論文2000-37SD-11-12

DES의 데이터 처리속도 향상을 위한 변형된 병렬 Feistel 구조에 관한 연구

(A Study of Modified Parallel Feistel Structure for Data Speed-up DES)

李 善 根*, 金 炯 均*, 金 煥 溶*

(Lee Seon Keun, Kim Hyeoung Kyun, and Kim Hwan Yong)

요 약

정보통신의 눈부신 발달과 인터넷의 급격한 확산으로 현대 네트워크 통신은 전자상거래 또는 전자화폐의 활성화, 전자서명등의 여러 가지 첨단기능을 수행하고 있으며 미래에는 더욱 진보된 서비스를 제공하게 될 것이다. 이러한 전자상거래와 같은 정보통신네트워크는 보다 안전하게, 보다 투명성이 있는 네트워크의 보장을 요구하게 되며, 보다 빠른 네트워크의 성능을 기대하게 된다. 본 논문에서는 이러한 여러 가지 요구에 부응하기 위하여 DES(Data Encryption Standard)의 기본 구조인 Feistel 구조를 병렬로 변화시킨 병렬 Feistel 구조를 가지는 DES를 제안한다. 제안된 병렬 Feistel 구조는 DES 자체의 구조적 문제(error의 propagation) 때문에 pipeline 방식을 사용할 수 없어 데이터 처리속도와 데이터 보안사이에서의 trade-off 관계를 가질 수밖에 없었던 DES의 성능을 크게 향상시킬 수 있으며 더불어 Feistel 구조를 채택한 SEED에 제안된 방식을 적용할 경우 지금보다 더욱 우월한 보안 기능 및 고속의 처리능력을 발휘하게 될 것이다. 여기에서 사용된 CAD Tool은 회로합성과 모의실험에 모두 Synopsys Ver.1999.10을 사용하였다.

Abstract

With the brilliant development of information communication and the rapid spread of internet, current network communication is carrying several up-to-date functions such as electronic commerce, activation of electro currency or electronic signature and will produce more advanced services in the future. Information communication network such as that electronic commerce would demand the more safe and transparent guard of network, and anticipate the more fast performance of network. In this paper, in order to meet the several demands, DES(data encryption standard) with parallel feistel structure, which feistel structure of the basic structure of DES is transformed into in parallel, is proposed. The existing feistel structure can't use pipeline method for the structural problem of DES itself-the propagation of error. therefore, this modified parallel feistel structure could improve largely the performance of DES which had to have the trade-off relation between data processing speed and data security and in addition a method proposed in SEED having adopted the modified parallel feistel structure shows more excellent secure function and/or fast processing ability. The used CAD Tool use Synopsys Ver. 1999. 10 in both of synthesis and simulation.

* 正會員, 圓光大學校 電子工學科 回路 및 시스템

(Dept. of Electronic Engineering, Wonkwang University)

※ 이 논문은 1999년도 원광대학교의 교비지원에 의해서 연구됨

接受日字 : 2000年6月20日, 수정완료일 : 2000年11月29日

I. 서 론

현대사회는 정보통신 네트워크 사회이다. 이러한 정보통신 네트워크 사회에서의 편리성이란 이루 헤아릴 수 없을 정도로 많다. 전자 상거래, 전자 쇼핑, 전자화폐, 전자투표 등의 단어들은 더 이상 낯설지 않다. 현

존하는 암호시스템들은 통계적 취약성을 모두 내포하고 있다. 그러므로 암호시스템자체가 해독되어지는 것은 시간과 노력의 차이만 있을 뿐, 완전하게 막아낼 수는 없다. 네트워크의 취약성을 막기 위해서는 키의 길이를 길게 하면 그만큼 해독하기에 어렵겠지만 키의 길이를 무작정 길게 하면 데이터를 처리하는데 걸리는 시간도 역시 비례적으로 길어지게 된다. 즉, 비도와 처리시간과는 trade-off관계가 존재하게 된다. 소프트웨어적으로 구현된 암호시스템의 경우, 암호시스템의 구현 및 운용이 용이하다는 장점이 있으나 데이터량이 많아지거나 폭주하게 될 경우 컴퓨터가 다운되거나 처리시간이 오래 걸리는 단점을 가지게 된다. 이러한 단점을 없애기 위하여 하드웨어로 구현한 암호시스템이 등장하였으나 아직까지는 크게 효율적이지 못하다. 이렇게 H/W와 S/W 암호시스템들이 효율적이지 못한 것은 암호시스템을 구현할 때 기본이 되는 암호 알고리즘에 문제점이 내포되어있기 때문이다. 즉, 효율적인 키분배와 키생성, 암호문과 평문사이에서의 변환과정 등이 수학적인 배경을 가지고 있기 때문에 고속 및 고비도의 암호시스템을 구현하는데 장애로써 존재하기 때문이다.

