E-DES 알고리즘을 이용한 암호칩 설계

김종우* 하태진* 김영진** 한승조*

*조선대학교 전자정보통신공학부 **데이콤

Design of Secure Chip Using E-DES Algorithm

Jong-woo Kim* Tai-jin Ha* Young-jin Kim** Seung-jo Han*

*School of Electronic and Information Communication Eng. Chosun Univ.

**DACOM

요 약

기 상용화되고 있는 소프트웨어/하드웨어 제품의 복제방지에 대한 강도가 부족하여 쉽게 락이 크랙될 뿐 아니라 복제방지의 기능을 수행할 수 없는 단점을 보안하여 본 논문은 세계적으로 가장 많이 사용하고 있는 암호알고리즘 중의 하나인 DES를 구조적으로 수정하고 키 길이를 확장하여 암호학적 강도를 개선한 E-DES(Extended DES)를 설계하고, 이를 하드웨어로 구현하기 위해서 시스템 설계 기술언어인 VHDL로 코딩하고, FPGA를 이용, test chip을 구현하여 성능테스트를 수행한 다음, 설계된 FPGA 칩을 ASIC으로 제작하여 강력한 암호알고리즘을 가진 보안칩을 설계한다.

I. 서론

오늘날 현대사회는 전 세계가 사상과 이념을 위해서 전쟁을 하는 것이 아니라 자국이 경제적 이익을 위해서 경쟁하고 있거나 대립하고 있다고 해도 과언이 아니다. 특히, 디지털콘텐츠 불법복제는 국내외 경제 발전의 저해 요인일 뿐만 아니라 디지털 문화 콘텐츠 개발업체에게 심각한 위협요소가 되고 있다.

디지털콘텐츠 개발업체는 자사제품이 불법 복제되어 유통되는 것을 예측하여 정품가격을 결정하기 때문에 불법복제에 대한 손실액을 정품사용자에게 부담함으로써 사용자는 정품가격이 비싸 불법복제품을 선호하게 되고 디지털콘텐츠 판매유통의 악순환이 되풀어 디지털문화 콘텐츠 개발업자와 소비자 모두가 손해를 보고 있는 실정이다.

또한, 기 상용화되고 있는 보안시스템은 소프트웨어나 하드웨어로 구현하게 되는데, 보안기능을 탑재한 소프트웨어는 보안모듈과 정해진 방법으로 통신함으로써 모듈의 유무나 오류 등을 파악하게 된다. 하드웨어로 제작된 보안 장치는 소프트웨어로 이루어진 장치보다는 보안의 강도가 높은 반면, 소프트웨어 제품의 가격에 큰 영향을 끼칠 정도로 고가이기 때문에 고가의 소프트웨어에만 적용되고 있는 실정이다. 또한 기 상용하드웨어 보안제품은 사용상의 불편함과 기능의 한계성을 가지고 있어서 소프트웨어 보호 측면에서 보면 개발업자들에게는 환영을 받지만 소프트웨어를 사용하는 최종사용자 입장에서는 외면 받기 십상이었다. 따라서 소프트웨어 복제방지 및 정보보호를 위해서는 사용이 편리하고 저가이며, 보안 강도가 높고 역기능 방지에 강력한 암호알고리즘을 가진 견고한 하드웨어적인 보안칩 개발이 절실히 필요한 실정이다.[5]

본 논문에서는 기존의 DES알고리즘을 개선한 E-DES 암호 알고리즘을 이용 하드웨어로 구현하기 위해서 시스템 설계 기술언어인 VHDL로 코딩하고, FPGA를 이용 ASIC으로 제작하여 E-DES 암호 알고리즘을 가진 보안칩을 설계한다.

II. 복제 방지 기법 및 문제점

1. 복제 방지 기법

프로그램을 보호하는 방법은 크게 법과 윤리에 의존하는 복제 금지 방법과 물리적인 방법을 이용한 복제 방지 기법으로 나눌 수 있다. 다시 물리적인 방법에는 크게 소프트웨어적인 방법과 하드웨어적인 방법이 존재한다. 소프트웨어적인 방법은 추가적인 주변기기를 필요로 하지 않고 단지 사용자의 물리적인 자원을 이용하게 된다. 즉, 제품번호와 같은 키 값을 사용하는 경우이다. 따라서 하드웨어적인 방법에 비해 소요되는 개발 및 유지비용이 적게드는 특징이 있다. 하드웨어적인 방법에는 외부에 특수한 물리적 요소를 두고 그 요소에 의한 복제방지를 하게 된다.

