DES의 하드웨어 구현

김영진*° , 염흥열**, 한승조*, 최광윤*
*조선대학교 전자·정보통신공학부, 정보통신 시스템 lab.
**순천향대학교 전자공학과.

H/W Implementation of DES Algorithm

YoungJin Kim*° , HeungYoul Youm**, SeungJo Han*, KwangYun Choi*
*Dept. of Electronic and Information communication Eng. Chosun Univ.
**Dept. of Electronics Eng. Soonchunhyang Univ.

Abstract

본 논문에서는 암호화 알고리즘의 표준으로 자리잡은 DES(Data Encryption Standard) 알고리즘을 시스템 설계 기술언어인 VHDL을 이용하여 top-down 방식으로 설계하고 시뮬레이션을 수행하여 암·복호화의 결과를 보여준다. 또한 이것을 FPGA로 구현함으로써 하드웨어가 차지하는 면적과 속도를 산출 비교하여 암호화 속도 및 크기의 최적화를 위한 설계 방식을 제안한다.

본 논문에서는 최종적으로 V-system을 이용하여 시뮬레이션을 수행하고 Synopsys의 EDA 툴을 이용하여 합성을 한 후에 Xilinx사의 xdm을 이용하여 XC4025E에 칩으로 구현하였다.

I. 서론

컴퓨터의 성능이 급속도로 발전하여 컴퓨터 통신망이 확산되고 개방화됨에 따라 다가오는 21세기 정보화 사회에서는 정보가 재화(財貨)의 가치로 인정받고 국가경제 발전의 성쇠를 좌우하는 중요한 변수로 작용하게 되었다. 또한 과거의 기업들은 축적된 기술(know how)을 중요시했지만 현재의 기업들에게는 정보를 보다 빨리 찾을 수 있는 능력(know where)이 중요하게 여겨지고 있다. 그러나 정보화의 순기능과 함께 비인가자에 의한 불법적인 사용 및 복제, 중요한 정보 유출 및 변경 그리고 훼손 등과 같은 역기능 현상들이 날로 증가되고 있으며, 그에 따른 피해규모가 심각한 수준에 도달하고 있다.

컴퓨터를 이용함으로서 편리한 점이 많이 있는 반면에, 권한이 없는 불법적인 침입자로부터 조직이나 개인의 중요한 자료를 보호하고 컴퓨터 통신망을 통해 전송되는 자료의 불법적인 도청이나 내용의 변조를 방지하기 위한 암호화가 요구되어진다. 암호화를 위해서는 암호알고리즘을 사용하는데, 암호알고리즘은 평문을 암호문으로 바꾸어주는 암호화 과정과 암호문을 본래의 평문으로 복원하는 복호화 과정으로 이루어지며, 암·복호에 필요한 키가 동일한 관용키 암호시스템과 암·복호화에 필요한 키가 각각 다른 공개키 암호시스템으로 나누어진다. 일반적으로 관용키 암호시스템은 수행속도는 빠르나 인증성이 결여되며, 공개키 암호시스템은 인증성은 있으나 수행속도가 늦은 단점이 있다.

현재 가장 보편적으로 실용화되어 사용되는 암호알고리즘이 IBM의 Lucifer 알고리즘을 기반으로 개발한 DES이다. DES는 암호화, 복호화 알고리즘이 대칭적이며 치환과 대치 그리고 S-Box로 구성된 블록암호화 시스템이다.

II. 이론적 배경

DES는 대치와 치환이 반복되는 알고리즘으로 원문의 한 블록을 64비트씩 읽어들여, 이를 2개의 32비트 서브 블록(L, R)으로 나누어 각 라운드마다 암호함수를 적용하여 암·복호화 한다.
암·복호화에 적용되는 단일 라운드를 식으로 표현하면 다음과 같다.

- 암호화 : $L_i = R_{i-1}$
  $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$

- 복호화 : $R_{i-1} = L_i$
  $L_{i-1} = R_i \oplus f(R_{i-1}, K_i)$
  $= R_i \oplus f(L_i, K_i)$

여기에서

$f(R_{i-1}, K_i) = P(S_1(B_1), \cdots\cdots, S_8(B_8))$
$B_1, B_2, B_3, \cdots\cdots, B_8 = E(R_{i-1}) \oplus K_i$

이다.

이것을 블록도로 나타내면 그림 1과 같다.

