Data Encryption Standard (DES)

- The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST). Most widely used block cipher in world .Adopted in 1977 by NBS as FIPS PUB 46 . Encrypts **64-bit data** using **56-bit key** .Has widespread use

DES STRUCTURE

The encryption process is made of two permutations (P-boxes), which we call initial and final permutations, and sixteen rounds.

DES CONSIST OF THE FOLLOWING:

Initial and Final Permutations

Rounds

DES Algorithm

The encryption algorithm involve 5 function

- 1:- Permutation [IP]
- 2:- Complex function labeled F_k involve both the permutation and depend on the Key input ;
- 3:- The permutation to switched SW the haves of the data
- 4:- the function F_K
- 5:- The inverse of the initial function permutation (IP^{-1})

Encryption (cont.)

- Plaintext:X
- Initial Permutation: IP( )
- $Round_i$: $1 \le i \le 16$ (FK)
- 32-bit switch: SW( )
- Inverse IP: IP^{-1}( )
- Ciphertext: $Y = IP^{-1}(SW(Round_i(IP(X), Key_i)))$

Figure 2.1 *General structure of DES*

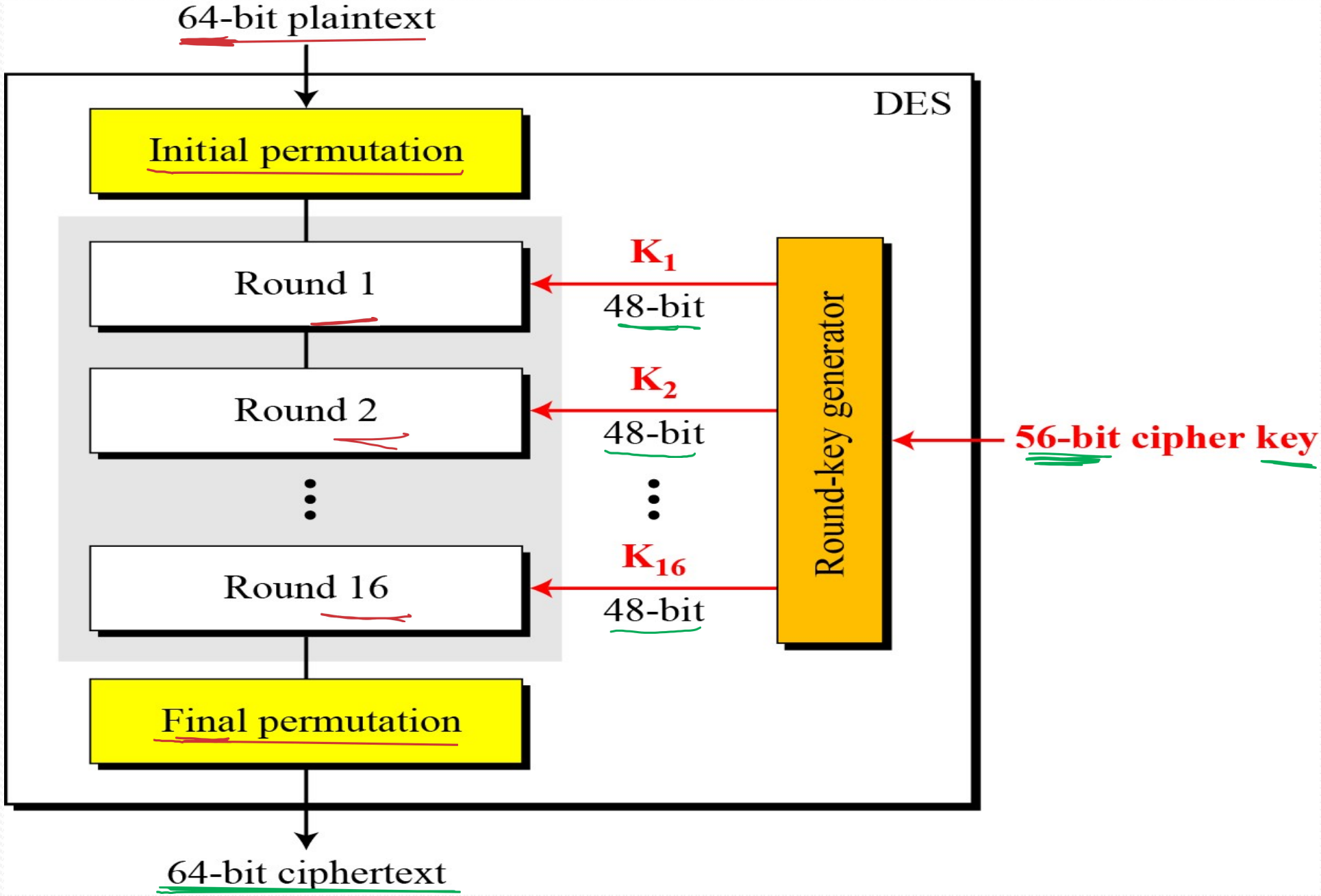
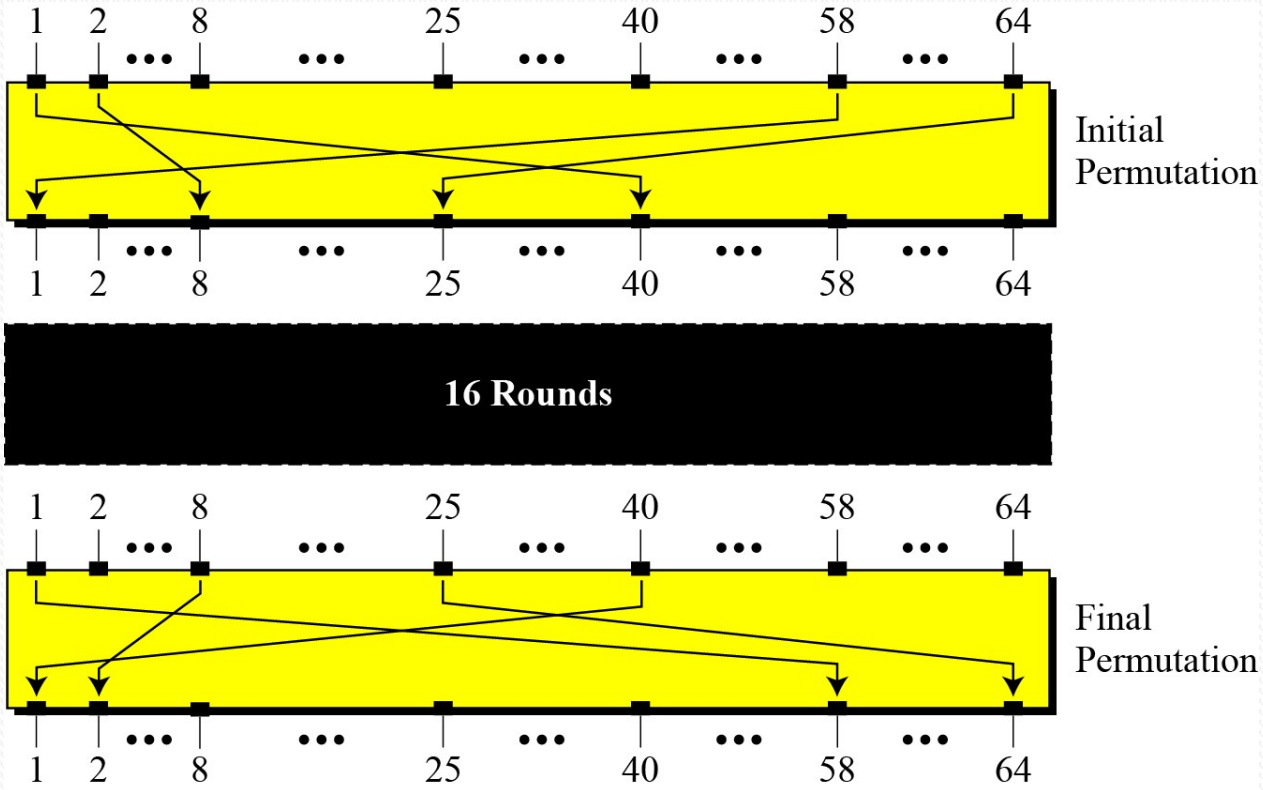