하드웨어를 이용한 상용 복제방지 시스템의 경우 다음에 기술된 여섯 가지의 기술적 흐름을 띄고 있다.

- 1개의 복제방지 장치에 의한 다수개의 프로그램 복제방지
- 복제방지 장치의 재활용
- 화학물질을 이용한 물리적인 복제방지 장치의 오픈방지
- ROM 등을 이용한 물리적인 복제를 사전에 방지
- 프로그램에 적용시키는 방법의 다양화
- 소프트웨어적인 역엔지니어링 체크루틴 삽입

2. 기 상용제품의 문제점

1) S/W Lock의 문제점

저장매체의 물리적인 특성을 이용한 경우 특수한 copy 프로그램(diskdup, DCF, copy II PC 등)으로 물리적인 특성까지 복사가 가능하다. 사용자 등록을 한 후 발급되는 key 또는 license file은 대부분 일반 copy 프로그램으로 복제가 가능하다.

사용일수에 제한을 가하는 경우는 사용기간이 만료되면 시스템의 날짜를 뒤로 돌려놓거나, 실행코드상의 시간검사 routine을 변경해 놓음으로써 재사용이 가능해 진다.

또한, shareware version은 사용자의 이름과 등록코드를 입력하는 방법을 사용하는데, 사용자의 이름을 가지고 올바른 코드를 생성하는 루틴이 프로그램 내부에 존재하기 때문에 이 비교 부분을 patch하면 정식 사용자라는 것을 인식한다. 더구나 심한 경우에는 등록코드 자체가 프로그램 내부에 존재하여 디버거를 통하여 이 등록코드를 확인할 수 있다.[3]

2) H/W 락의 문제점

락의 존재 여부만을 체크하는 방식은 가장 고전적이며 단순한 방법이다. 이 경우 디버깅 툴을 이용하여 프로그램의 락 체크루틴을 찾아낸 후 비교구문 등을 변경하면 락의 무력화가 가능하다. 또는 락의 입출력 단자를 모니터링 하여 락으로부터 넘어오는 값을 알아낸 후 S/W의 뒤에 락을 대신하는 실행코드(에뮬레이션 코드)를 붙임으로써(컴퓨터 virus처럼) 실제로 하드웨어 락을 검사하는 것이 아니라 소프트웨어적으로 검사를 통과시키는 방법도 존재한다.[1][2]

락에 중요한 정보를 저장해 놓는 경우 그 정보가 고정적이면 락의 복제(PAL이나 ROM등으로)로 쉽게 락의 무력화가 가능하다. 그러나 락에서 넘어오는 정보가 고정적이지 않을 경우 락의 복제보다는 디버거를 이용하여 S/W코드 중 락에서 가져오는 데이터를 저장하는 곳에 고정적인 값을 넣어주는 에뮬레이션 코드를 추가하는 등의 방법으로 무력화(완벽하지는 않지만 동작에는 지장이 없을 정도)가 가능하다.

락에서 가져온 값으로 실행코드나 데이터를 변형하는 방법(scrambling)은 S/W상에서 코드의 변경이 불가능하기 때문에 실행코드 상에서는 가장 강력한 방법에 속한다. 그러나 락으로부터 넘어오는 scramble을 위한 code값이 일정할 경우 락을 쉽게 에뮬레이션 할 수 있기 때문에 락의 보호가 취약하다. 또한 에뮬레이션 코드를 실행 S/W의 뒤에 추가시킴으로써도 무력화가 쉽게 가능하다.

III. 복제 방지 보안칩 설계

1. 보안칩 알고리즘

1) E-DES 암호알고리즘

가. 기본구조

96비트 한 블록의 데이터를 32비트의 3개의 서브블록으로 분할함으로써 비대칭으로 만든다.

$$B_i = C_{i-1} \oplus f(B_{i-1}, K_i)$$

$$C_i = A_{i-1} \oplus f(B_{i-1}, K_i) \qquad (1)$$

이것을 초기조건 $C_i = B_{i-1}$ 와 같이 단순화 시켜 적용하여 보면, 완전히 같은 특성으로 반대편에 위치함을 알 수 있다. 식(1)에 의한 조합은 대칭적이며, 따라서 본 E-DES 알고리즘에서 그림 1과 같이 적용하였다. 암호화와 복호화는 각각의 관계를 서로 비교함으로써 동치임을 확인할 수 있다. 그림 1로부터 암호화 공식을 유도하면 다음과 같다.