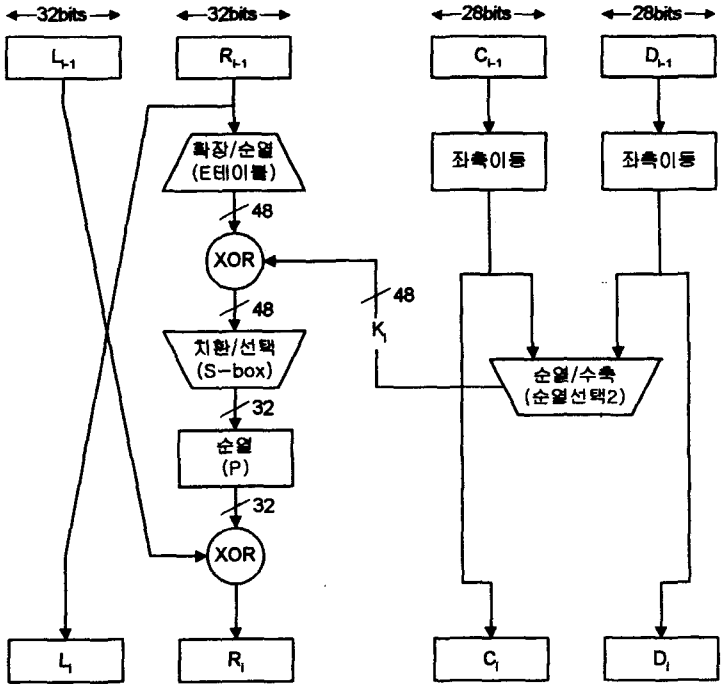

그림 1. DES의 단일 round 반복과정

III. 칩의 설계

본 논문에서는 DES의 H/W 설계를 다음과 같이 기능별로 나누어 Synopsys사의 EDA 툴을 사용하여 테스트용 칩인 DES-FPGA를 설계한다.

- 블록단위(64비트)로 데이터를 암호화시키는 암호 처리부
- 암호화된 데이터를 8비트 단위로 입출력하는 입출력부
- 데이터를 암호화시키기 위해 각 라운드에 필요한 키 값을 생성시키는 키 생성부

암호 처리부, 입출력부 그리고 키생성부는 하나의 칩으로 구현하게 되며, 외부에서 제어신호에 의해 암·복호화가 이루어지게 된다.

본 과제에서 개발하고자 하는 암호칩은 암호칩의 입출력을 통제하는 Control Block, key를 생성하기 위한 Key Block 그리고 DES의 각 라운드를 수행하는 Round Block 등 크게 3개의 블록으로 나누어진다. 암호칩은 앞으로 사용하게 될 key를 먼저 입력받아 각 라운드별로 key generator에 의해 key를 생성해 놓고 이 key를 이용하여 입력되는 데이터를 암·복호화 한다. 데이터는 한번에 64비트씩 처리되며 외부와의 입출력은 8비트 단위로 이루어지는 범용성 칩을 제작하였다.

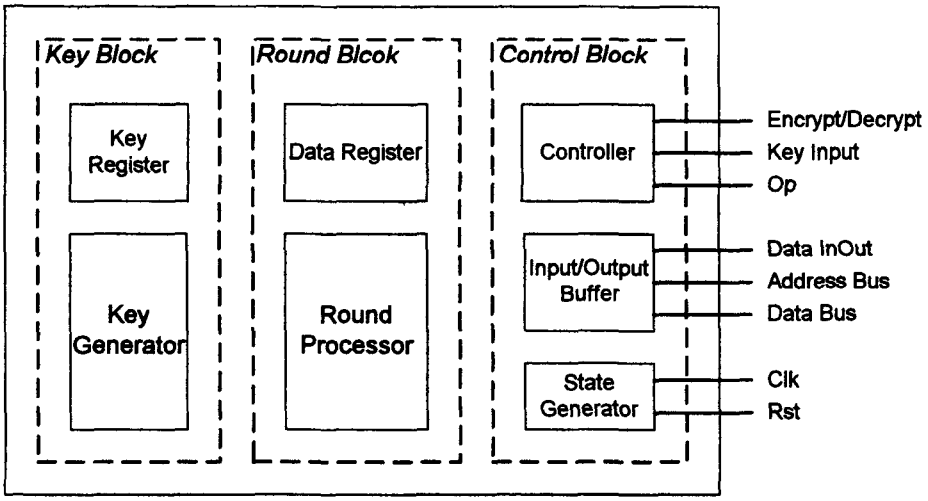

그림 2. 암호칩의 블록도

1. Control Block

Control 블록에서는 데이터의 입출력을 위한 Buffer와 각 라운드를 카운트하는 Round counter, 각 블록들의 동작을 콘트롤하는 Controller 그리고 칩의 전체적인 타이밍을 조절하기 위한 State generator 등으로 세분되어 진다.