Figure 2.2 *Initial and final permutation steps in DES*



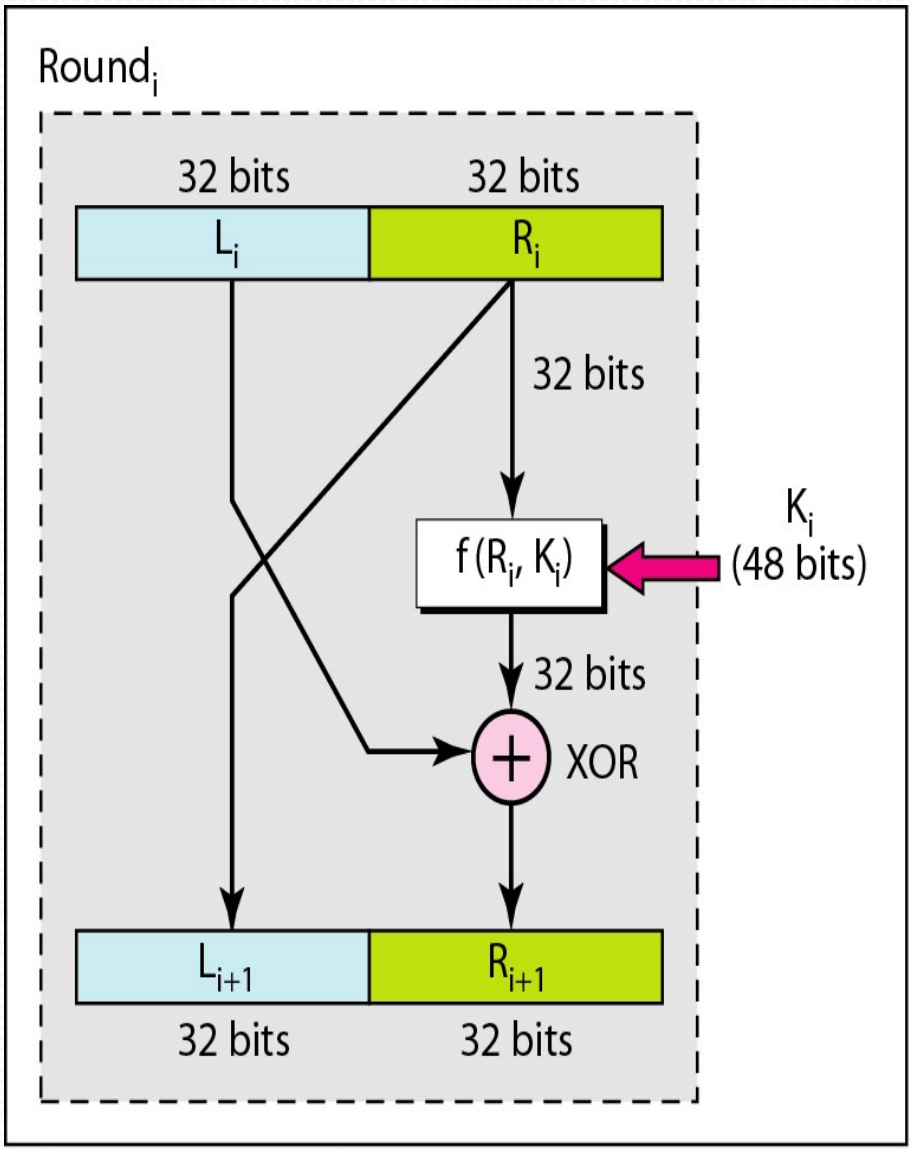
Permutation is an operation performed by a function, which moves an element at place *j* to the place *k*.