$$A_i = B_{i-1}$$

$$B_i = C_{i-1} \oplus f(B_{i-1}, K_i)$$

$$C_i = A_{i-1} \oplus f(B_{i-1}, K_i) \qquad (2)$$

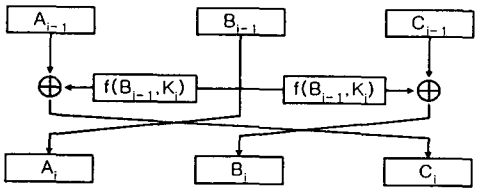

그림 1 개선된 DES의 구조 (1 라운드)

식 (2)로부터, XOR 특성과 $f(B_{i-1}, K_i) = f(A_i, K_i)$를 이용하면 다음의 복호화 식을 얻는다.

$$B_{i-1} = A_i$$

$$C_{i-1} = B_i \oplus f(A_i, K_i)$$

$$A_{i-1} = C_i \oplus f(A_i, K_i) \qquad (3)$$

A와 B를 서로 바꾸면 다음과 같은 관계를 얻을 수 있다.

$$A_{i-1} = B_i$$

$$B_{i-1} = C_i \oplus f(B_i, K_i)$$

$$C_{i-1} = A_i \oplus f(B_i, K_i) \qquad (4)$$

식 (4)에 보여진 것과 같이 복호화 식도 식 (2)과 같은 구조를 이루고 있다. 그러므로 하나의 하드웨어로 암호화와 복호화를 같이 적용할 수 있다. 개선된 DES의 암호화 알고리즘이 그림 2와 같으며 3개의 32비트 서브블록으로 나누어진다. 또한 초기순열(IP)과 역초기순열 (IP-1)도 96비트로 확장되었다.

f 함수는 각 구조에서 왼쪽과 오른쪽에서 동작하며, 왼쪽과 오른쪽 각각에서 f1과 f2로 명명되어 있다. S-box의 수는 16개로 증가시키는데, S1에서 S16까지로 명명되며 왼쪽 f 함수에 S1에서 S8, 오른쪽 f 함수에 S9에서 S16이 사용된다. 마지막으로 식을 적용하는데 있어서 암호화의 마지막 라운드 결과인 A16과 B16은 서로 교환되어야 한다.

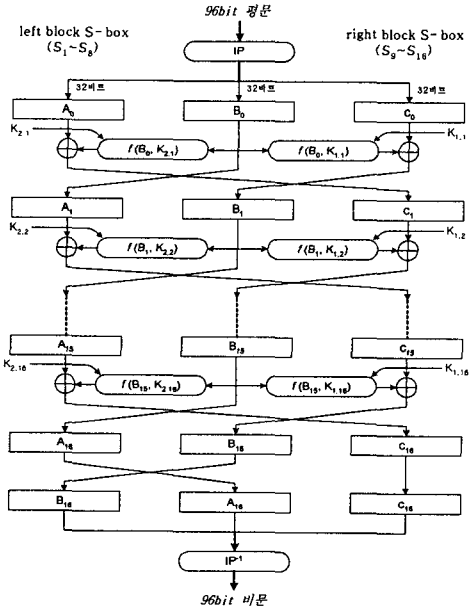

그림 2 개선된 DES의 암호화 알고리즘

공식에 의한 개선된 DES의 복호화 알고리즘이 그림 12와 같다. 복호화 알고리즘은 암호화 알고리즘과 같으나 첫 번째 라운드에서 A0과 B0을 서로 바꾸어 주어야 한다. 각 라운드의 서브블록에 다른 f 함수를 가지기 위하여 입력키를 64비트에서 128비트로 확장하였다. 16개의 패리티비트를 제거한 후 남은 입력키는 왼쪽의 K1과 오른쪽의 K2로 각각 56비트씩 2개의키로 나눈다. 또한 PC-1(Permuted Choice 1)을 거친 후 56비트의 키는 좌우로 28비트씩 나뉘어 지고, 매 라운드 동안 K1과 K2는 키 스케줄에 따라 왼쪽으로 쉬프트 된다. 각 라운드에 해당하는 키의 56비트 출력은 PC-2(Permuted Choice 2)를 통과한 후 48비트로 줄어들고, 왼쪽과 오른쪽 각각 K1,i와 K2,i로 명명된 후 각 라운드에 적용된다. (i=1 to 16)