1.1 Buffer

버퍼는 데이터의 입출력시 데이터의 크기를 맞추기 위한 임시 저장 공간으로서 데이터의 입력시는 address_bus가 지정하는 위치에 8비트씩 채우는 역할을 한다.

데이터의 입출력 방향은 dir 신호에 의해서 결정되어지게 되는데, dir이 '1'이고 in_out이 '1'로 trigger 되면 address_bus가 지정하는 buffer의 위치에 data_bus에 있는 값을 저장하게 된다. 또 dir이 '0'이고

in_out이 '1'로 trigger되면 address_bus가 지정하는 buffer의 위치에 있는 8비트 데이터가 data_bus상에 있게 된다. buffer는 64비트로 이루어져 있으며 데이터의 입력 동작(dir = '1')일 때 버퍼에 데이터가 차게 되면 key_in 신호가 '1'로 trigger될 때 buffer의 64비트 데이터는 key generator로 전송되게 되고, key_in 대신 op가 '1'로 trigger 되면 buffer의 데이터는 round register로 전송된다.

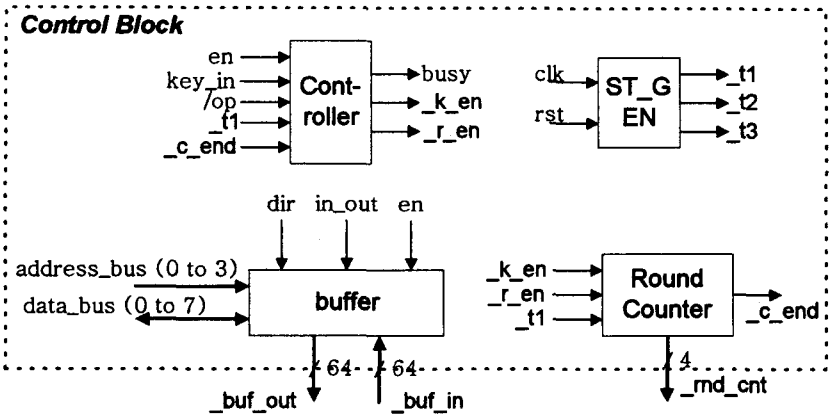

그림 3. 컨트롤 블록의 내부구조

1.2 Round Counter

암호칩은 key를 생성할 때나 16라운드의 수행중일 때 key register의 현재 위치를 지시할 포인터가 필요하다. round counter는 현재 수행중인 라운드가 몇 번째인지를 나타내는 4비트의 신호선으로 key 생성시에는 왼쪽이동(left shift) 횟수나 key register의 저장위치를 지정하는데 쓰인다. 또한 암호라운드를 수행 중에는 현재 수행중인 라운드에 맞는 sub-key를 제공하기 위해 사용된다.

1.3 Controller

칩을 이루고 있는 각각의 블록들은 서로 배타적인 동작을 할 필요가 있다. 예로 sub-key를 생성중일 때는 암호 라운드는 동작을 하지 않아야 하고 16라운드가 모두 처리된 후에만 데이터의 입출력이 이루어 져야 한다. 이렇듯 외부의 제어신호에 따라 칩내부의 블록들에게 제어신호를 전송해주는 부분이 controller이다.

이 블록은 en이 '1'이고 _t1이 '1'로 trigger될 때 key_in이 '1' 이면 _k_en을 op가 '1' 이면 _r_en을 '1'로 trigger 시킨다.

그리고 busy 신호선은 칩 외부로 현재의 작업 완료 여부를 알리는 신호선으로서 _c_end로부터 영향을 받는다.

1.4 State Generator

암호칩의 모든 동작은 3 cycle 동안에 수행되어 진다. 암호 과정의 1라운드 수행이나 key의 생성시에는 fetch, execute, store 동작으로 이루어진다. state generator는 이 3가지 동작을 구분하기 위해 입력되는 clock은 _t1, _t2, _t3로 분리된다.