본 논문에서는 MAC(Message Authentication Code)등과 같은 네트워크 서비스 실현을 위해서 사용되어지고 있는 대칭형 암호시스템인 DES에서 DES의 기본구조인 Feistel구조를 병렬처리가 가능하도록 변형시켜 키의 길이에 처리속도가 비례관계를 갖지 않도록 하며, 데이터의 고속처리가 가능하도록 하는 병렬 Feistel 구조를 제안한다. 이러한 병렬 Feistel 구조는 DES의 기본구조에서만 적용되는 것이 아니고 키생성 및 분배에 해당하는 키 생성 알고리즘에도 적용하며 DES의 안전도에 절대적으로 기인하는 키생성 단계를 보다 효율적으로 구성함으로써 키의 길이는 짧고 데이터처리는 고속으로 수행할 수 있도록 하는데 주요한 기능을 수행하게 될 것이다.

데이터를 입력받아 64비트 또는 128비트로 포맷팅을 수행하고 다시 permutation, expansion, compaction, substitution, 그리고 iteration을 순서대로 수행하는 DES의 근본적인 구조는 파이프라인 방식을 행할 수 없다는 구조적 단점을 가지고 있다. 그래서 DES의 하드웨어 구현 방법에는 여러 가지가 등장하게 된다. 즉, iteration을 수행할 때 한 round만을 구현하여 한 round 내에서 16번의 iteration을 수행하는 형태의 하드웨어를 구현하거나 또는 16round를 모두 구현하여 DES의 일상적인 데이터를 처리하는 방법 등이 있으나 이들 모두 키의 길이가 길어지면 처리시간도 역시 비례적으로 길어지게 되며 특히, 16round 모두를 하드웨어로 구현하게 될 경우, 하드웨어의 크기가 무시하지 못할 정도로 커지게 된다는 부가적 단점도 있다.

그러므로 본 논문에서는 이러한 여러 가지 방법으로 구현되어지는 DES를 보다 효율적으로 사용하기 위하여 DES의 기본 구조인 Feistel 구조를 변형하여 1 round 또는 16 round를 하드웨어로 구현하더라도 키 길이에 비례적으로 증가하는 처리시간 및 하드웨어 크기를 감소시키며 더 나아가 대칭형 암호시스템의 효율적인 구현 및 비도를 더욱 증가시키고자 하는데 그 목적이 있다.

Ⅱ. 일반적인 DES 시스템

현대 암호화의 기술은 암호알고리즘의 공개와 키의 보호라는 특징을 가지고 있다. 암호시스템은 알려진 알고리즘에 입력되는 평문과 키를 섞어 암호문을 만들어 낸다. 즉, 키를 모르면 도저히 원래의 정보를 회복할 수 없으나 키를 알면 손쉽게 원래의 정보를 복원한다는 것이다.

DES는 64비트의 평문을 64비트의 암호문으로 만드는 블록 암호 시스템으로 64비트의 키를 사용한다. 이 64비트의 키(외부키) 중 56비트는 실제키(내부키)가 되고 나머지 8비트는 검사용 비트가 된다.

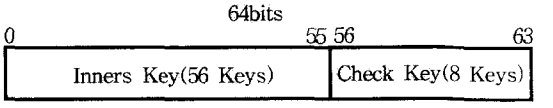


그림 1. DES 키의 구조
Fig. 1. Structure of DES Key.

DES의 구조는 16round의 iteration을 수행하며 각 round마다 transposition과 substitution을 거친 평문과 56비트의 내부키에서 나온 48비트의 키가 섞여 암호문을 형성하고 복호화는 암호화 과정과 동일하나 사용되는 키만 역순으로 작용한다.

