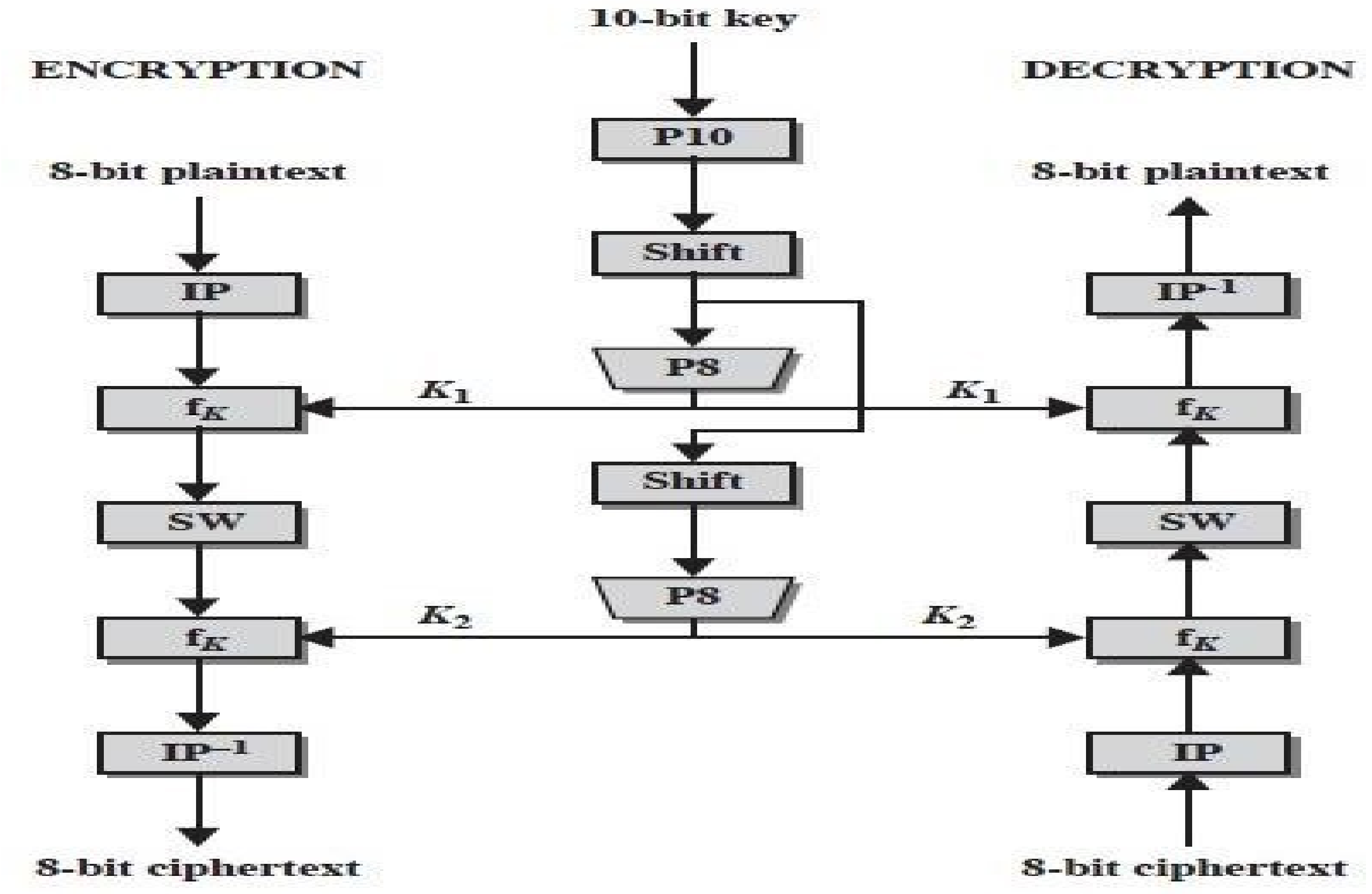
Simplified DES (S-DES), DES

SIMPLIFIED DES

- **Encryption**
- **Simplified Data Encryption Standard**
- The S-DES encryption algorithm takes an 8-bit block of plaintext (example: 10111101) and a 10-bit key as input
- produces an 8-bit block of ciphertext as output.

Overall structure of the simplified DES



S-DES encryption algorithm involves five functions:

1) an initial permutation (IP)
2) a complex function labeled f_K, which involves both permutation and substitution operations and depends on a key input
3) a simple permutation function that switches (SW) the two halves of the data
4) the function f_K again
5) a permutation function that is the inverse of the initial permutation (IP^{-1}).

Simplified DES

- SDES ‘s five steps:
- IP - an initial permutation
- f_k - a complex, 2-input function
- SW - switches the left half and the right half of a data string
- f_k - a complex, 2-input function; again
- IP^{-1} - inverse permutation of the initial permutation

SDES may be said to have two ROUNDS of the function $f_{k.}$

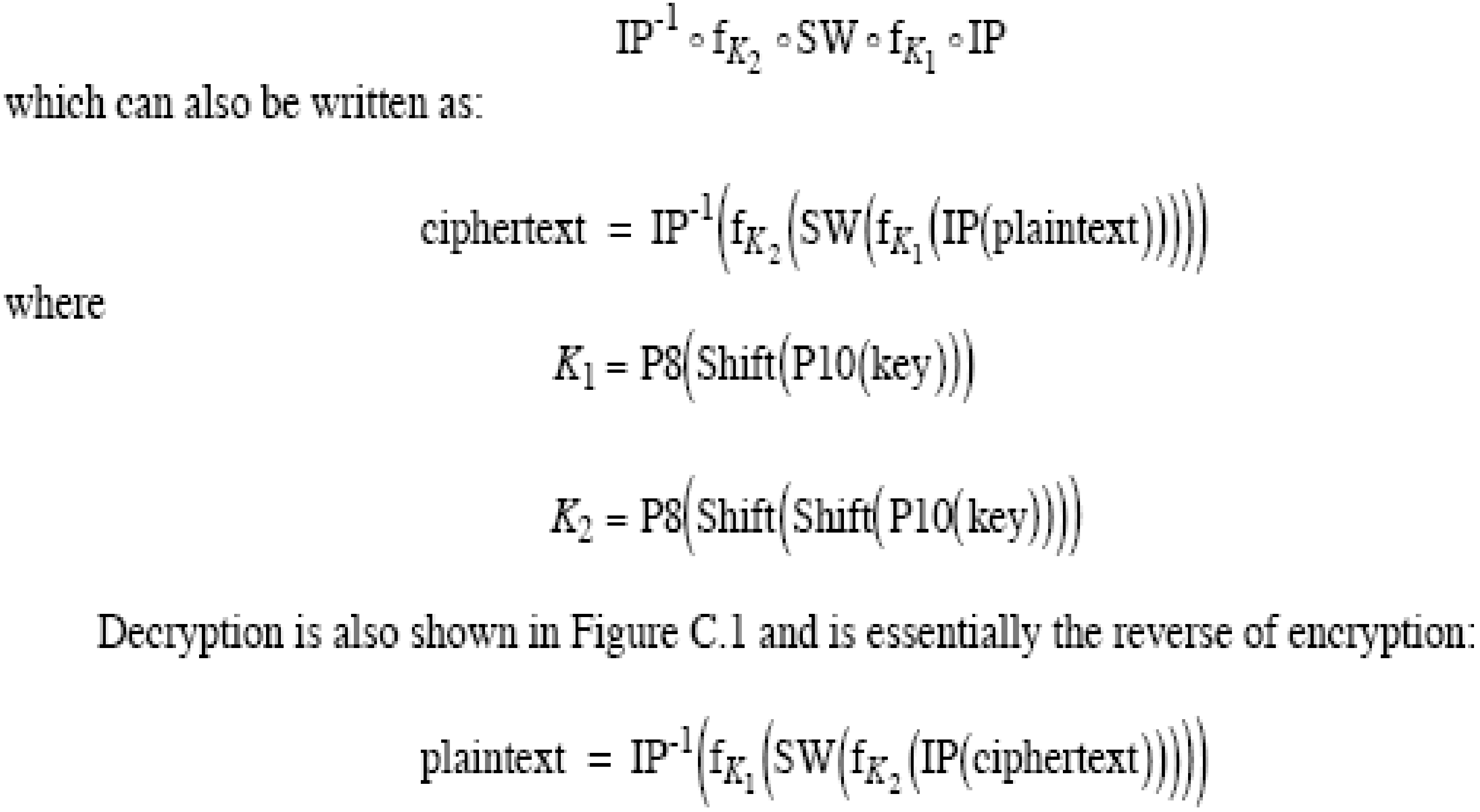
S-DES

- The use of multiple stages of permutation and substitution results in a more complex algorithm, which increases the difficulty of cryptanalysis.
- The function f_K takes as input not only the data passing through the encryption algorithm, but also an 8-bit key.

S-DES

- In this case, the key is first subjected to a permutation (P10).
- Then a shift operation is performed.
- The output of the shift operation then passes through a permutation function that produces an 8-bit output (P8) for the first subkey (K_1).
- The output of the shift operation also feeds into another shift and another instance of P8 to produce the second subkey (K_2).

The encryption algorithm is expressed as a composition of functions:



$$\mathrm{IP}^{-1} \circ \mathrm{f}_{K_2} \circ \mathrm{SW} \circ \mathrm{f}_{K_1} \circ \mathrm{IP}$$

which can also be written as:

$$\text{ciphertext} = \mathrm{IP}^{-1}\left(\mathrm{f}_{K_2}\left(\mathrm{SW}\left(\mathrm{f}_{K_1}\left(\mathrm{IP}(\text{plaintext})\right)\right)\right)\right)$$

where

$$K_1 = \mathrm{P8}\left(\mathrm{Shift}\left(\mathrm{P10}(\text{key})\right)\right)$$

$$K_2 = \mathrm{P8}\left(\mathrm{Shift}\left(\mathrm{Shift}\left(\mathrm{P10}(\text{key})\right)\right)\right)$$

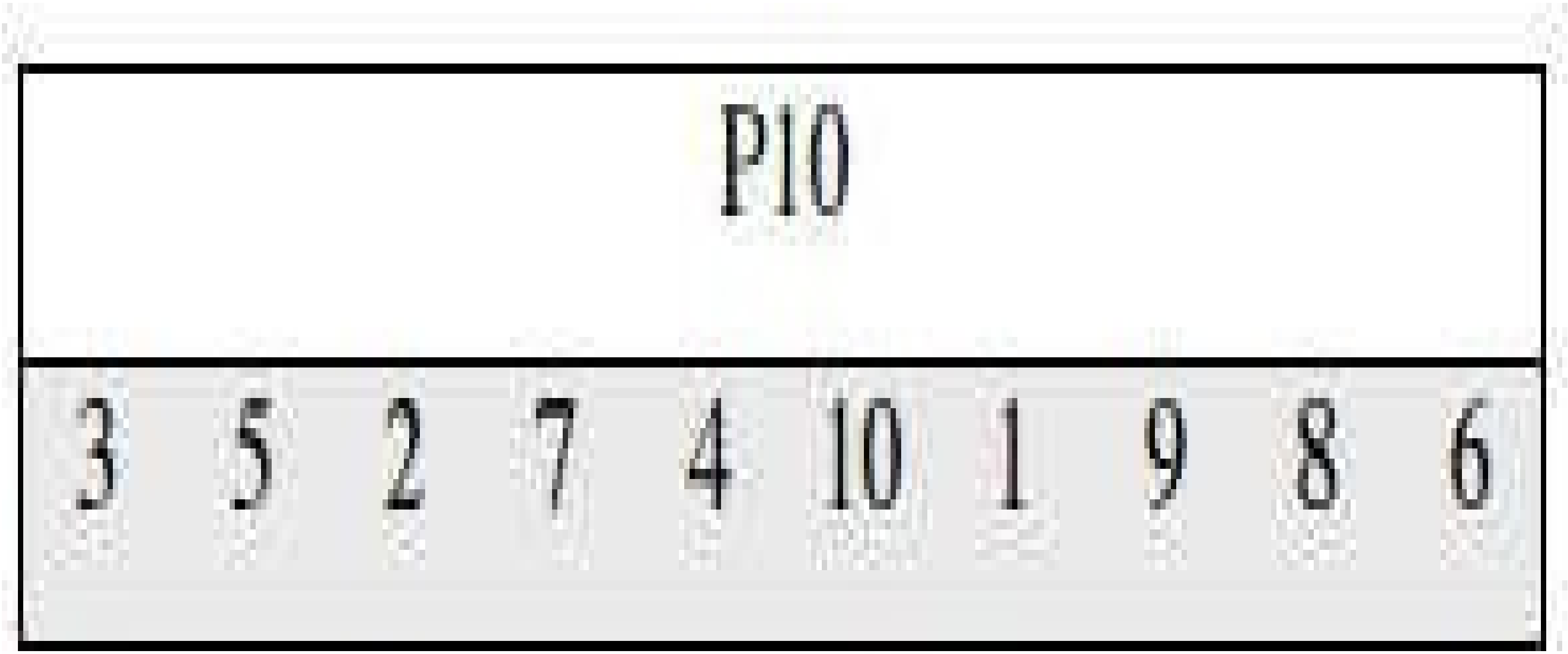
Decryption is also shown in Figure C.1 and is essentially the reverse of encryption:

$$\text{plaintext} = \mathrm{IP}^{-1}\left(\mathrm{f}_{K_1}\left(\mathrm{SW}\left(\mathrm{f}_{K_2}\left(\mathrm{IP}(\text{ciphertext})\right)\right)\right)\right)$$

S-DES KEY GENERATION

- First, permute the key in the following fashion.
- Let the 10-bit key be designated as (k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{10}).
- Then the permutation P10 is defined as:
- P10 (k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{10}) = (k_3, k_5, k_2, k_7, k_4, k_{10}, k_1, k_9, k_8, k_6)

P10 can be defined by:

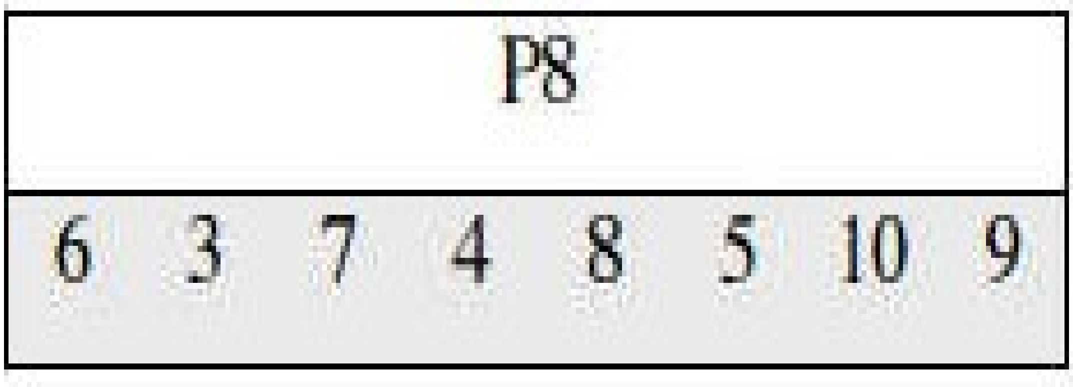


P10									
3	5	2	7	4	10	1	9	8	6

- This table is read from left to right
- each position in the table gives the identity of the input bit that produces the output bit in that position.
- So the first output bit is bit 3 of the input; the second output bit is bit 5 of the input, and so on.