2. Key Block

이 블록은 입력된 64비트의 데이터를 가지고 16개의 서브키를 생성하는 블록으로 _k_en이 '1' 일 때는 _rnd_cnt가 지시하는 라운드의 서브키를 생성하며, _k_en이 '0' 이면 _rnd_cnt가 지시하는 라운드의 서브키를 _sub_key(48 비트)로 출력한다.

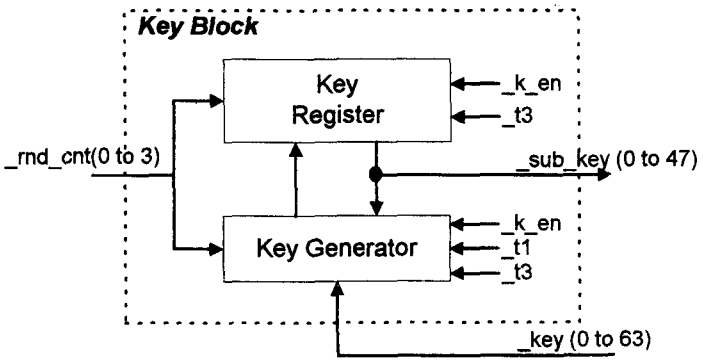

그림 4. 키 블록의 내부구조

3. Round Block

round block은 각 라운드에서 생성된 data를 저장하기 위한 Register와 단일라운드의 수행블록으로 구성되어 있다.

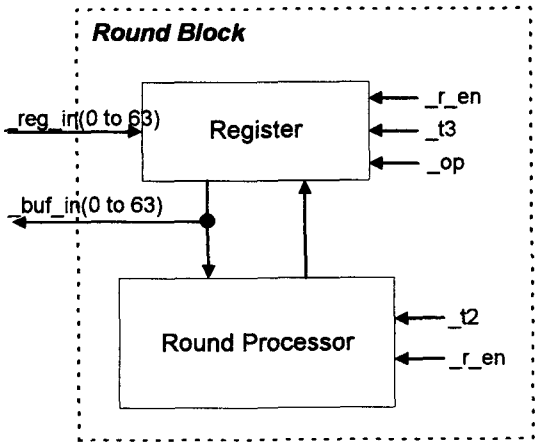

그림 5. 라운드 블록의 내부구조

1라운드의 동작은 _r_en이 '1' 일 때 이루어지며 이 동작을 16회 반복한 후에 _buf_in으로 결과값을 출력한다.

round processor는 실질적인 암호화 과정을 수행하는 부분으로 E table, S-box, P table 들로 구성되어 있다.

IV. 칩의 구현

본 논문에서는 암호화칩의 target으로 Xilinx FPGA를 사용하였다. 따라서 칩의 최적화는 FPGA의 cell 개수를 평가 대상으로 이용하였으며 면적 뿐 만 아니라 암·복호화의 속도의 최적화도 함께 고려하였다.

1. 칩면적의 최적화

면적을 가장 많이 차지하는 부분은 Key Block으로서 16라운드에 사용되는 sub key를 저장하기 위한 램이 차지하는 공간이 크기 때문이다. 이는 FPGA로 램을 구현하게 되는 결과로 인해 면적이 커질 수 밖에 없다. 다음으로 많은 면적을 차지하는 부분은 Round Block 내의 S-box로서 여기서는 S-box 내의 table이 ROM처럼 쓰여졌기 때문이다.

	cell area	arrival time
Control Block	61	16.64
Key Block	1835	3.26
Round Block	391	4.39
Total	2275	16.64

표 1. 암호칩의 구현 결과

2. 속도의 최적화

암호칩의 제작시 따르는 속도상의 지연 문제에서 가장 큰 부분을 차지하는 곳은 16라운드 부분으로 pipe line처리를 위해서는 16개의 독립된 라운드가 존재하는 것이 바람직하겠으나 한 라운드가 391개의 cell을 필요로 하는 점을 고려하였을 때 FPGA 칩으로 16라운드를 구현하는 것은 비현실적이다. 따라서 본 논문에서는 단일 라운드 처리기를 16회 반복수행하여 16라운드의 결과를 얻는 방식을 이용한다. 이 결과 64비트의 데이터를 16라운드 수행하는데 필요한 cycle은 48 cycle이 된다.

V. 실험 및 고찰

1. 합성결과

우선 암호칩의 가장 많은 부분을 차지하는 부분은 key generator 부분임이 위와 같이 보여졌다. 그러나 key generator에서 차지하는 면적의 대부분은 sub key를 저장하고 있는 ROM으로써 이는 최적화가 더 이상은 불가능하다.