Table 2.1 *Initial and final permutation tables*

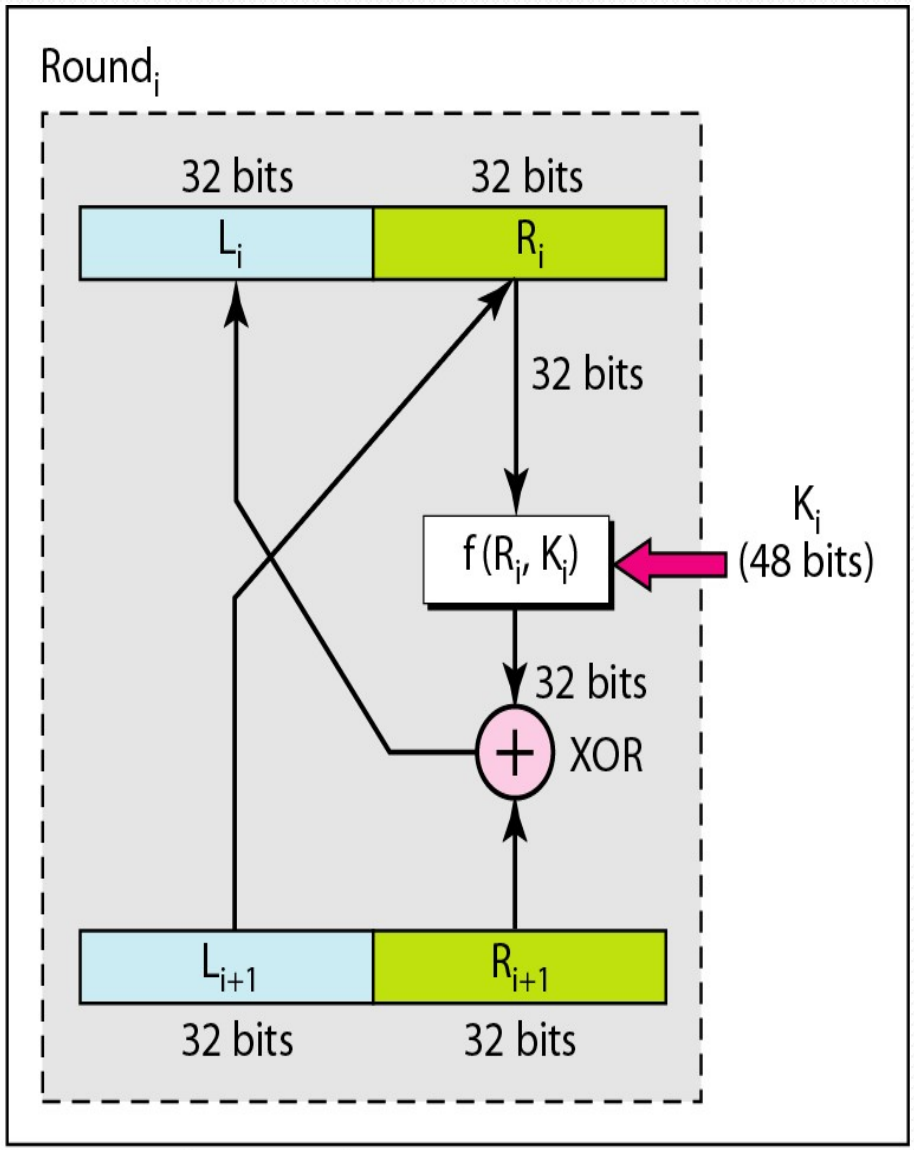
Initial Permutation	*Final Permutation*
58 50 42 34 26 18 10 02	40 08 48 16 56 24 64 32
60 52 44 36 28 20 12 04	39 07 47 15 55 23 63 31
62 54 46 38 30 22 14 06	38 06 46 14 54 22 62 30
64 56 48 40 32 24 16 08	37 05 45 13 53 21 61 29
57 49 41 33 25 17 09 01	36 04 44 12 52 20 60 28
59 51 43 35 27 19 11 03	35 03 43 11 51 19 59 27
61 53 45 37 29 21 13 05	34 02 42 10 50 18 58 26
63 55 47 39 31 23 15 07	33 01 41 09 49 17 57 25

Figure 2.3 each round of DES

DES uses 16 rounds. Each round of DES is a Feistel cipher.



a. Encryption round



b. Decryption round

Encryption (Round)