- For example, the key (1010000010) is permuted to (1000001100).
- Next, perform a circular left shift (LS-1), or rotation, separately on the first
- five bits and the second five bits. In our example, the result is (00001 11000).

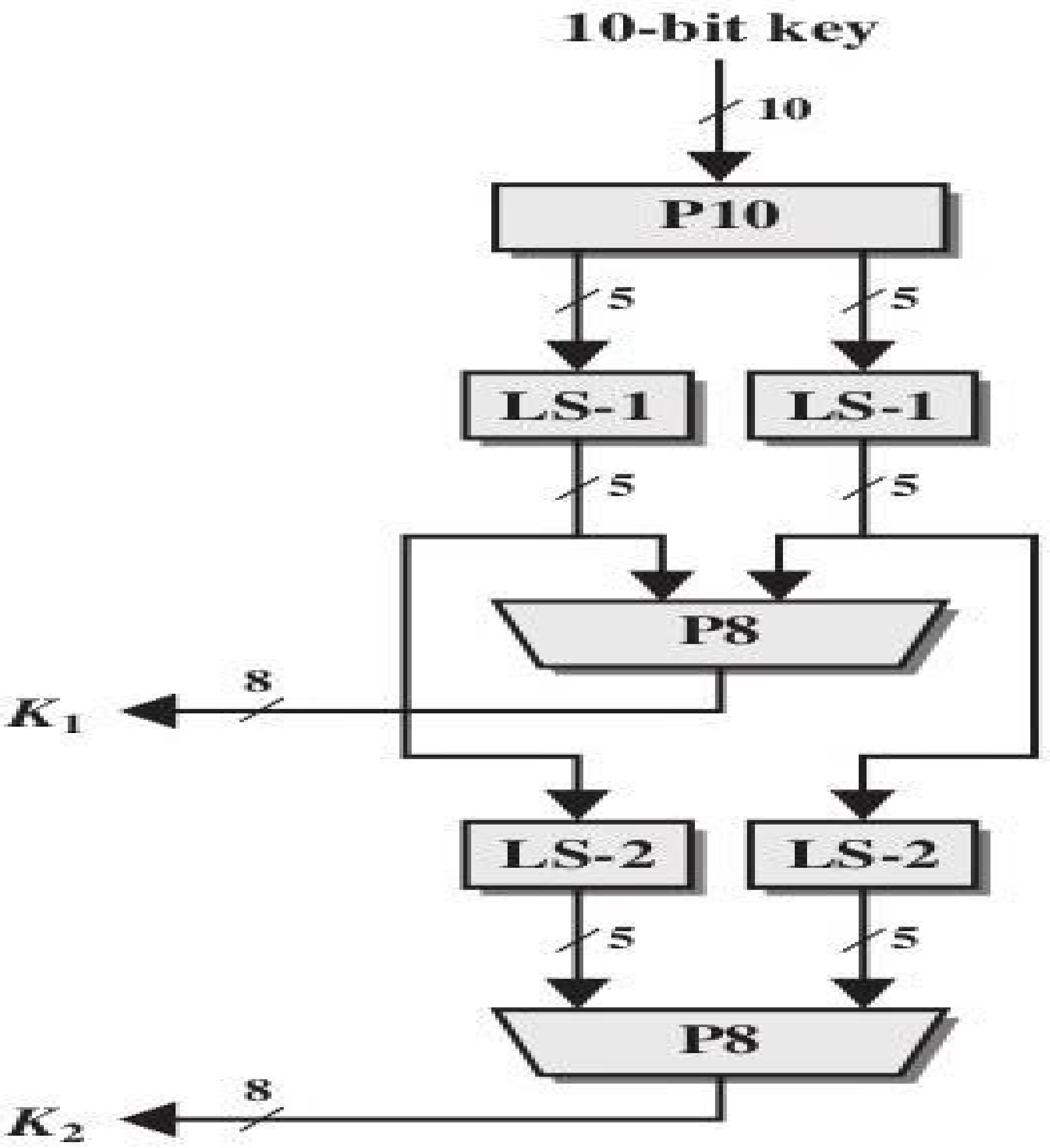
Next apply P8, which picks out and permutes 8 of the 10 bits according to the following rule:



P8							
6	3	7	4	8	5	10	9

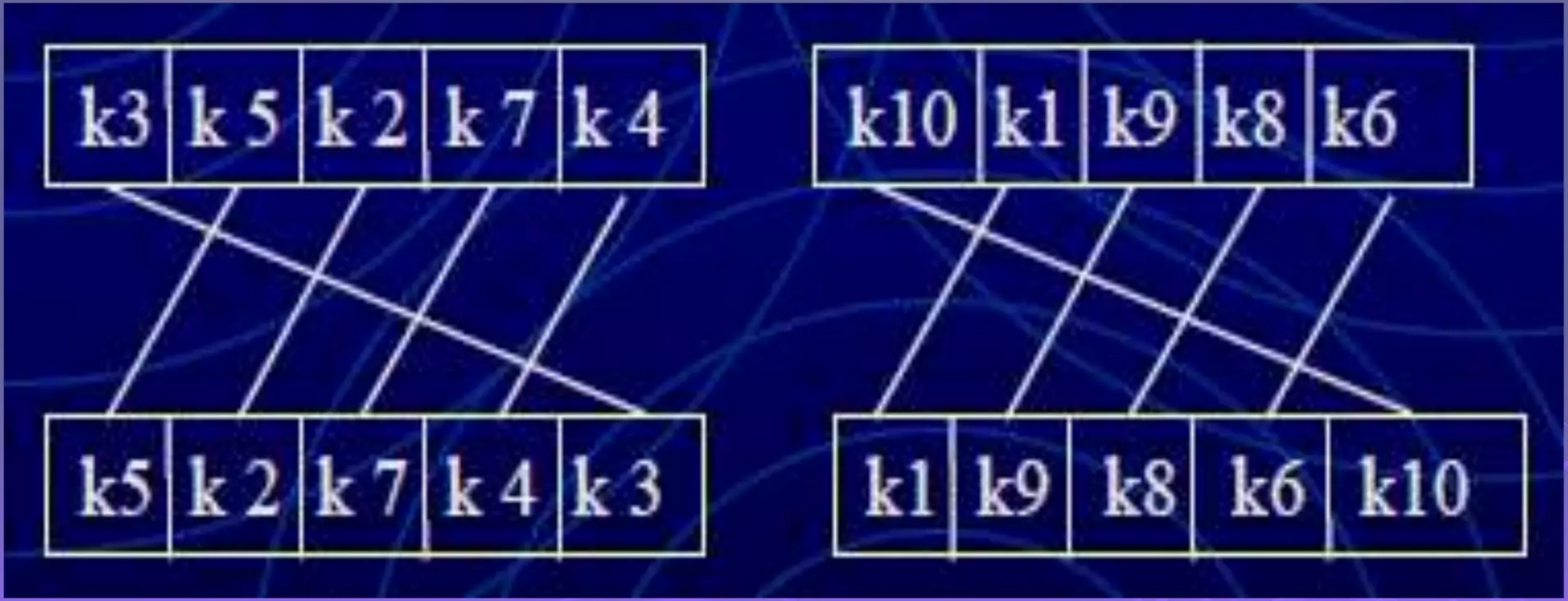
- The result is subkey 1 (K1). In our example, this yields (10100100).
- We then go back to the pair of 5-bit strings produced by the two LS-1
- functions and performs a circular left shift of 2 bit positions on each string. In our
- example, the value (00001 11000) becomes (00100 00011). Finally, P8 is applied
- again to produce K2. In our example, the result is (01000011).

S-DES Key



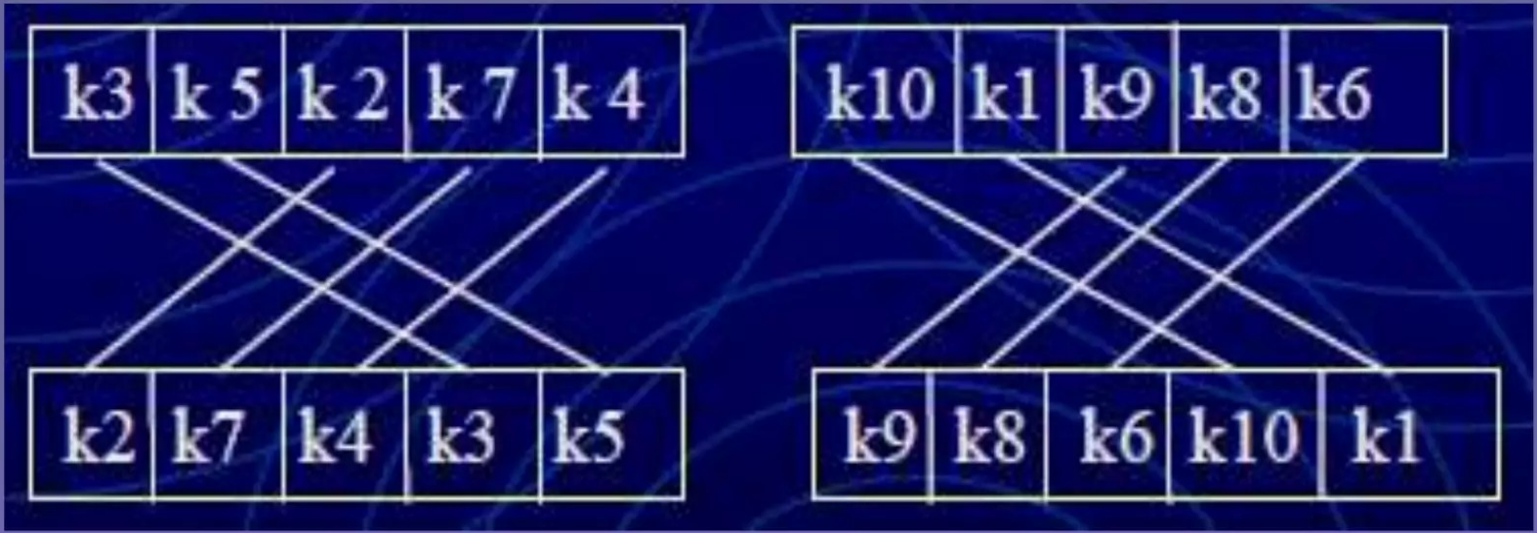
LS-1

- Left circular shift 1 each 5 bit group



LS-2

- Left circular shift 2 each 5 bit group



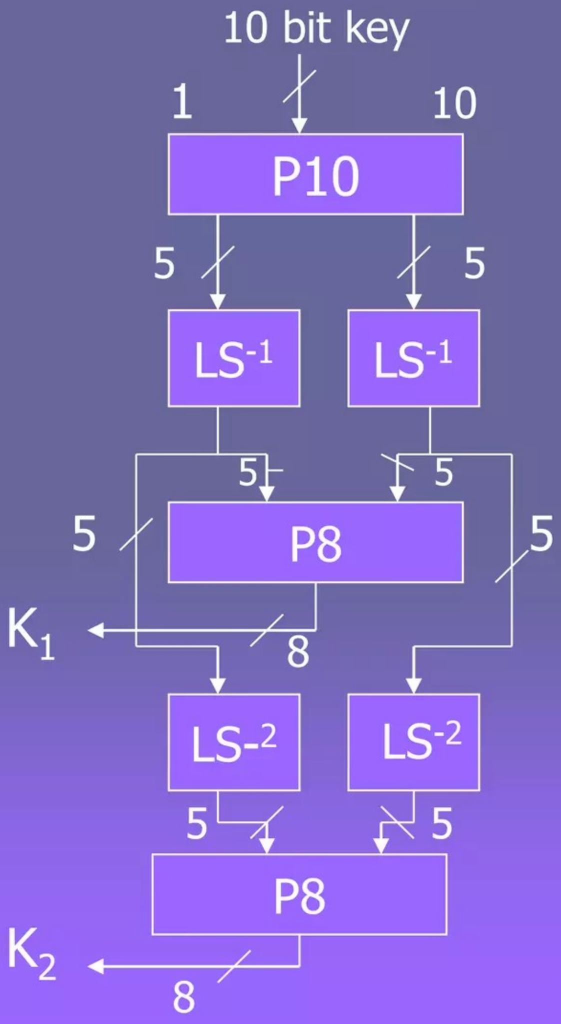
k3 | k 5 | k 2 | k 7 | k 4 k10 | k1 | k9 | k8 | k6

