

MODULE-II

SYMMETRIC KEY CIPHER

- Block Cipher principles, DES, AES, Blowfish, RC5, IDEA, Block cipher operation, Stream ciphers, RC4.

ASYMMETRIC KEY CIPHERS

- Principles of public key cryptosystems, RSA algorithm, Elgamal Cryptography, Diffie-Hellman Key Exchange, Knapsack Algorithm.

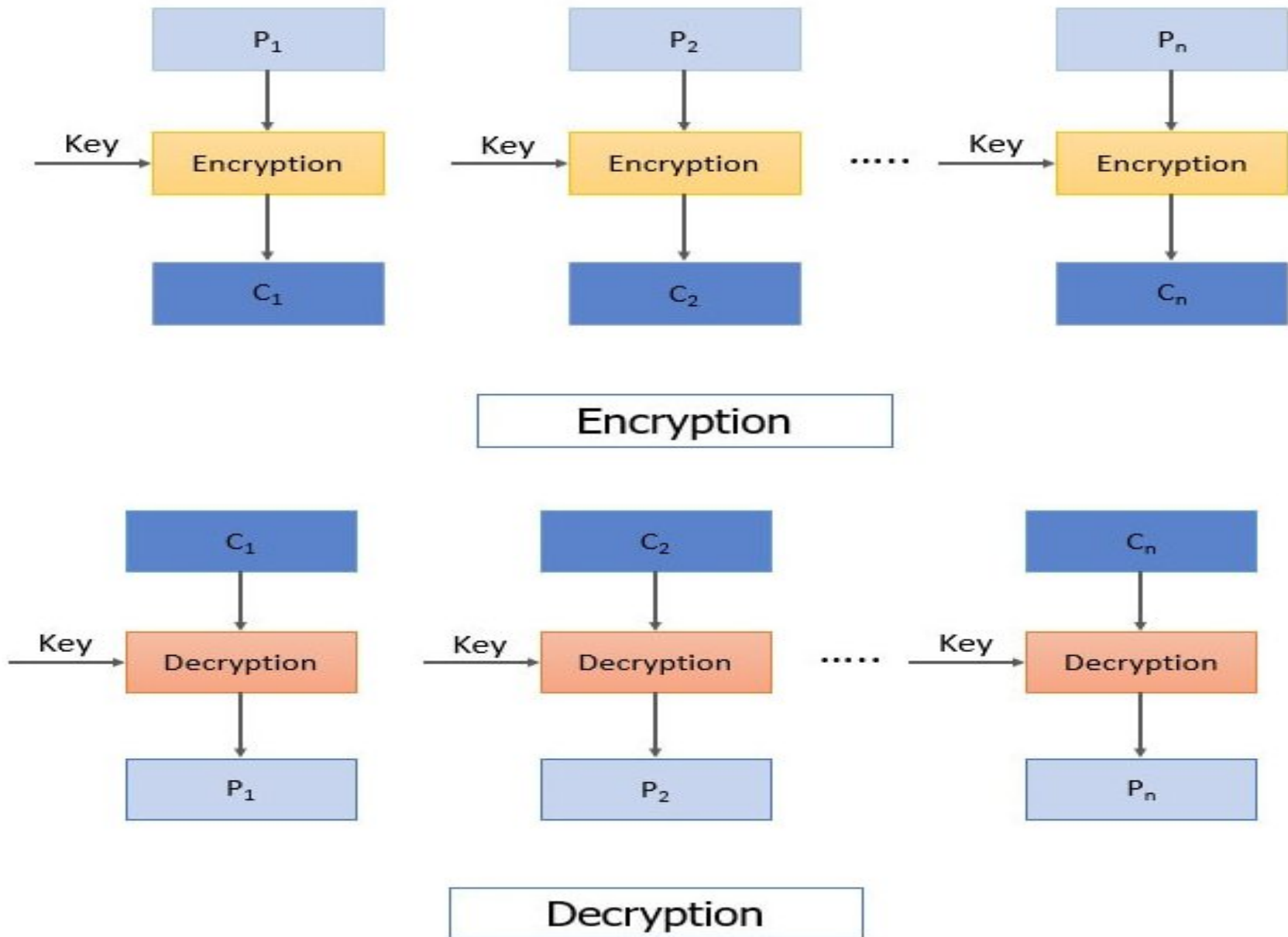
Block Cipher principles

- Block cipher is an encryption method which divides the plain text into blocks of fixed size. Each block has an equal number of bits. At a time, block cipher operates only on one block of plain text and applies key on it to produce the corresponding block of ciphertext.
- **Block Cipher Principles**
 - Number of Rounds
 - Design of Function F
 - Key Schedule Algorithm

