

DES 구현

심민주

목차

- Git & Github 사용방법
- C언어 공부
- DES 구현

1) Git & GitHub 사용방법

0. Git이란?

컴퓨터 프로그램 소스를 공유하고
협업하여 개발할 수 있는 버전 관리 시스템

0. GitHub이란?

Git 에 프로젝트 관리 지원기능을
확장하여 제공하는 웹 호스팅 서비스

0. GitHub이란?

Showing 1 changed file with 4 additions and 2 deletions.

6 README.md

...	...	@@ -1,7 +1,9 @@
1	-	# hello-world
1	+	# Hello-world
2	2	### 안녕하세요
3	3	저는 ForteDev 입니다.
4	4	
5	5	Github에서는 체계적으로 수정할 수 있고 많은 사람들과 효율적으로 동시에 작업할 수 있습니다.
6	6	
7	-	정말 멋지죠!
7	+	바로 이렇게 수정된 모습을 하나하나 확인할 수 있죠!!
8	+	
9	+	## HAYO

- 원본 프로젝트를 건드리지 않고, 새 수정본을 만들어 수정가능
- 수정할 때마다 기록(commit)이 남고, 원할 때 특정 변경 위치로 돌아갈 수 있음
- 어떤 팀원이 수정을 잘 했더라도 팀원에게 승인을 받아야 원본 프로젝트 변경 가능

1. GitHub 계정 만들기

1 단계

<https://github.com>

Built for
developers

GitHub is a development platform inspired by the way you work. From **open source** to **business**, you can host and review code, manage projects, and build software alongside millions of other developers.

Username

Pick a username

Email

you@example.com

Password

Create a password

Use at least one letter, one numeral, and seven characters.

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy policy](#). We'll occasionally send you account related emails.

1. GitHub 계정 만들기



2단계

무료 이용

Welcome to GitHub
You've taken your first step into a larger world, @ForteDevBlog.

Completed Set up a personal account Step 2: Choose your plan Step 3: Tailor your experience

Choose your personal plan
Every plan comes with GitHub's most-loved features: Collaborative code review, issue tracking, the open source community, and the ability to join organizations.

 Free	 Developer
\$0 per month	\$7 per month (view in KRW)
Includes: Personal account Unlimited public repositories Unlimited collaborators	Includes: Personal account Unlimited public repositories Unlimited private repositories Unlimited collaborators Free for students as part of the Student Developer Pack .

☒ **Help me set up an organization next**
Organizations are separate from personal accounts and are best suited for businesses who need to manage permissions for many employees. [Learn more about organizations](#)

☐ **Send me updates on GitHub news, offers, and events**
Unsubscribe anytime in your email preferences. [Learn more](#)

[Continue](#)

매달 \$7
(비공개 저장소 사용 가능)

1. GitHub 계정 만들기

3단계

Welcome to GitHub

You'll find endless opportunities to learn, code, and create, @ForteDevBlog.

✓ Completed
Set up a personal account

📁 Step 2:
Choose your plan

⚙️ Step 3:
Tailor your experience

How would you describe your level of programming experience?

☐ Very experienced ☐ Somewhat experienced ☐ Totally new to programming

What do you plan to use GitHub for? (check all that apply)

☐ Research
☐ Project Management

Which is closest to how you would describe yourself?

☐ I'm a hobbyist ☐ I'm a professional ☐ I'm a student
☐ Other (please specify)

What are you interested in?

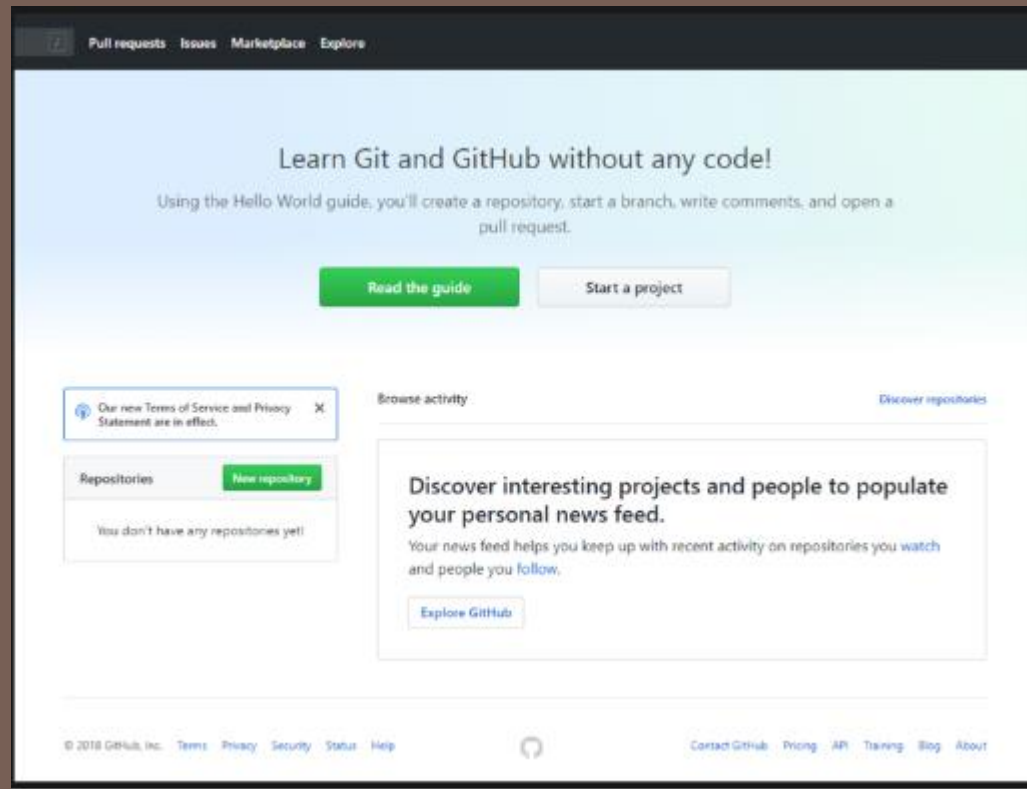
e.g. tutorials, android, ruby, web-development, machine-learning, open-source

[Submit](#) [skip this step](#)

Skip 가능

1. GitHub 계정 만들기

완료



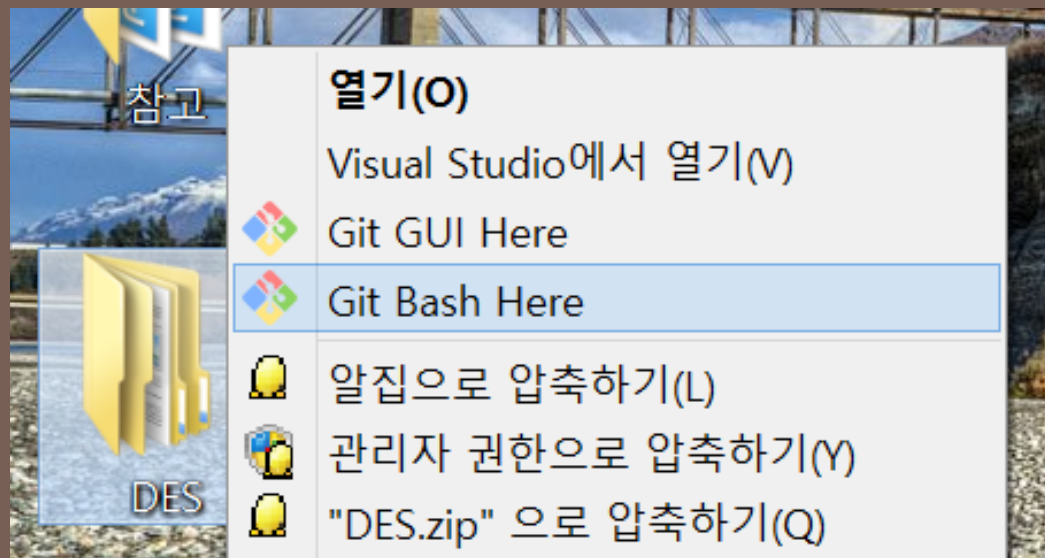
2. Git 다운

<https://git-scm.com/downloads>



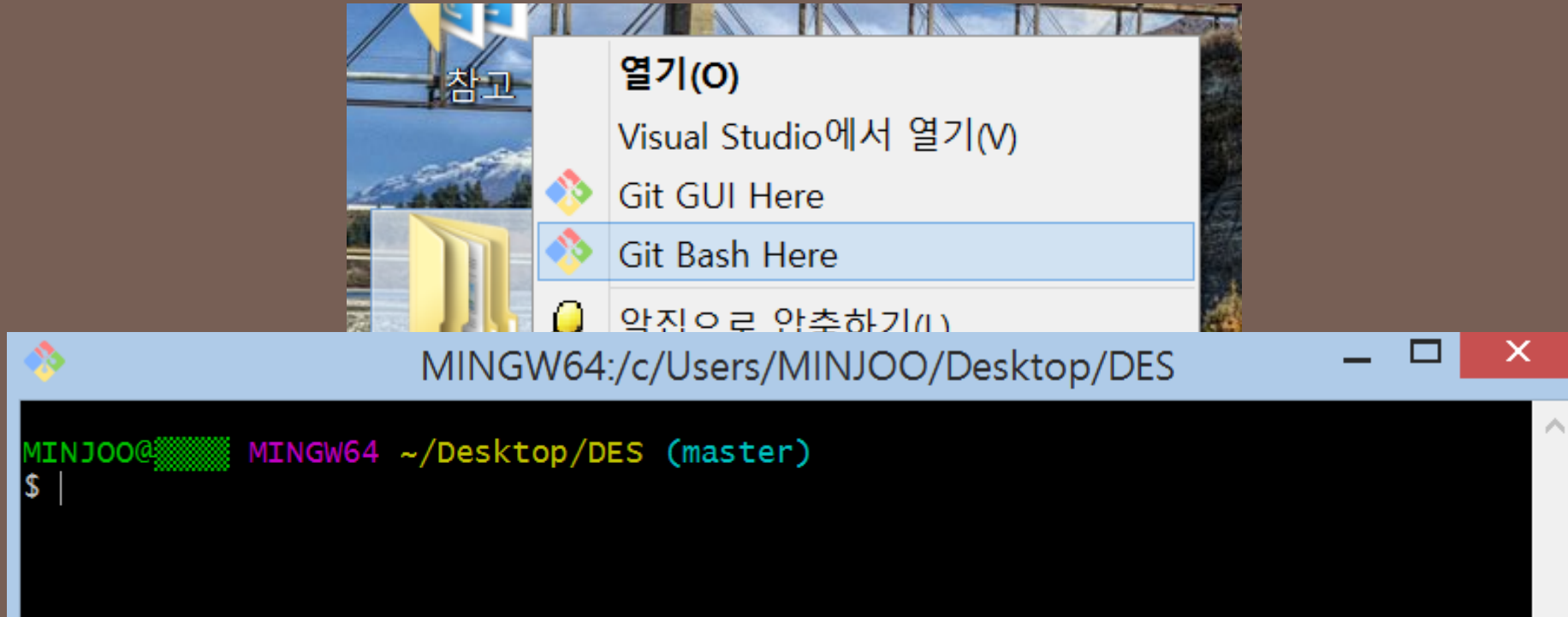
The screenshot shows the Git website's Downloads page. At the top left is the Git logo (a red diamond with a white branching diagram) followed by the text "git --local-branching-on-the-cheap". To the right of the logo is a search bar with the placeholder text "Search entire site...". Below the logo, there is a sidebar with links: "About", "Documentation", "Downloads" (highlighted in red), "GUI Clients", "Logos", and "Community". The main content area is titled "Downloads" and features three operating system icons: "Mac OS X" (Apple logo), "Windows" (Windows logo), and "Linux/Unix" (Tux penguin logo). To the right of these icons is a large monitor graphic displaying the text: "Latest source Release", "2.20.1", "Release Notes (2018-12-15)", and a button that says "Download 2.20.1 for Windows". At the bottom of the main content area, there is a link that says "Older releases are available and the Git source repository is on GitHub."