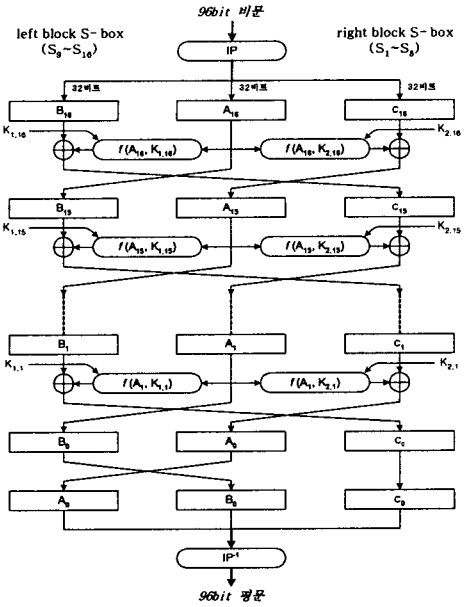

그림 3 개선된 DES의 복호화 알고리즘

나. 키 관리 및 f 함수

복호화 과정의 키는 암호화 과정에서 적용된 키와 반대로 입력되어야 한다. 또한 S-box도 왼쪽과 오른쪽이 서로 바뀌어야 한다.

그림 4는 키 스케줄 블록도 이다.

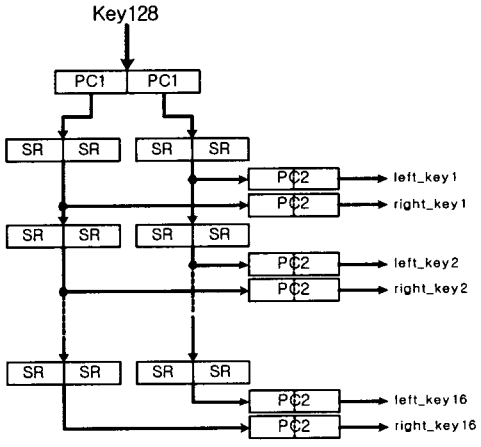

그림 4 키 스케줄링

SR은 Shift Rotate로서 암호화에는 shift rotate left를 수행하고 복호화에는 shift rotate right를 수행한다. shift rotate 횟수는 라운드마다 다르며, 전부 shift rotate 되었을 경우 다시 원래의 값으로 되돌아오게 된다. 이것은 복호화는 암호화에서 사용한 키를 역순으로 넣어주어야 하는 것을 쉽게 구현가능 하게 한다.

키 스케줄에는 두 개의 PC1로 각각 64 비트를 입력하여 PC2를 통하여 48비트의 키를 생성해 낸다.

개선된 DES가 differential cryptanalysis에 보다 견고하게 하기 위하여 f함수의 수를 증가시켰다. 증가된 수치는 3개의 서브블록과는 차이가 있다. 본래의 DES에 사용된 f 함수는 R0에서 L16까지 그리고 L0에서 R16까지 변환되는 과정에서 8번 반복된다. 반면에 개선된 DES의 f 함수는 A0에서 C16 까지 처리될 때 11번 반복된다. 이것은 같은 E(Bit-selection table), P(Permutation table) 그리고 S-box에서 개선된 DES가 본래의 DES에 비해 differential cryptanalysis에 보다 더 견고함을 보여준다.

다. S-Box

S-Box의 설계는 알고리즘의 강도에 결정적인 영향을 미친다. DES에 있어서 비선형적인 모듈러-2 알고리즘이 단지 하나의 단계에 불과한 반면 실제적인 알고리즘의 암호강도는 S-Box에 의존하고 있다. S-Box의 엔트리를 개선하기 위해 DES는 잘 분석된 엔트리를 사용한다. 이 분석은 평균 비트확률(Pi,j)과 그 상호계수에 의해 나타나는 SAC(Strict Avalanche Criterion)에 근거하고 있다.

2) 칩의 구조

본 개발에서 설계한 칩은 크게 5가지의 기능별 블록을 갖는다.

가. E-DES 암호화 블록을 이용한 실행코드 보호

보조저장매체에 저장되어 있는 실행코드는 프로그램의 개발자에 의해 미리 암호화되어 있으며 최종사용자는 락에 내장되어 있는 복호화 블록에 의해 프로그램의 실행단계에서 복호화 작업을 수행함으로써 인증된 사용을 하게 된다. 이를 위하여 핵심칩에는 E-DES 알고리즘을 내장한 Cipher block을 갖는다.

나. 실행코드의 변경을 검사

복제방지 된 프로그램이 실행을 위하여 메모리에 적재되어 있을 때 가장 취약한 부분은 적재된 프로그램의 변경이며, 이것을 방지하기 위한 방법으로 실행코드의 변경을 감시하기 위한 코드를 삽입하는 방법이 사용된다. 본 과제에서 코드의 변경감시는 Hash 함수에 의한 출력 코드의 변경 검사기법을 이용하였다. 복제방지 된 프로그램은 보조 메모리에 적재될 때에 저장된 코드에 대한 Hash값을 함께 저장하고 있게되며 핵심칩 내에서 보조저장매체에 존재하는 Hash값과 칩 내부에서 계산된 Hash값을 비교 수행함으로써 코드의 변경 유무를 검사하게 된다.