다음으로 많은 면적을 차지하는 부분은 S-box로서 단일 라운드를 반복 수행하는 기법을 16개로 독립시키는 것이 수행속도면에서 바람직하기 때문에 실제로 최적화가 필요한 부분으로서 앞으로 면적을 줄이는 연구가 더 이루어져야 할 것이다.

그림 6은 Synopsys를 이용하여 FPGA cell로 합성한 결과이다.

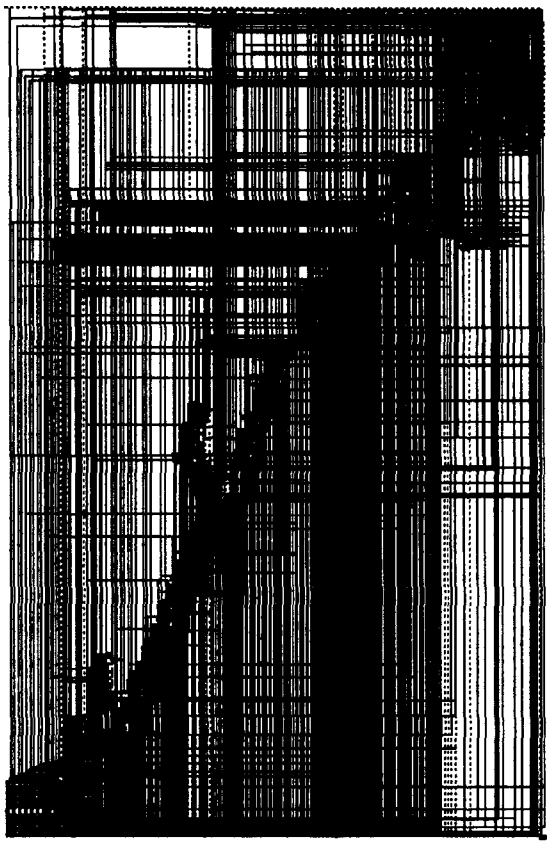

그림 6 암호칩의 합성결과

2. 시뮬레이션 결과

본 논문에서 제시한 최적화 된 설계를 다음과 같이 시뮬레이션을 거쳐 검증하였다.

- 적용키 : "YoungJin" = $(596F756E674A696E)_{16}$
- 적용데이타 : "DES-chip" = $(4445532D63686970)_{16}$
- 시뮬레이터 : V-system for IBM PC
- 수행시간 : 4 μs
- 입력클럭 : 33 MHz

위와 같은 조건에서 그림7, 그림8과 같이 암·복호화가 정상적으로 수행되었다. 한번에 64비트의 데이터를 암호화시키며 수행시간은 총 1355 ns가 걸렸다. 따라서 암호화 속도는 약 47.2 Mbps가 된다..

VI. 결론

본 논문에서는 DES 암호화 알고리즘을 Xilinx FPGA를 이용한 하드웨어 chip으로 구현함으로써 암·

복호화 수행 속도 및 칩이 차지하는 면적의 최적화를 제안하였다. 본 알고리즘을 주문형 반도체로 제작하게 되면 레지스터나 buffer를 고속의 램으로, S-box를 고속의 ROM으로 대체할 수 있어 data의 pipe line 처리 등, 크기와 속도를 수십배 높이는 것이 가능하다. 따라서 암호칩을 ASIC으로 설계하고 최적화하는 것은 앞으로의 연구과제이다.