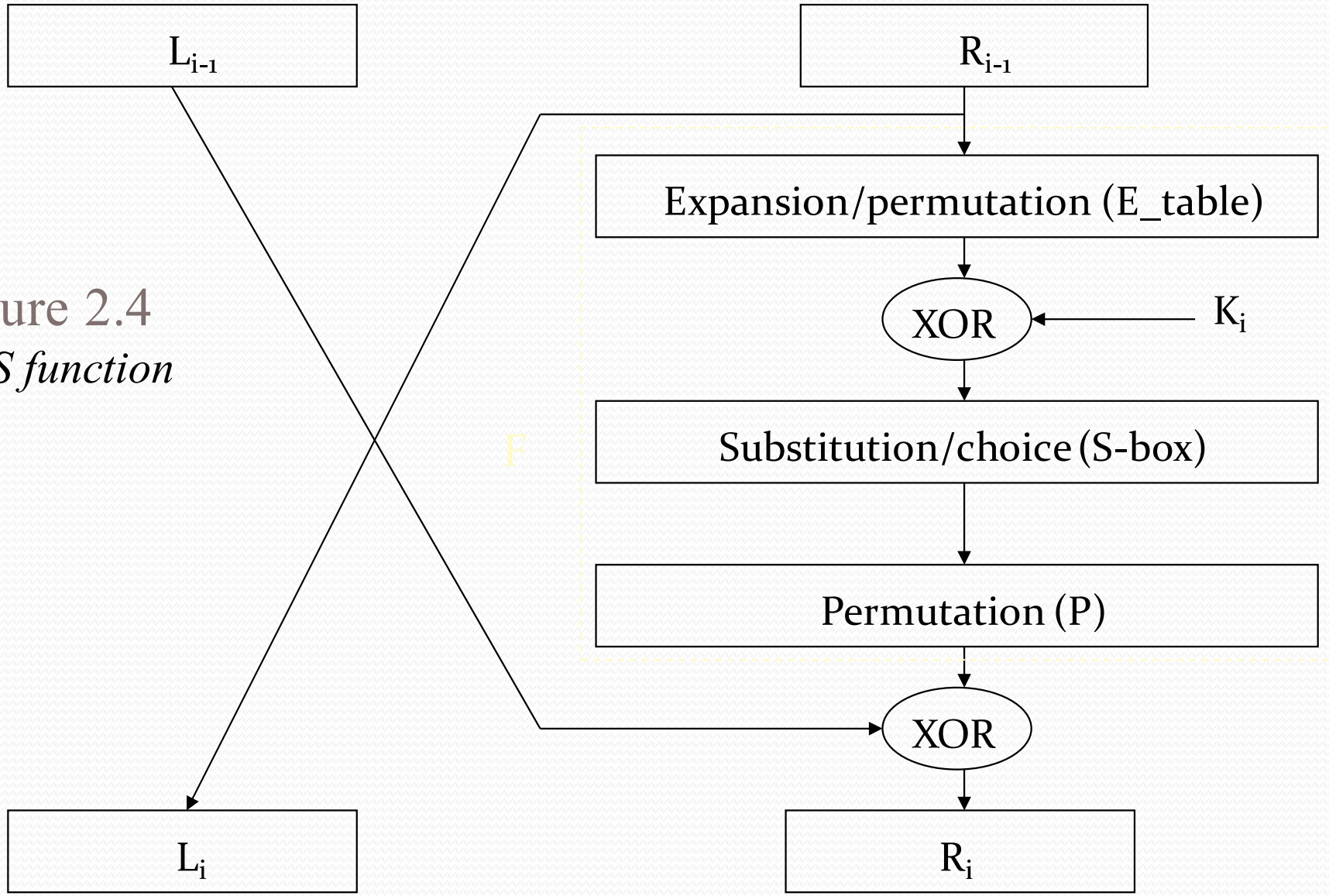


Figure 2.4
DES function

DES Round Structure

- uses two 32-bit L & R halves
- as for any Feistel cipher can describe as:

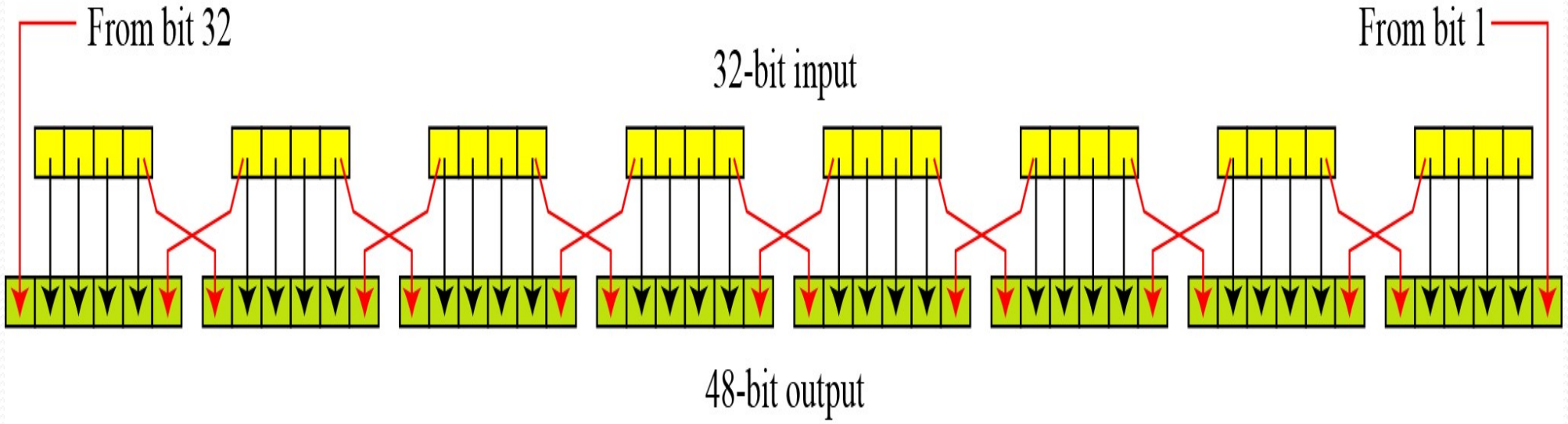
$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus \mathrm{F}(R_{i-1}, K_i)$$