다. Scrambler를 사용

Cipher Block에 의해 복호화 된 실행코드는 PC로 전송되어 지기 전에 다시 암호화되어진다. 이는 락으로부터 반환되는 값이 일정할 경우 락을 하드웨어적으로 에뮬레이션 하는 것이 가능하기 때문이며, 이를 방지하기 위한 방법으로 Scrambler를 사용하였다.

라. 락을 제어하는 소프트웨어를 보호하기 위한 PC의 실행 모드 검사기를 칩에 포함

모든 프로그램은 디버거에 의해 실행 시점에서 노출이 되어지며, 실행코드의 변경도 가능해지게 된다. 따라서 이를 방지하기 위한 디버거 검사기를 칩에 내장함으로써 디버거에 의한 노출을 방지할 수 있다.

락 제어용 프로그램은 복제방지 된 프로그램의 실행시 PC의 동작환경 중 디버거와 관계된 정보를 락으로 전송하게 되며, 칩 내부의 모드 검사기는 이를 검사하여 디버거가 존재하지 않는 환경에서만 락이 동작을 하게 제어한다.

마. 외부 입출력을 위한 버퍼

설계된 칩은 외부의 8비트 입출력 포트를 갖고 있다. E-DES 암호알고리즘은 96비트 블록이며 이를 위하여 칩의 내부는 96비트로 구성되어 있다. 따라서 외부와의 입출력을 위한 8 to 96 비트 컨버터를 위한 버퍼가 필요하다.

이와 같은 5개의 블록과 이를 제어하기 위한 controller 그리고 key register block이 그림 5에 보여지고 있다.

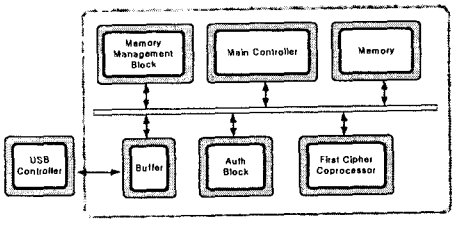

그림 5 핵심칩의 블록도

Ⅳ. 복제 방지 보안칩 설계

1. 칩의 설계

1) SPE128

SPE128은 Round Block, Key Block 그리고 Control Block 등 크게 3부분으로 나누어진다. Round Block은 E-DES 알고리즘 중 1개의 라운드를 가지고 있으며 2분주 된 Clk에 의해 16회 반복 수행하게 된다. Key Block은 16라운드에서 사용될 Kl과 Kr을 생성하는 블록으로 데이터의 입력 전에 미리 키 레지스터로 입력받게 되며 새로운 키가 입력되기 전까지 계속 유지된다. 입력된 키는 라운드블록의 반복에 맞추어 서브키를 생성해서 출력해 준다. Control Block은 Key Block과 Round Block이 유기적인 동작을 할 수 있도록 외부 제어선의 제어하에 내부 각 블록들을 세부적으로 제어하는 블록이며, 데이터의 반복에 따른 현재의 루프를 카운트하는 기능 등을 갖는다.

그림 6은 E-DES 알고리즘의 블록도이다.

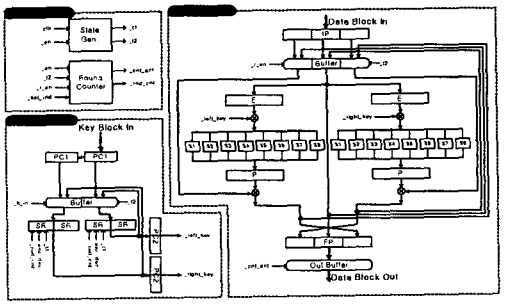

그림 6 E-DES 알고리즘의 칩구현 블록도

가. Round Block

라운드 블록은 16라운드의 E-DES 알고리즘을

실질적으로 처리하는 블록으로서 데이터는 96비트로 입력받아 96비트로 출력한다. _r_en이 상승에지일 때 초기순열(IP)은 데이터 버스로부터 값을 입력받아 초기순열을 수행한 후 Round Processor의 입력으로 보낸다. Round Processor는 좌측과 우측으로 분리되어 있으며 Processor 내에서 사용되는 각종 테이블을 서로 공유한다. Round Processor는 단일 라운드 블록만으로 구성되어 있으며 16회 반복수행을 하게된다. 반복라운드의 수행시 좌우측 각각의 Round Processor에는 현재의 수행 라운드 횟수에 맞는 서브키가 입력된다. 서브키는 키 블록으로부터 전송되며, 각각 48비트로 구성되어 있다.