k2 | k7 | k4 | k3 | k5 k9 | k8 | k6 | k10 | k1

Key generation for simplified DES:

$$K_1 = P8\ (\ \text{Shift}\ (P10\ (\text{Key})))$$

$$K_2 = P8\ (\ \text{Shift}\ (\ \text{Shift}\ (P10\ (\text{Key}))))$$



3	5	2	7	4	10	1	9	8	6	P10

Circular left shift by 1, separately on the left and the right halves

6	3	7	4	8	5	10	9	P8

Circular left shift by 2 , separately on the left and the right halves

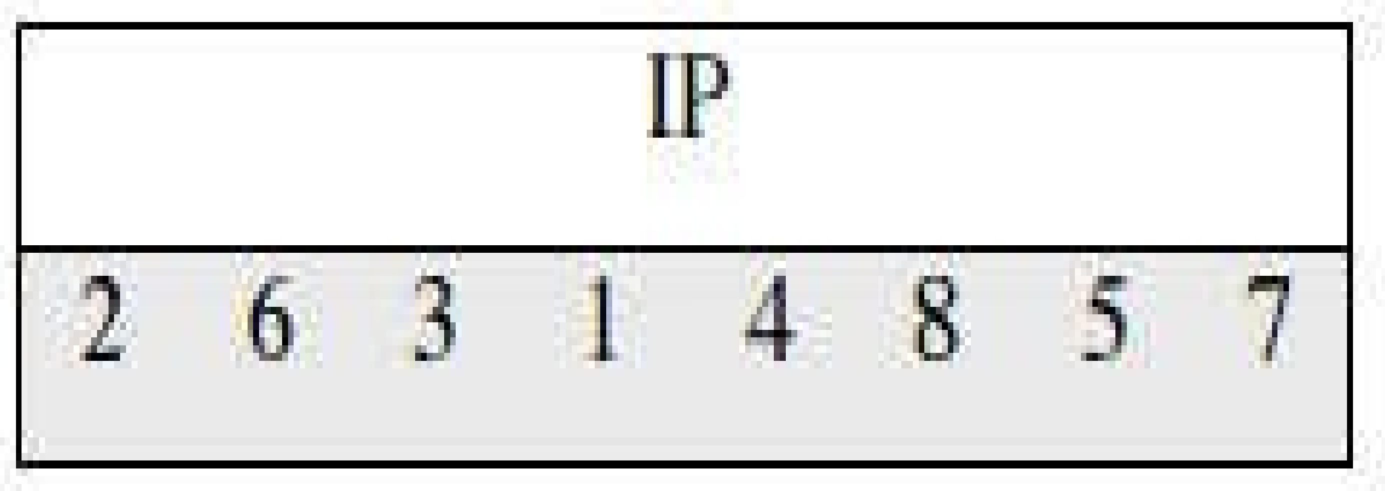
6	3	7	4	8	5	10	9	P8

S-DES ENCRYPTION

1) Initial and Final Permutations

- The input to the algorithm is an 8-bit block of plaintext, which we first permute using the IP function:

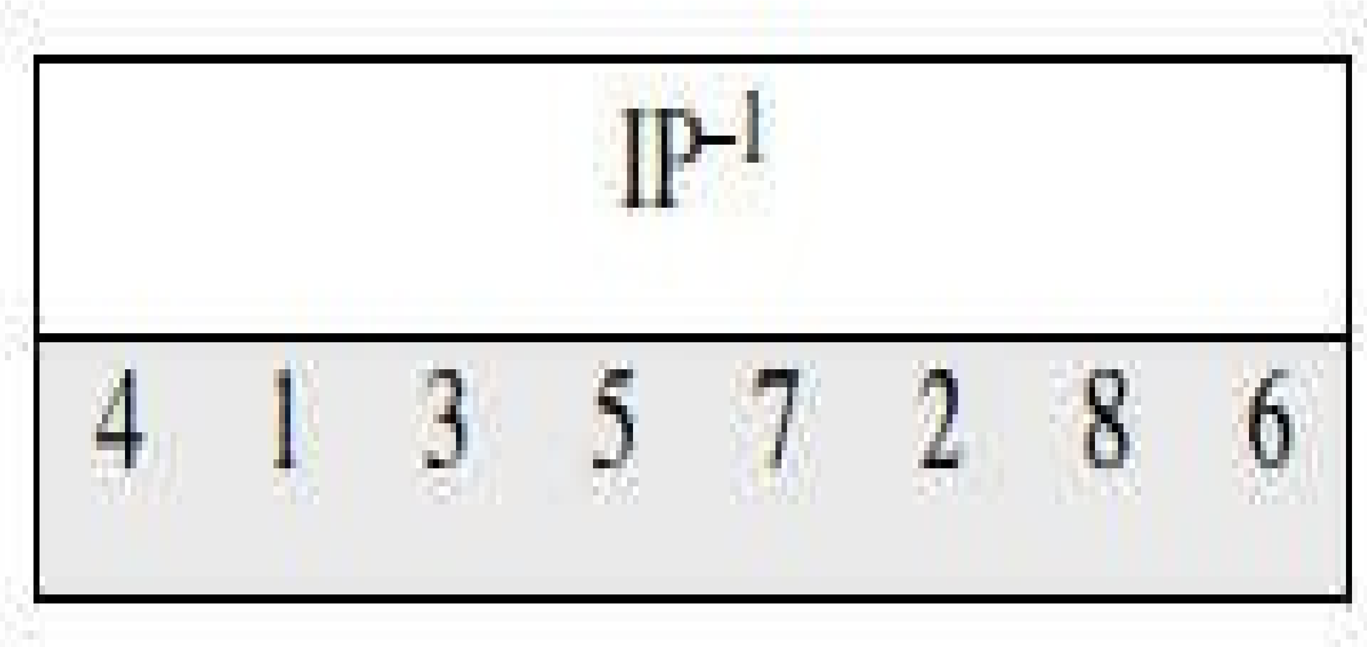
Initial Permutation



IP							
2	6	3	1	4	8	5	7

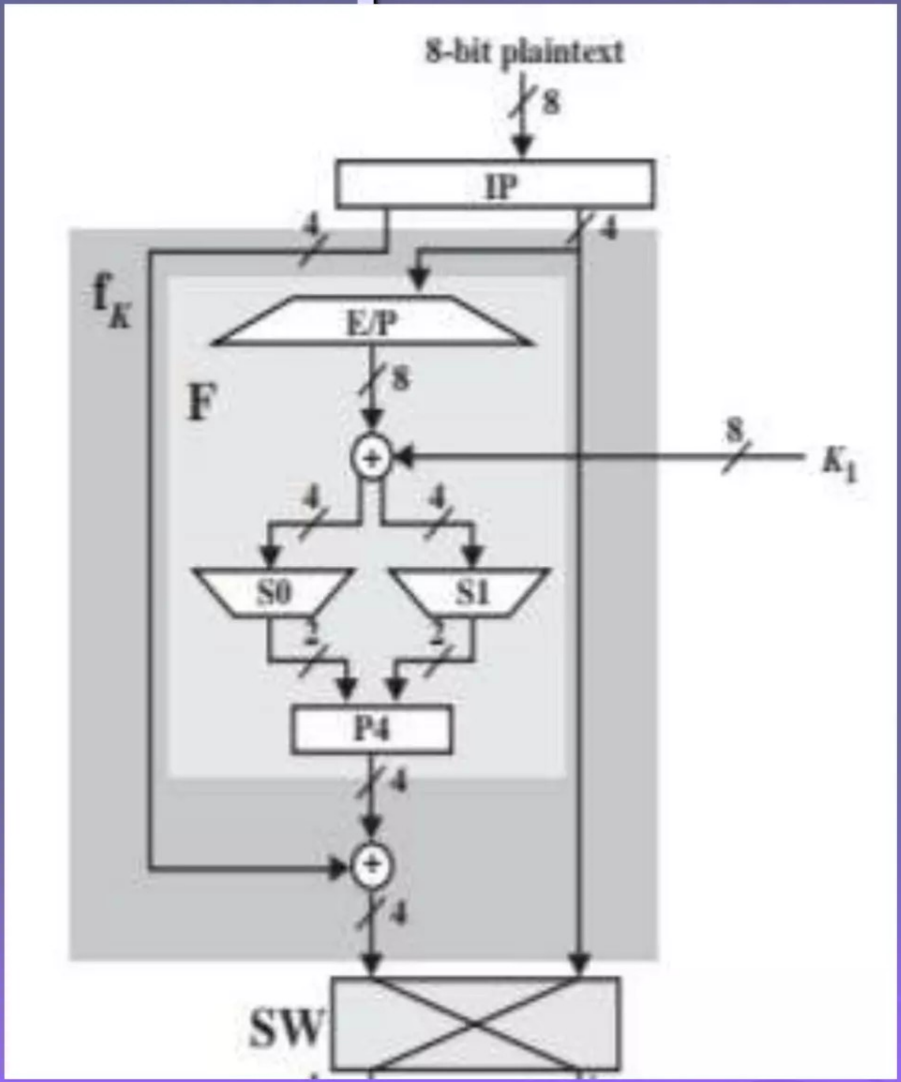
Inverse of IP

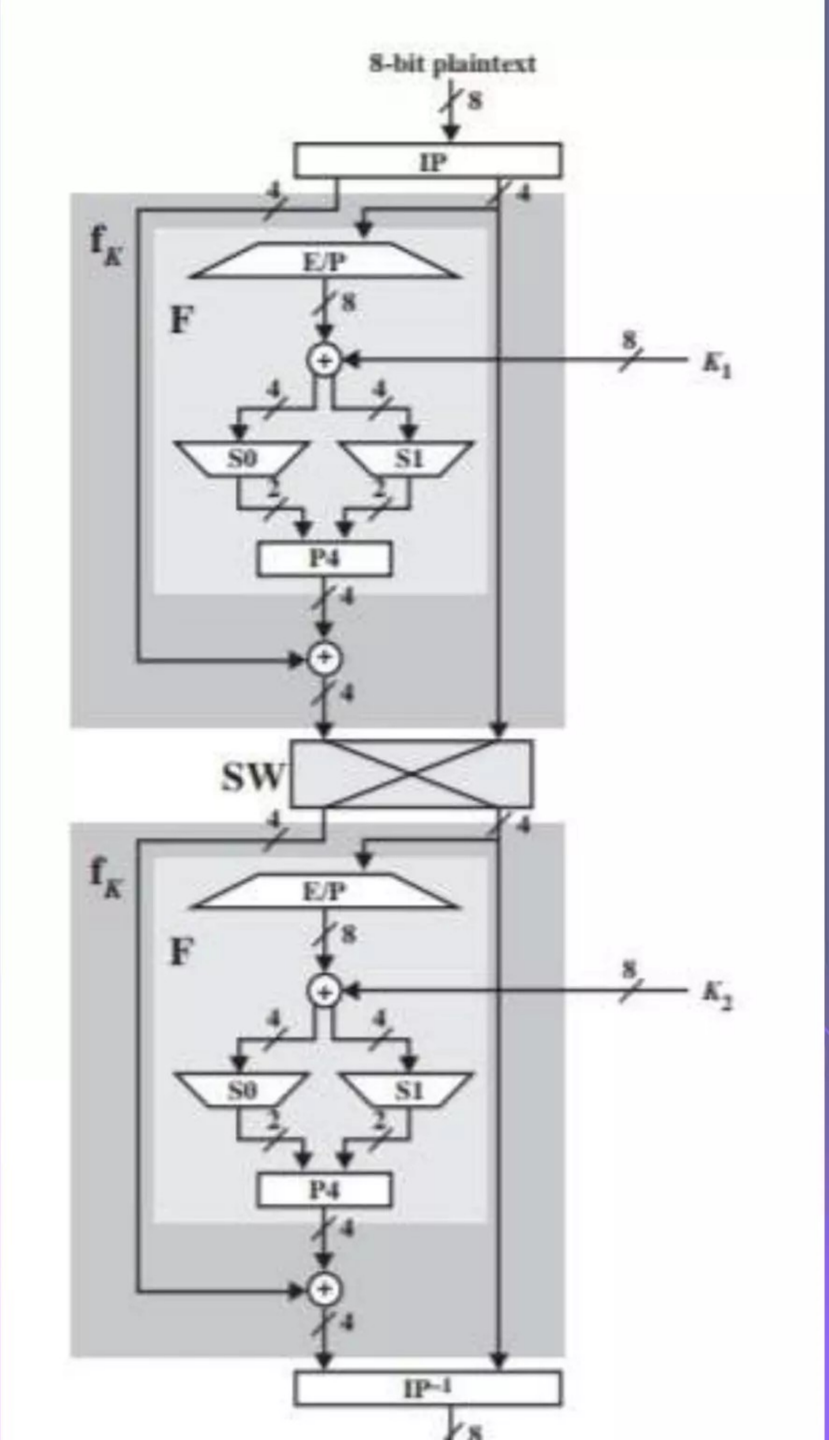
$IP^{-1}(IP(X)) = X$



IP^{-1}							
4	1	3	5	7	2	8	6

Simplified DES



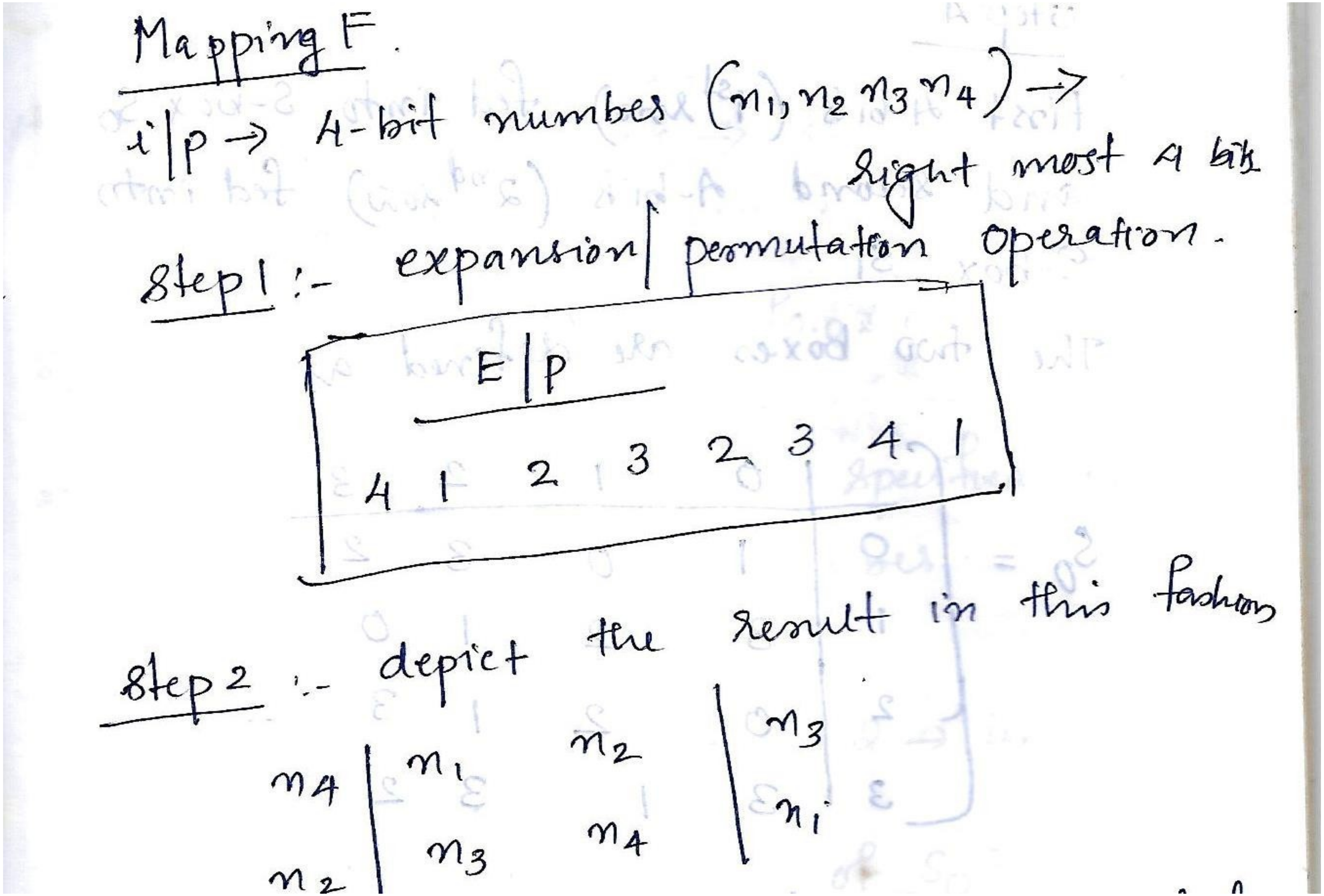


2) The Function f_K

- The function $\mathbf{f_K}$, consists of a combination of permutation and substitution functions. The functions can be expressed as follows:
- Let L and R be the leftmost 4 bits and rightmost 4 bits of the 8-bit input to $\mathbf{f_K}$, and let F be a mapping (not necessarily one to one) from 4-bit strings to 4-bit strings.

- SK – Sub key

$$f_K(L, R) = (L \oplus F(R, SK), R)$$



Mapping F.

i/p → 4-bit numbers $(n_1, n_2 n_3 n_4)$ →

right most 4 bits

Step 1 :- expansion/permutation operation.

E/P

4 1 2 3 2 3 4 1

Step 2 :- depict the result in this fashion

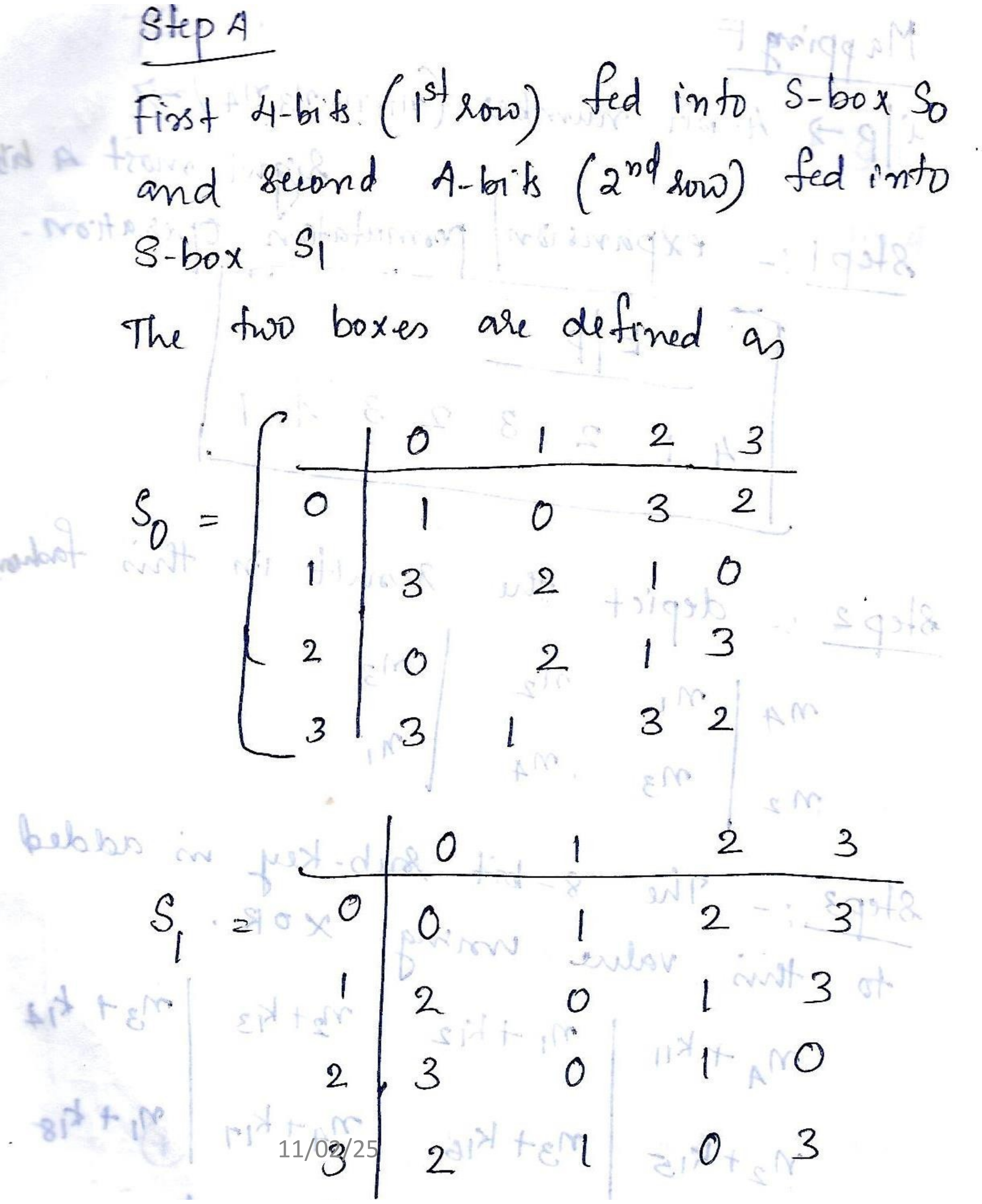
$$
\begin{array}{c|cc|c}
n_4 & n_1 & n_2 & n_3 \\
n_2 & n_3 & n_4 & n_1
\end{array}
$$

Step3 :- The 8-bit sub-key is add
to this value using XOR.

$n_4 + k_{11}$	$n_1 + k_{12}$	$n_2 + k_{13}$	$n_3 +$
$n_2 + k_{15}$	$n_3 + k_{16}$	$n_4 + k_{17}$	$n_1 + k$

$\downarrow$ result renamed as

$P_{0.0}$	$P_{0.1}$	$P_{0.2}$	$P_{0.3}$ → fed into S-bo
$P_{1.0}$	$P_{1.1}$	$P_{1.2}$	$P_{1.3}$ → fed into S-box



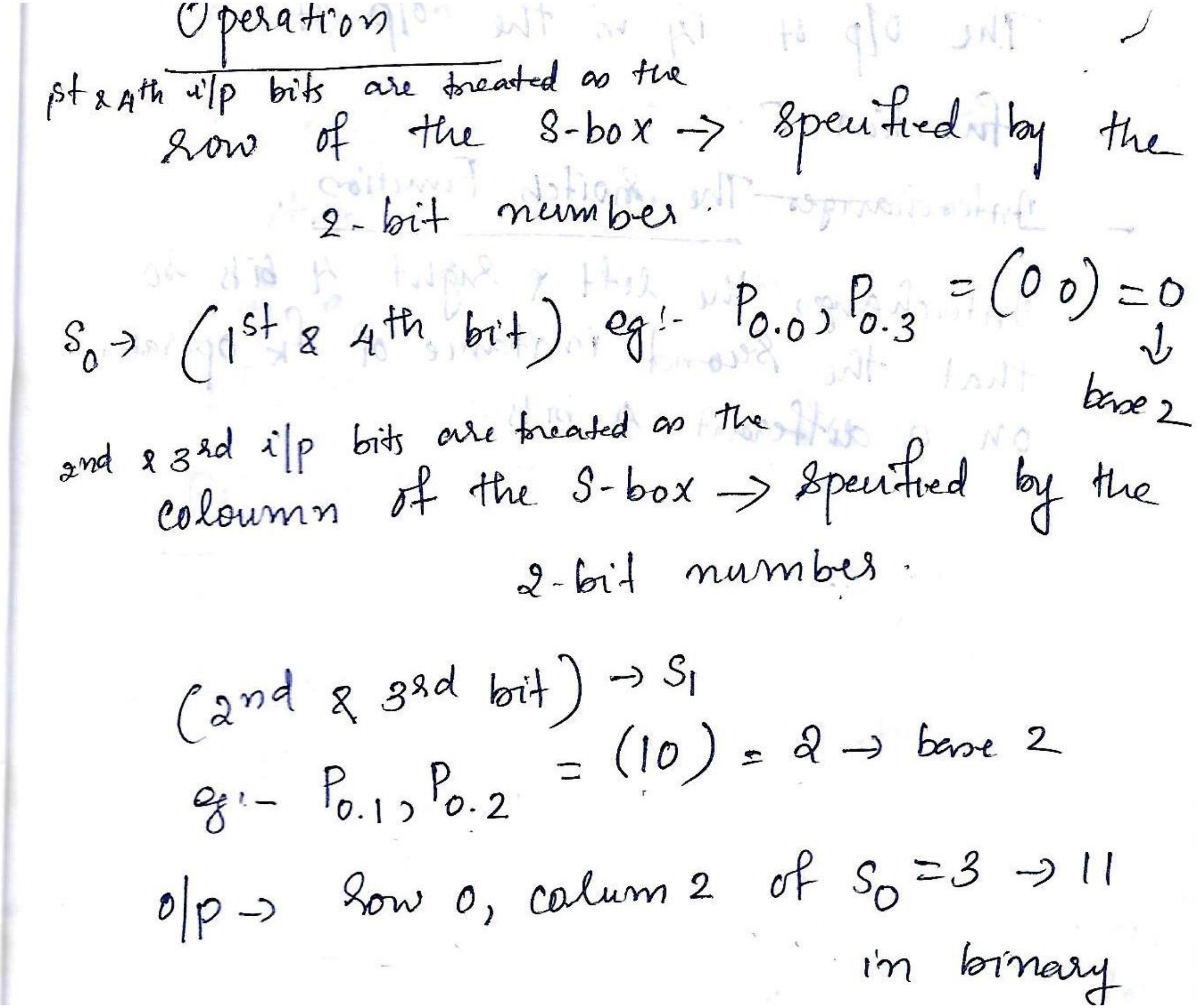
Step A

First 4-bits (1^{st} row) fed into S-box S_0 and second 4-bits (2^{nd} row) fed into S-box S_1

The two boxes are defined as

$$S_0 = \begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 1 & 0 & 3 & 2 \\ 1 & 3 & 2 & 1 & 0 \\ 2 & 0 & 2 & 1 & 3 \\ 3 & 3 & 1 & 3 & 2 \end{array}$$

$$S_1 = \begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 1 & 2 & 3 \\ 1 & 2 & 0 & 1 & 3 \\ 2 & 3 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 & 3 \end{array}$$



Operation

1st & 4th i/p bits are treated as the row of the S-box → specified by the 2-bit number.