2. 로컬 저장소 만들기 위한 준비



로컬 저장소 : 개인적인 공간에 있는 저장소이기 때문에 다른 사람이 접근 할 수 없음

2. 로컬 저장소 만들기 위한 준비



3. \$ git init : 깃 저장소 초기화

```
MINJOO@MINGW64 ~/Desktop/DES (master)  
$ git init  
Reinitialized existing Git repository in C:/Users/MINJOO/Desktop/DES/.git/
```

4. \$ git status : 저장소 상태 체크

```
MINJOO@MINGW64 ~/Desktop/DES (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   HW/DES_HW_1.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    HW/DES_HW_2.py

no changes added to commit (use "git add" and/or "git commit -a")
```

4. \$ git status : 저장소 상태 체크

```
MINJOO@MINGW64 ~/Desktop/DES (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   HW/DES_HW_1.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        HW/DES_HW_2.py

no changes added to commit (use "git add" and/or "git commit -a")
```


4. \$ git status : 저장소 상태 체크

```
MINJOO@MINGW64 ~/Desktop/DES (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   HW/DES_HW_1.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        HW/DES_HW_2.py

no changes added to commit (use "git add" and/or "git commit -a")
```

4. \$ git status : 저장소 상태 체크

```
MINJOO@MINGW64 ~/Desktop/DES (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   HW/DES_HW_1.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        HW/DES_HW_2.py

no changes added to commit (use "git add" and/or "git commit -a")
```

5. \$ git add 원하는 파일/폴더 이름

```
MINJOO@MINGW64 ~/Desktop/DES (master)  
$ git add HW
```

6. \$ git commit -m “원하는 내용”

- Commit 하기
- 현재 상태를 스냅샷처럼 저장
- Commit 명령어를 이용하여 원하는 메모와 함께 현재 상태 저장

```
MINJOO@MINGW64 ~/Desktop/DES (master)
$ git commit -m "All of DES Source Code"
[master a35f00f] All of DES Source Code
2 files changed, 571 insertions(+), 1 deletion(-)
create mode 100644 HW/DES_HW_2.py
```

6. \$ git commit -m “원하는 내용”

- Commit 하기
- 현재 상태를 스냅샷처럼 저장
- Commit 명령어를 이용하여 원하는 메모와 함께 현재 상태 저장

```
MINJOO@MINGW64 ~/Desktop/DES (master)
$ git commit -m "All of DES Source Code"
[master a35f00f] All of DES Source Code
2 files changed 571 insertions(+), 1 deletion(-)
create mode 100644 HW/DES_HW_2.py
```

6. \$ git commit -m “원하는 내용”

- Commit 하기
- 현재 상태를 스냅샷처럼 저장
- Commit 명령어를 이용하여 원하는 메모와 함께 현재 상태 저장

```
MINJOO@MINGW64 ~/Desktop/DES (master)
$ git commit -m "All of DES Source Code"
[master a35f00f] All of DES Source Code
2 files changed, 571 insertions(+), 1 deletion(-)
create mode 100644 HW/DES_HW_2.py
```

7. 원격 저장소로 연결하기

```
MINJOO@MINGW64 ~/Desktop/DES (master)
$ git remote add origin https://github.com/MinjooSim/DES_tutorial.git
fatal: remote origin already exists.
```

원격 저장소 : 온라인으로 접근해야 하는 데스크탑 외부의 저장소

7. 원격 저장소로 연결하기

```
MINJOO@MINGW64 ~/Desktop/DES (master)  
$ git remote add origin https://github.com/MinjooSim/DES_tutorial.git  
fatal: remote origin already exists.
```

https://github.com/MinjooSim/DES_tutorial



[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)

[MinjooSim](#) / [DES_tutorial](#)

원격 저장소 : 온라인으로 접근해야 하는 데스크탑 외부의 저장소

7. 원격 저장소로 연결하기

연동 확인하기

```
MINJOO@MINGW64 ~/Desktop/DES (master)  
$ git remote -v  
origin  https://github.com/MinjooSim/Des_tutorial (fetch)  
origin  https://github.com/MinjooSim/Des_tutorial (push)
```

8. Push 하여 원격장소로 commit 내용 올리기

```
MTN100@MTNGW64 ~/Desktop/DES (master)
```

```
$ git push origin master
```

```
Enumerating objects: 8, done.
```

```
Counting objects: 100% (8/8), done.
```

```
Delta compression using up to 4 threads
```

```
Compressing objects: 100% (4/4), done.
```

```
Writing objects: 100% (5/5), 5.76 KiB | 1.15 MiB/s, done.
```

```
Total 5 (delta 1), reused 0 (delta 0)
```

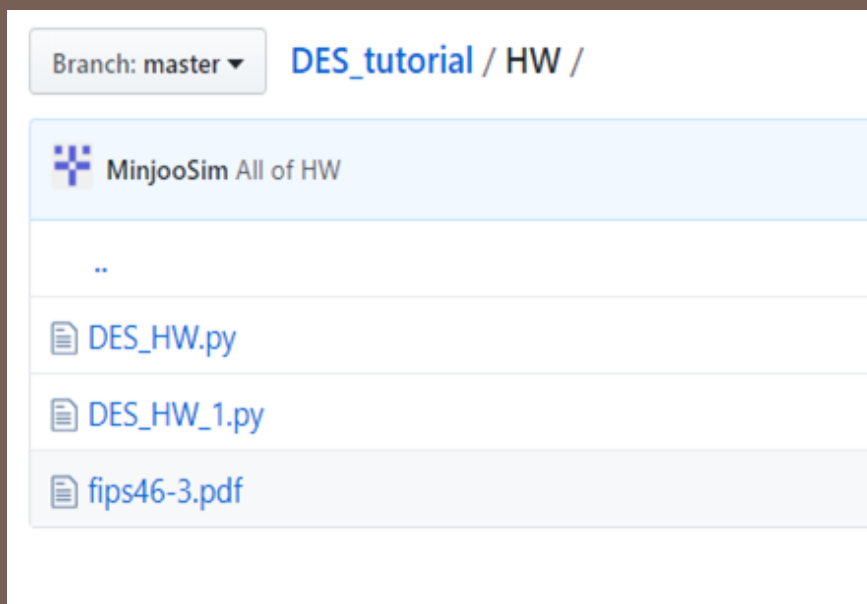
```
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
```

```
To https://github.com/MinjooSim/Des_tutorial
```

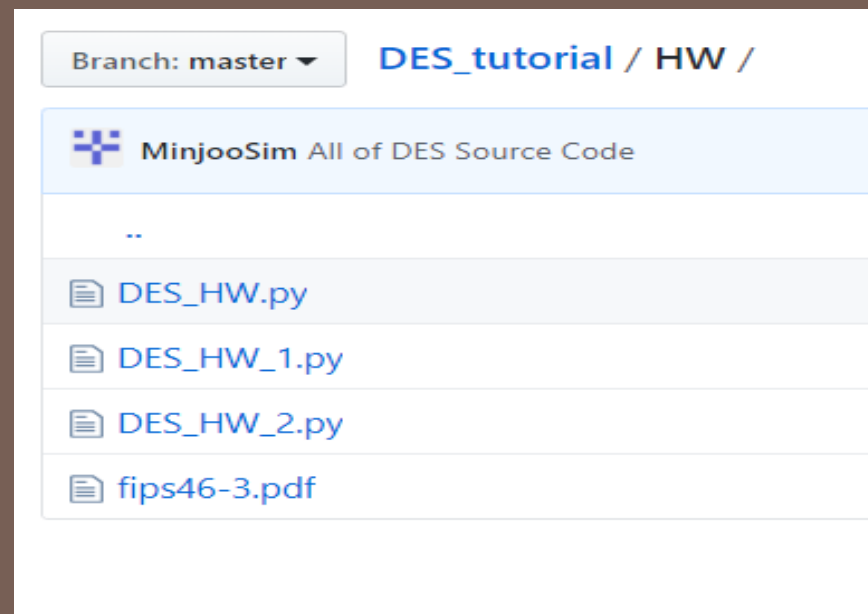
```
612afd0..a35f00f master -> master
```

9. git허브에서 확인하기

추가 전



추가 후



C언어 공부



○ 4장 ~ 9장 : ~ 1/9

4장 : 연산자

5장 : 선택문

6장 : 반복문

7장 : 함수

8장 : 배열

9장 : 포인터

● 10장 ~ 18장 : ~ 1/18

10장 : 배열과 포인터

11장 : 문자

12장 : 문자열

13장 : 변수의 영역과 데이터 공유

16장 : 메모리 동적 할당

17장 : 사용자 정의 자료형



10.1연습4	tutorial	25 days ago
10.2.연습2	tutorial	25 days ago
10.2.연습4	tutorial	25 days ago
10.2.연습5	tutorial	25 days ago
10.도전1	tutorial	25 days ago
10.도전2	tutorial	25 days ago
10.도전3	tutorial	25 days ago
11.1.연습5	tutorial	25 days ago
11.2.연습3	tutorial	25 days ago
11.2.연습4	tutorial	25 days ago
11.2.연습5	tutorial	25 days ago
11.도전1.1	tutorial	25 days ago
11.도전1	tutorial	25 days ago
11.도전2	tutorial	25 days ago
11.도전3	tutorial	25 days ago
12.도전1.1	tutorial	25 days ago
12.도전1	tutorial	25 days ago



MinjooSim tutorial

Latest commit 5cc020a 25 days ago

..

📁 .vs/10.도전3/v15	tutorial	25 days ago
📁 Debug	tutorial	25 days ago
📄 10.3.c	tutorial	25 days ago
📄 10.도전3.sln	tutorial	25 days ago
📄 10.도전3.vcxproj	tutorial	25 days ago
📄 10.도전3.vcxproj.filters	tutorial	25 days ago

```

1  #include<stdio.h>
2
3  void input_nums(int *lotto_nums)//배열에 로또 번호를 입력하는 함수
4  {
5      int i, j;
6      int str, n;
7      printf("로또 번호를 입력하세요 : ");
8      scanf("%d", &lotto_nums[0]);
9
10     i = 1;
11     while (i != 6)
12     {
13         printf("로또 번호를 입력하세요 : ");
14         scanf("%d", &n);
15
16         str = 1;
17         for (j = 0; j < 6; j++)
18         {
19             if (n == lotto_nums[j])
20             {
21                 printf("같은 번호가 있습니다.\n");
22                 str = 0;
23                 break;
24             }
25         }
26         if (str == 1)
27         {
28             lotto_nums[i] = n;
29             i++;
30         }
31     }
32 }
33
34 void print_nums(int *lotto_nums)//배열에 저장된 값을 출력하는 함수
35 {

```

```

34
35 void print_nums(int *lotto_nums)//배열에 저장된 값을 출력하는 함수
36 {
37     int i, j;
38     int num;
39
40     for (i = 0; i < 6; i++)
41     {
42         for (j = i + 1; j < 6; j++)
43         {
44             if (lotto_nums[i] > lotto_nums[j])
45             {
46                 num = lotto_nums[j];
47                 lotto_nums[j] = lotto_nums[i];
48                 lotto_nums[i] = num;
49             }//오름차순 배열
50         }
51     }
52     printf("입력된 로또번호 : ");
53     for (i = 0; i < 6; i++)
54         printf("%3d\t", lotto_nums[i]);
55     printf("\n");
56 }
57 int main(void)
58 {
59     int lotto_nums[6] = {0}; //로또 번호를 저장할 배열
60
61     input_nums(lotto_nums);// 입력함수 호출
62     print_nums(lotto_nums);//출력함수 호출
63     return 0;
64 }

```



```

1  #include<stdio.h>
2
3  struct money_box
4  {
5      int w500;
6      int w100;
7      int w50;
8      int w10;
9  };
10
11 typedef struct money_box MoneyBox;
12
13 void init(MoneyBox *pmoney)//MoneyBox 변수 초기화 //매개변수는 구조체 포인터
14 {
15     pmoney->w10 = 0;//pmoney = 구조체 포인터//->구조체 변수 간접참조() 대신 사용
16     pmoney->w50 = 0;
17     pmoney->w100 = 0;
18     pmoney->w500 = 0;
19 }
20
21 void save(MoneyBox *pmoney, int unit, int cnt)//unit 동전을 cnt개 저금
22 {
23     switch(unit)
24     {
25     case 500:
26     {
27         pmoney->w500 = pmoney->w500 + cnt;
28         break;
29     }
30     case 100:
31     {
32         pmoney->w100 = pmoney->w100 + cnt;
33         break;
34     }
35     case 50:
36     {
37         pmoney->w50 = pmoney->w50 + cnt;
38         break;
39     }
40     case 10:
41     {
42         pmoney->w10 = pmoney->w10 + cnt;

```

```

43         break;
44     }
45
46     }
47
48 }
49
50 int total(MoneyBox *pmoney)//저금통의 총 저축액 반환
51 {
52     int tot =0;
53     tot = pmoney->w500 * 500 + pmoney->w100 * 100 + pmoney->w50 * 50 + pmoney->w10 * 10;
54     return tot;
55 }
56
57 int main(void)
58 {
59     MoneyBox money;
60     MoneyBox *pmoney = &money;
61
62     int money_total = 0;
63     int count = 0;
64     int coin = 0;
65
66     init(pmone);
67
68     while (coin != -1)
69     {
70         printf("동전의 금액과 개수 : ");
71         scanf("%d", &coin);
72         if (coin == -1)
73         {
74             break;
75         }
76         scanf("%d", &count);
77
78         save(pmone, coin, count);
79     }
80
81     money_total = total(pmone);
82
83     printf("총 저금액 : %d원\n", money_total);
84
85     return 0;
86
87 }

```

DES구현

0. DES(Data Encryption Standard) 정의

미국 표준의 56bit 암호 키를 사용하는
대칭키 블록 암호화 알고리즘

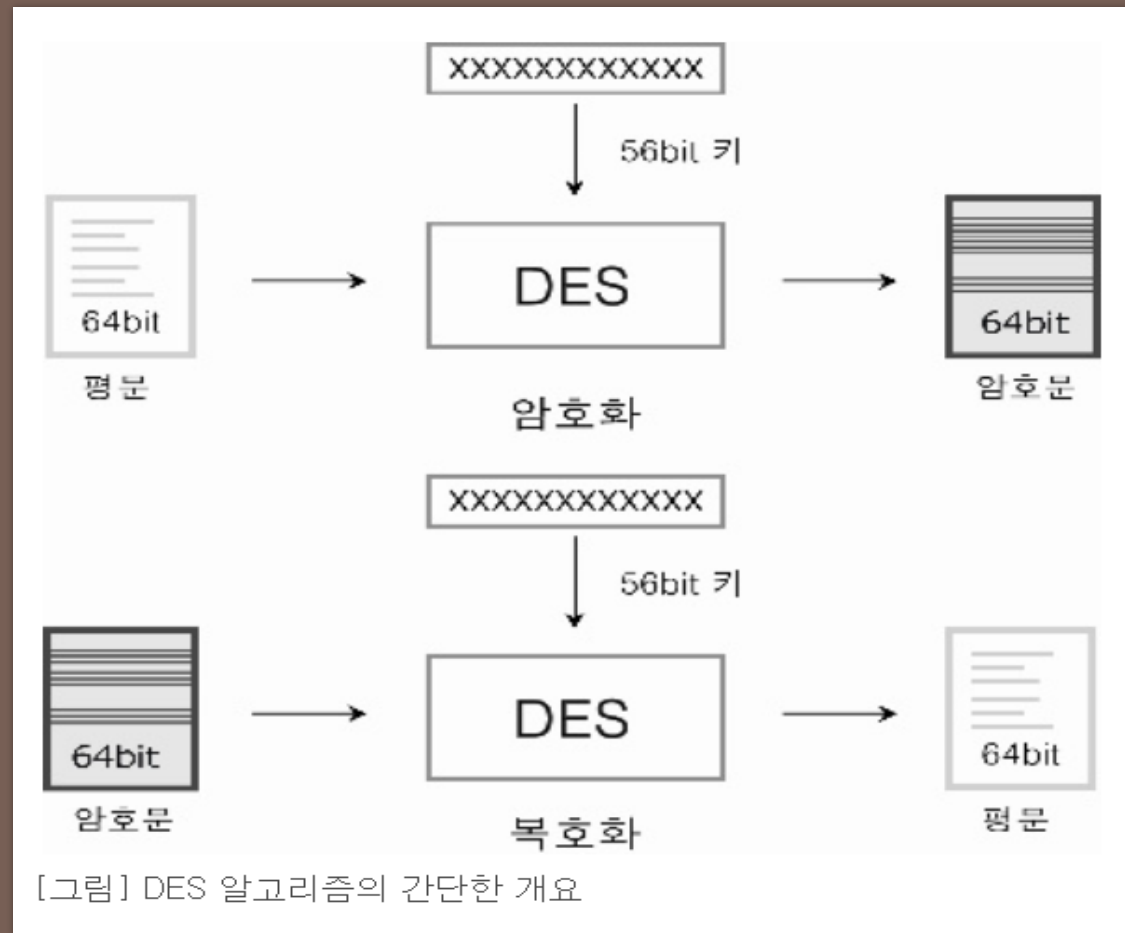
**30년간 전세계적으로 널리 통용되어온 실질적인 블록 암호화 방식

0. DES의 한계점

- 컴퓨터의 처리 속도가 급속도로 발전하면서 암호 키와 암호화 루틴을 해독하는 것이 쉬워짐.
- 보안을 위해 Triple DES(3DES)개발
 - DES의 암호 키 2개 & 암호화 작업 3중으로 처리