그림 7은 라운드블록의 블록도이다.

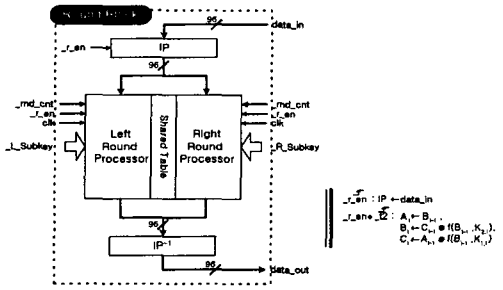

그림 7 Round Block의 블록도

Round Processor는 E(Bit-selection table), mod 2 (XOR), S-box , P(Permutation table) 그리고 mod 2 (XOR)의 5단계로 구성되어 있으며, _r_en이 액티브일 때 클록(clk)의 상승 에지에서 제공된 서브키를 이용하여 5단계 1라운드씩 암호화를 수행한다.

5단계의 각라운드가 16회 반복 수행된 데이터는 역초기순열(IP-1)을 거쳐 96비트의 데이터 버스를 통해 출력된다.

나. Key Block

키 블록은 입력된 128비트의 키를 PC-1을 통해 패리티 비트를 제거하여 112비트로 줄인 후 키 스케줄에 따라 라운드별로 좌측 쉬프트를 수행한 후 PC-2를 거쳐서 각 라운드별 서브키를 생성한다. 키 레지스터는 16라운드에 해당하는 좌측 서브키와 우측 서브키를 보관하고 있게 된다. 키의 입력은 외부제어선인 key_in에 의해 이루어진다.

enc 제어시그널은 키가 출력될 때 현재의 동작이 암호화에 따른 정순 출력인지 복호화 처리에 따른 역순 출력인지를 지시한다. 그림 8은 키 블록의 블록도이다.

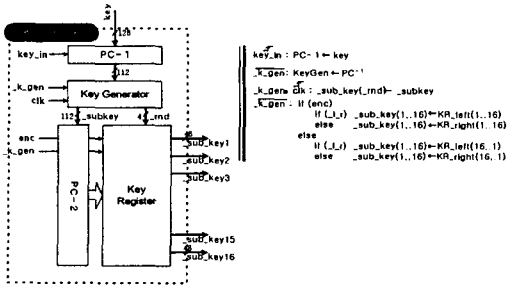

그림 8 Key Block의 블록도

라운드 블록에서 사용되는 키는 다음의 새로운 키가 입력될 때까지 내부 레지스터에 유지되며, 키 제너레이터는 _k_gen이 액티브일 때 클록(clk)에 따라 내부 카운터가 1부터 16까지 증가하면서 서브키를 생성하여 키 레지스터로 전송하게 된다.

다. Control Block

컨트롤 블록은 크게 2가지의 기능을 갖는다. 첫째, 외부의 제어선으로부터 입력을 받아 전체 블록들을 제어하는 기능과, 둘째, 입력된 데이터가 몇 번째 단계를 수행 중인지를 카운트하여 최종 데이터가 중간단계에 잔류하지 않도록 하는 기능으로 분류된다. 컨트롤 블록의 블록도가 그림 9에 나타나 있다.

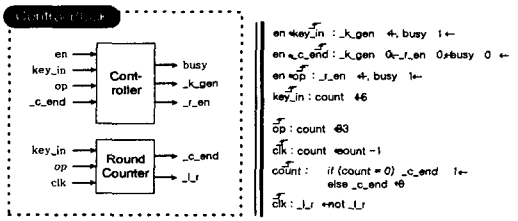

그림 9 Control Block의 블록도

컨트롤 블록은 현재 데이터가 중간의 단계에 아직 남아 있을 경우 busy 신호선을 액티브 상태로 유지한다. 그리고 마지막 데이터가 출력이 된 후에는 _c_end 신호선이 액티브 되어 busy 신호선을 not busy로 유도한다.

라. Hash Block

Hash 블록은 암호화되어 있던 실행코드의 불법적인 변경을 감시하기 위한 블록으로 _in_bus에 입력된 코드의 H 값과 _hash에 입력된 값을 비교한 후 같으면 _cert에 '1'을 출력한다.

그림 10은 Hash Block의 블록도이다.