S_0 → (1st & 4th bit) eg:- $P_{0.0}, P_{0.3} = (0\ 0) = 0$ ↓ base 2

2nd & 3rd i/p bits are treated as the coloumn of the S-box → specified by the 2-bit numbers.

(2nd & 3rd bit) → S_1

eg:- $P_{0.1}, P_{0.2} = (10) = 2$ → base 2

o/p → row 0, calum 2 of $S_0 = 3 \rightarrow 11$ in binary

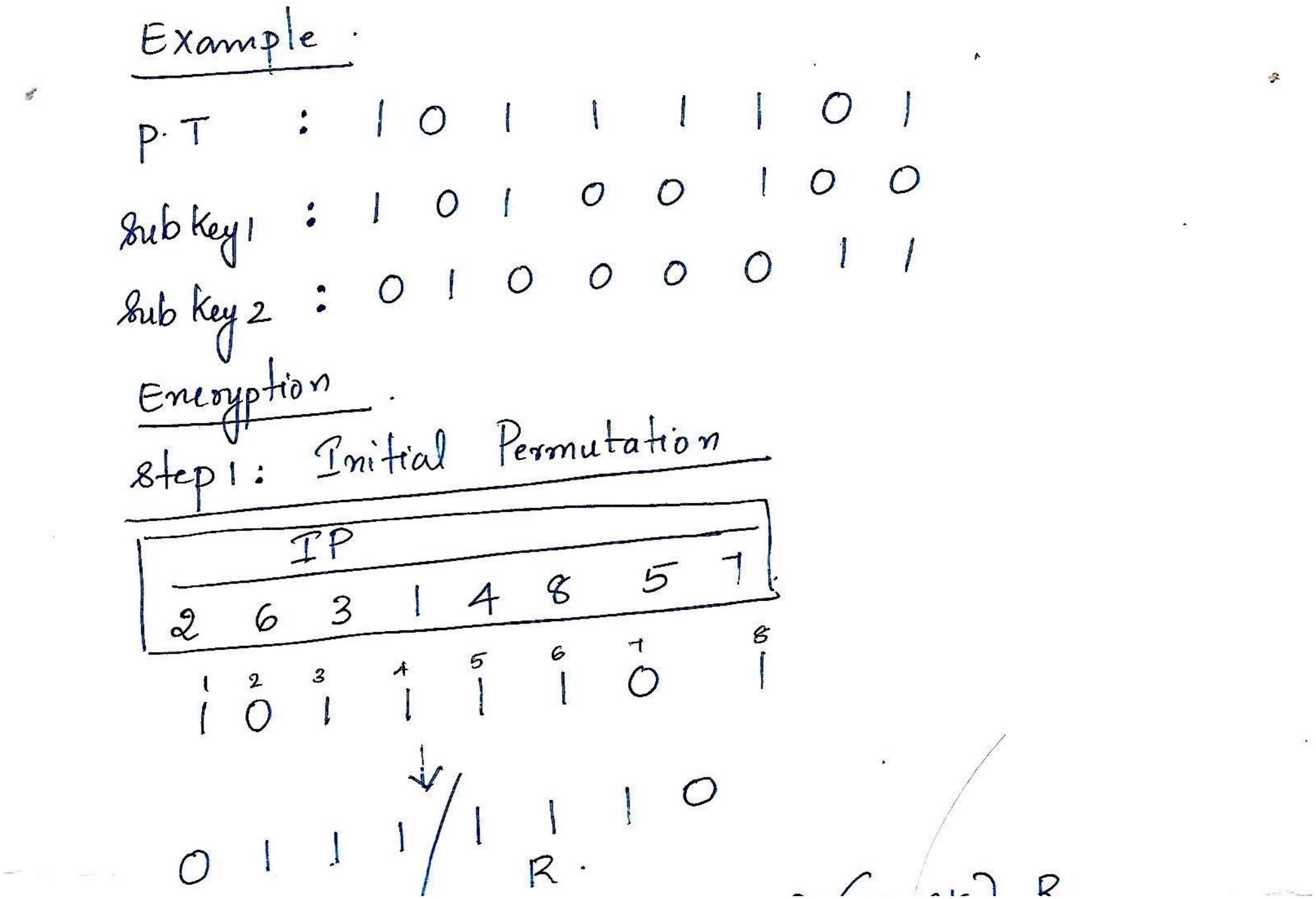
Similarly operation for S1 also.

Step 5

The 4 bits produced by S_0 & S_1 undergo a further permutation.

P_4			
2	4	3	1

- The output of p4 is the output of function F
- Switch function
  - interchanges left & right 4 bits so that the second instance of fk operates on different 4 bits



Example.

P.T : 1 0 1 1 1 1 0 1

Sub Key1 : 1 0 1 0 0 1 0 0

Sub Key 2 : 0 1 0 0 0 0 1 1

Encryption.

Step 1: Initial Permutation

IP							
2	6	3	1	4	8	5	7

1	2	3	4	5	6	7	8
1	0	1	1	1	1	0	1

↓

0 1 1 1 / 1 1 1 0

R.

L

Step 2: Function f_K mapping $f(R, SK), K$

1) Expansion / permutation

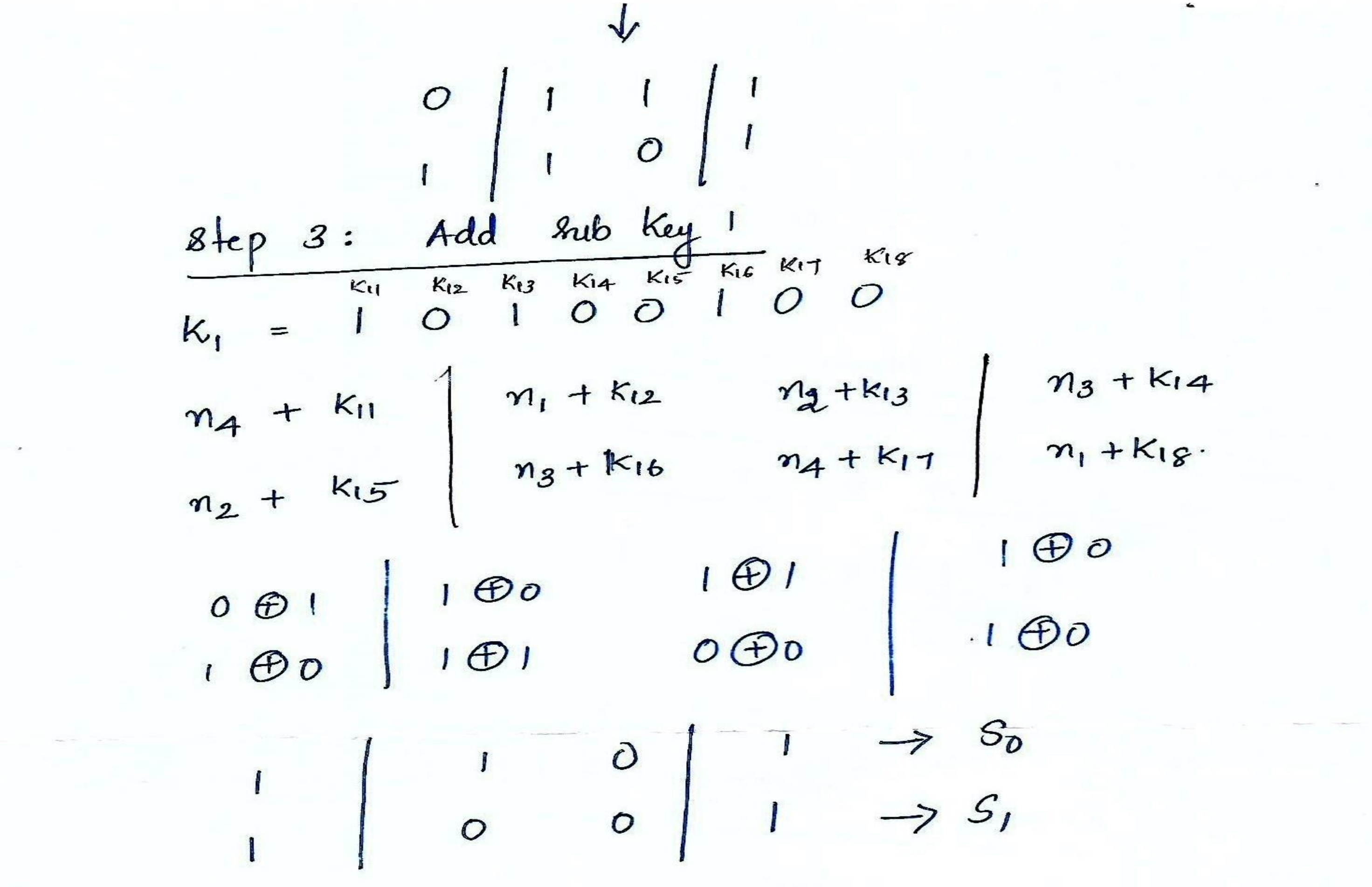
E/P							
4	1	2	3	2	3	4	1

	n_1	n_2	n_3	n_4
R →	1	1	1	0
	1	2	3	4

↓

0	1	1	1	1	1	0	1
1	2	3	4	5	6	7	8

n_4	n_1	n_2	n_3
n_2	n_3	n_4	n_1



↓

$$\left.\begin{array}{c|cc|c} 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{array}\right.$$

Step 3: Add sub Key 1

$$\begin{array}{ccccccccc} & K_{11} & K_{12} & K_{13} & K_{14} & K_{15} & K_{16} & K_{17} & K_{18} \\ K_1 = & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{array}$$

$$\left.\begin{array}{c|cc|c} n_4 + K_{11} & n_1 + K_{12} & n_2 + k_{13} & n_3 + K_{14} \\ n_2 + K_{15} & n_3 + K_{16} & n_4 + K_{17} & n_1 + K_{18}. \end{array}\right.$$

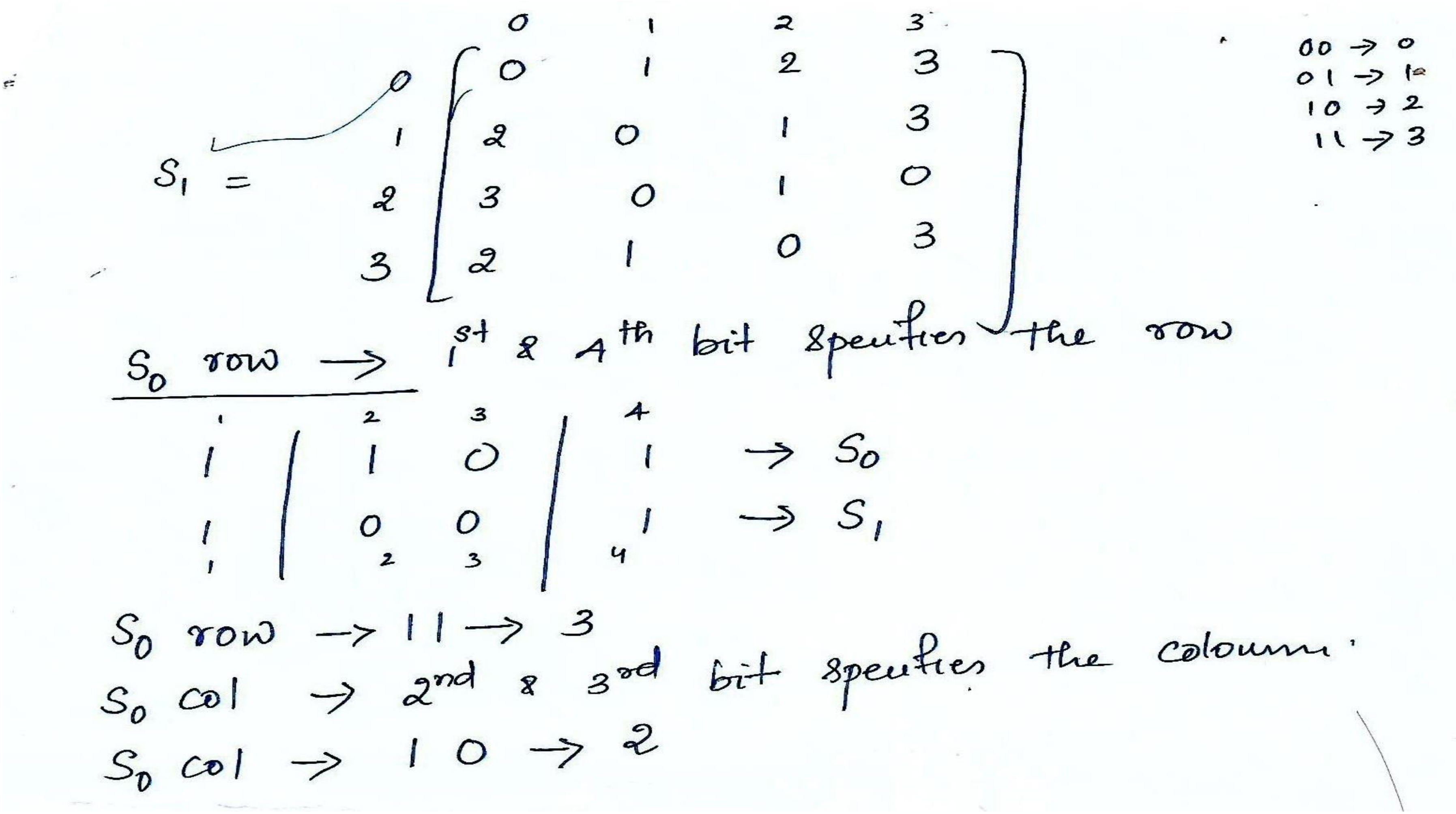
$$\left.\begin{array}{c|cc|c} 0 \oplus 1 & 1 \oplus 0 & 1 \oplus 1 & 1 \oplus 0 \\ 1 \oplus 0 & 1 \oplus 1 & 0 \oplus 0 & 1 \oplus 0 \end{array}\right.$$

$$\left.\begin{array}{c|cc|cl} 1 & 1 & 0 & 1 & \rightarrow S_0 \\ 1 & 0 & 0 & 1 & \rightarrow S_1 \end{array}\right.$$

$$\begin{array}{c|cc|c} P_{0,0} & P_{0.1} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \end{array}$$

Step 4 : S-box operation

$$S_0 = \begin{array}{cc|cccc|} & & 0 & 1 & 2 & 3 \\ 00 & 0 & 1 & 0 & 3 & 2 \\ 01 & 1 & 3 & 2 & 1 & 0 \\ 02 & 2 & 0 & 2 & 1 & 3 \\ 03 & 3 & 3 & 1 & 3 & 2 \end{array}$$



$$S_1 = \begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 1 & 2 & 3 \\ 1 & 2 & 0 & 1 & 3 \\ 2 & 3 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 & 3 \end{array}$$

00 → 0
01 → 1
10 → 2
11 → 3

S_0 row → 1st & 4th bit specifies the row

$$\begin{array}{c|cc|c} 1 & 2 & 3 & 4 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 2 & 3 & 4 \end{array} \quad \begin{array}{l} \\ \to S_0 \\ \to S_1 \\ \\ \end{array}$$

S_0 row → 11 → 3

S_0 col → 2nd & 3rd bit specifies the coloumn.