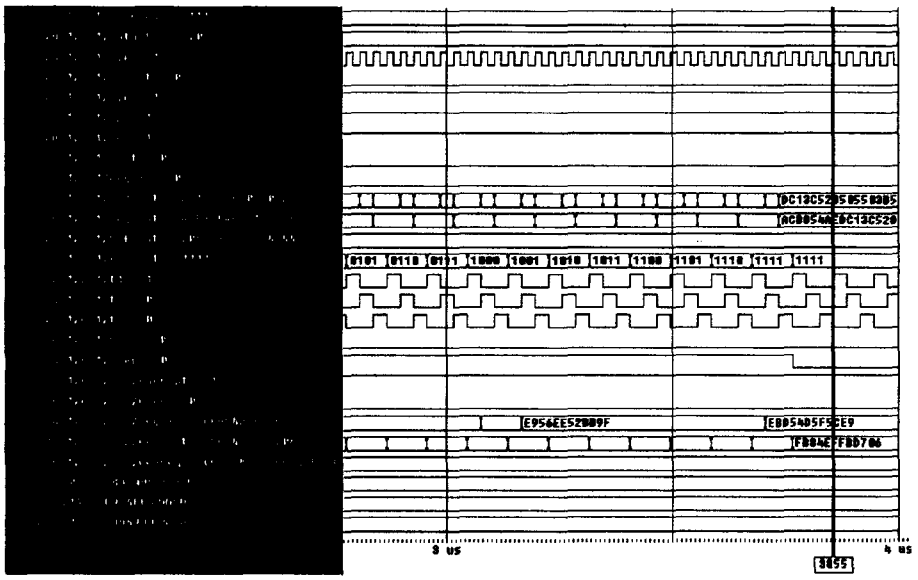

그림 7 암호화 시뮬레이션

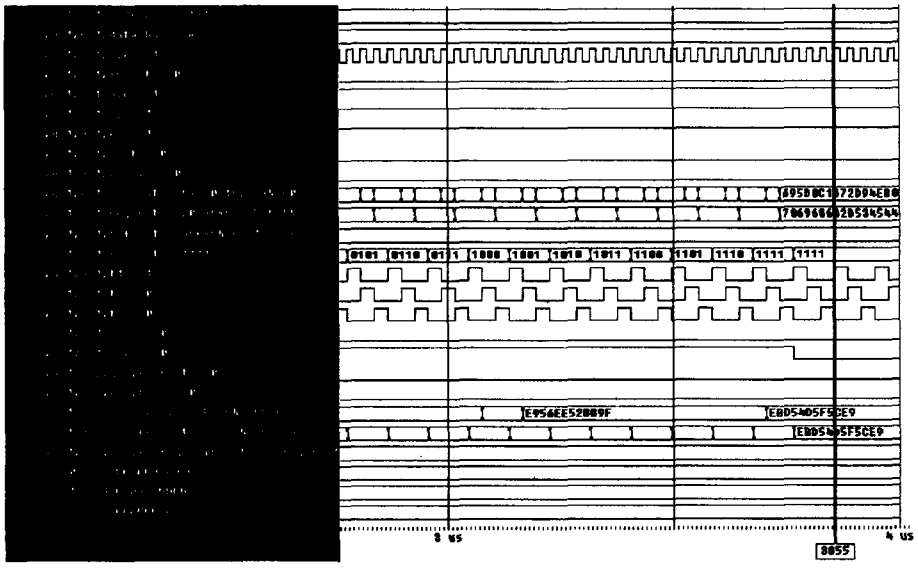

그림 8 복호화 시뮬레이션

<참 고 문 헌>

1. M. E. Hellman, "DES will be totally Insecure within ten Years," IEEE Spectrum, Vol.16, no.7, pp.32-39, 1979.

2. E. Biham & A. Shamir, "Differential Cryptanalysis of the Full 16-Round DES," Adavances in Cryptology-CRYPTO '92 Proceedings. Berlin:Springer -Verlag, 1993.

3. H. H. Evertse, "Liner Structures in Block Ciphers," Advabeces in Cryptology-EUROCRYPT '87, Proceedings. Berlin:Springer-Verlag, pp.249-266, 1987.

4. G. I. Davida, D. J. Linton, C. P. Szelag & D. L. Well, "Data base security," IEEE Transactions on Software Engineering, Vol.SE-4, no.6, pp.531-533, 1978.

5. J. H. Moore & G. J. Simmons, "Cycle Structure of the DES for Keys Having Palindromic (or Antipal-indromic) Sequences of Round Keys," IEEE Transaction on Software Engineering, Vol.13, no.6, pp.858-864, 1982.

6. Schneirer Bruce, Applied Cryptography, John Wiley & Sons, Inc., pp.219-296

7. J. B. Kam & G. I. David, "Structured Design of Substitution permution Encryption Networks," IEEE Transactions on Computer, Vol.28, no.10, pp.747-753, 1989.

8. A. F. Webster, & S. E. Tavares, "On the design of S-boxes," Proceedings of Crypto '85, 1985.

9. J. B. Kam, & G. I. Davida, "Structured Design of Substitution Permutation Encryption Network," IEEE Transactions on Compiters, Vol.28, no.10, pp.747-753, 1989.

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