**현재는 AES(Advanced Encryption Standard) 가 새로운 표준
(2000~)

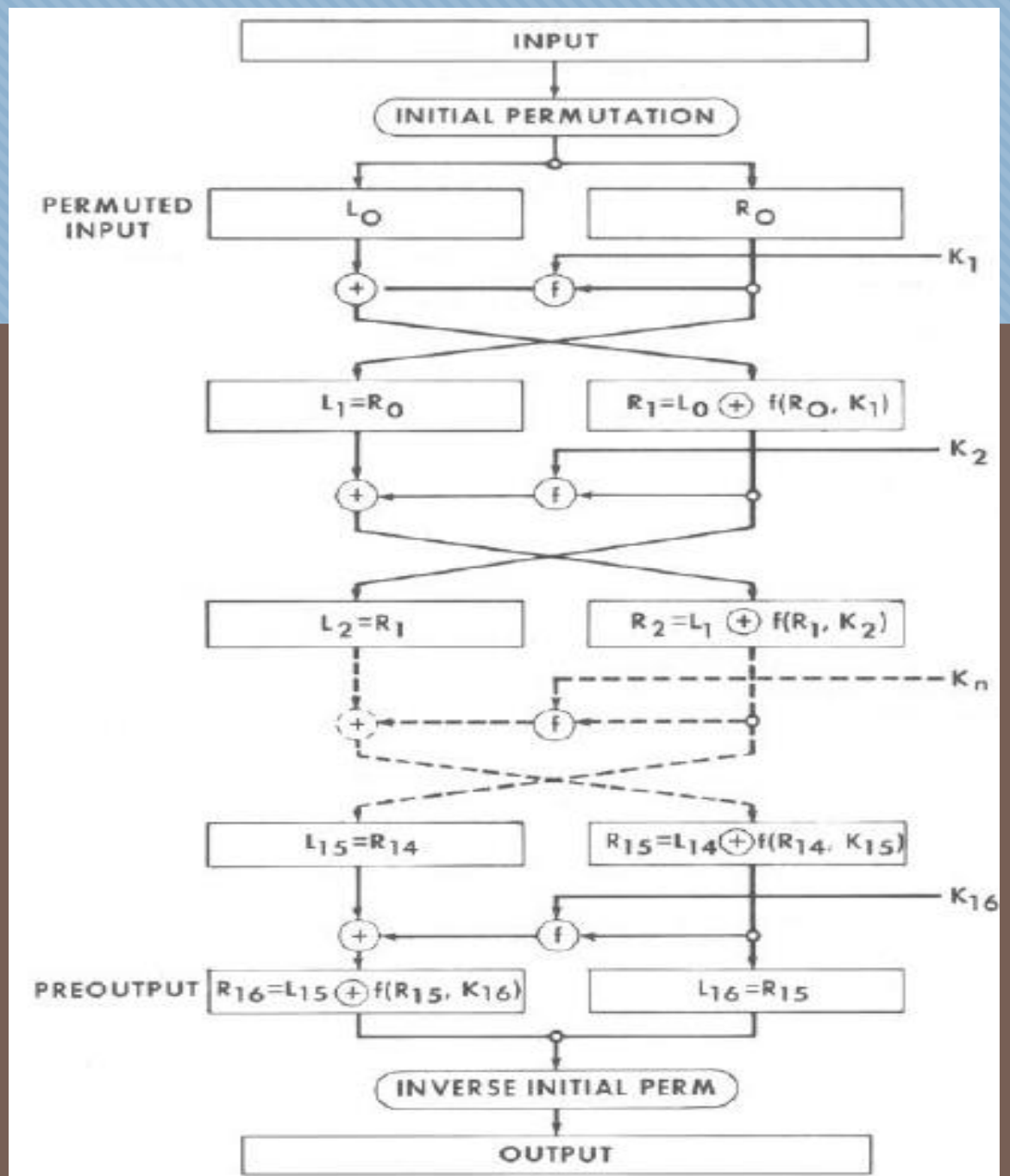
1. DES 암호화 과정



1. DES 암호화 과정

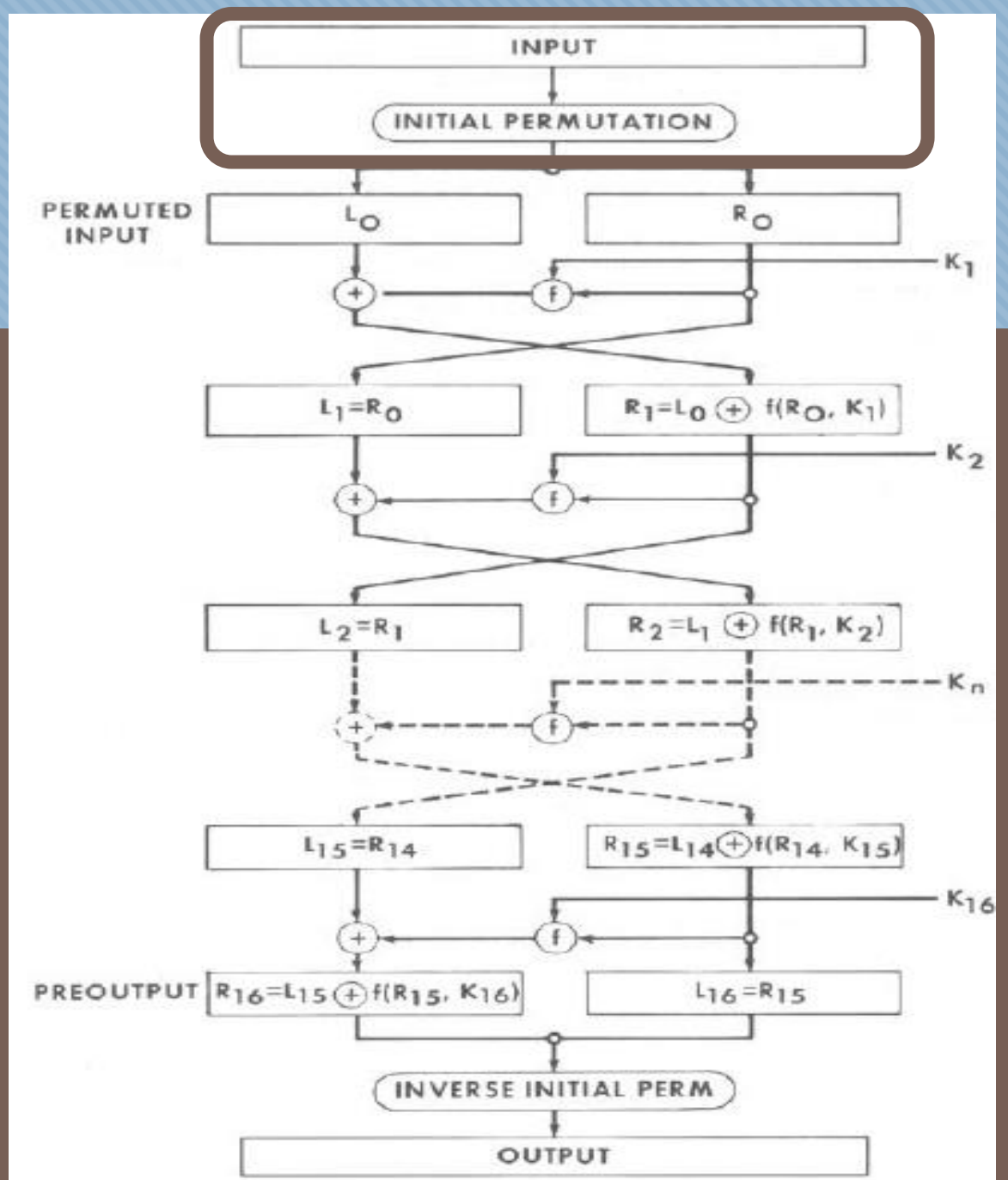
○Feistel 암호 구조

- n-bit의 평문을 입력으로 했을 때,
n-bit의 암호문이 나오게 되고, 각각의
평문 블록은 유일한 암호문 블록을 생성



1. DES 암호화 과정

1) 64bit 평문
초기 치환(IP)단계 통과

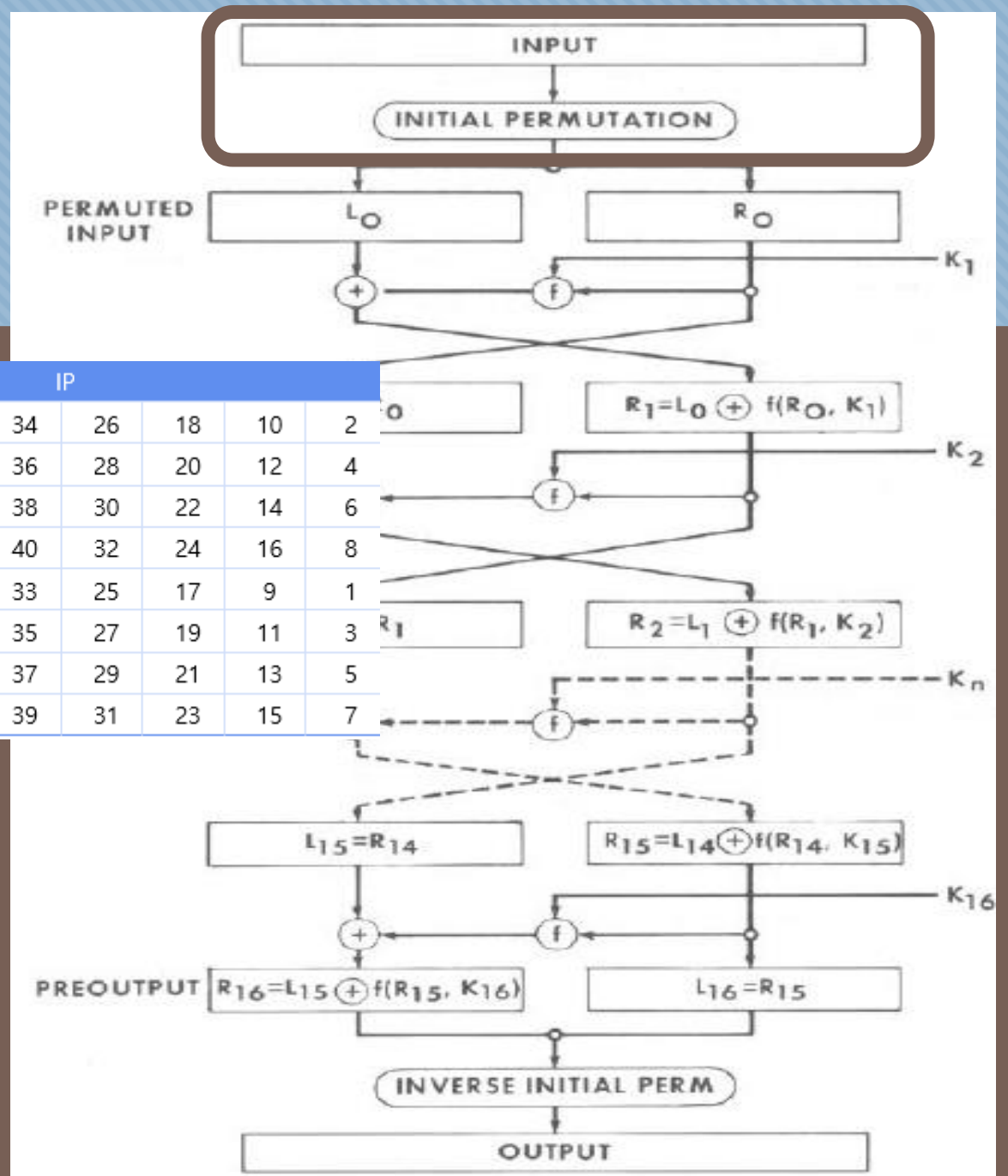


1. DES 암호화 과정

1->58 치환

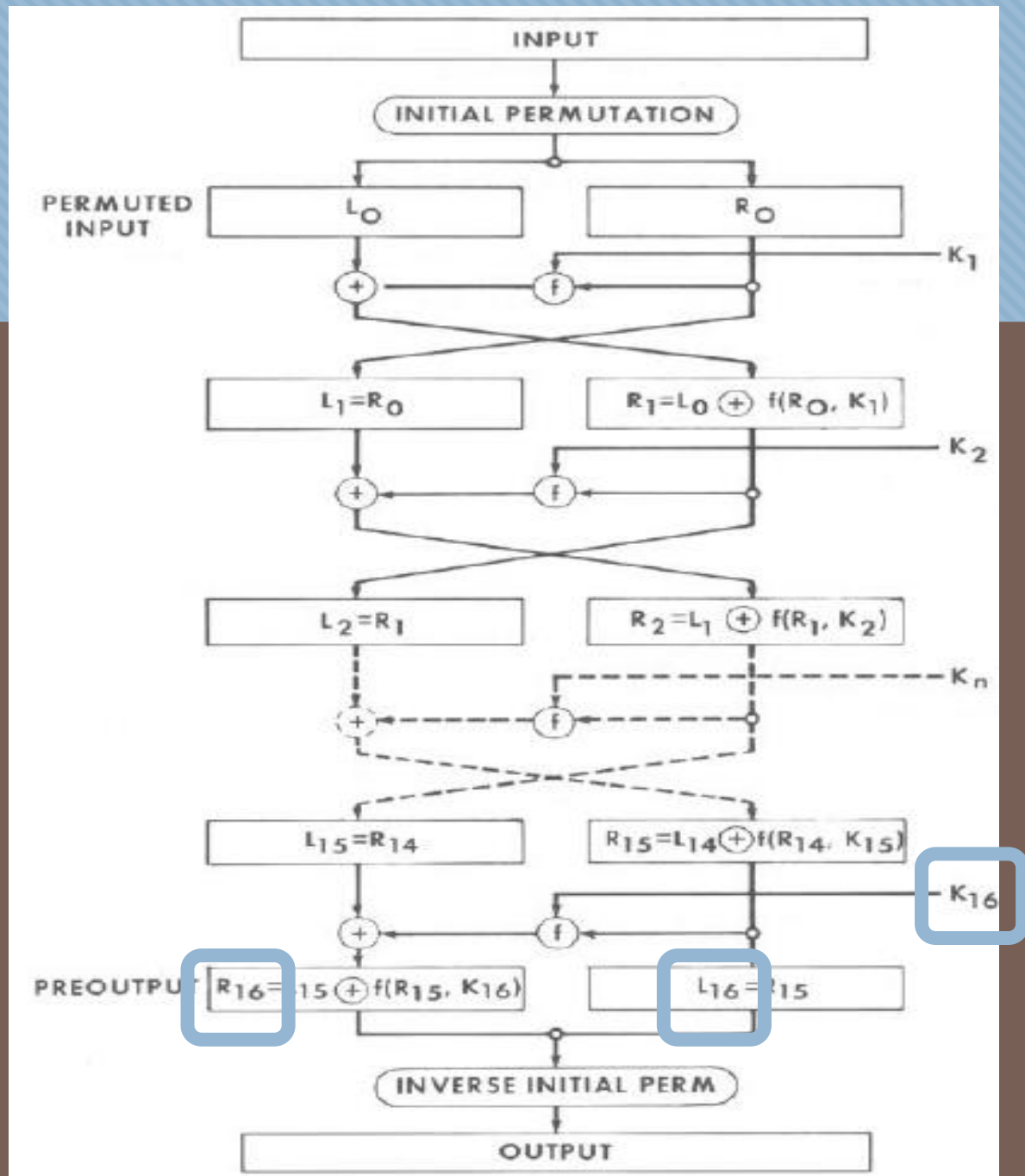
IP								
58	50	42	34	26	18	10	2	0
60	52	44	36	28	20	12	4	