S_0 col → 1 0 → 2

S_1 row → 11 → 3

S_0 col → 00 → 0

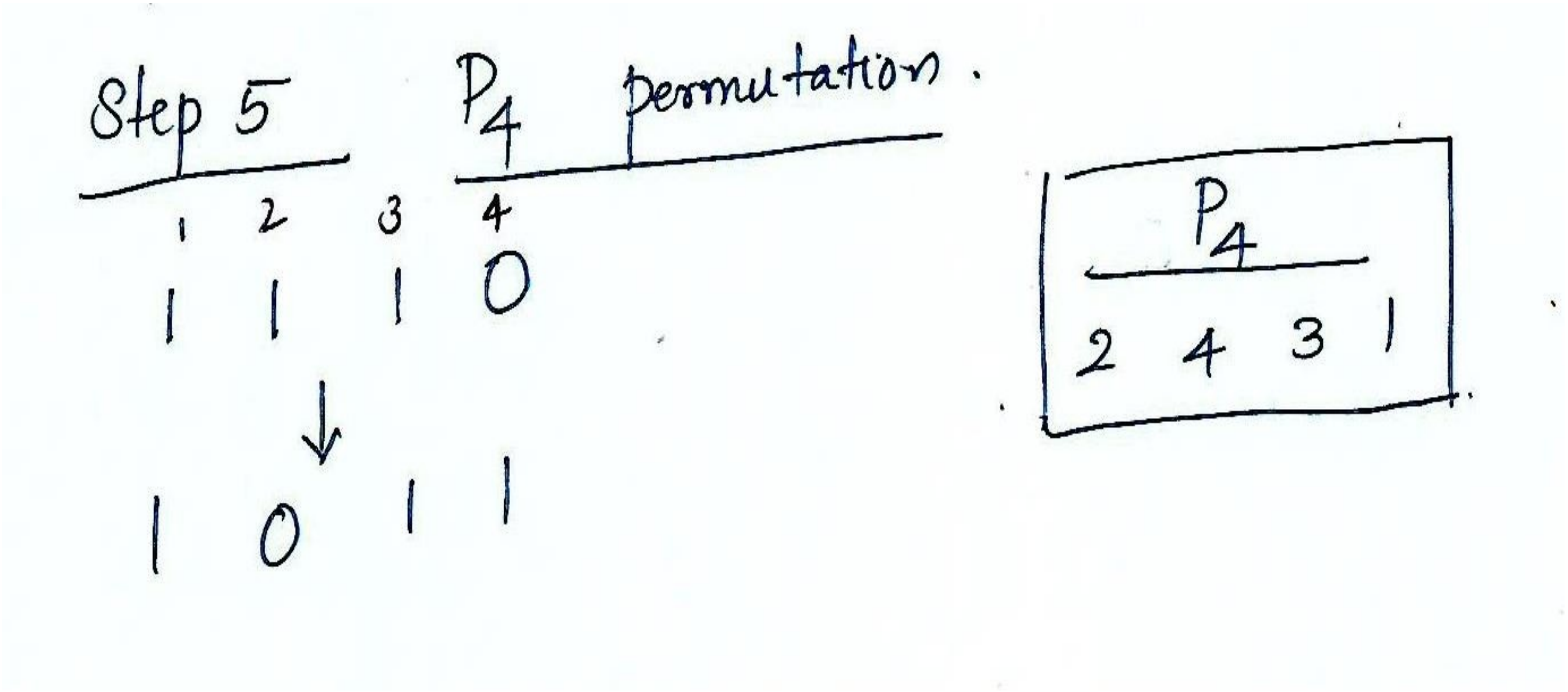
o/p of S_0 box.

row - 3, col - 2 → 3 → 11

o/p of S_1 box.

row - 3, col - 0 → 2 → 10

o/p of S box → $\underline{\underline{1110}}$



Step 5 P_4 permutation.

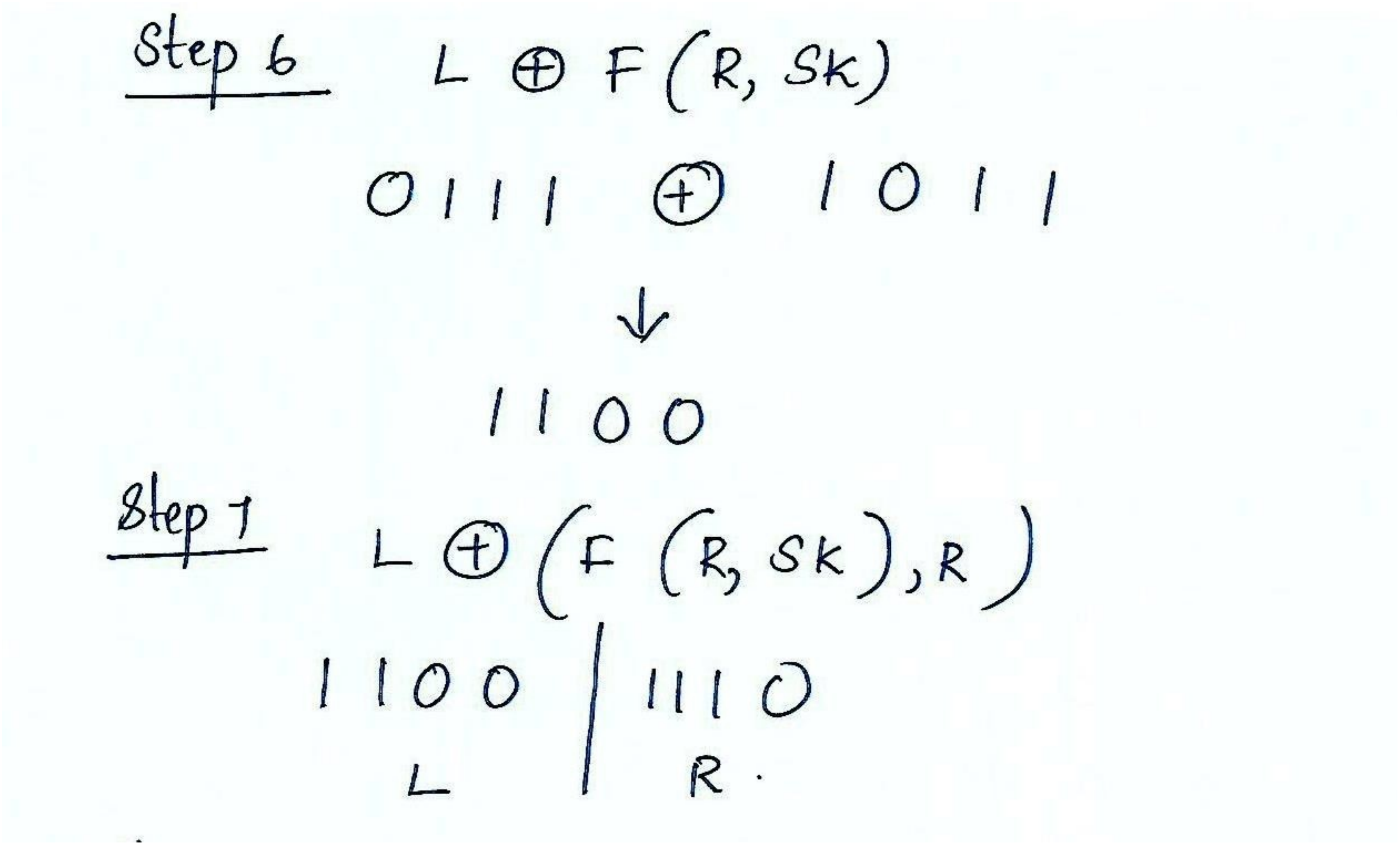
1	2	3	4
1	1	1	0

↓

1 0 1 1

P_4

2 4 3 1



<u>Step 6</u> $L \oplus F(R, SK)$

$0111 \oplus 1011$

$\downarrow$

1100

<u>Step 7</u> $L \oplus (F(R, SK), R)$

1100	1110
L	R .

Step 8

Switch function SW

1100 1110

↓

1110 1100

Step 9 E/P

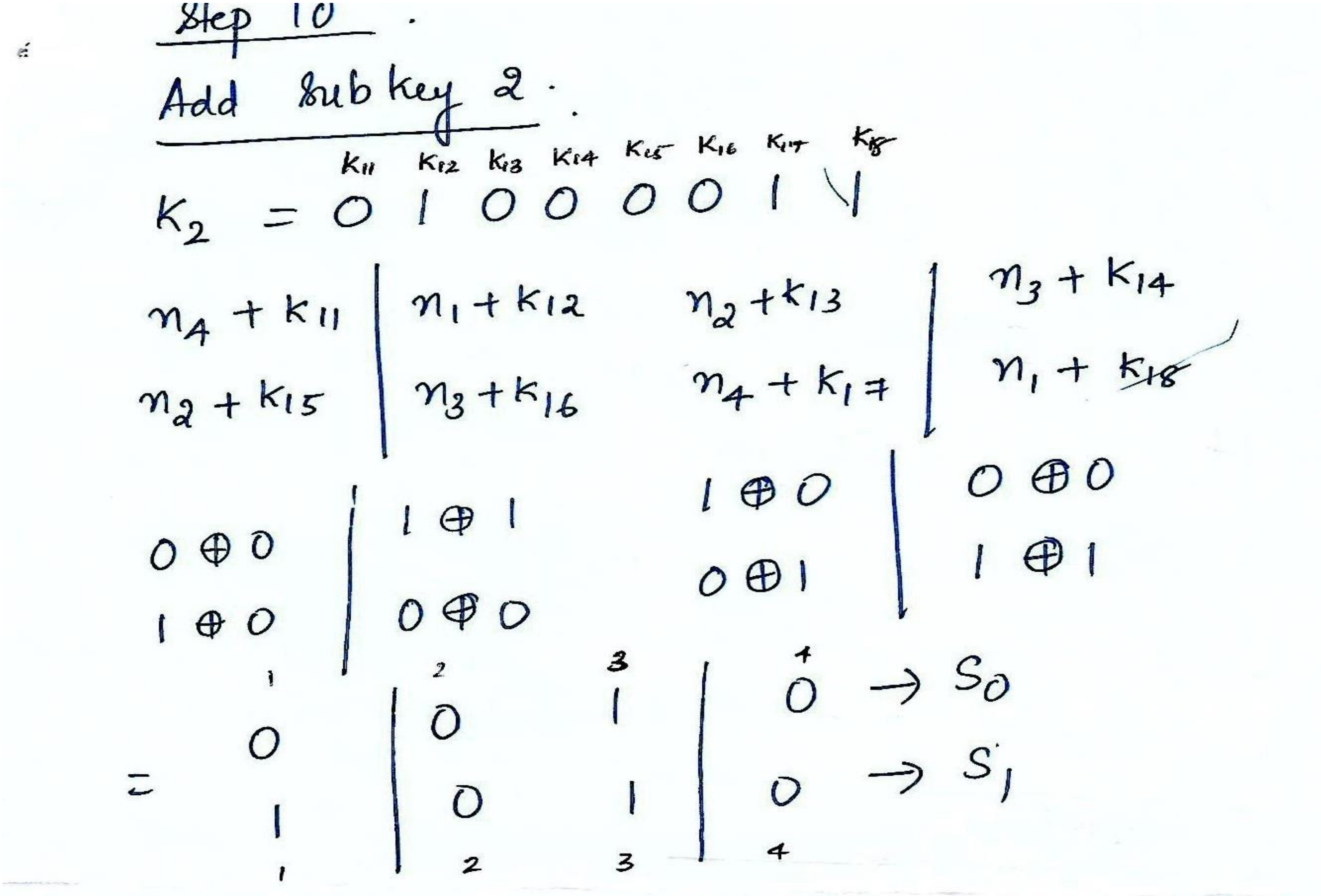
$$\boxed{\begin{array}{cccccccc} \multicolumn{8}{c}{E/P} \\ \hline 4 & 1 & 2 & 3 & 2 & 3 & 4 & 1 \end{array}}$$

$R \rightarrow \overset{1}{1}\ \overset{2}{1}\ \overset{3}{0}\ \overset{4}{0}$

0 1 1 0, 1 0 0 1

$$\begin{array}{c|cc|c} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{array}$$

$$\begin{array}{c|cc|c} n_4 & n_1 & n_2 & n_3 \\ n_2 & n_3 & n_4 & n_1 \end{array}$$



Step 10 .