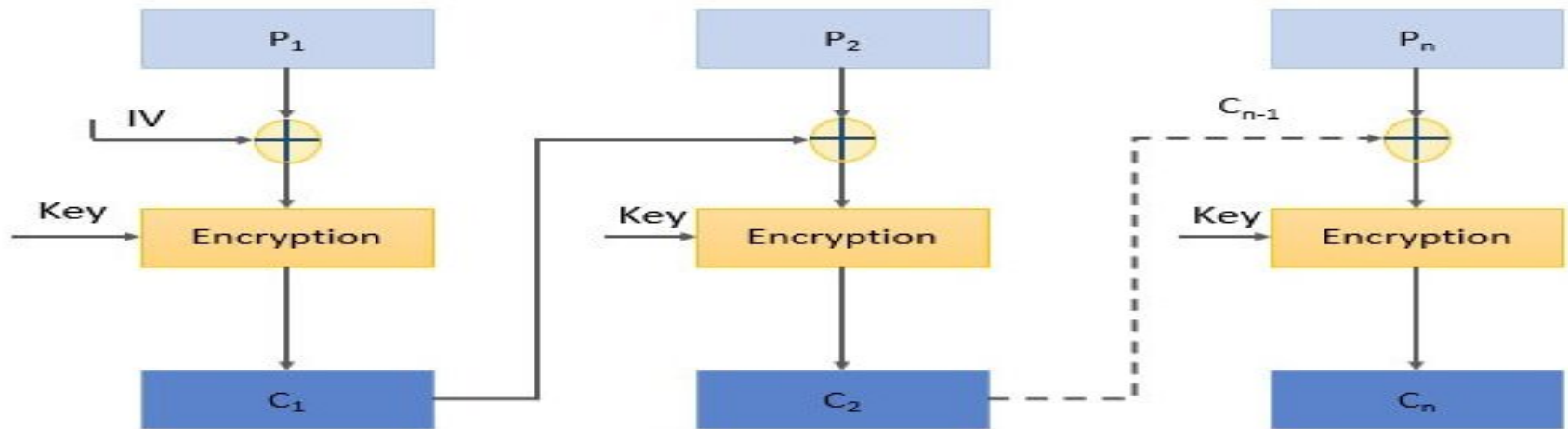
Cipher Block modes of Operation

- **Electronic Code Book (ECB)**
- **Cipher Block Chaining Mode (CBC)**
- **Cipher Feed Back Mode (CFB)**
- **Output Feedback Mode (OFB)**
- **Counter Mode (CTR)**

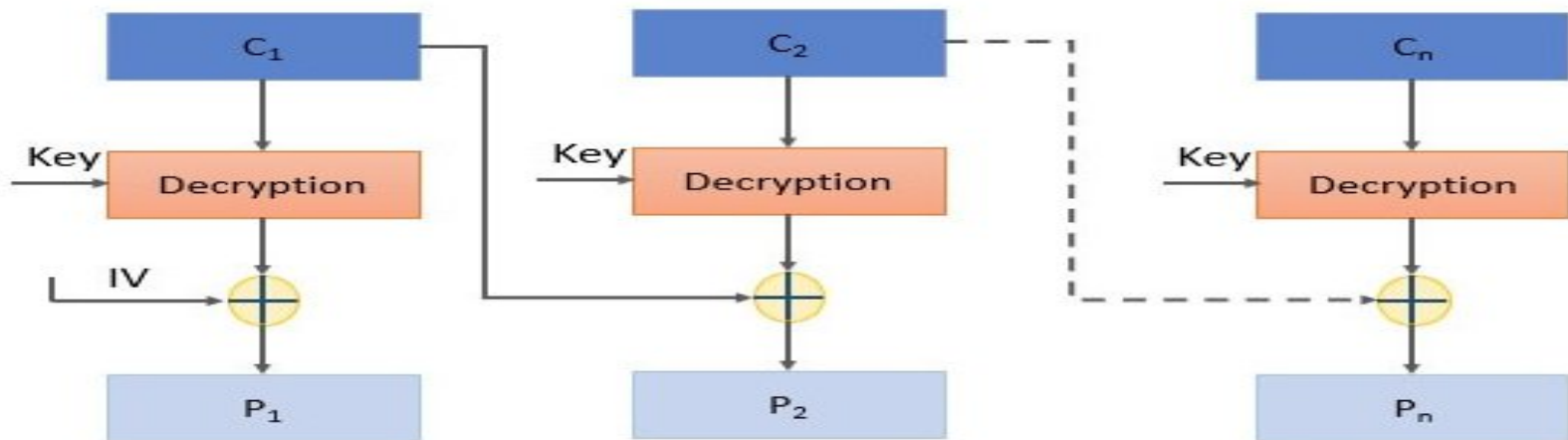
Electronic Code Book (ECB)



