 입력받도록 입력 인터페이스를 정의하여 PC나 마이크로콘트롤로와 연동할 수 있도록 하였다.

향후 연구방향으로는 회로의 타이밍과 면적을 좀더 개선하기 위해 모듈러 $2^{16}+1$에 대한 곱셈과 모듈러 $2^{16}+1$에 대한 곱셈의 역원 연산에 대한 게이트 레벨에서의 최적화에 관한 연구와 ATM과 FDDI와 같은 고속 네트웍 통신망 등에 적용할 수 있도록 전 라운드 방식을 사용하여 파이프라인 처리가 가능하도록 하는 연구 등이 있다.

참 고 문 헌

[1] Kris Gaj, Pawel Chodowiec, "Fast implementation and fair comparison of the final candidates for Advanced Encryption Standard using Field Programmable Gate Arrays," *Proc. RSA Security Conference-Cryptographer's Track San Francisco, CA*, April 8-12, 2001.

[2] Bruce Schneier, *Applied Cryptography*, John Wiey & Sons, Inc, 1994.

[3] X. Lai and J. L. Messay, "A proposal for a new block encryption standard," *In Advance in Cryptology-EUROCRYPT'90*, pp 389-404, Berlin, 1990.

[4] Helger Lipmaa homepage, *http://www.tml.hut.fi/~helger/fastidea*, 2001. 6. 1.

[5] H. Bonnenberg, A. Curiger, N. Felber, H. Kaeslin, X. Lai, "VLSI implementation of a new block cipher," *1991 IEEE Int Conf. Computer Design: VLSI in computer and Processor*. pp 510-513, 1991.

[6] 김응태, 박승안, *정수론*, 경문사, 1997년 4월.

[7] Zhongde Wan, Jullien G.A, Miller W.C. "An algorithm for multiplication modulo (2^n+1)," Signals, Systems and Computers, Conference Record of the Twenty-Ninth Asilomar Conference, pp 956-960, 1995.

[8] A. Curiger, H. Bonnenberg, and H. Kaeslin, "Regular VLSI architectures for multiplication modulo (2^n+1)," *IEEE Solid-State Circuits*, vol. 26, no. 7. pp. 990-994, July 1991.

[9] Ashur. A.S., Ibrahim. M.K., Aggoun, A, "Area-time efficient diminished-1 multiplier for Fermat number transform," *Electronics Letters*, Vol 30, pp: 1640 -1641. Sep. 1994.

[10] Salomao. S.L.C , de Alcantara, J.M.S, Alves, V.C., Franca, F.M.G., "Improved IDEA," *Integrated Circuits and Systems Design, 2000. Proceedings. 13th Symposium on*, pp. 47-52. 2000.

[11] Philip Zimmermann, *PGP Source Code and Internals*, MIT Press, 1994.

[12] 최영민, 권용진, "FPGA를 이용한 효율적인 IDEA 칩 설계", *전자정보통신공학 논문지*, 1999년 12월.

[13] MediaCrypt Homepage, *http://www.media-crypt.com*, 2001. 6. 1.

최 영 민

1976년 9월 26일생. 1999년 2월 한국항공대학교 항공통신정보공학과 졸업. 1999년 3월 ~ 현재 한국항공대학교 항공통신정보공학과 대학원 석사과정 재학중. 관심분야는 논리회로 설계 및 합성, 통신용 ASIC 설계

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 항공항공대학교항공전자정보통신 컴퓨터 공학부 부교수. 관심분야는 정보 보호, 논리회로 설계 및 합성, 알고리즘 개발