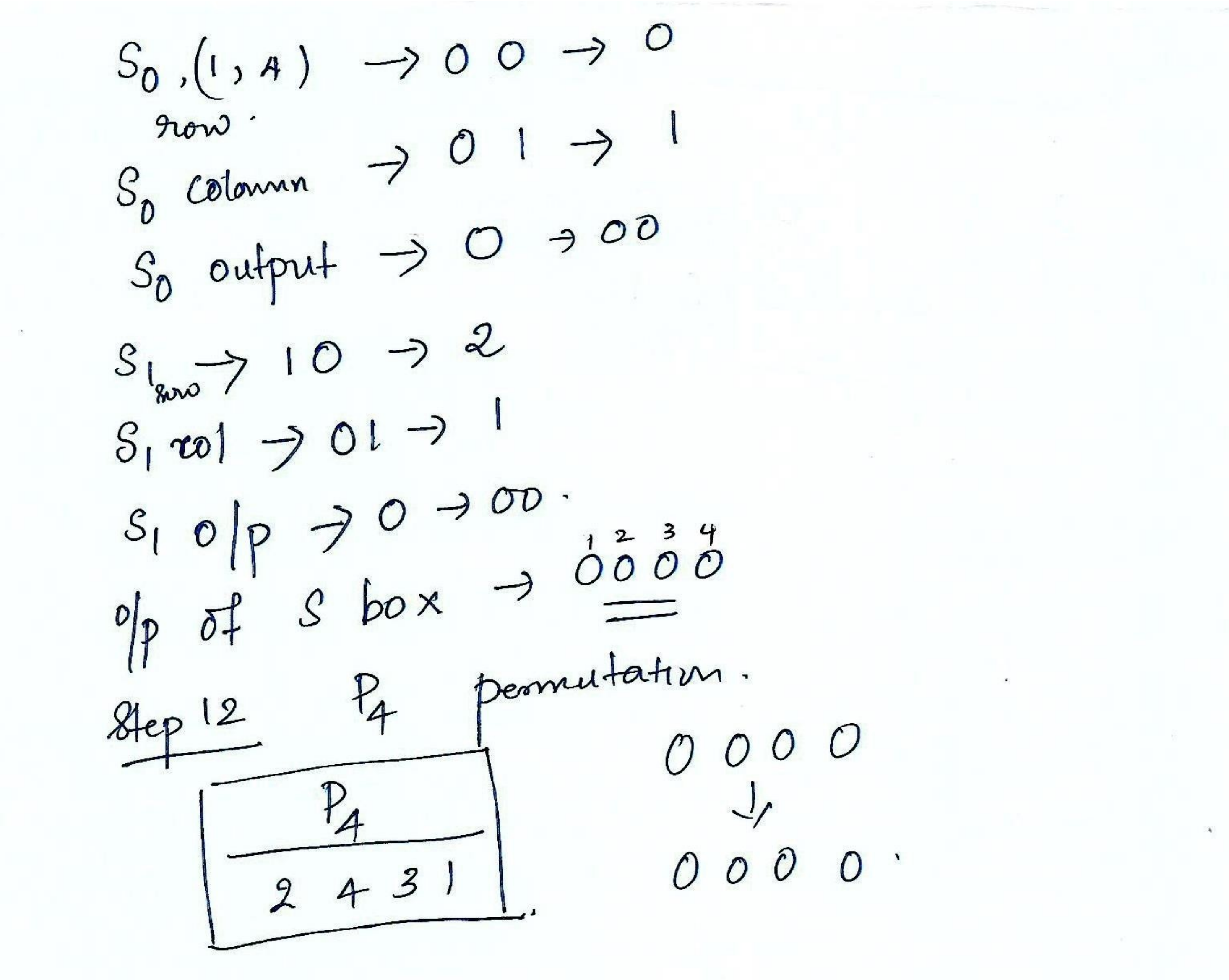
Add Subkey 2 .

$$\begin{array}{cccccccccc} & & K_{11} & K_{12} & K_{13} & K_{14} & K_{15} & K_{16} & K_{17} & K_{18} \\ K_2 & = & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}$$

$$\left.\begin{array}{c} n_4 + k_{11} \\ n_2 + k_{15} \end{array}\right| \begin{array}{cc} n_1 + k_{12} & n_2 + k_{13} \\ n_3 + k_{16} & n_4 + k_{17} \end{array} \left| \begin{array}{c} n_3 + k_{14} \\ n_1 + k_{18} \end{array}\right.$$

$$\left.\begin{array}{c} 0 \oplus 0 \\ 1 \oplus 0 \end{array}\right| \begin{array}{cc} 1 \oplus 1 & 1 \oplus 0 \\ 0 \oplus 0 & 0 \oplus 1 \end{array} \left| \begin{array}{c} 0 \oplus 0 \\ 1 \oplus 1 \end{array}\right.$$

$$= \begin{array}{c|cc|cl} 1 & 2 & 3 & 4 & \\ 0 & 0 & 1 & 0 & \rightarrow S_0 \\ 1 & 0 & 1 & 0 & \rightarrow S_1 \\ 1 & 2 & 3 & 4 & \end{array}$$



S_0 ,(1, 4) → 0 0 → 0
row.

S_0 column → 0 1 → 1

S_0 output → 0 → 00

$S_{1_{row}}$ → 1 0 → 2

S_1 col → 01 → 1

S_1 o/p → 0 → 00.

o/p of S box → $\underline{\underline{\overset{1}{0}\overset{2}{0}\overset{3}{0}\overset{4}{0}}}$

Step 12 P_4 permutation.

P_4
2 4 3 1

0 0 0 0
↓
0 0 0 0.

Step 13

$L \oplus F(R, S, K)$

$$
\begin{array}{cccc}
1 & 1 & 1 & 0 \quad \oplus \\
0 & 0 & 0 & 0 \\
\hline
1 & 1 & 1 & 0
\end{array}
$$

Step 14

$F(R, SK), R$.

1110 1100

Step 15

Inverse permutation, IP^{-1}

1	2	3	4	5	6	7	8
1	1	1	0	1	1	0	0

IP^{-1}							
4	1	3	5	7	2	8	6

↓

0 1 1 1 0 1 0 1

C.T → 01110101

Analysis of S-DES

- Brute-force attack on S-DES is certainly feasible
- With a 10 bit key, there are only 2^{10} possiblities

Relationship to DES

- DES operates on 64 bit blocks of input

The three critical aspects of block cipher design are:

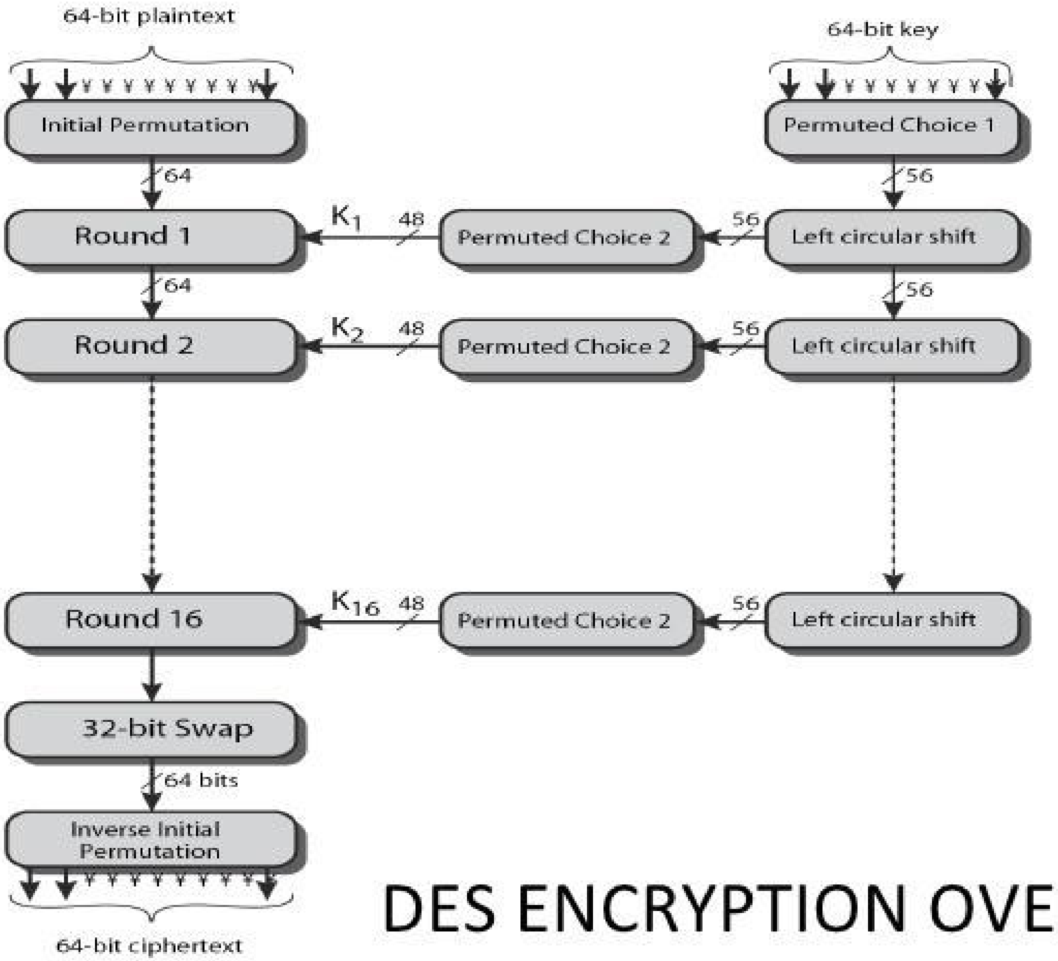
- **The number of rounds**
- **Design of the function f**
- **Key scheduling**

Number of Rounds

- The greater the number of rounds, the more difficult it is to perform cryptanalysis,even for a relatively weak F.
- the number of rounds is chosen so that known cryptanalytic efforts require greater effort than a simple brute-force key search attack.

THE DATA ENCRYPTION STANDARD

- data are encrypted in 64-bit blocks
- 56-bit key.
- The algorithm transforms 64-bit input in a series of steps into a 64-bit output.
- The same steps, with the same key, are used to reverse the encryption.



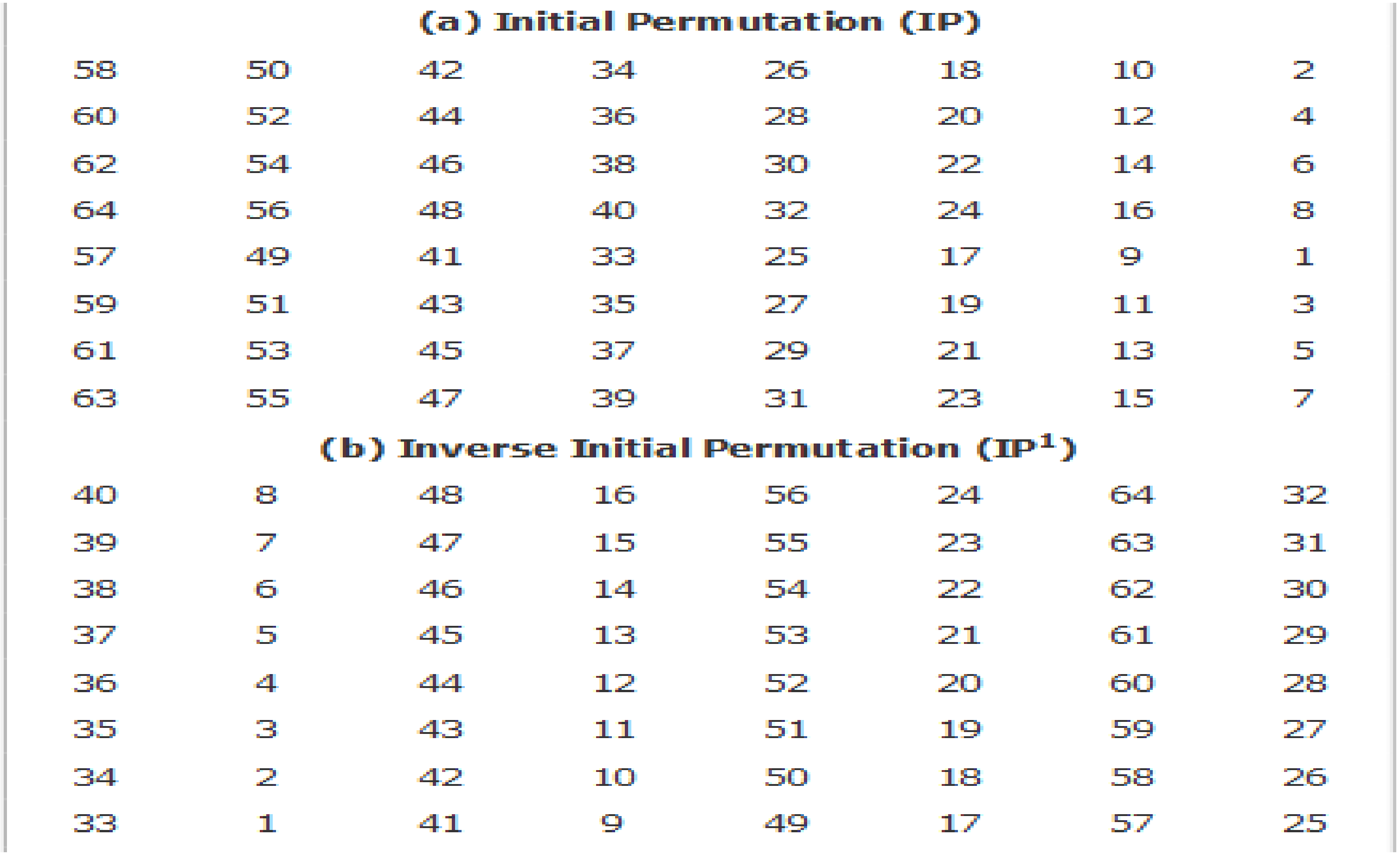
DES ENCRYPTION OVERVIEW

Processing of Plaintext

- At the left-hand side of the figure, the processing of the plaintext proceeds

in **three phases.**

- ❖ First, the 64-bit plaintext passes through an initial permutation (IP) that rearranges the bits to produce the permuted input**.**
- ❖ This is followed by a phase consisting of sixteen rounds of the same function, which involves both permutation and substitution functions.
- ❖ The output of the last (sixteenth)round consists of 64 bits that are a function of the input plaintext and the key.
- ❖ The left and right halves of the output are swapped to produce the preoutput.
- ❖ Finally, the preoutput is passed through the inverse of the initial permutation function, to produce the 64-bit C.T