- **F** takes 32-bit R half and 48-bit subkey:
  - expands R to 48-bits using permutation **E**
  - adds to subkey using XOR
  - split into 8 segment of 6 bits each, passes through 8 S-boxes to get 4-bit from each S-box then obtain 32-bit result.
  - finally permutes using 32-bit permutation **P**

Expansion P-box

Since R_{I-1} is a 32-bit input and K_I is a 48-bit key, we first need to expand R_{I-1} to 48 bits.

Figure 2.5 *Expansion permutation*



Encryption (Round)

- E

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	45	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Expansion

Since R_{I-1} is a 32-bit input and K_I is a 48-bit key, we first need to expand R_{I-1} to 48 bits.

- 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
9	13	30	6	22	11	4	25

Expansion

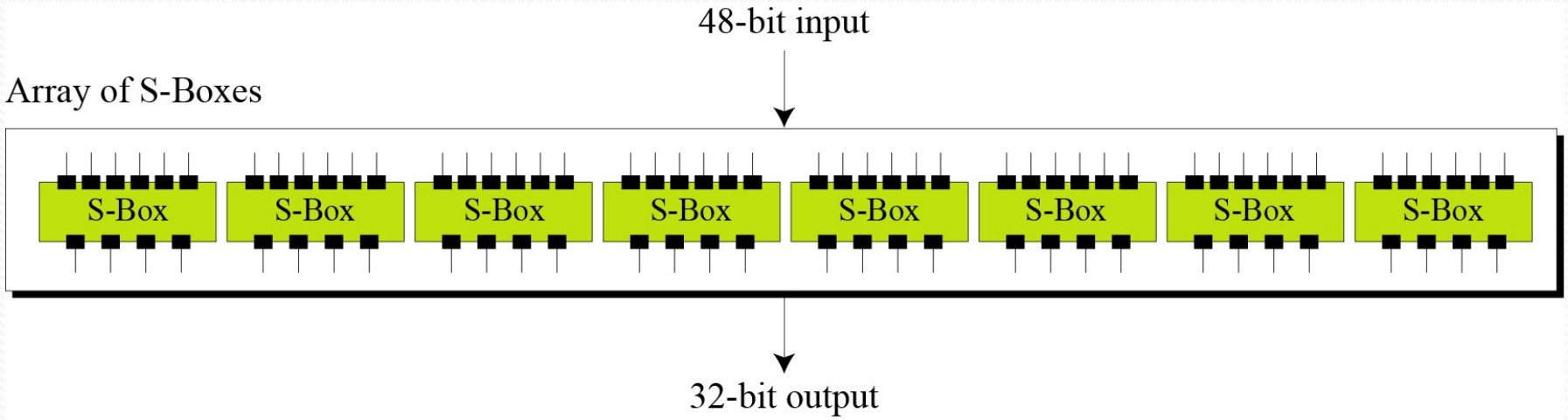
(XOR) Key with the right Section

- *Whitener (XOR)*
- *After the expansion permutation, DES uses the XOR operation on the expanded right section and the round key. Note that both the right section and the key are 48-bits in length. Also note that the round key is used only in this operation.*