62	54	46	38	30	22	14	6	
64	56	48	40	32	24	16	8	
57	49	41	33	25	17	9	1	
59	51	43	35	27	19	11	3	
61	53	45	37	29	21	13	5	
63	55	47	39	31	23	15	7	

1) 64bit 평문
초기 치환(IP)단계 통과



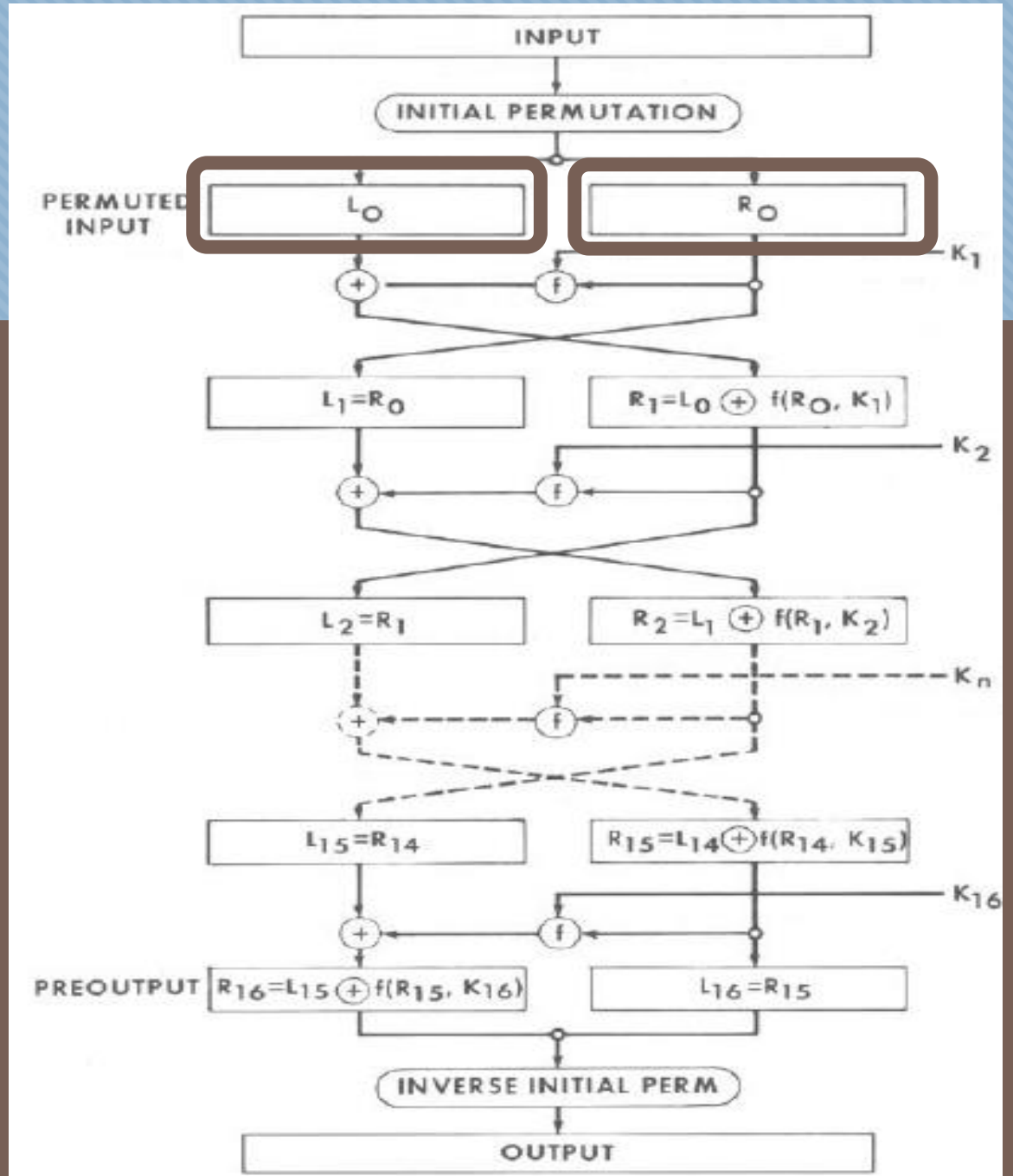
1. DES 암호화 과정

2) 초기 치환을 거친
평문을 총 16번의
라운드를 거쳐 암호화



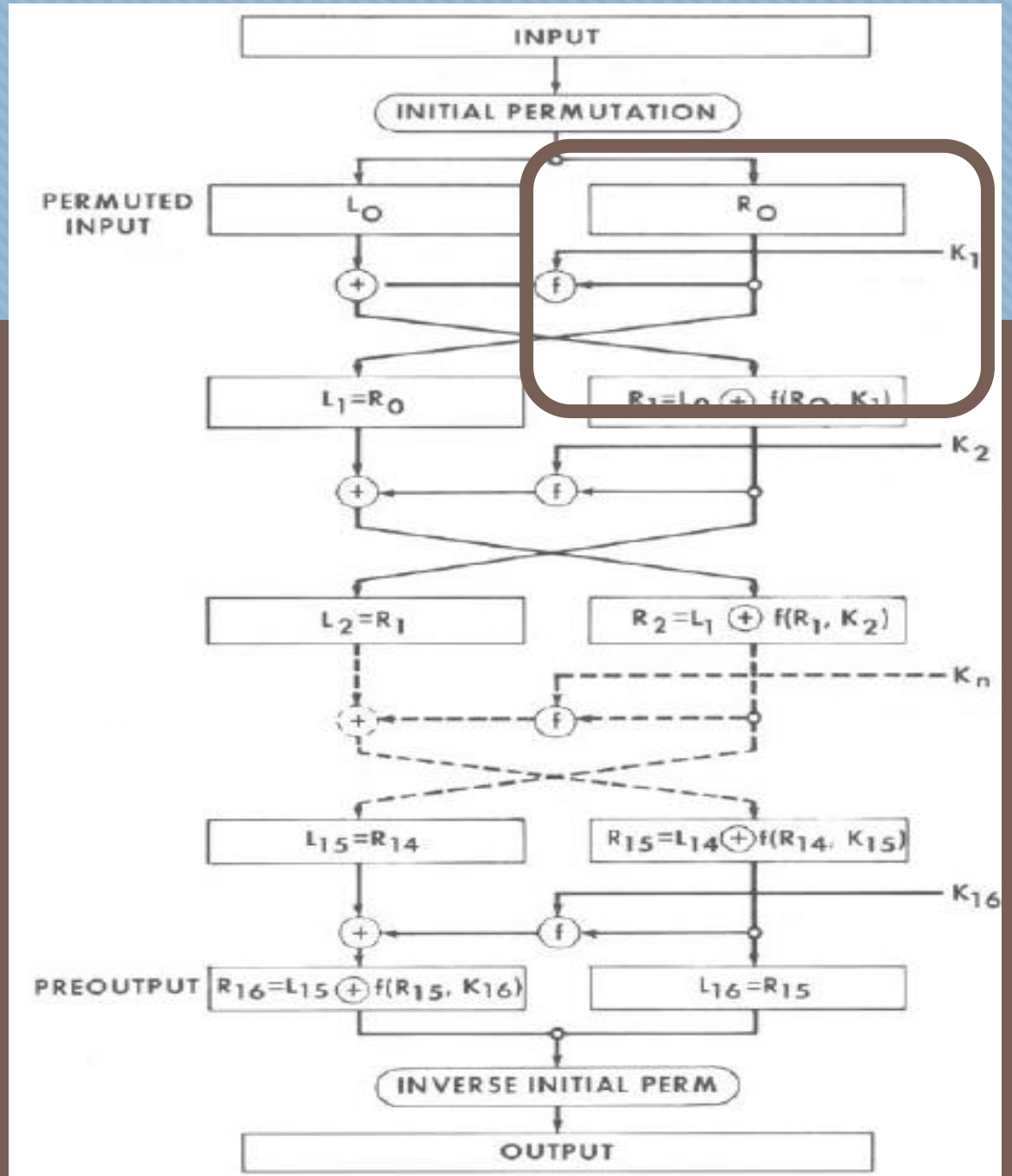
1. DES 암호화 과정

3) 각 라운드는
32bit씩 (L, R) 나누어
들어감



1. DES 암호화 과정

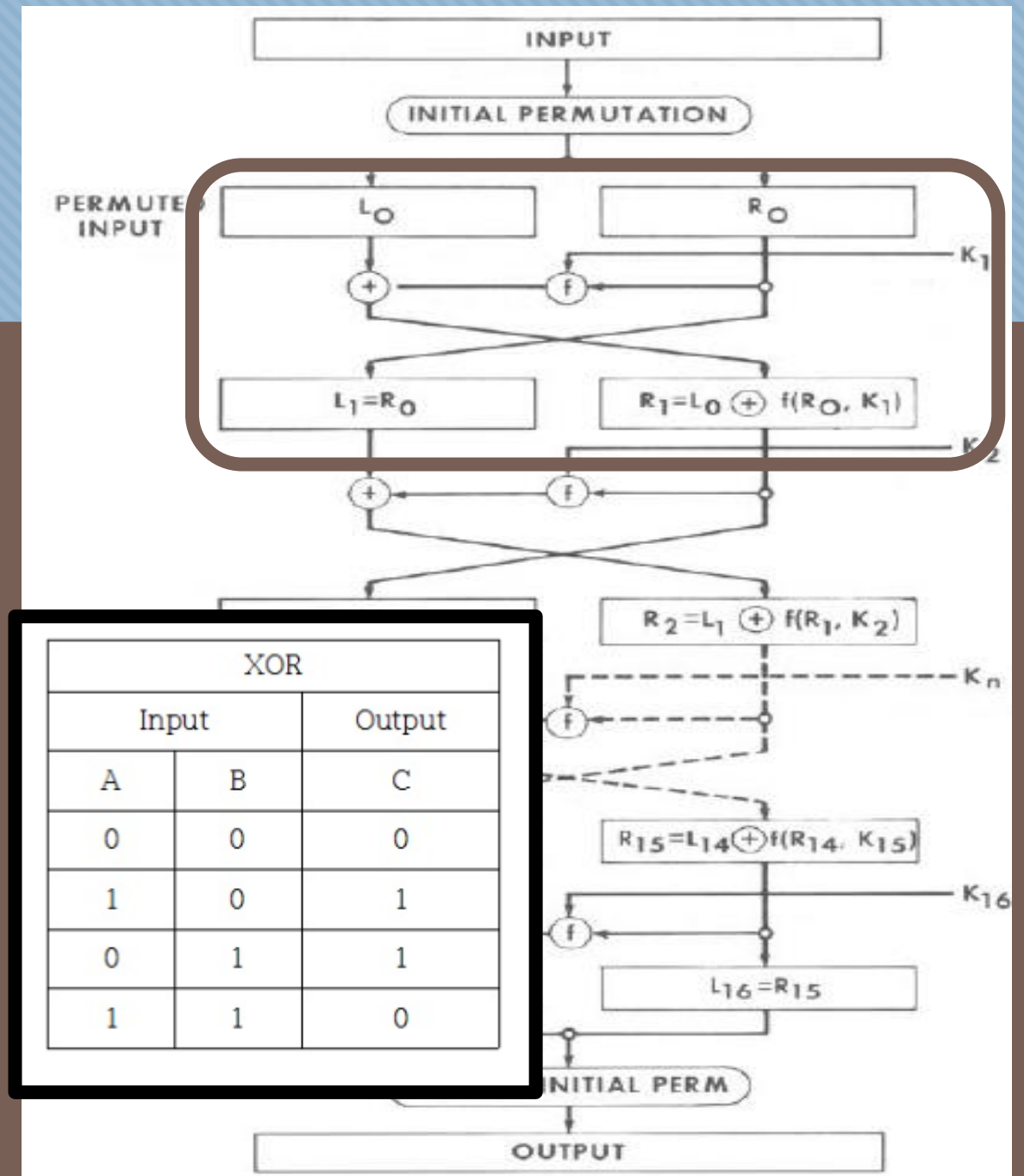
4) 오른쪽 32bit는
키 스케줄에 의해
만들어진 48bit의 키와
함께 f함수에 들어감