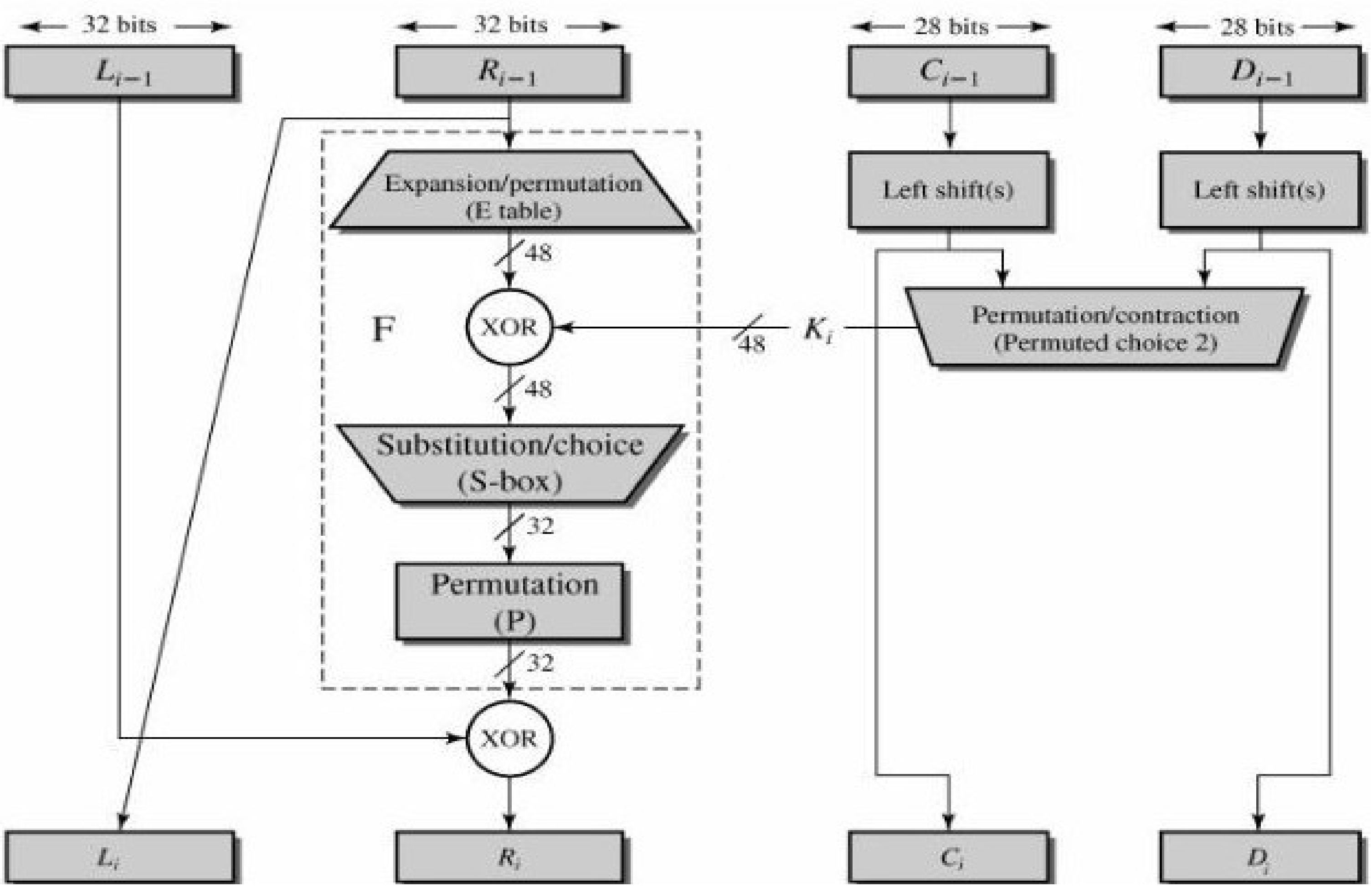
(a) Initial Permutation (IP)

58	50	42	34	26	18	10	2
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

(b) Inverse Initial Permutation (IP^{1})

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

Details of Single Round

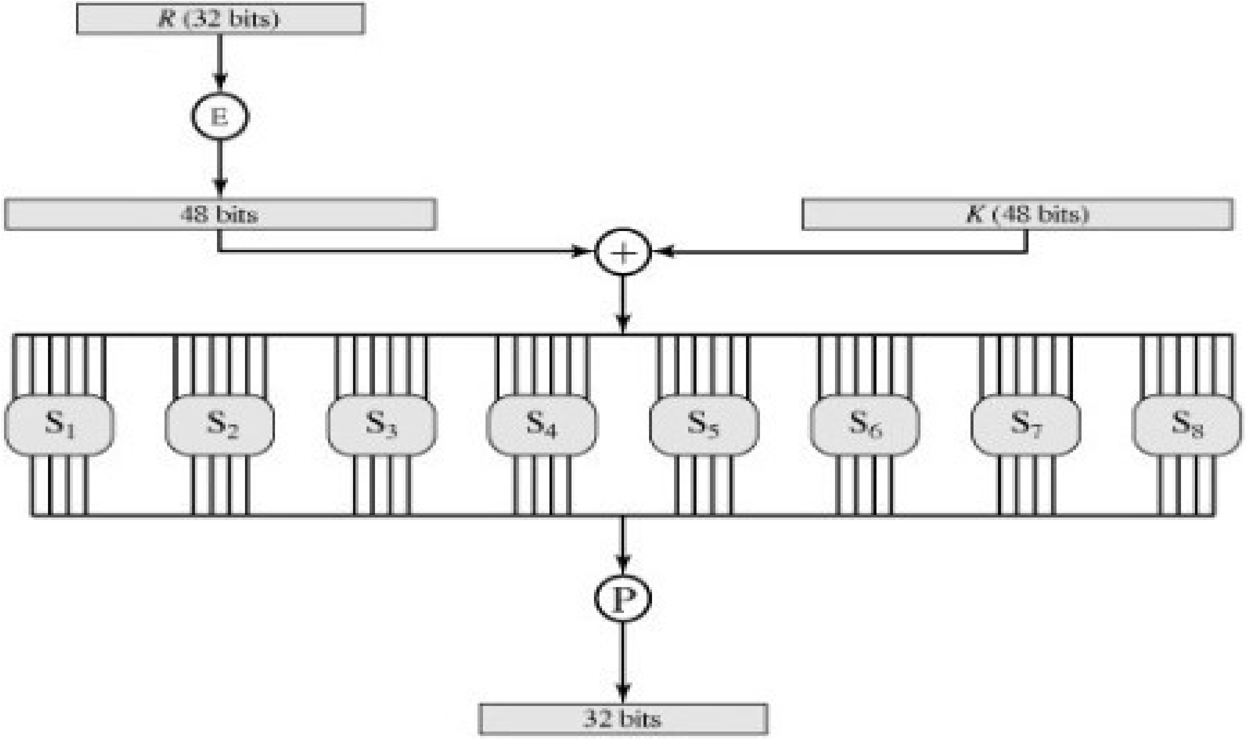


The overall processing at each round can be summarized in the following formulas:

$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \text{ x } F(R_{i-1}, K_i)$$

The role of the S-boxes in the function F is illustrated in Figure below



The Avalanche Effect

- A desirable property of any encryption algorithm is that a small change in either the plaintext or the key should produce a significant change in the ciphertext.
- In particular, a change in one bit of the plaintext or one bit of the key should produce a change in many bits of the ciphertext.

- For example, in S1 for input 011001, the row is 01 (row 1) and the column is 1100 (column 12). The value in row 1, column 12 is 9, so the output is 1001.

THE STRENGTH OF DES

- Use of 56-Bit Keys
- brute-force attack appears impractical.
- With current technology, it is not even necessary to use special purpose built in hardware.
- Rather, the speed of commercial processors threatens the security of DES.

The time required fora brute-force attack for various key sizes:

- A single PC can break DES in about a year.
- If multiple PCs work in parallel, the time is drastically shortened.
- Today's super computers should be able to find a key in about an hour.
- Key sizes of 128 bits or greater are effectively unbreakable using simply a brute force approach.

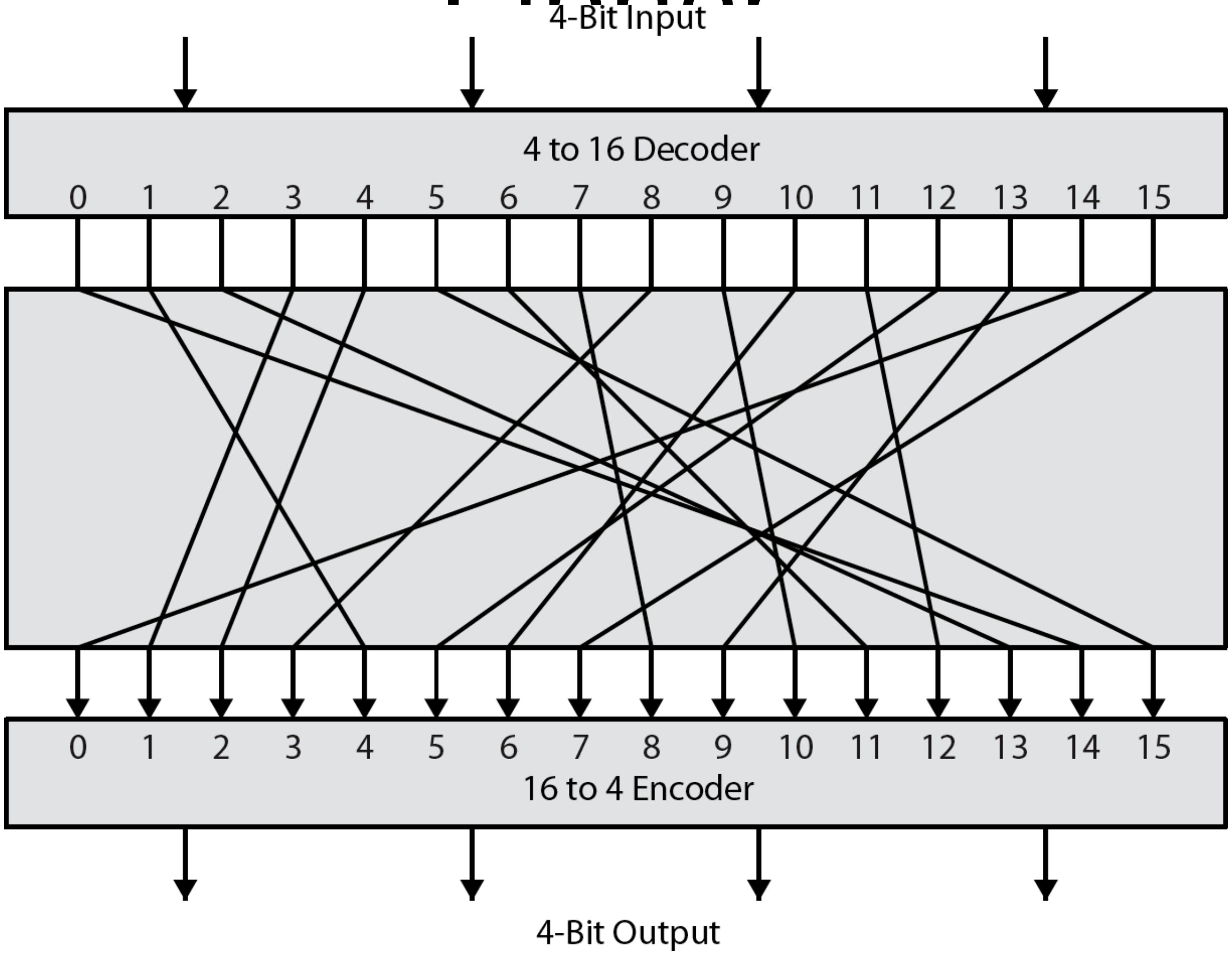
Block Cipher Principles

Block Cipher Principles

- most symmetric block ciphers are based on a Feistel Cipher Structure
- needed since must be able to decrypt ciphertext to recover messages efficiently
- block ciphers look like an extremely large substitution
- would need table of 2^{64} entries for a 64-bit block
- instead create from smaller building blocks
- using idea of a product cipher

Ideal Block Cipher

Ideal Block Cipher



permutation