ICS 1

SDES

Gayatri Dhekane 41323

In [46]:

```
P10 = (3, 5, 2, 7, 4, 10, 1, 9, 8, 6)
P8 = (6, 3, 7, 4, 8, 5, 10, 9)
P4 = (2, 4, 3, 1)

IP = (2, 6, 3, 1, 4, 8, 5, 7)
IPi = (4, 1, 3, 5, 7, 2, 8, 6)

E = (4, 1, 2, 3, 2, 3, 4, 1)

S0 = [
        [1, 0, 3, 2],
        [3, 2, 1, 0],
        [0, 2, 1, 3],
        [3, 1, 3, 2]
     ]

S1 = [
        [0, 1, 2, 3],
        [2, 0, 1, 3],
        [3, 0, 1, 0],
        [2, 1, 0, 3]
     ]

def permutation(pattern, key):
    permuted = ""

    for i in pattern:
        permuted += key[i-1]

    return permuted

def generate_first(left, right):
    left = left[1:] + left[:1]
    right = right[1:] + right[:1]
    key = left + right

    return permutation(P8, key)

def generate_second(left, right):
    left = left[3:] + left[:3]
    right = right[3:] + right[:3]
    key = left + right

    return permutation(P8, key)
```

In [47]:

```python

def transform(right, key):
    extended = permutation(E, right)
    xor_cipher = bin(int(extended, 2) ^ int(key, 2))[2:].zfill(8)
    xor_left = xor_cipher[:4]
    xor_right = xor_cipher[4:]
    print("Xor left",xor_left)
    print("Xor right",xor_right)

    new_left = Sbox(xor_left, S0)
    new_right = Sbox(xor_right, S1)
    print(new_left[2:],new_right[2:])
    #print("cipher from fk1",permutation(P4, new_left[2:] + new_right[2:]))

    return permutation(P4, new_left[2:] + new_right[2:])

def Sbox(data, box):
    row = int(data[0] + data[3], 2)
    column = int(data[1] + data[2], 2)

    return bin(box[row][column])[2:].zfill(4)

def encrypt(left, right, key):
    cipher = int(left, 2) ^ int(transform(right, key), 2)

    return right, bin(cipher)[2:].zfill(4)
```

In [48]:

```
key = input("Enter a 10-bit key: ")
if len(key) != 10:
    raise Exception("Check the input")

plaintext = input("Enter 8-bit plaintext: ")
if len(plaintext) != 8:
    raise Exception("Check the input")
#key generation
p10key = permutation(P10, key)
print("First Permutation")
print(p10key)
left_key = p10key[:len(p10key)//2]
print("Left key",left_key)
right_key = p10key[len(p10key)//2:]
print("Right key",right_key)
first_key = generate_first(left_key, right_key)
print("*****")
print("First key")
print(first_key)
second_key = generate_second(left_key, right_key)
print("*****")
print("Second key")
print(second_key)
initial_permutation = permutation(IP, plaintext)
print("Initial Permutation",initial_permutation)
left_data = initial_permutation[:len(initial_permutation)//2]
right_data = initial_permutation[len(initial_permutation)//2:]

left, right = encrypt(left_data, right_data, first_key)
print(left,right)
left, right = encrypt(left, right, second_key)
print(left,right)
print("Ciphertext:", permutation(IPi, left + right))
```

```
Enter a 10-bit key: 1010000010
Enter 8-bit plaintext: 10010111
First Permutation
1000001100
Left key 10000
Right key 01100
*****
First key
10100100
*****
Second key
01000011
Initial Permutation 01011101
Xor left 0100
Xor right 1111
11 11
cipher from fk1 1111
1101 1010
Xor left 0001
Xor right 0110
11 11
cipher from fk1 1111
1010 0010
Ciphertext: 01101000
```

In [49]:

```
# key: 1010000010
# 10100100 01000011
# PT: 10010111
```