1. DES 암호화 과정

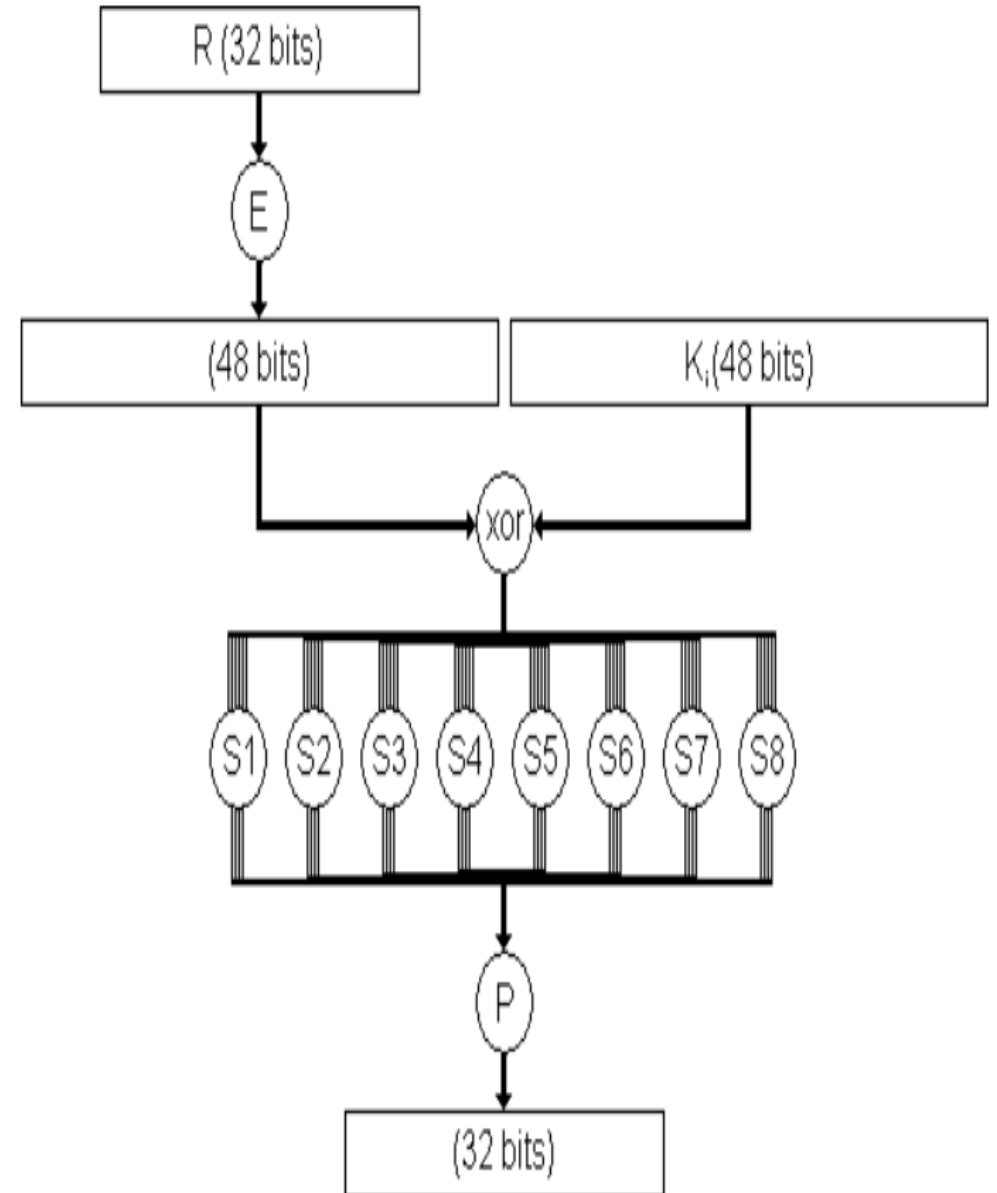
5) $L1 = R0$

$R1 = L0 \text{ XOR } F(R0, K1)$



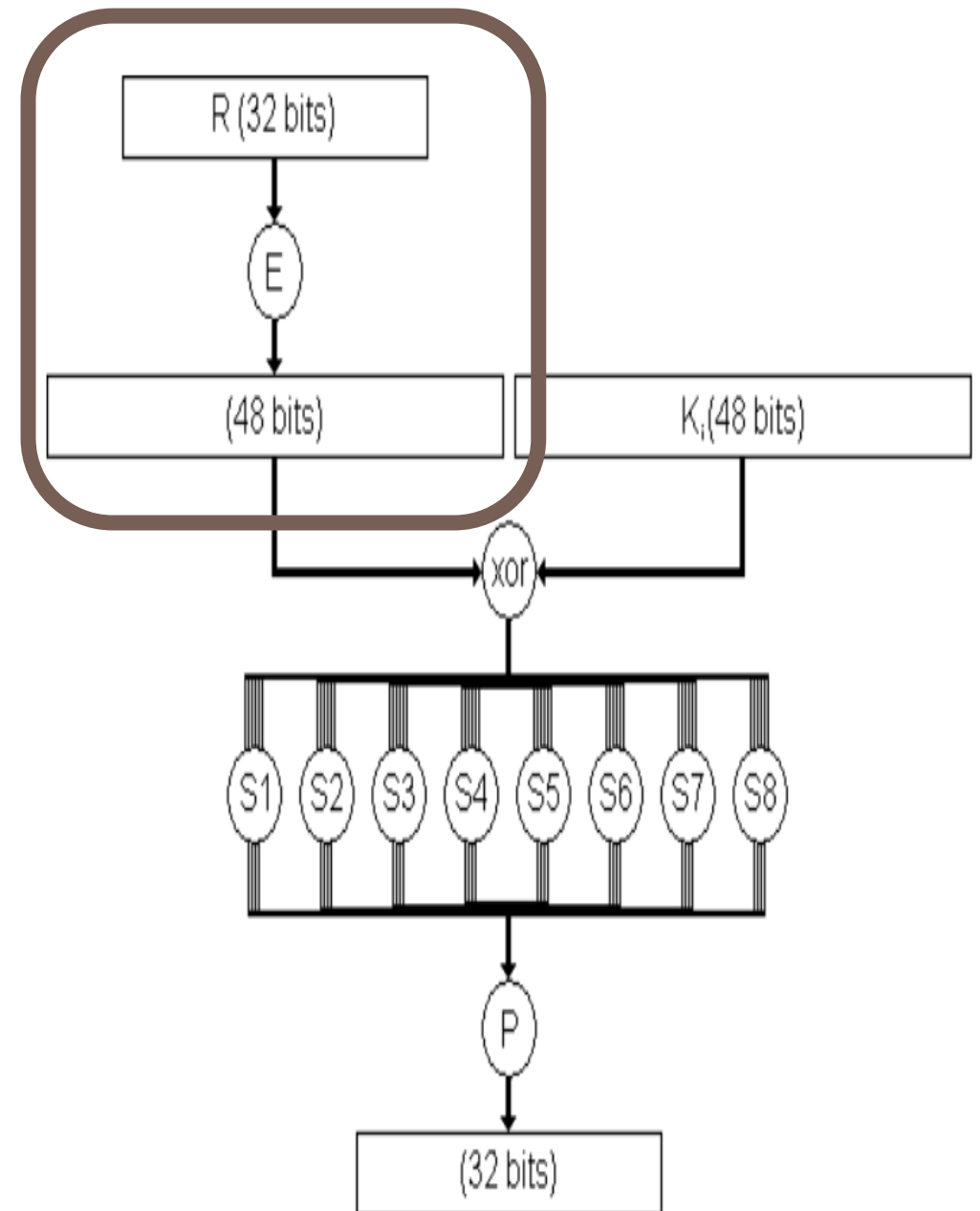
1. DES 암호화 과정

○ F 함수



1. DES 암호화 과정

5) 32bit의 평문을
48bit로 확장



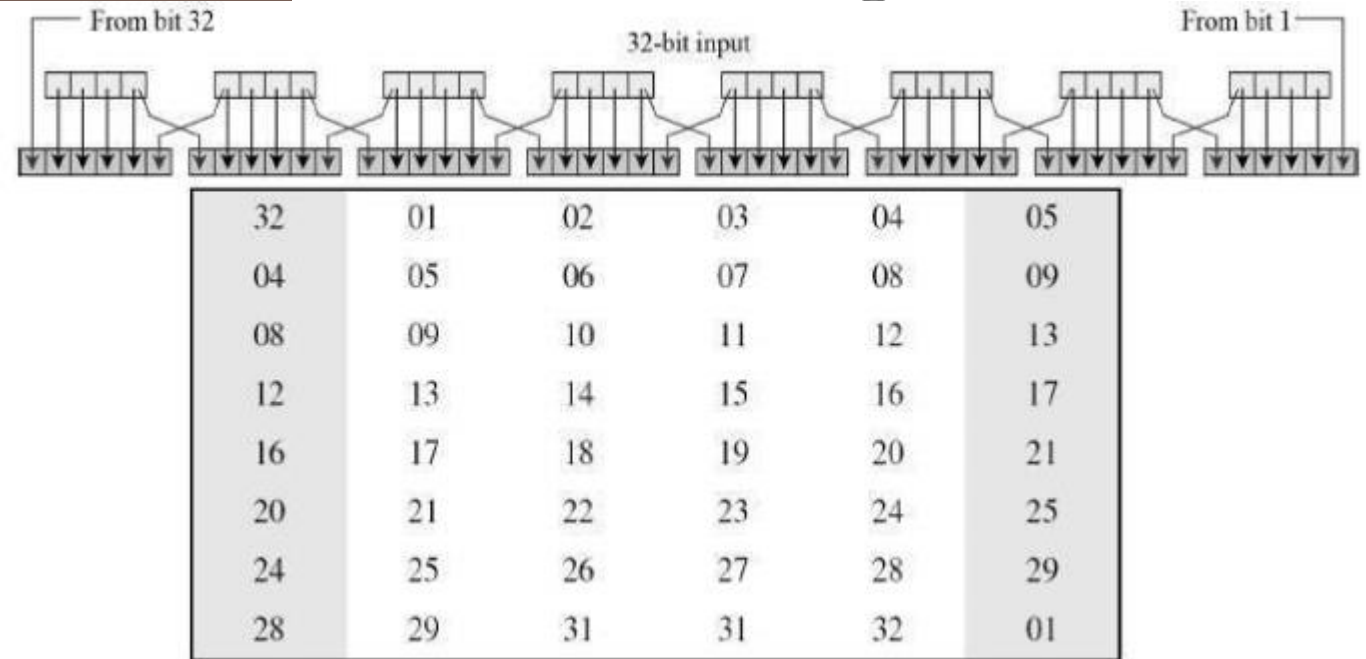
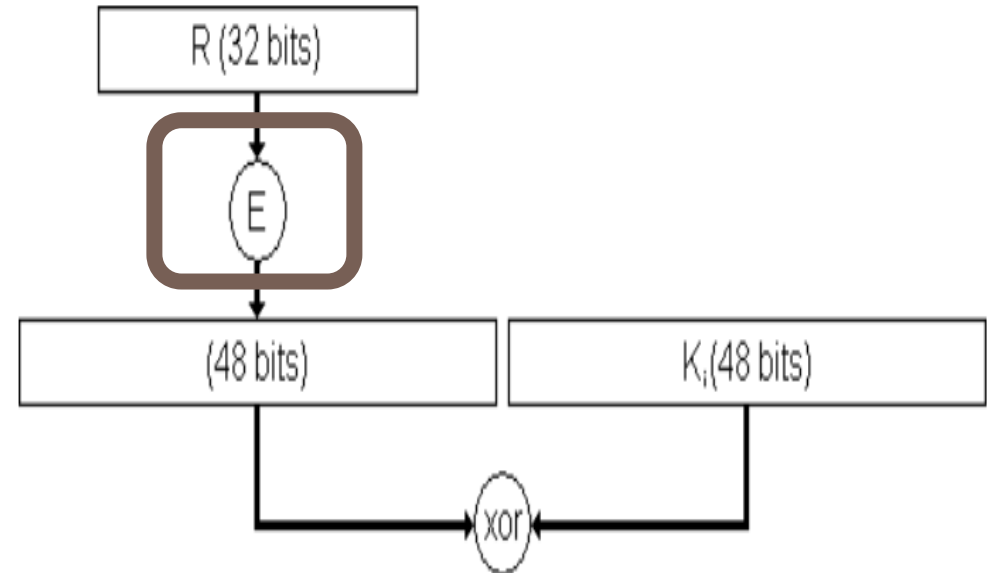
1. DES 암호화 과정

