S-DES KEY Generation

- input ⇒ 10 bit key
- output ⇒ two 8 bit key

Steps:

1º P10 ⇒ Permutation of 10 bits

1	2	3	4	5	6	7	8	9	10
P10									
3	5	2	7	4	10	1	9	8	6

example :

1 0 1 0 0 0 0 0 1 0

⇓ P10

1 0 0 0 0 0 0 1 1 0

2º LS-1 ⇒ Divide the previous output by two, which will result in two blocks of 5 bits and then perform a left circular shift in both.

example :

$\boxed{1\ 0\ 0\ 0\ 0}\ \boxed{0\ 1\ 1\ 0\ 0}$

⇓ LS-1 ⇓ LS-1

$\boxed{0\ 0\ 0\ 0\ 1}\ \boxed{1\ 1\ 0\ 0\ 0}$

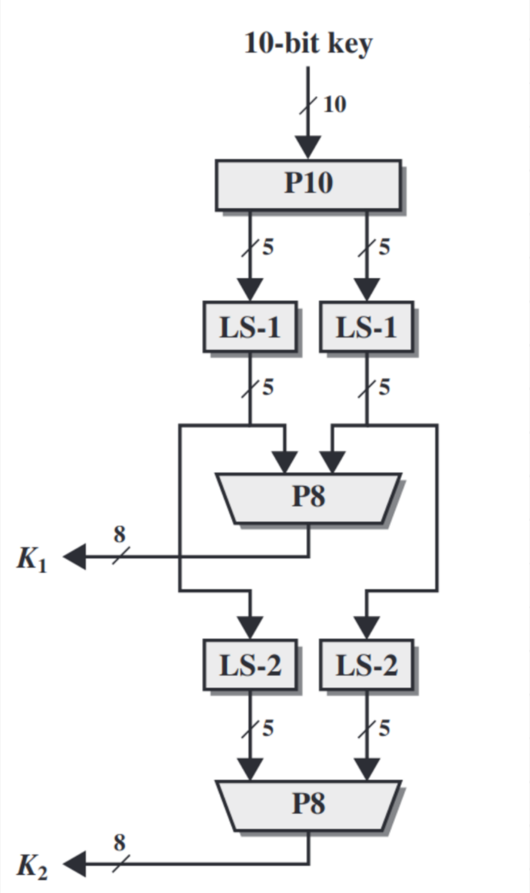
3º P8 ⇒ Permutation of 10 bits in which 8 are "selected"

P8							
6	3	7	4	8	5	10	9

example : 0 0 0 0 1 1 1 0 0 0

$\Downarrow$ P8

1 0 1 0 0 1 0 0 (subkey 1)



Encryption:

- input = 8 bit plaintext
- output: 8 bit ciphertext

Steps:

Initial and Final Permutation

1º It is very similar with P8.

IP							
2	6	3	1	4	8	5	7

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

example:

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

⇓ ip

plaintext ⇐ 1 1 0 1 1 1 0 1

🔵 left part

🔴 right part

2º FK Function

First divide the input in two parts

1101 1101

E/P expansion permutation

· take the left part and do the E/P

E/P							
4	1	2	3	2	3	4	1

example:

$$\begin{array}{cccc} 1 & 2 & 3 & 4 \\ 1 & 1 & 0 & 1 \end{array}$$

$\Downarrow$ E/P

$$\begin{array}{r} \boxed{1\,1\,1\,0}\,\boxed{1\,0\,1\,1} \\ \underline{1\,0\,1\,0\,0\,1\,0\,0} \\ 0\,1\,0\,0\,1\,1\,1\,1 \end{array} \quad \oplus \text{ subkey } 1$$

- divide both parts again, each part will be feed to a S-box

$$\begin{array}{cccc} 0 & 1 & 0 & 0 \end{array} \qquad\qquad \begin{array}{cccc} 1 & 1 & 1 & 1 \end{array}$$

$$S_0 = \begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 1 & 0 & 3 & 2 \\ 1 & 3 & 2 & 1 & 0 \\ 2 & 0 & 2 & 1 & 3 \\ 3 & 3 & 1 & 3 & 2 \end{array} \Rightarrow 11 \qquad S_1 = \begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 1 & 2 & 3 \\ 1 & 2 & 0 & 1 & 3 \\ 2 & 3 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 & 3 \end{array} \Rightarrow 11$$