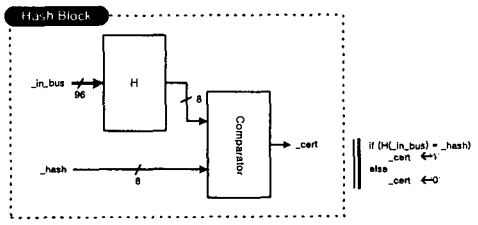

그림 10 Hash 블록의 블록도

마. Scrambler

cipher block에서 암호화되어진 데이터는 PC로 전송되어지기 전에 Scramble 과정을 거치게 된다. 키는 Key Generator에 의해 생성되며, _s_ken이 active될 때 _in_bus를 입력으로 하여 SK(Scramble Key)를 생성한다. 생성된 SK는 _scr_en 이 active될 때 _out_bus의 데이터를 Scramble 하는데 사용되어지며 _out_buf의 출력을 얻게 된다.

그림 11은 Scrambler의 블록도이다.

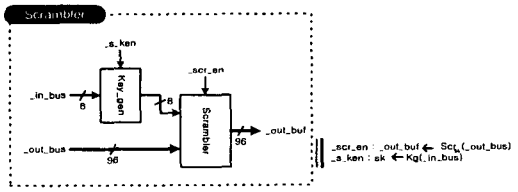

그림 11 Scrambler의 블록도

사. Buffer

설계된 칩은 내부 96비트, 외부 8비트의 입출력을 갖는다. 따라서 데이터의 입출력을 위한 버퍼가 필요하며 8 to 96비트 컨버터 역할을 하게 된다. 버퍼는 hash code를 위한 8비트를 포함하여 13개의 8비트 버퍼 블록으로 나뉘어 진다. 또한 외부 입출력 버스는 buf1과 연결되어 있고, 데이터의 입출력시 13개의 버퍼는 쉬프트를 수행함으로써 내부 버퍼를 채우게 된다. 설계된 버퍼는 _d_in의 상승에지에서 하위버퍼에서 상위버퍼로 쉬프트 되며, _d_out의 상승에지에서 하위버퍼로 쉬프트 된다.

그림 12는 버퍼의 블록도이다.

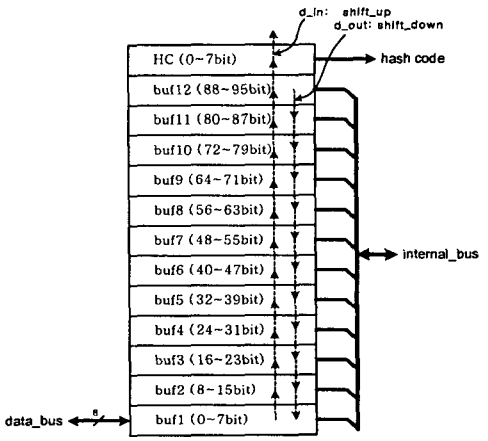

그림 12 Buffer의 블록도

아. Controller

콘트롤러는 각각의 블록들의 유기적인 동작을 위한 제어를 담당하며, 그 외에 외부의 제어워드를 해석하고 키 레지스터를 포함하고 있으며, 디버그 모드를 감지하기 위한 모드 검사기를 갖고 있다.

키 레지스터는 총 10개의 키 공간을 내장하고 있으며, 그중 1개는 키의 고유한 ID를 갖게되고, 8개는 외부에서 writing이 가능한 ROM으로 구성되며, 1개는 임시 키를 위한 RAM의 형태를 갖는다. 키의 선택은 _key_sel에 의해 이루어지며 키의 입력과 출력에서 공통이다. 키의 입력은 _k_in의 상승에지에서 _In_key의 값을 입력으로 받게되며, 키의 출력은 내부의 10개의 키 중 _key_sel에 의해 선택되어진 후 _cur_key로 출력한다.

키 레지스터의 블록도가 그림 13에 나타나 있다.

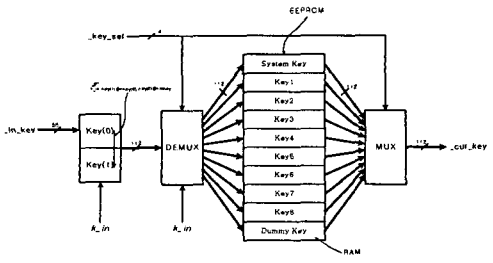

그림 13 Key Register의 블록도

자. 제어워드(Control Word)

제어워드는 칩의 환경을 세팅하는 명령어 그룹과 암·복호화를 수행하기 위한 명령어 그룹으로 나뉜다. 설계된 칩은 IBM PC 환경에서 동작할 경우 칩과의 통신내용을 디버거(Debugger)를 통하여 모니터를 할 수 있는 단점을 보완하기 위하여 디버거의 특징을 판단하여 칩의 동작유무를 결정하는 함수를 내장하고 있다. 또한 칩으로부터 되돌아오는 암·복호화 된 데이터의 분석을 방지하기 위하여 스크램블러(Scrambler)를 내장하고 있으며, 이 두 가지의 기능은 필요에 따라 On/Off 할 수 있다.