**E구조

-32bit->4bit 씩 8세트

-앞뒤로 한 비트씩

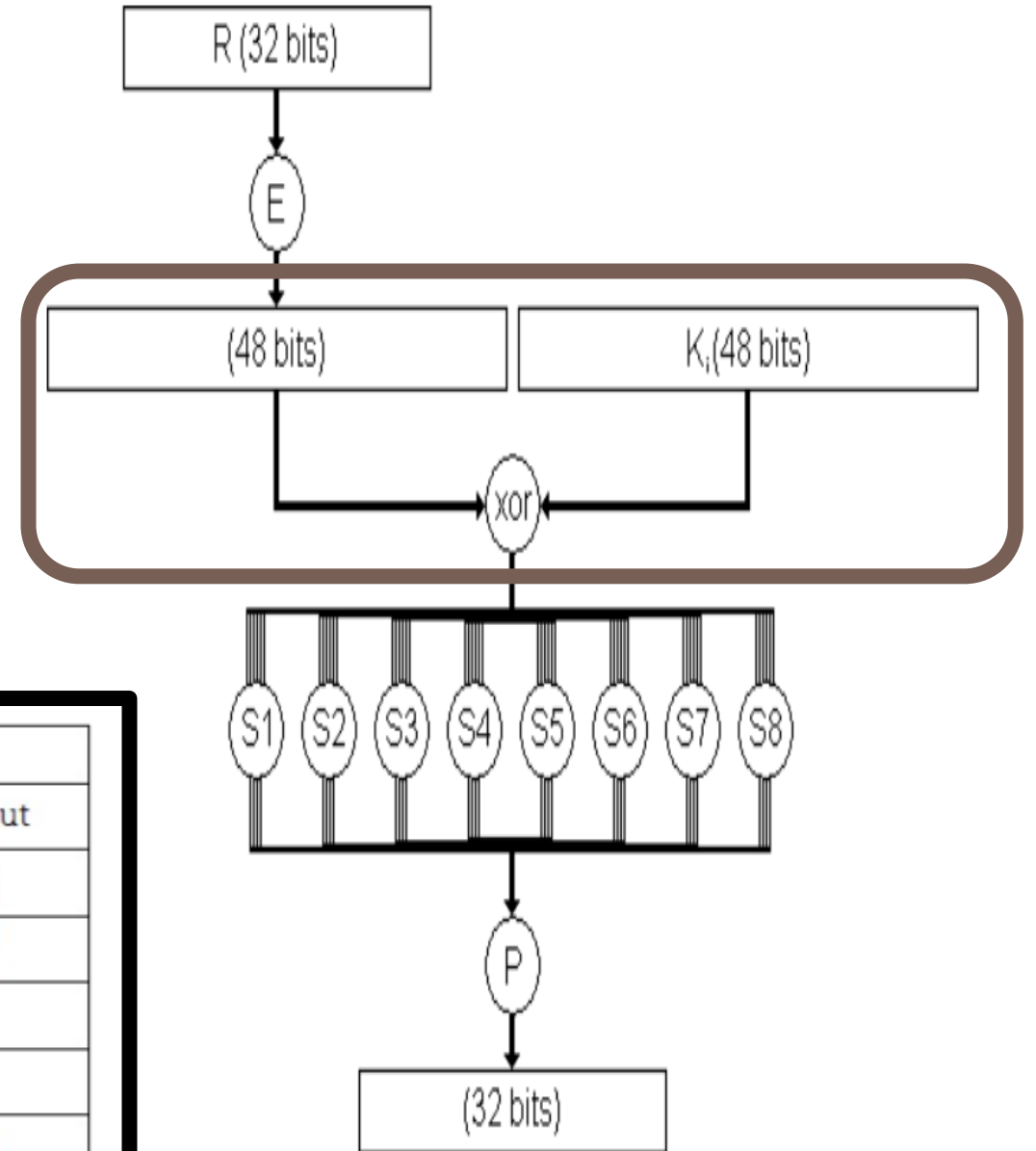
붙여준다



1. DES 암호화 과정

6) 확장된 문장과
암호키를 XOR

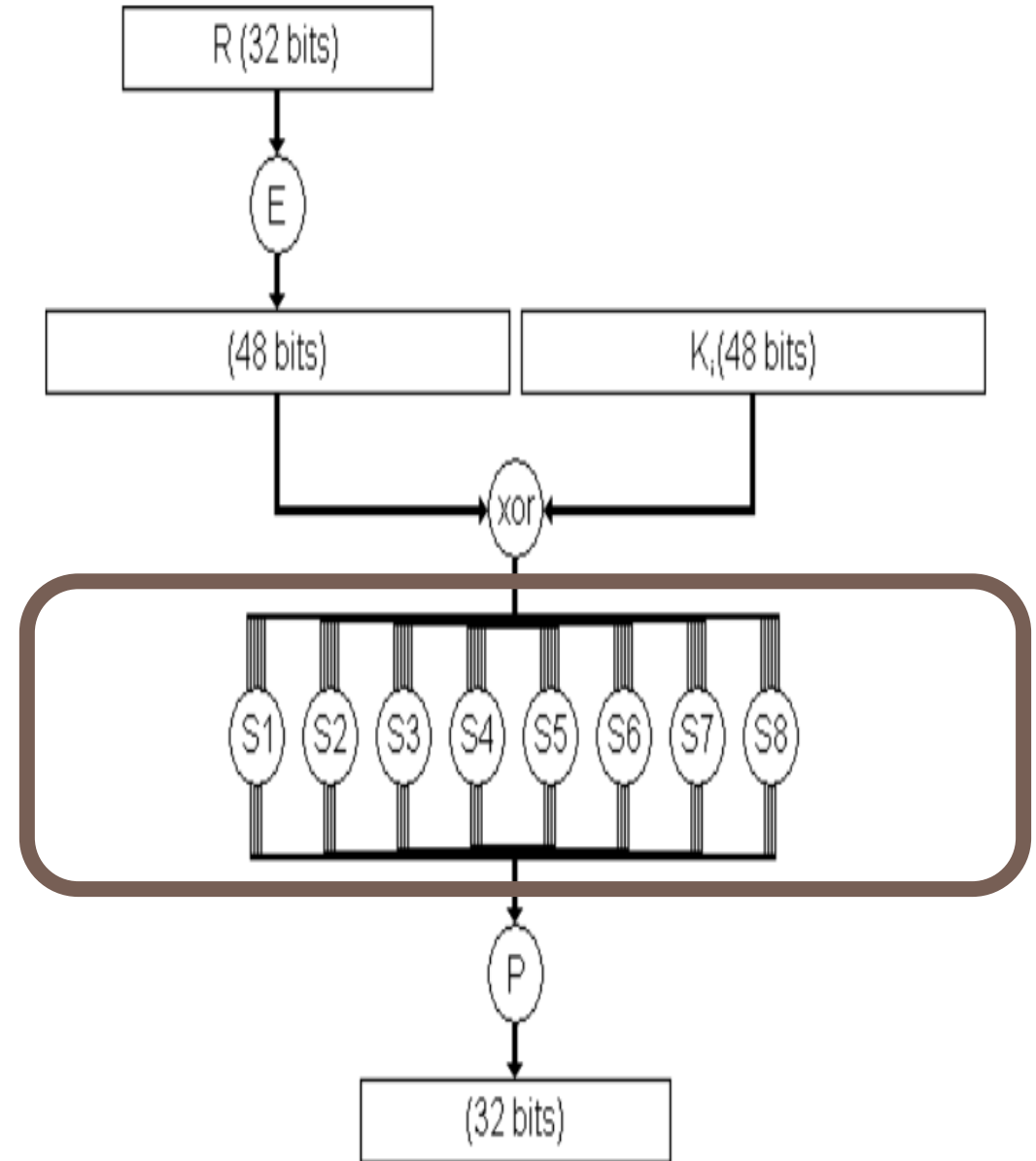
XOR		
Input		Output
A	B	C
0	0	0
1	0	1
0	1	1
1	1	0



1. DES 암호화 과정

7) 6bit씩 8개로 나눈 후,
s-box를 사용하여 암호화
진행

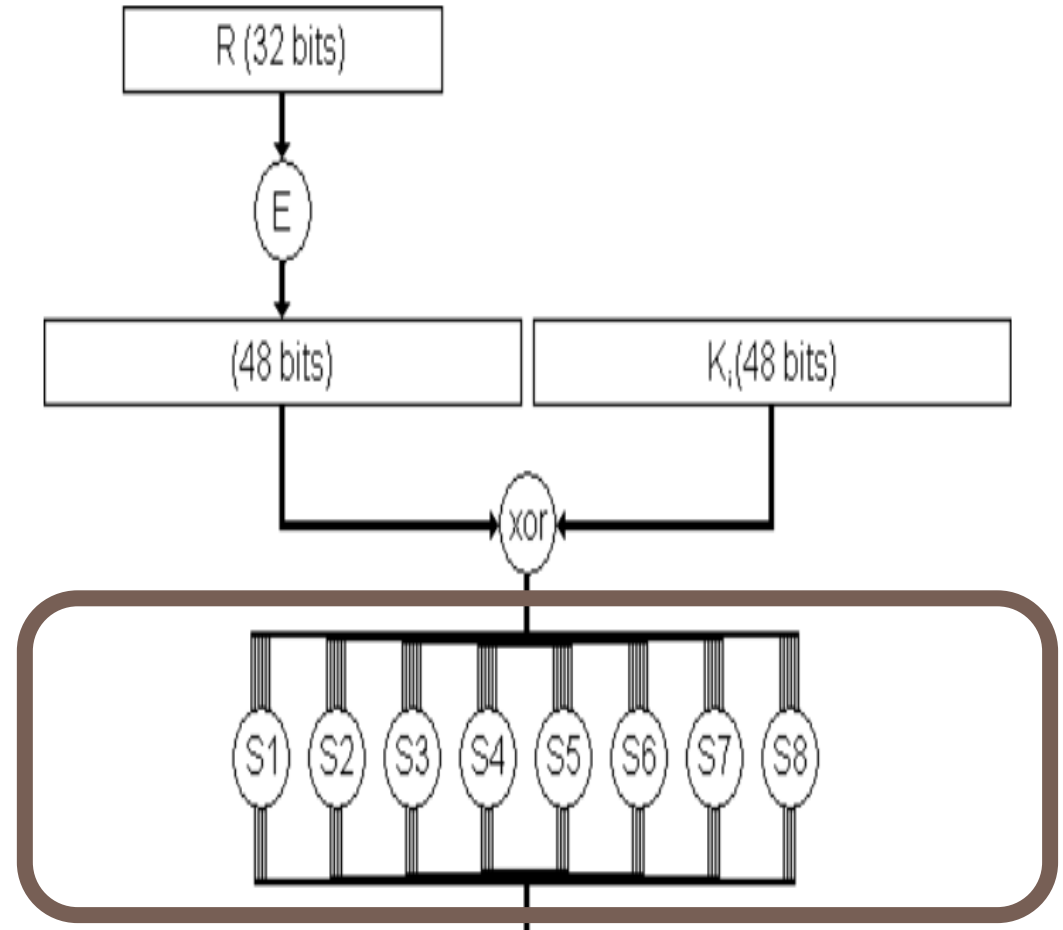
-4bit 8개의 결과가 나옴



1. DES 암호화 과정

7) 6bit씩 8개로 나눈 후,
s-box를 사용하여 암호화
진행

-4bit 8개의 결과가 나옴

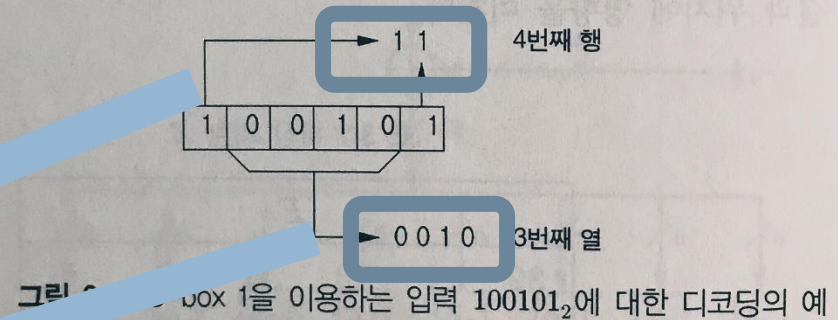


S1																
행 \ 열	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	3	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	13	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

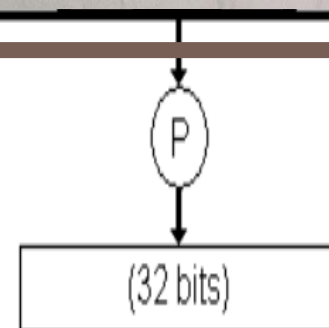
1. DES 암호화 과정

**s-box

예제 3.2 S-Box 입력 $b = (100101)_2$ 는 행 $11_2 = 3$ (즉, 숫자가 00_2 로 시작하므로 4번째 행)과 열 $0010_2 = 2$ (즉, 3번째 열)을 의미한다. 입력 b 가 S-box 1에 입력되면 결과는 $S_1(37 = 100101_2) = 8 = 1000_2$ 이다.



행 \ 열	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	10	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	14	2	13	1	10	6	12	11	9	5	3	8	
2	4	14	8	13	6	2	11	15	12	9	7	13	10	5	0	
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13



1. DES 암호화 과정

**s-box

예제 3.2 S-Box 입력 $b = (100101)_2$ 는 행 $11_2 = 3$ (즉, 숫자가 00₂로 시작하므로 4번째 행)과 열 $0010_2 = 2$ (즉, 3번째 열)을 의미한다. 입력 b 가 S-box 1에 입력되면 결과는 $S_1(37 = 100101_2) = 8 = 1000_2$ 이다.