$\Downarrow$ $\qquad\qquad\qquad\qquad$ $\Downarrow$

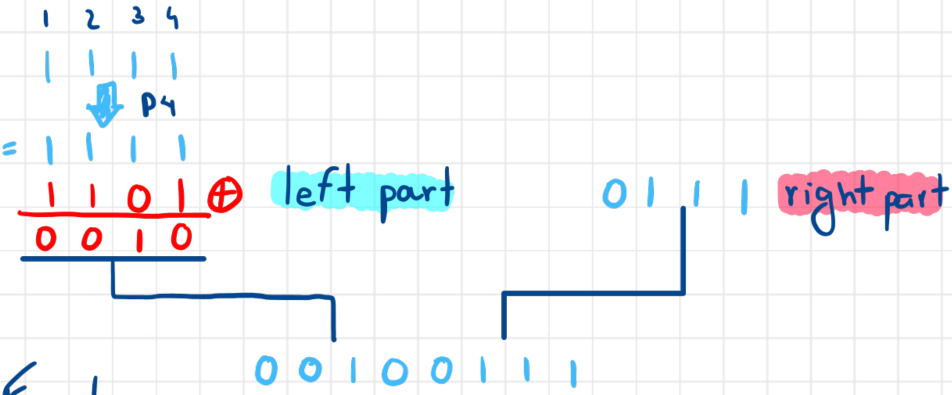
These values in the S's boxes are set arbitrary

- output = 1 1 1 1

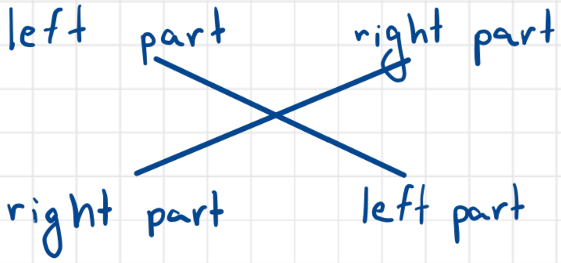
  - pass another permutation

P4			
2	4	3	1

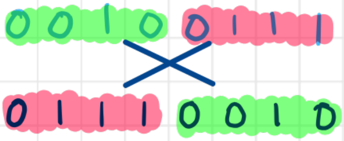
example :