Cipher Block Chaining Mode (CBC)

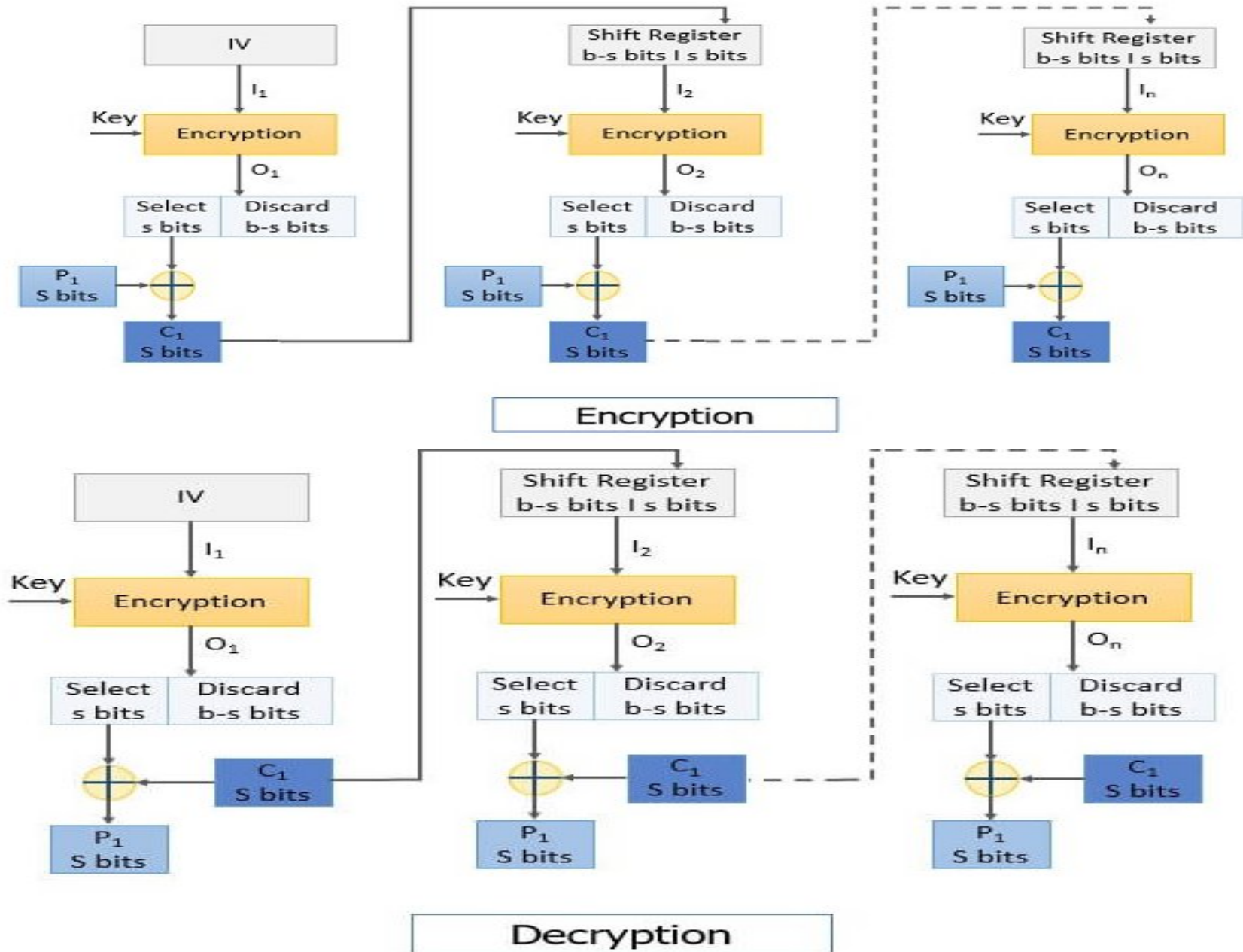


Encryption