DES의 기본 요소는 평문을 64비트로 blocking 하는 과정이 있으며 transposition, substitution를 DES에서는 table에 의해 행하게 된다. 또한 확장(expansion),

압축(compaction)과 배타적 논리합(exclusive-OR)이 기본 연산이 된다. DES의 키는 0과 127사이의 8개의 십진수로 구성되어 있는데 키는 어떤 규칙성이 나타나지 않도록 난수 발생기가 무작위로 골라낸 숫자들로 만들어진다. DES는 평문 64비트들이 치환(permutation)을 거친 후 32비트씩 두 부분으로 나뉘어진다. 오른쪽 부분은 키와 f라고 표시되는 알고리즘에 의해 섞이고, 그 결과는 왼쪽부분과 xor 연산으로 결합되어 새로운 round의 출발선상에 서게 된다. 이런 과정으로 각 블록은 16round의 처리과정을 반복하게 되며 키 역시 각 round를 지날 때마다 새로운 형태로 바뀌어진다. 매 round 각각의 키는 그 선행된 키에 의해 그 형태가 규정되어 이것이 다시 배타적 논리합으로 암호화되고 있는 평문의 내용과 결합된다. 평문의 암호화 알고리즘의 전체적인 구조는 그림 2와 같다. 64비트의 입력은 미리 정해진 표에 의해 initial permutation을 수행한다. 치환된 결과의 64비트 중 왼쪽 32비트를 L_0, 오른쪽 32비트를 R_0라 한다. 16round 끝난 후 마지막에서 다시 합쳐져서 정해진 표와 같은 역 초기 치환을 거쳐 64비트의 출력을 산출한다. i번째 round가 시작되기 전의 왼쪽 32비트를 L_{i-1}이라 표시하고, 오른쪽 32비트를 R_{i-1}일 경우, i번째 round의 왼쪽 32비트 L_i는 바로 이전 round의 오른쪽 32비트 R_{i-1}을 그대로 쓴다. f에서는 S-box가 있으며 compaction, expansion등의 방법이 쓰인다. 알고리즘 f는 그림 3과 같다. 이 알고리즘은 R_{i-1}과 키 K_i를 받아들여서 $f(R_{i-1},K_i)$를 출력하며 이 출력결과와 L_{i-1}과의 Ex-OR에 의하여 R_i를 생성하게 된다.

그림 3의 expansion은 미리 정해진 표에서 행하여진다. 이 결과 48비트는 6비트씩 나누어져 처음 6비트는 S-box S_1에, 다음 6비트는 S_2에, …순으로 입력되고 각 S_1, S_2, …, S_8들은 각각 4비트를 출력하게 되며 이 $8\times4=32$비트들은 치환표에 의해 치환된 후 32비트의 출력 $f(R_{i-1},K_i)$로 된다.

이러한 S-box의 역할은 미리 정해진 표에 의해 각각 6비트를 받아들여서 4비트로 출력을 하게 되는데 이러한 S-box들은 비선형 특성을 가지고 있다. DES의 키 생성 알고리즘은 그림 4와 같다. 여기에서 키는 암호화/복호화 키이다.

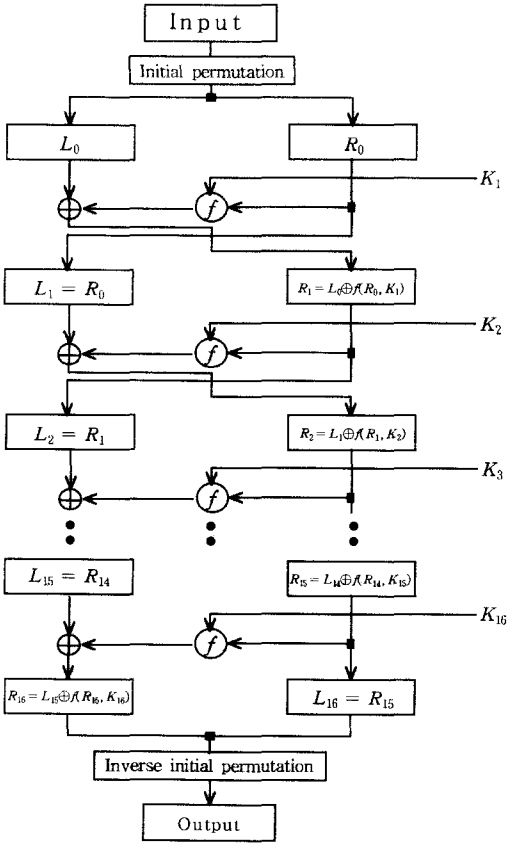


그림 2. DES의 암호화 과정
Fig. 2. Encryption Processing of DES.