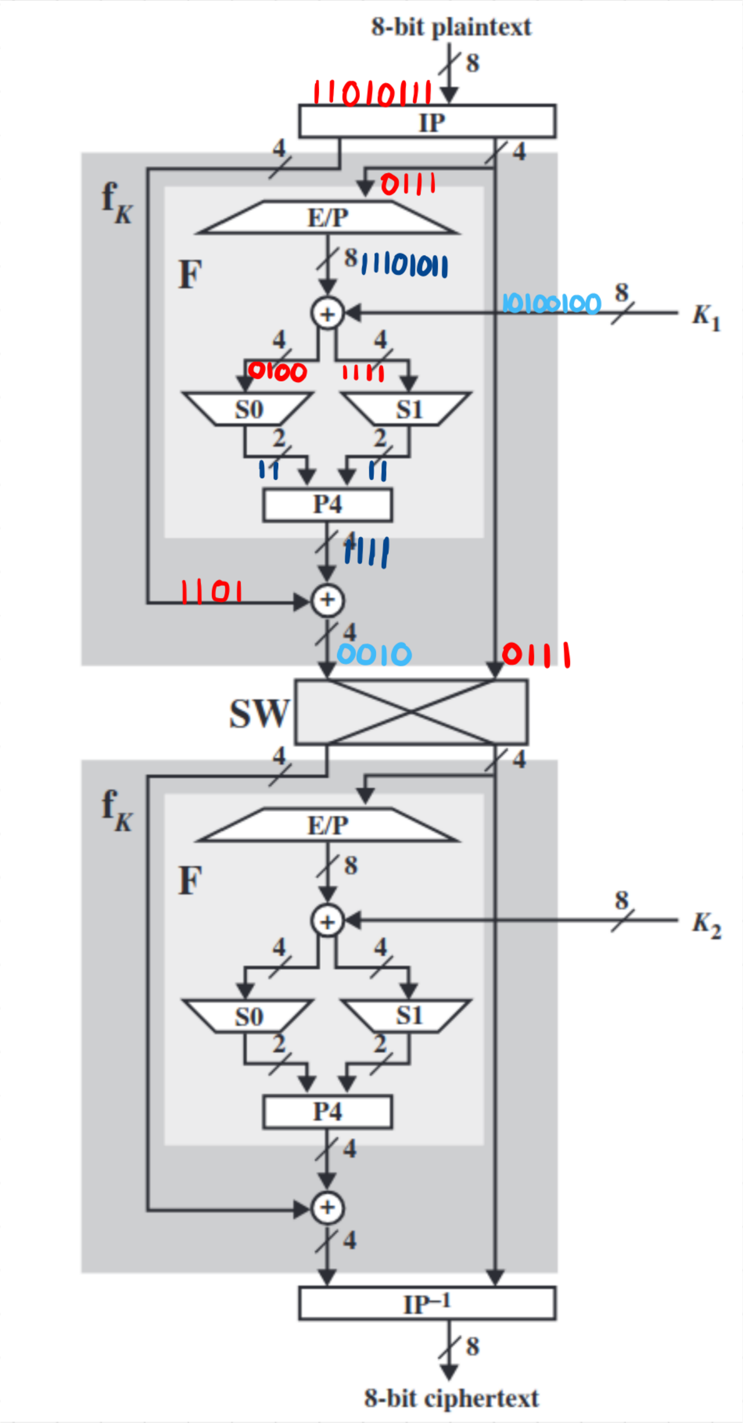


Switch Function

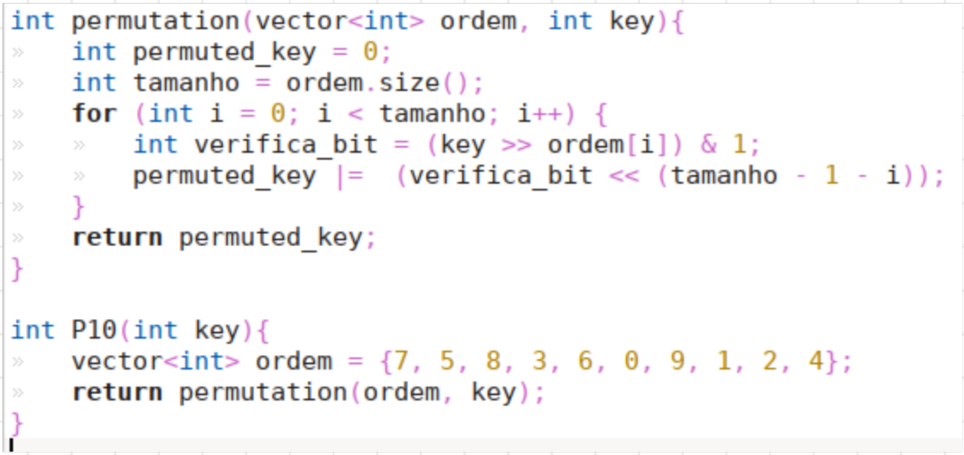


example :




8-bit plaintext
8
11010111
IP
4
4
0111
f_K
E/P
F
8
11101011
10100100
8
K_1
4
4
0100
1111
S0
S1
2
2
11
11
P4
4
1111
1101
4
0010
0111
SW
4
4
f_K
E/P
F
8
8
K_2
4
4
S0
S1
2
2
P4
4
4
IP–1
8
8-bit ciphertext

Observação:



```
int permutation(vector<int> ordem, int key){
    int permuted_key = 0;
    int tamanho = ordem.size();
    for (int i = 0; i < tamanho; i++) {
        int verifica_bit = (key >> ordem[i]) & 1;
        permuted_key |=  (verifica_bit << (tamanho - 1 - i));
    }
    return permuted_key;
}

int P10(int key){
    vector<int> ordem = {7, 5, 8, 3, 6, 0, 9, 1, 2, 4};
    return permutation(ordem, key);
}
```

- para a implementação em c++ a ordem foi "alterada", isso se dá pelo fato de que no livro os índices dos bits é diferente no c++.

exemplo:

Livro

1	2	3	4	5	6	7	8	9	10
1	0	1	0	0	0	0	0	1	0

c++

9	8	7	6	5	4	3	2	1	0
1	0	1	0	0	0	0	0	1	0

P10									
3	5	2	7	4	10	1	9	8	6

mapeando se torna

7 5 8 3 6 0 9 1 2 4