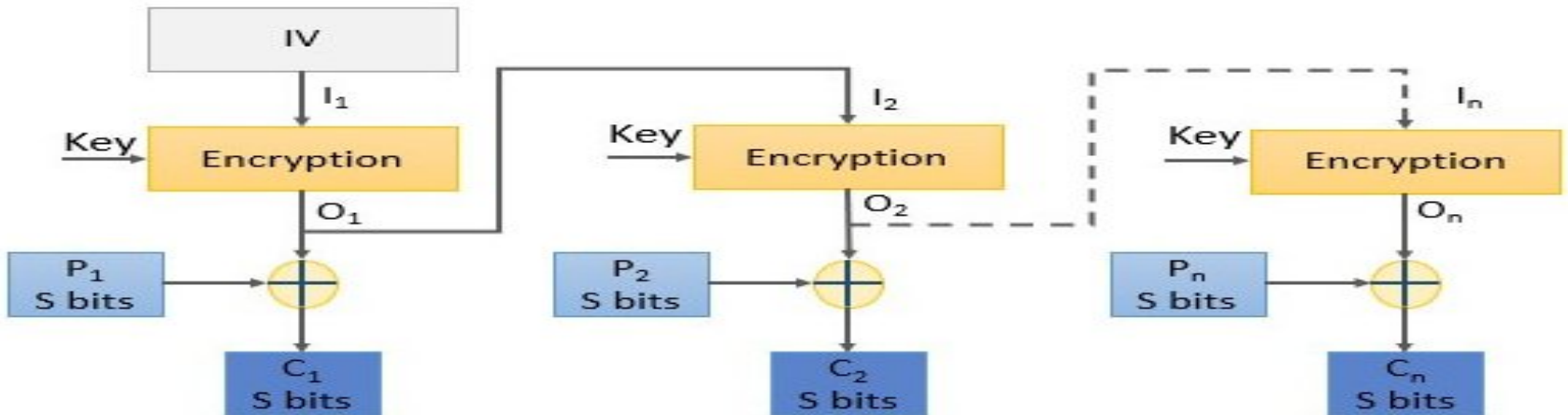


Decryption

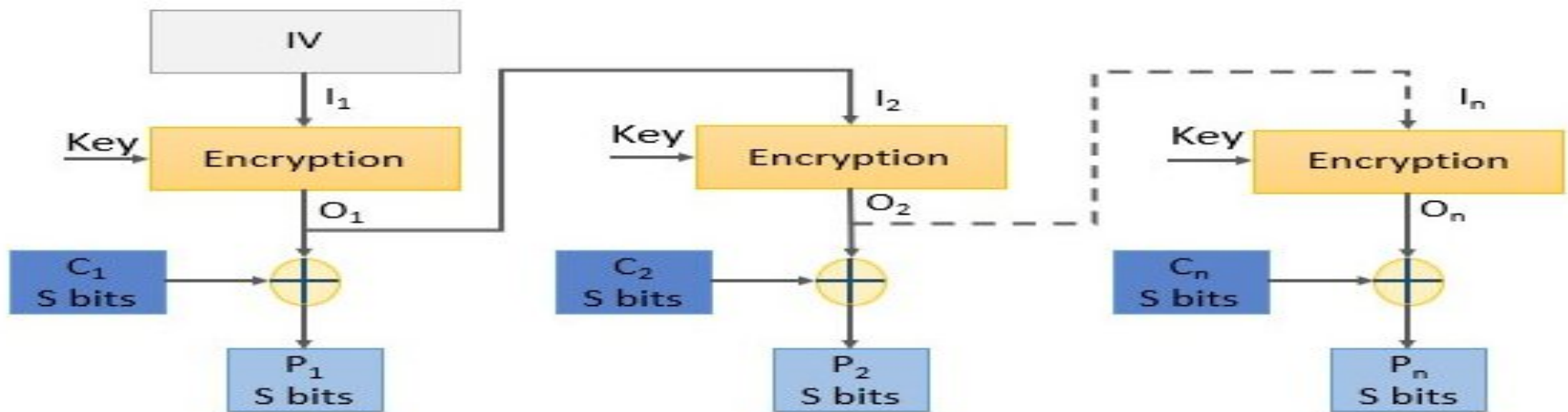
Cipher Feed Back Mode (CFB)