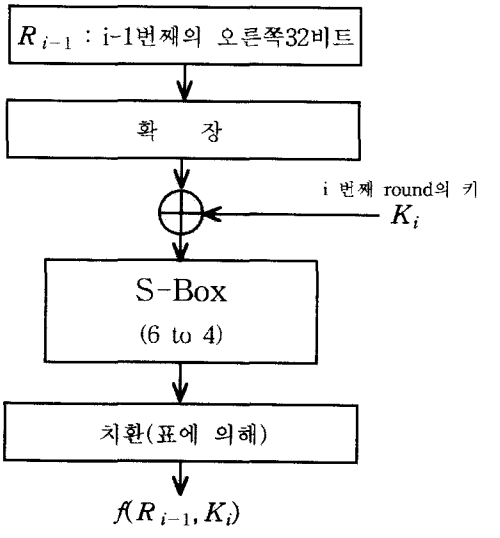


그림 3. f함수의 구조
Fig. 3. Structure of F-function.

그림 4에서 56bits의 키는 표에 의해 치환이 수행되어지고 그후 키는 28비트씩으로 나뉜다. 만약 각 round 키가 동일하다면, DES의 수학적 복잡도, 즉 비도가 저하될 수 있다.

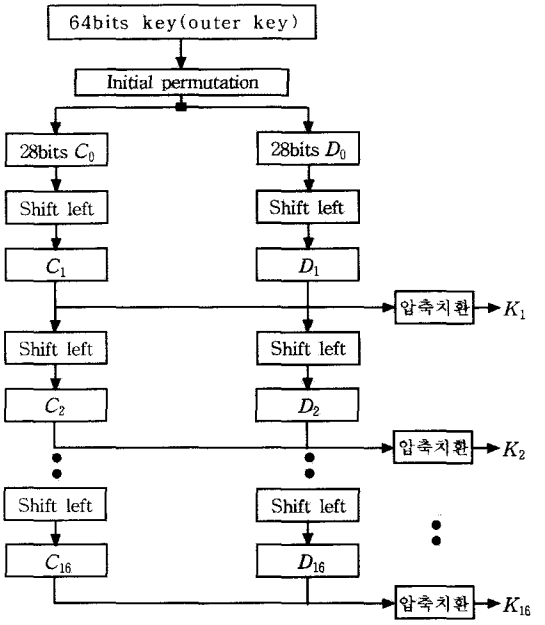


그림 4. DES의 키 생성 과정
Fig. 4. Key generated processing of DES.

Ⅲ. 병렬 Feistel 구조를 가지는 DES 설계

Feistel 암호화 시스템은 r-round의 과정을 거치는 동안에 평문 2m 비트(L_0, R_0)에 대하여 암호문 m 비트(L_r, R_r)를 생성하는 구조로 되어있다.

여기에서 $r \geq 1$, For $1 \leq i \leq r$.

$$L_i = R_{i-1}, R_i = L_{i-1} \oplus F(R_{i-1}, K_i) \quad (1)$$

그림 5는 기본적인 Feistel 구조를 나타낸 것으로서 기존 DES의 경우 그림 5를 채택하고 있다. 입력값에 대하여 좌우 대칭형으로 데이터를 나누고 f함수를 이용하여 신호변환(암호, 복호)을 수행하고 다시 합하는 형태이다. 이러한 구조는 DC(Differential Cryptanalysis) 해석법에 의하여 쉽게 해석이 가능해진다. DC 해석법은 입력과 출력의 관계로부터 Chosen-plaintext을 이용하여 공격하는 방법으로써 그림 6에서와 같은 특징을 Feistel 구조가 가지고 있기 때문에 Feistel 구조에 대한 변형은 DC 해석법에 대한 방어의 기능을 향