S-Boxes

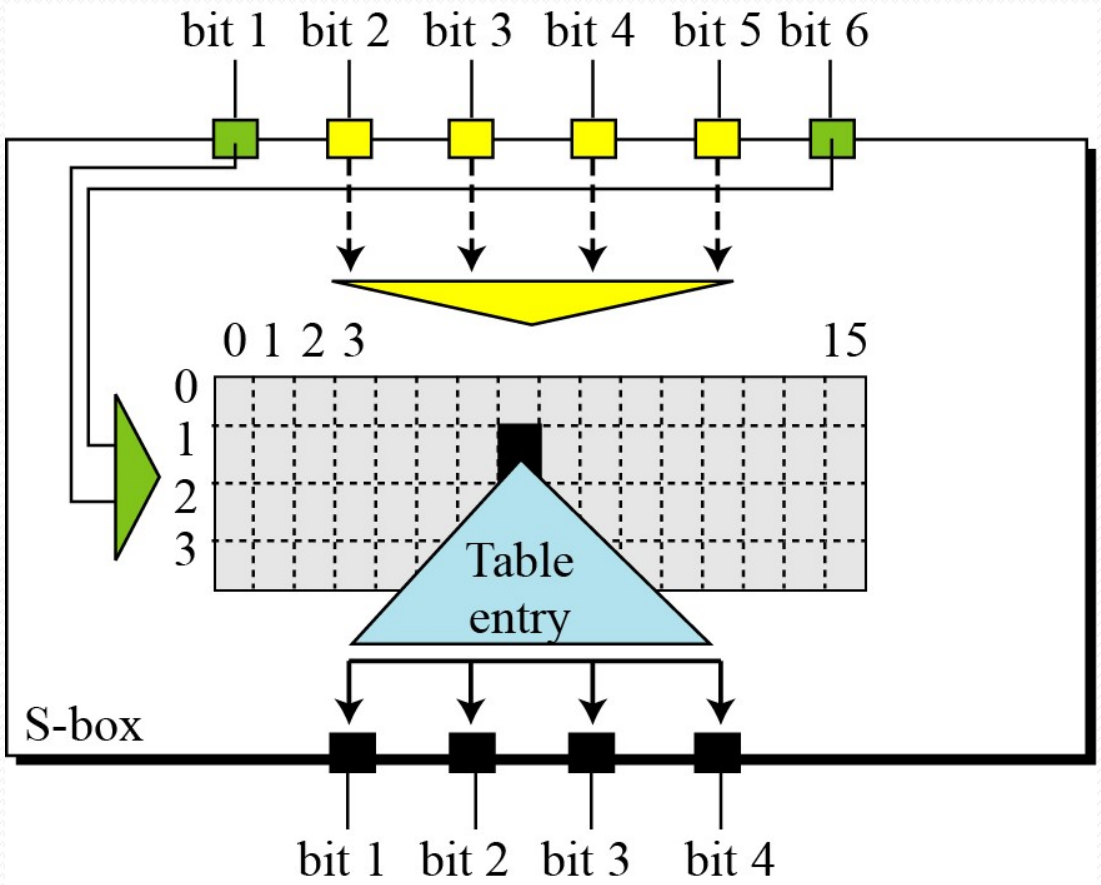
- have eight S-boxes which map 6 to 4 bits
- in each S-box
  - outer bits **1 & 6** (**row** bits) select one row of 4
  - inner bits **2-5** (**column** bits) are substituted
  - result is 8 lots of 4 bits, or 32 bits
- row selection depends on both data & key
  - feature known as autoclaving (autokeying)

Figure *S-boxes*



S-Box Rule

The S-boxes do the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output . Figure 2.7 *S-box rule*



Encryption (Round) (s-box.)

- S-box

S_1

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

S_2

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

S_3

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

S_4

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

S_5

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

S_6

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

S_7

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

S_8

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

[1]

Example 2.1

The input to S-box 1 is 100011. What is the output?

Solution

If we write the first and the sixth bits together, we get 11 in binary, which is 3 in decimal. The remaining bits are 0001 in binary, which is 1 in decimal. We look for the value in row 3, column 1, in Table (S-box 1). The result is 12 in decimal, which in binary is 1100. So the input 100011 yields the output 1100.

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

Example 2.2

The input to S-box 1 is 000000. What is the output?

Solution

If we write the first and the sixth bits together, we get 00 in binary, which is 0 in decimal. The remaining bits are 0000 in binary, which is 0 in decimal. We look for the value in row 0, column 0, in Table (S-box 1). The result is 14 in decimal, which is 1110 in binary. So the input 000000 yields the output 1110.