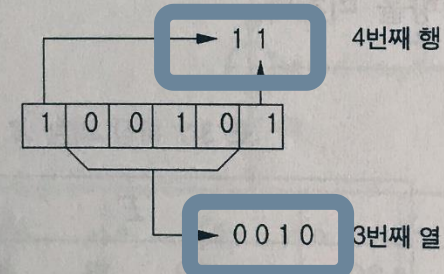
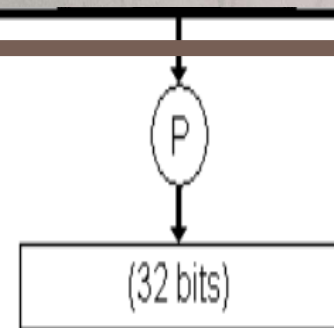


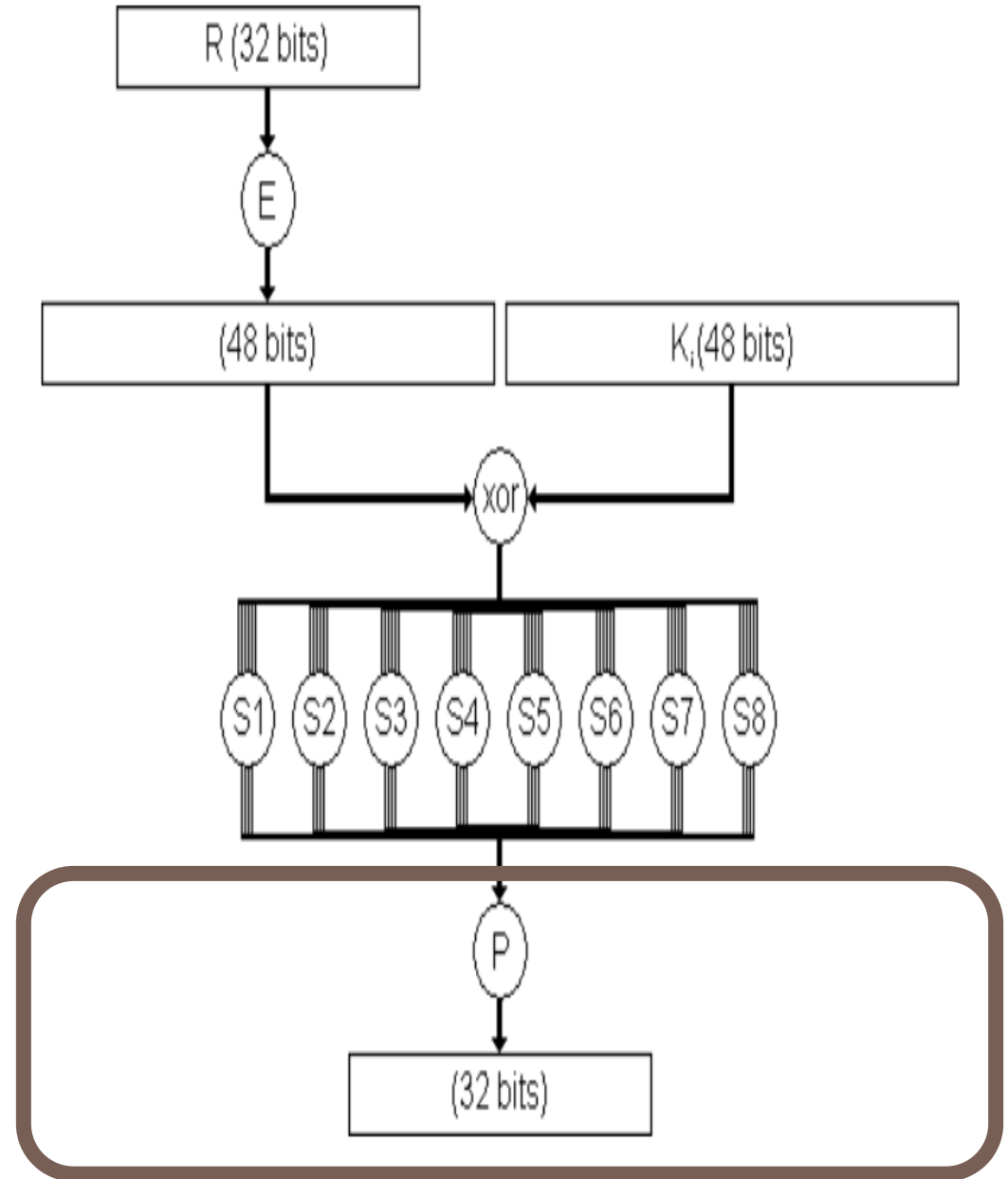
그림 3.12 S-box 1을 이용하는 입력 100101₂에 대한 디코딩의 예

행 \ 열	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	3	14	12	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	13	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13



1. DES 암호화 과정

8) S-box를 거친 결과 값을 치환한 후 반환



1. DES 암호화 과정

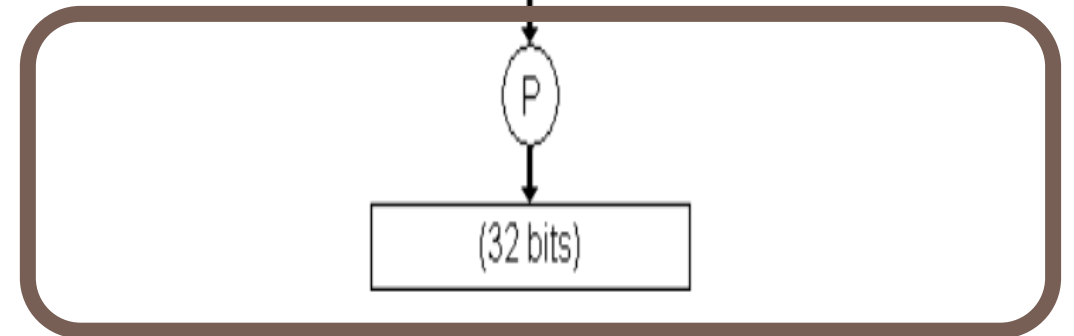
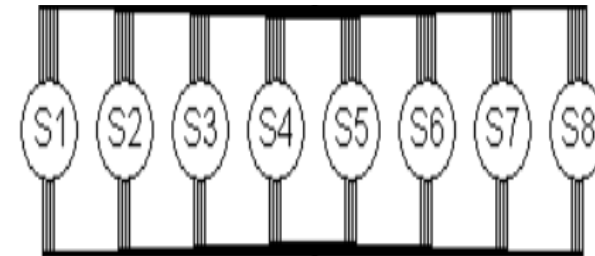
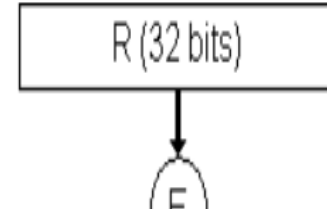
1->16으로 치환

- DES P(Permutation, 비트 치환) 구조

32 비트 길이의 데이터의 순서를 섞는다.

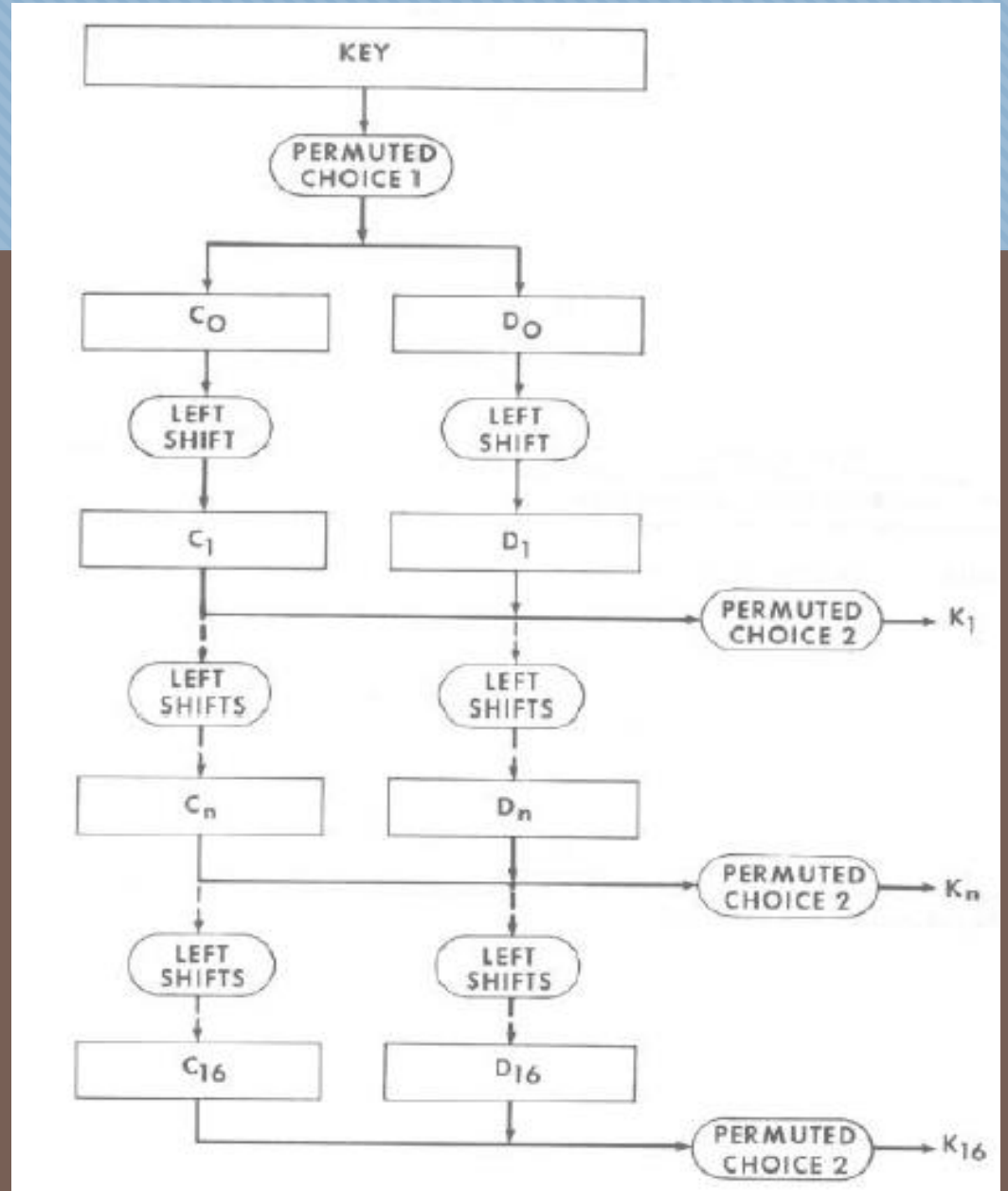
P							
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

8) S-box를 거친 결과 값을 치환한 후 반환



1. DES 암호화 과정

○ 키 스케줄

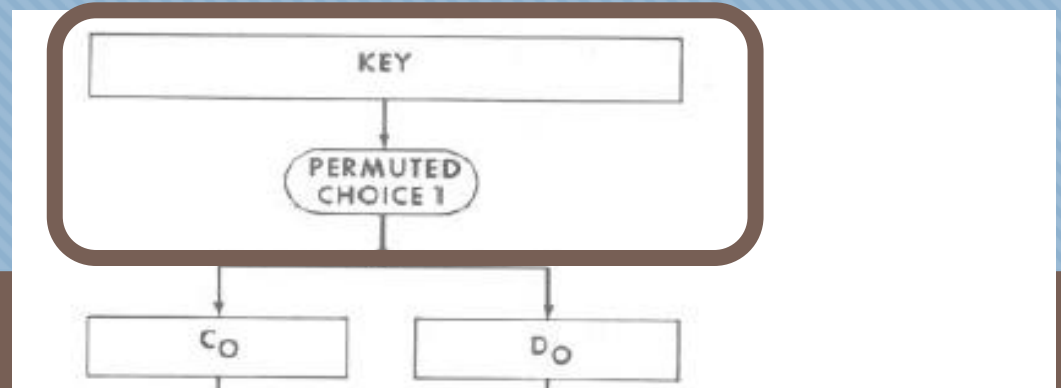


1. DES 암호화 과정

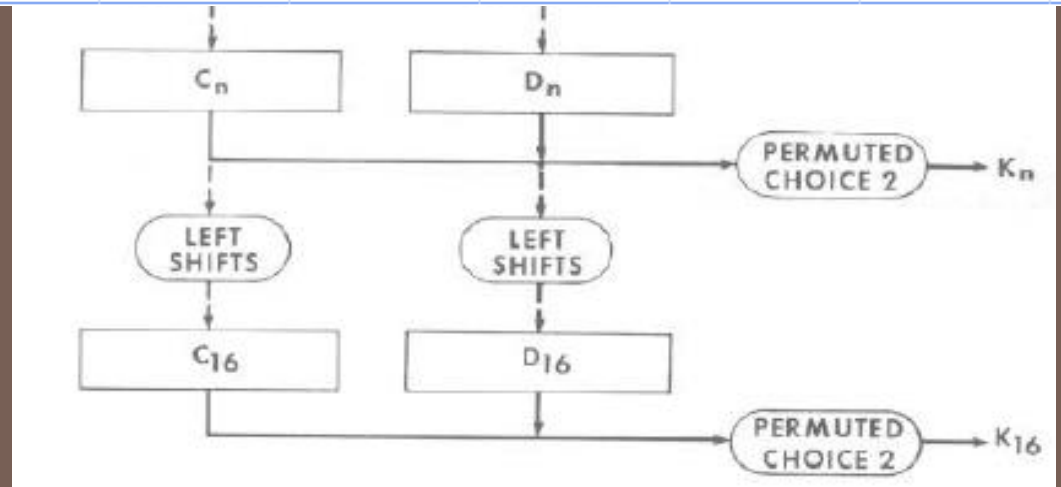
9) key(64bit)