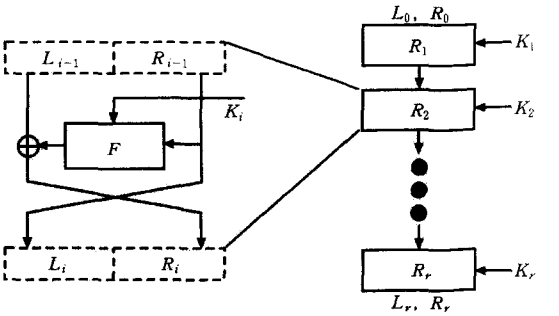


그림 5. 기본 Feistel 구조
Fig. 5. Basic Feistel Structure.

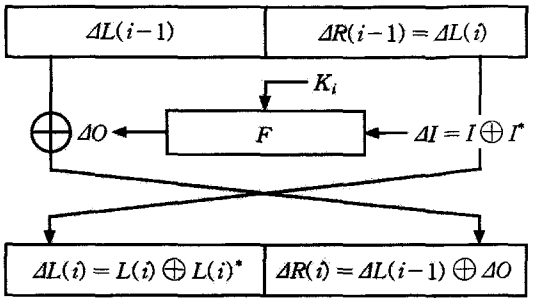
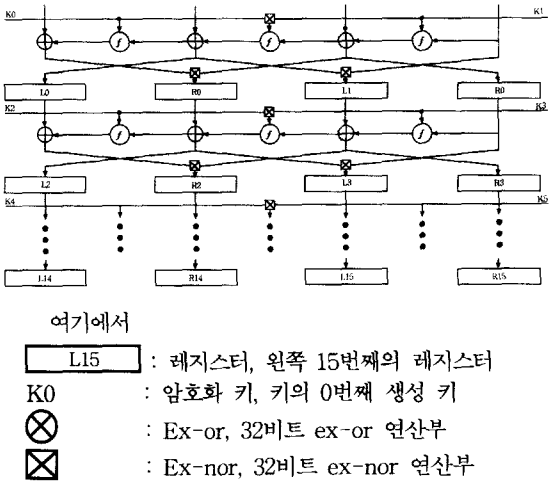


그림 6. Feistel 구조에서의 DC 특성
Fig. 6. DC characteristics in Feistel structure.

상시킬 수 있다. DC는 Key Exhaustive Search 보다 효과적이며 DC의 가장 큰 특징은 2개 입력에 대한 xor 값과 2개출력의 xor 값 사이의 xor 분포관계를 반복적으로 이용하는 방법으로써 DES에 사용되는 Feistel 구조가 xor 기능만을 가진다면 피할 수 없는 공격방법이 된다. 그래서 DES에서는 이러한 단점을 보완하기 위하여 16번의 iteration을 수행하게 되는데 이러한 iteration 후에도 Feistel의 DC 특성은 그대로 유지가 된다.

식 (1)에서와 같은 Feistel 구조는 DC의 특성에 의해 노출될 가능성이 높기 때문에 본 논문에서는 그림 7과 같은 구조를 가지는 구조로써 Feistel 구조를 변형하였다. 그림 7은 일반적인 DES에 대한 보안기능 강화 및 구현을 용이하게 하기 위한 방법으로써 Feistel 구조를 병렬처리가 가능하도록 한 구조이다. 입력에 대하여 초기 치환을 수행하고 128비트들에 대하여 32비트 4개의 데이터를 일반 DES의 Feistel 구조에서와 같이 처리하도록 하였다. 그러므로 본 논문에서 제안한 방식은 기본 32비트 처리를 이용하기 때문에 별도의 데이터 포맷이 필요 없고 초기 및 역 치환 수행시에도 비도의 차이가 국부적으로는 기존의

DES와 같다. 또한 round 처리시에도 16round를 수행할 필요가 없이 단지 8회의 iteration만으로도 기존 DES와 같은 등급의 보안기능을 수행할 수가 있다. 본 시스템은 이러한 기능을 수행하면서도 제안된 구조의 특징은 단순히 Feistel 구조 2개를 병렬 처리하는 형태가 아닌 DC 공격으로부터 더욱 안전성을 보장받을 수 있도록 한 것에 주안점이 있다. 16번의 round에 대하여 2 블록의 8round로 나누어 병렬로 만든 것이 아니고 그림 6과 같은 Feistel 구조에 대한 DC 특성이 나타나지 않도록 하기 위하여 변형된 구조인 것이다.