Output Feedback Mode (OFB)

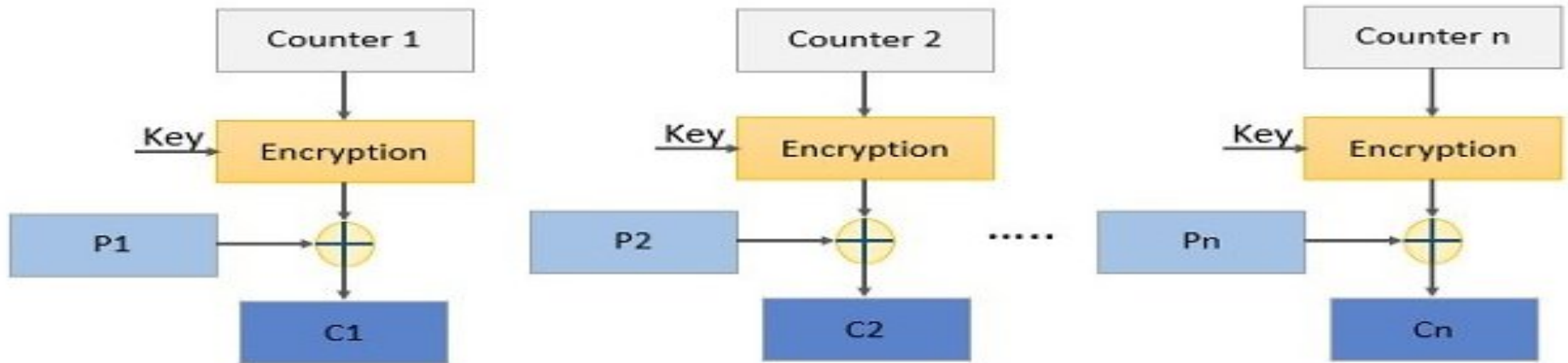


Encryption

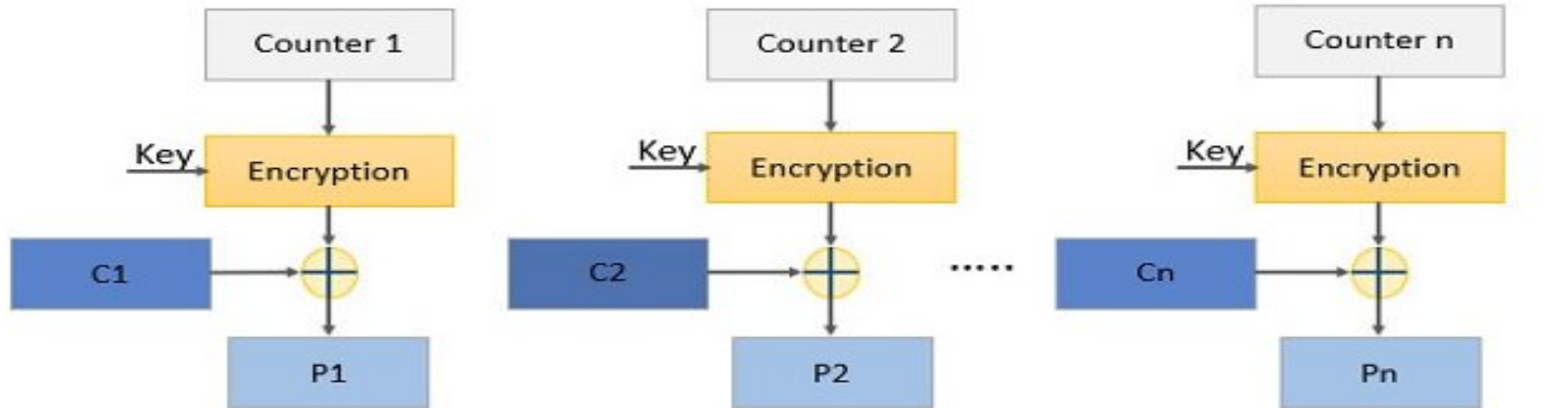


Decryption

Counter Mode (CTR)