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

DES-Key generation

Shifting

Rounds	Shift
1, 2, 9, 16	one bit
Others	two bits

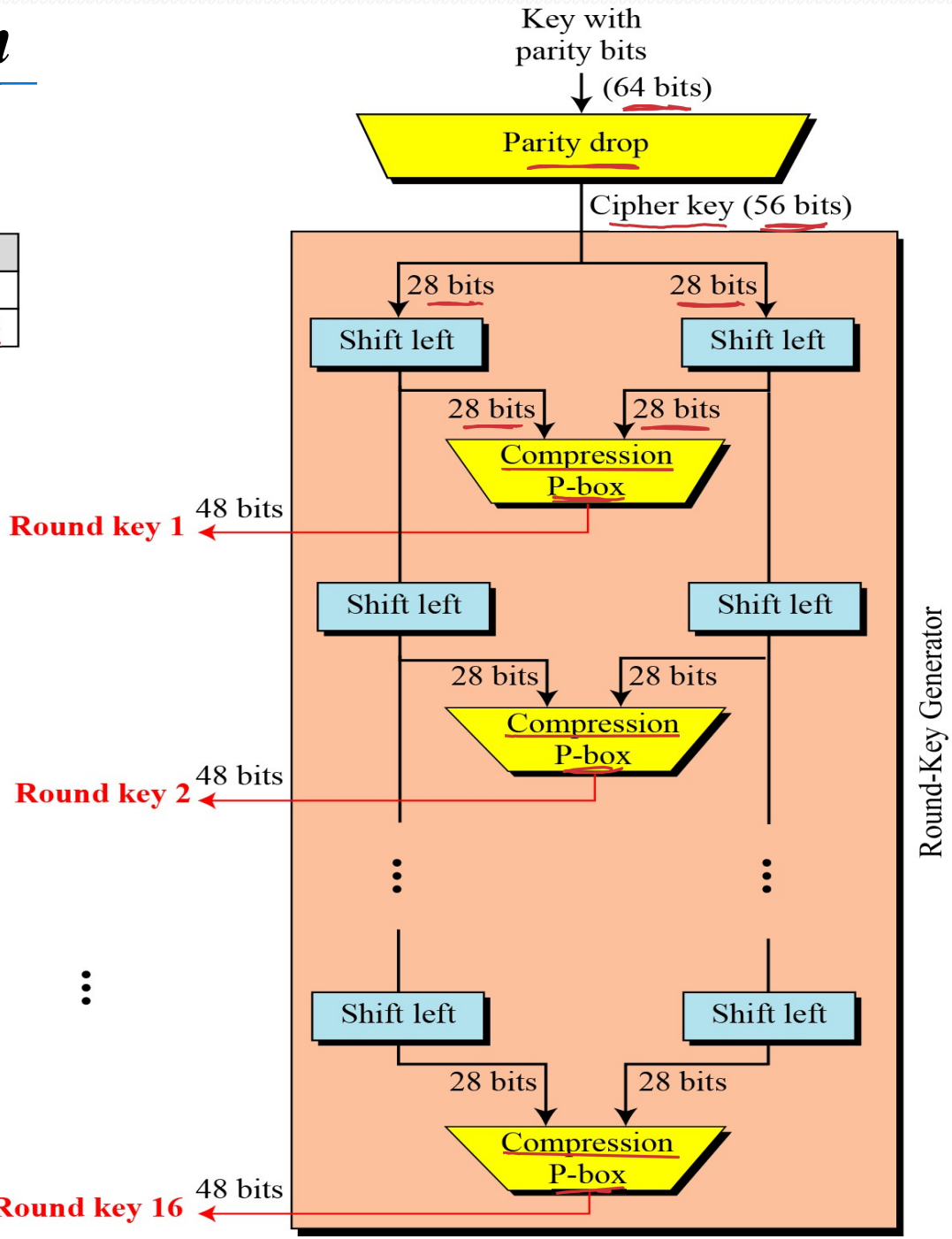


Figure 2.10

Key generation

DES Key Schedule

- Step 1: pass 64-bit key through a permutation called permutation choice1 or PC-1. (every eighth bit is ignored, for parity bit) This obtain 56-bit.
- Step 2: use this key to generate 16x48-bit key which is used in the 16 rounds of DES.
- Step 3: split the current 56-bit key up into 28-bit blocks (**L**eft-hand half and **R**ight-hand half).
- Step 4: rotate L and R by the number of bits specified in the shift table.
- Step 5: join L and R together to generate new K.
- Step 6: apply permuted choice2 or PC-2 to $\mathbf{K_R}$