여기에서

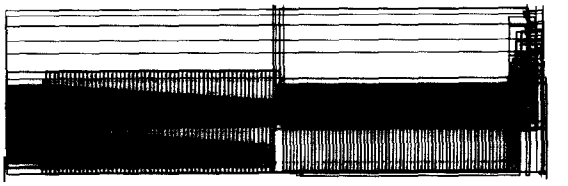
L15 : 레지스터, 왼쪽 15번째의 레지스터
K0 : 암호화 키, 키의 0번째 생성 키
⊗ : Ex-or, 32비트 ex-or 연산부
⊠ : Ex-nor, 32비트 ex-nor 연산부

그림 7. 변형된 DES Feistel 구조
Fig. 7. Modified DES Feistel structure.

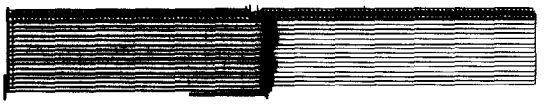
또한, 제안된 Feistel 구조에 대한 생성방정식은 다음 식 (2)와 같이 표현되어지며 식 (2)의 결과식을 보면 단일구조의 Feistel과 차이가 없는 Feistel 구조 형태를 따르게 된다. 그러나 이러한 구조를 따른다 하더라도 보다 높은 비도를 유지하기 위한 함수 ⊠가 포함되어 있어 암호측면에서 볼때는 더욱 높은 비도를 유지하게 된다. 실제적으로 암호의 비도측면에서는 ⊗ 또는 ⊠의 연산결과는 반대이지만 결과값으로는 원래의 값을 판별하는 것은 매우 어렵기 때문이다. 즉, ⊗와 ⊠를 병행하여 사용하였을 경우에는 hash 방정식과 같은 단방향 특성을 가지기 때문에 비인가자의 DC에 의한 데이터 해석은 매우 어렵게 된다. 식 (1)과 식 (2)를 비교해보면 식 (1)의 경우 각 round를 거칠 때마다의 방정식 변화의 폭이 1씩인데 반하여 식 (2)의 경우는 2씩 변화를 가져오게 된다. 즉, 한 round를 처리하는 과정은 기존 Feistel 구조에서 2 round의 과정을 수행하여야만 하는 결과와 같다. 그러므로 동일환경 내에서의 round만을 비교할 경우 기존보다 제안된 구조가 1/2배의 처리이득을 가져온다.

$$L_n = R_{n-2} \oplus F[(k_n \boxtimes k_{n+1}),\ L_{n-1} \oplus F[k_{n+1},\ R_{n-2}]]$$
$$R_n = L_{n-2} \oplus F[k_n,\ F[(k_n \boxtimes k_{n+1}),\ L_{n-1} \oplus F[k_{n+1},\ R_{n-2}]]]$$
$$= L_{n-2} \oplus F[k_n,\ L_n] \quad (2)$$

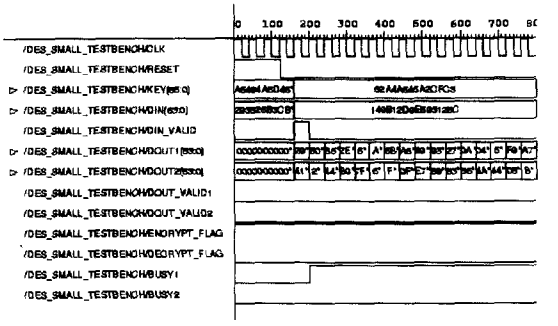
그림 8은 VHDL coding 및 Synopsys를 이용하여 일반 DES의 H/W 구현을 수행한 것으로써 그림 8(a)는 단일 round만을 구현하여 제어 신호에 의하여 16번의 iteration 처리를 수행한 것이며 그림 8(b)는 전체 16round에 대하여 설계된 것이다. 그림 8의 모든 부분들의 시스템은 round처리에 대한 부분을 수행하는 동안 출력을 얻을 수 없기 때문에 encryption을 수행하기 위해서는 시스템 처리시간이 절대적으로 늘어나게 되며(총 8round 처리에 걸리는 시간) H/W의 구현시에도 구현의 어려움(제어신호 구성)이 존재하게