표 1은 제어워드에 대한 설명과 수행에 필요한 cycle이다.

표 1 Control Word

CW	Function	Op. Cycle	
00000001b	System check	3	System Control
00000010b	Scramble function enable/disable	1	
00000100b	Data crc check enable/disable	1	
00001000b	Internal key buffer clear	3	
00010000b	Seed Number(RN) input	3	Key Control
00100000b	External key input	3	
01000000b	Seed of internal key input	3	
10000000b	Data for crc check	16	Data Control
10000100b	Data decryption (external key)	16	
10001000b	Data decryption (internal key)	16	
10010000b	Data encryption (external key)	16	
10100000b	Data encryption (internal key)	16	

V. 결 과

1. 칩의 제작

암호칩은 0.45㎛ 공정의 Gate Array Rule을 따랐으며, 실 Gate 수가 32011 gate였다. 칩의 설계 단계에서 동작주파수의 평균을 33MHz로 설계하였으나, 최대 동작 주파수는 50MHz까지 오류 없이 동작하였다.

50MHz의 동작 주파수는 현재 보편화되고 있는 100MHz 이상의 주변 동작 주파수에 비해 낮으며, 이는 Design Rule을 Full Custom으로 하여 CMOS 공정이 아닌 0.45㎛ 이하의 GaAs 공정으로 제작하게 되면 350 MHz 이상의 속도로 동작 가능한 점을 들어 충분히 해결 가능하다.

그림 14는 암호칩의 사진이다.

그림 14 암호칩

VI. 결 론

본 논문은 소프트웨어의 불법복제와 개인정보의 침해 등을 방지하기 위한 정보보호용 보안모듈의 설계를 위해 기존의 DES 암호알고리즘을 구조적으로 수정하고 키 길이를 확장하여 암호강도를 증가시킨 E-DES를 설계하였으며, 제안된 암호화 알고리즘을 하드웨어로 구현함에 있어서 적용 가능한 전 라운드 구현형, S-box 공유형, 단일라운드 반복형의 설계방식을 VHDL을 이용하여 설계하였다.

이는 개인정보와 컴퓨터 프로그램에 대한 보호가 시급한 현시점에서 암호칩을 기반으로 하는 보안모듈을 통하여 기밀성이 요구되는 개인정보의 보호와 소프트웨어의 불법복제방지를 가능하게 하였으며, 전자상거래나 전자화폐의 활용이 많은 최근 위·변조의 방지를 위한 개인용 보안장치, 인터넷 뱅킹의 인증서 저장장치, 전자상거래 인증장치 등으로 활용할 수 있을 것이며, 또한 급속히 초고속화 되어가고 있는 컴퓨터 통신망에서 비인가자에 대한 정보전송의 보호에도 효과적으로 활용될 수 있을 것이다.

참고문헌

[1] David Aucsmith, "Tamper resistant software" Information Hiding-Proceedings of the First International Workshop, pp.317-333, 1996

[2] M. G. Arnold and Mark D. Winkel, "Computer systems to inhibit unauthorized copying, unauthorized usage, and automated cracking of protected software" U.S. Patent No.4 558 176. issued Dec. 1985.

[3] A. Herzberg, and G. Karmi, "on software protection" In Proc. Fourth JCIT, pp.388, Apr. 1984.

[4] S. J. Han, H. S. Oh, "Design of Extended-DES cryptography" Proceeding of the IEEE International symposium on Information Theory, pp.353-359, July 1995.

[5] Roger Lipsett, Carl Schaefer, Cary Ussery,

VHDL : "Hardware Description and Design", Kluwer Academic Publishers, Norwell, Massachusetts, 1989.

[6] Frank Hoornart, Jo Goubert & Yvo Desmedt, "Efficient hardware implementation of the DES", Journal of Cryptology, Vol.4, no.1, pp.148 -151.

[7] B. Den Boer, "Cryptanalysis of FEAL", Advances in Cryptology, Proc. of EURCRYPT '87, Springer-Verlag, pp.167-173, 1989.

[8] D. Chaumm, "Design concepts for tamper responding systems", In Advances in Cryptology: Proc. Crypto 83. D. Chaum. Ed. New York: Plenum. pp.387-390. 1984..