PC1치환

8bit 패리티 체크
비트 제거(56bit)



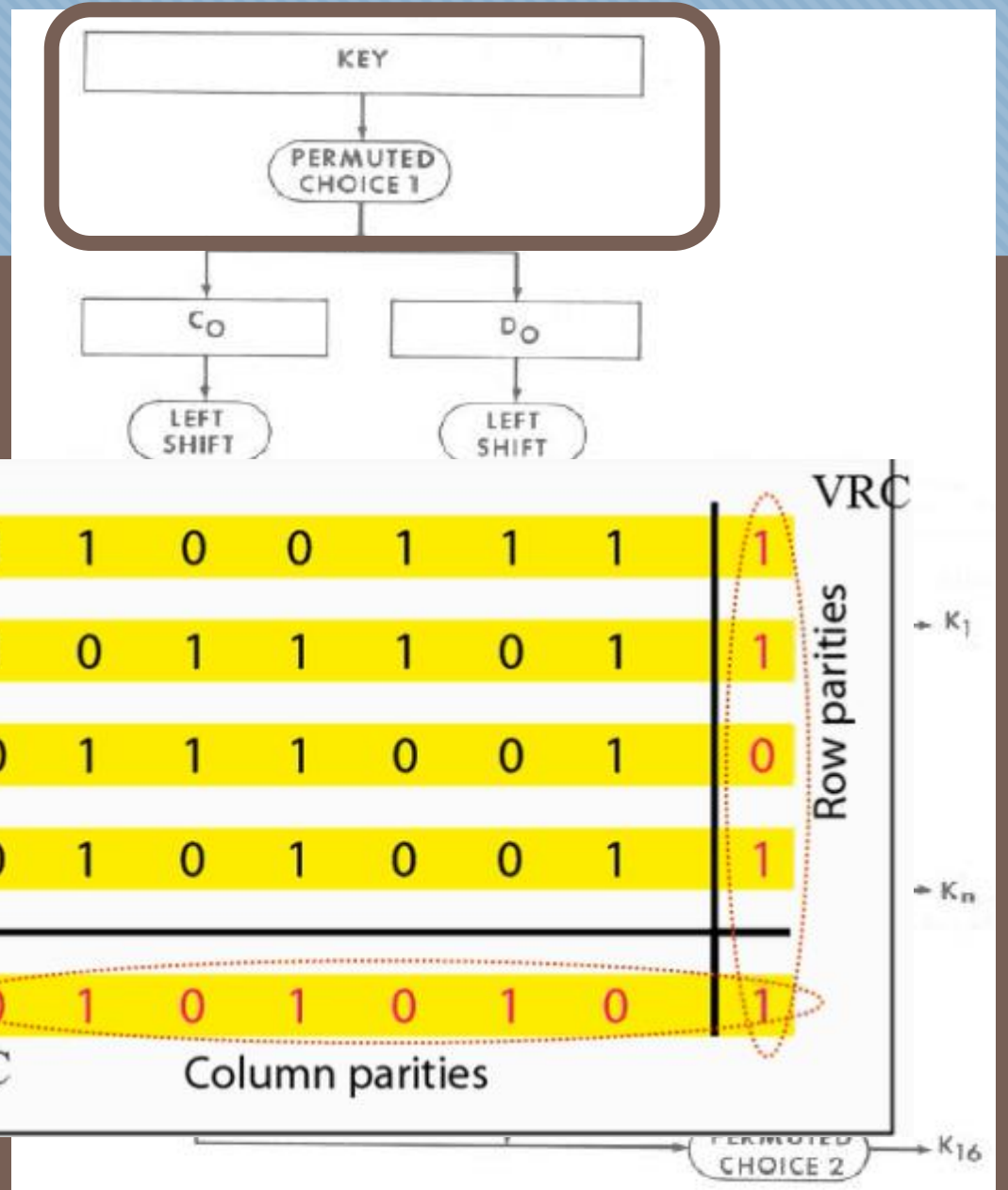
PC1							
57	49	41	33	25	17	9	1
58	50	42	34	26	18	10	2
59	51	43	35	27	19	11	3
60	52	44	36	63	55	47	39
31	23	15	7	62	54	46	38
30	22	14	6	61	53	45	37
29	21	13	5	28	20	12	4



1. DES 암호화 과정

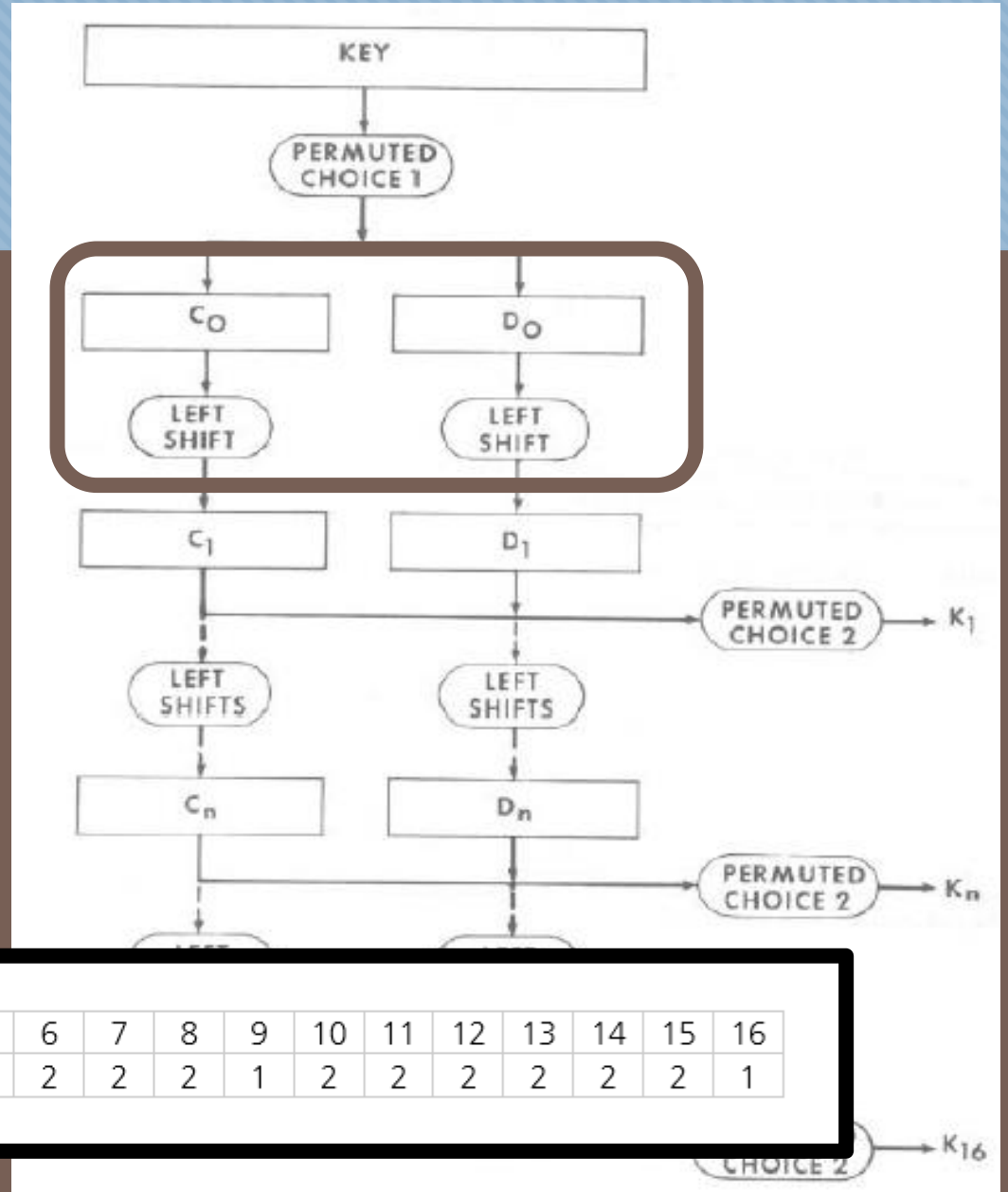
9) key(64bit)

8bit 패리티 체크
비트 제거(56bit)



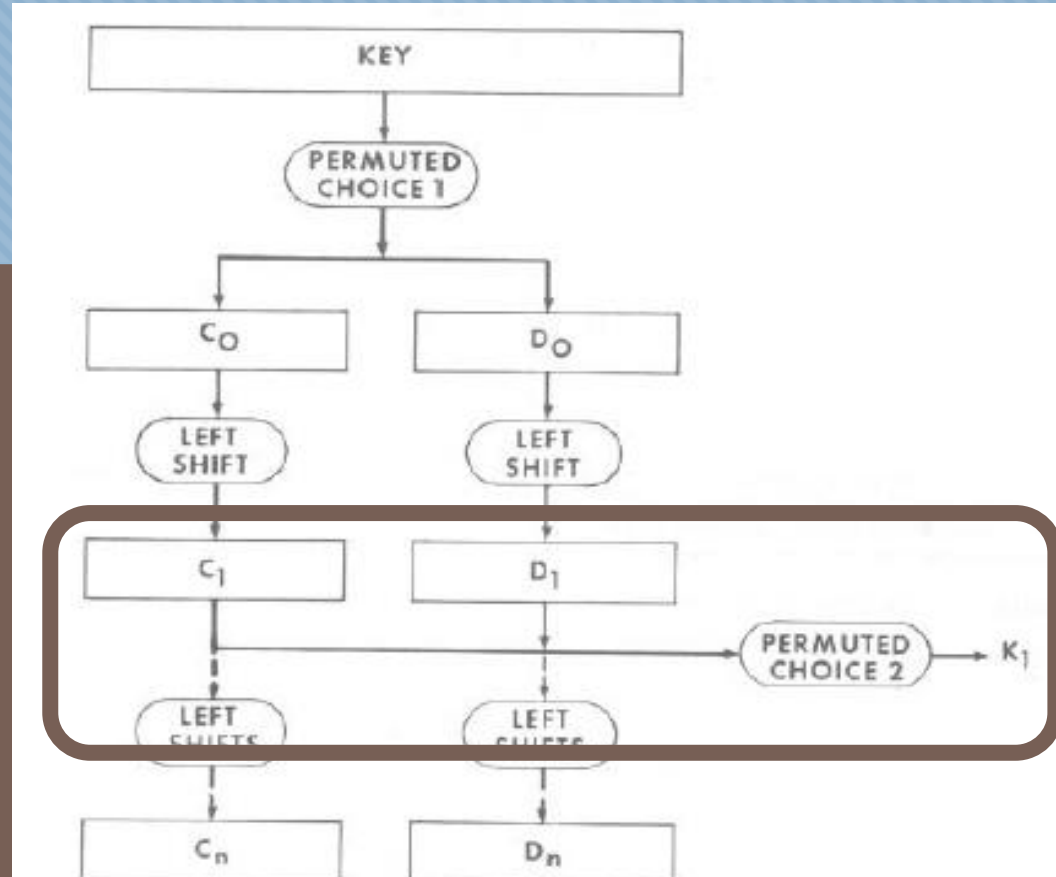
1. DES 암호화 과정

10) C0, D0 28bit
를 각각 왼쪽으로
비트 이동
(1라운드-> 1번)



1. DES 암호화 과정

11) PC2 치환 후,
K(48bit) 생성



- DES PC2(Permuted choice 2, 선택 치환 2)

PC2							
14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

2. DES 구현 (진행 중)

```
def swapper(leftBlock, rightBlock):  
    ### Function swapper: switch leftBlock and rightBlock  
    ###  
    ### BEGIN - description of parameters  
    ###     leftBlock:           left sub-block as input  
    ###                               but the contents would be that of rig  
    ###     rightBlock:          right sub-block as input  
    ###                               but the contents would be that of lef  
    ### END - description of parameters  
    assert(len(leftBlock) == 32)  
    assert(len(rightBlock) == 32)  
  
    T = []  
  
    ### BEGIN - TODO (insert code here)  
  
    ### END - TODO
```

2. DES 구현

```

print ("Round %d - %s\n" % (round, BinaryArrayToHexString(outBlock, 16)))
#      ((round + 1), #
#      BinaryArrayToHexString(leftBlock, 8), #
#      BinaryArrayToHexString(rightBlock, 8), #
#      BinaryArrayToHexString(RoundKeys[round], 12)) )
### END - Uncomment when you test result for each round in the report

combine(32, 64, leftBlock, rightBlock, outBlock)
### BEGIN - Uncomment when you test result for each round in the report
#print ("-" * 60 + "\nAfter combination: %s" % #
#      BinaryArrayToHexString(outBlock, 16) )
### END - Uncomment when you test result for each round in the report

permute(64, 64, outBlock, cipherBlock, FinalPermutationTable)
### BEGIN - Uncomment when you test result for each round in the report
#print ("Ciphertext: %s\n" % #
#      BinaryArrayToHexString(cipherBlock, 16))
### END - Uncomment when you test result for each round in the report

assert(len(cipherBlock) == 64)

def Key_Generator(keyWithParities, RoundKeys, ShiftTable):
    ### Function Key_Generator: round key generation algorithm
    ###
    ### BEGIN - description of parameters
    ### keyWithParities:      64-bit input key
    ### RoundKeys:           16 48-bit round keys to be generated from keyWithParities
    ### ShiftTable:          shift table indicating the amount of circular shift left
    assert(len(keyWithParities) == 64)
    assert(len(ShiftTable) == 16)

    cipherKey = []
    leftKey = []
    rightKey = []
    preRoundKey = []

    ### BEGIN - TODO (insert code here)

    cipherKey = ParityDropTable
    preRoundKey = KeyCompressionTable

    assert(len(cipherKey) == 56)

    leftKey[:] = cipherKey[:28]
    rightKey[:] = cipherKey[28:]

```

감사합니다.