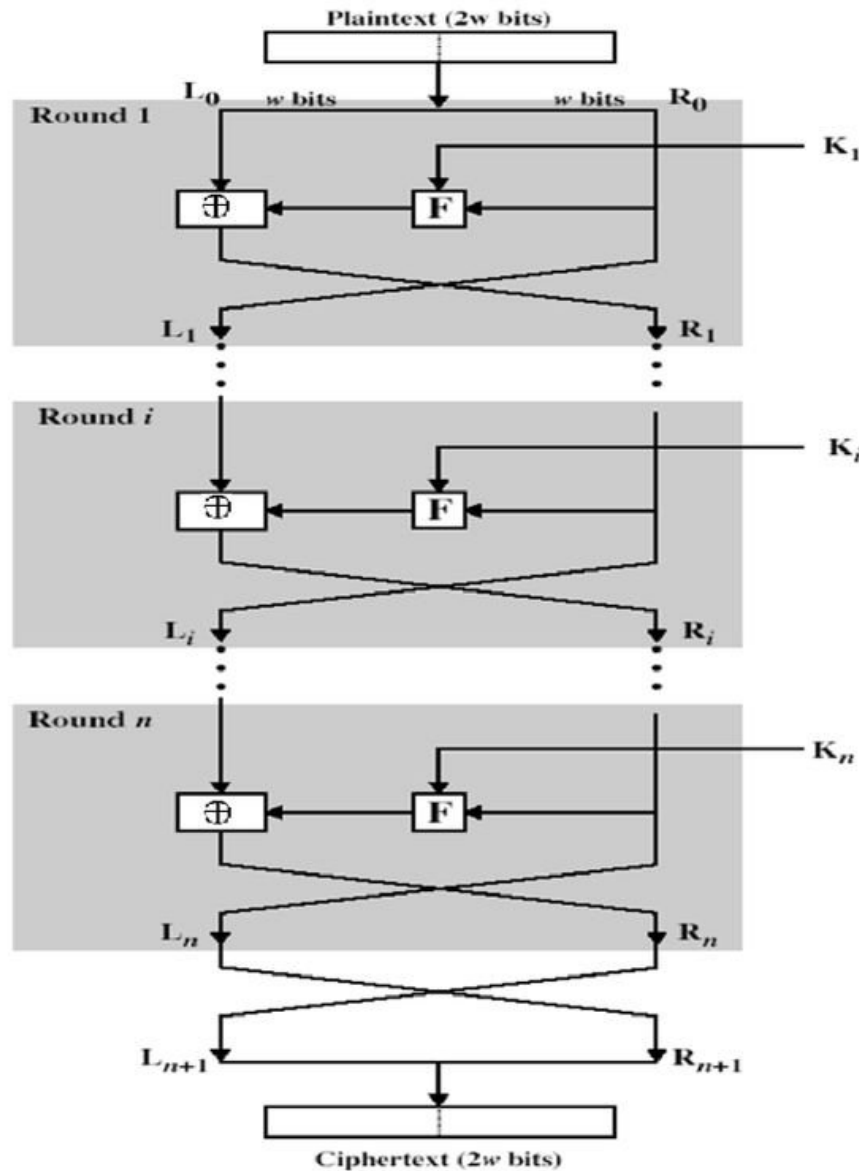


Encryption