(a) 단일 round 사용한 DES
(a) DES using unit round



(b) 16 round를 사용한 DES
(b) DES using 16th round

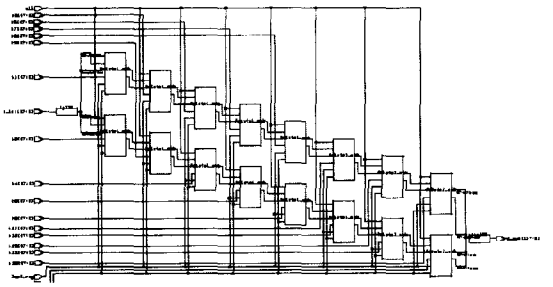


(c) 기존 시스템의 모의실험결과
(c) Simulation waveform of existing DES

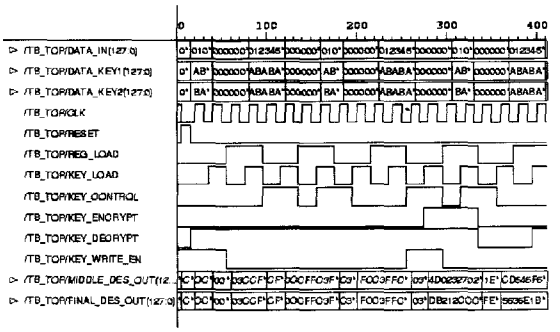
그림 8. 기존의 DES 시스템
Fig. 8. Existing DES system.

된다. 그림 8(c)는 이러한 기존의 DES에 대한 모의실험 결과이다.

그림 9(a)는 본 논문에서 제안한 병렬처리 DES의 구조를 VHDL 및 Synopsys를 이용하여 구현하였다. 입력값에 대한 초기 치환을 수행한 후 바로 32비트씩 4 블록으로 분할되어 동시에 f 함수와의 계산을 수행하게 되고 8round에 대한 iteration만을 수행하고 나서 바로 출력이 나타나게 된다. 그럼으로써 시스템 처리시간 및 H/W의 구현상의 문제를 용이하게 해결할 수 있었다. 그림 9(b)는 제안된 시스템의 모의실험결과이다.



(a) 제안한 병렬처리 DES의 구조
(a) DES structure of proposed parallel processing method



(b) 제안된 시스템의 모의실험결과
(b) Simulation waveform of proposed DES

그림 9. 제안된 병렬처리 DES 시스템
Fig. 9. Proposed parallel processing DES system.

이상과 같은 모의실험 결과, 단일 round와 16 round, 그리고 제안된 병렬구조에 대한 데이터 처리율은 다음과 같다.

단일 round 사용한 DES : 1(round 수) × 3(1round 처리 clock) × 2(control block 처리 clock) × 16 (iteration 수) × 64 bits(한번 처리되어질 때의 데이터량) = 96 × 64 bits = 416kbps이며, 16round를 사용한 DES : 16(round 수) × 3(1round 처리 clock) × 2 (control block 처리 clock) × 1(iteration 수) × 64 bits(한번 처리되어질 때의 데이터량) = 96 × 64bits = 416kbps, 제안된 병렬처리 DES : 8(round 수) × 3(1 round 처리 clock) × 2(control block 처리 clock) ×1 (iteration 수) × 64bits(한번 처리되어질 때의 데이터량) = 48 × 64bits = 833kbps임을 알 수 있었다. 또한 128bit에 대한 처리율은 16round 및 1round에 대해서는 832kbps이며, 제안된 방법을 사용하는 경우 1.666Mbps임을 알 수 있었다.

이러한 결과에 대하여 기존의 방법(그림 8의 a, b)과 제안된 방법(그림 9의 a)을 상호 비교하여 분석한 후 표로 작성하면 다음 표 1과 같다. 또한 표 1에서는 기존에 사용된 2가지 방법과 제안된 방법에 대한 회로 합성에서의 게이트 수를 포함하였다. 비교 분석한 결과 게이트 수에서는 기존과 거의 차이가 없음을 알 수 있었다. 여기에서는 처리되어지는 데이터의 크기를 64 bits와 128bits로 구별하여 표시하였다.

표 1. 기존 및 제안된 방법으로 설계된 DES의 크기 및 처리율 비교
Table 1. Comparing gate count and processing rate for existing and proposal DES.

64/128bits processing	Gate counting	Processing method	Processing rate(@40Mhz)
DES using unit round with control block(그림 8-a)	1090	-feedback after 1 round processing -16 round iteration	416kbps (64bits) 832kbps (128bits)
DES using 16th round (그림 8-b)	6159	-each 16th processing	416kbps (64bits) 832kbps (128bits)
DES using modified parallel feistel structure (그림 9-a)	6638	-8 round processing -simultaneously left and right block processing	833kbps (64bits) 1.66Mbps (128bits)

Ⅳ. 결 론

본 논문에서는 정보통신의 눈부신 발달과 인터넷의 급격한 확산으로 인한 여러 가지 첨단기능을 수행함에 있어 보다 안전하게, 보다 투명성이 있는 네트워크의 보장을 요구하며, 보다 빠른 네트워크의 성능을 기대

하기 위하여 기존의 DES를 H/W의 구현에 대하여 병렬로써 처리하도록 하였다. 본 논문에서 사용된 병렬기법의 Feistel 구조는 처리시간(8round에 대한 1 iteration) 및 H/W 구현의 용이성(제어신호 구성)등을 매우 효율적으로 처리하는 것으로 사료됨으로써 전자상거래, 전자화폐, 전자서명 등의 여러 가지 첨단기능을 수행하며 미래에는 더욱 진보된 서비스를 제공하게 될 것이다.

본 논문에서는 DES의 기본 구조인 Feistel 구조를 병렬로 변화시킨 병렬 Feistel 구조를 가지는 DES를 제안하였다. 제안된 병렬 Feistel 구조는 자체의 구조적 문제 때문에 pipeline 방식을 사용할 수 없어 데이터 처리속도와 데이터 보안사이에서의 trade-off관계를 가질 수밖에 없었던 DES의 처리과정중의 round 수행을 기존과 비교하여 1/2배로 감소시켜 수행할 수 있으며, 대용량 데이터(128비트)에 대하여 한번의 처리로써 동시에 수행할 수 있다는 빠른 처리에 대한 성능을 크게 향상시킬 수 있고 또한 DC에 의한 공격으로부터 더욱 자유로울 수 있도록 xnor, xor를 병행하여 사용하였으며 더불어 Feistel 구조를 적용한 국내 암호시스템의 표준인 SEED에 제안된 방식을 적용할 경우 지금보다 더욱 우월한 보안 기능 및 고속의 처리 능력을 발휘하게 될 것이다.

참 고 문 헌

[1] D. E. Denning, "The data encryption standard : Fifteen years of public security", Dist.Lecture, 6th Annual Comp. Security Appl. Conf., IEEE Comp, Soc. Press, pp. x-xv, 1990.

[2] A. Shimizu, S. Miyaguchi, "Fast Data Encipherment Algorithm, FEAL", Proc. of Eurocrypt 91, 1991.

[3] E. Biham and A. Shamir, "Differential Cryptanalysis of DES-like Cryptosystems," Journal of Cryptology, Vol. 4, No. 1, pp. 3-72, 1991.

[4] X. Lai, "On the Design and Security of Block Ciphers", ETH Series in Information Processing, vol. 1, 1992.

[5] E. Biham, A. Shamir, "Differential Cryptanalysis of Data Encryption Standard", Springer-Verlag, 1993.

저 자 소개

李 善 根(正會員)

Journal of Electrical Engineer and Information Science Vol. 4, No. 6, December 1999

金 炯 均(正會員)

大韓電子工學會 論文誌, Vol. 36-C, No. 2, pp. 46-53

金 煥 溶(正會員)

大韓電子工學會 論文誌, Vol. 36-C, No. 2, pp. 46-53