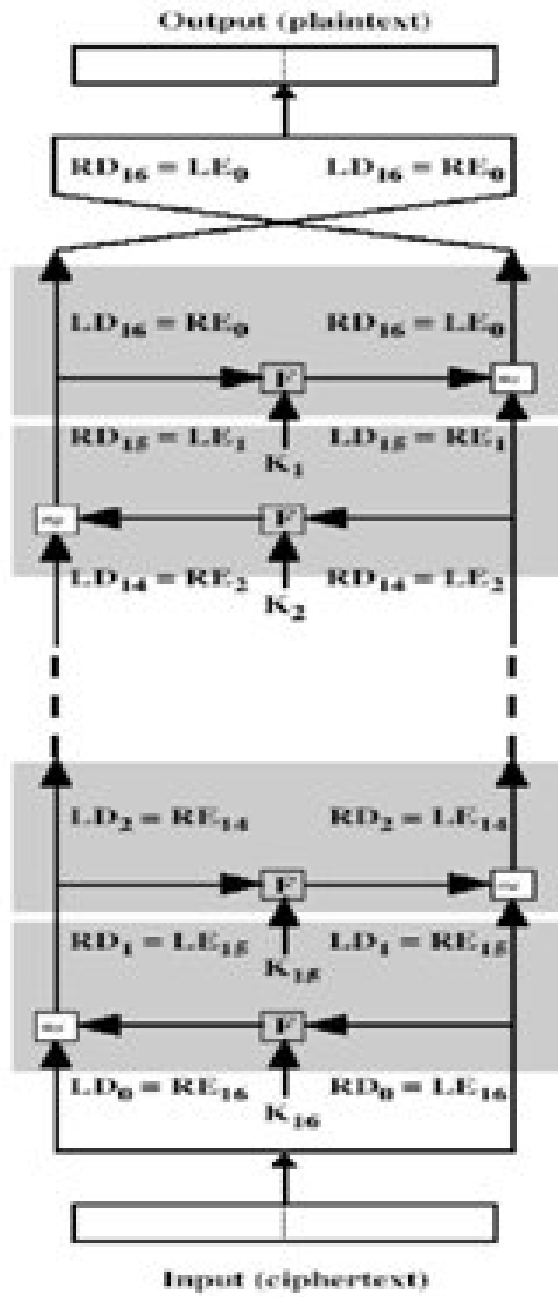
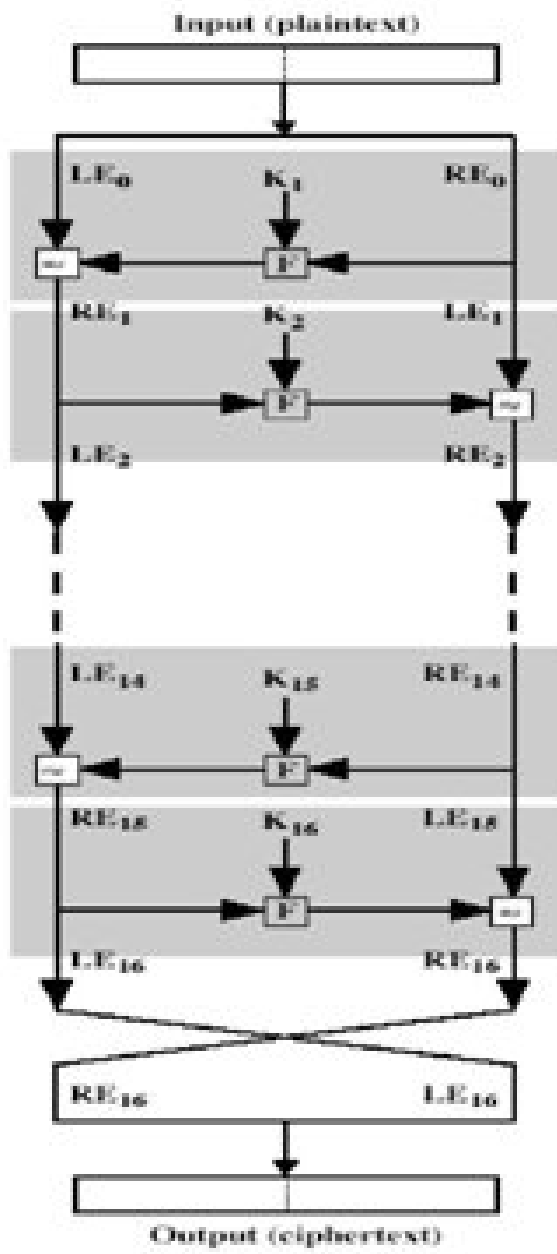


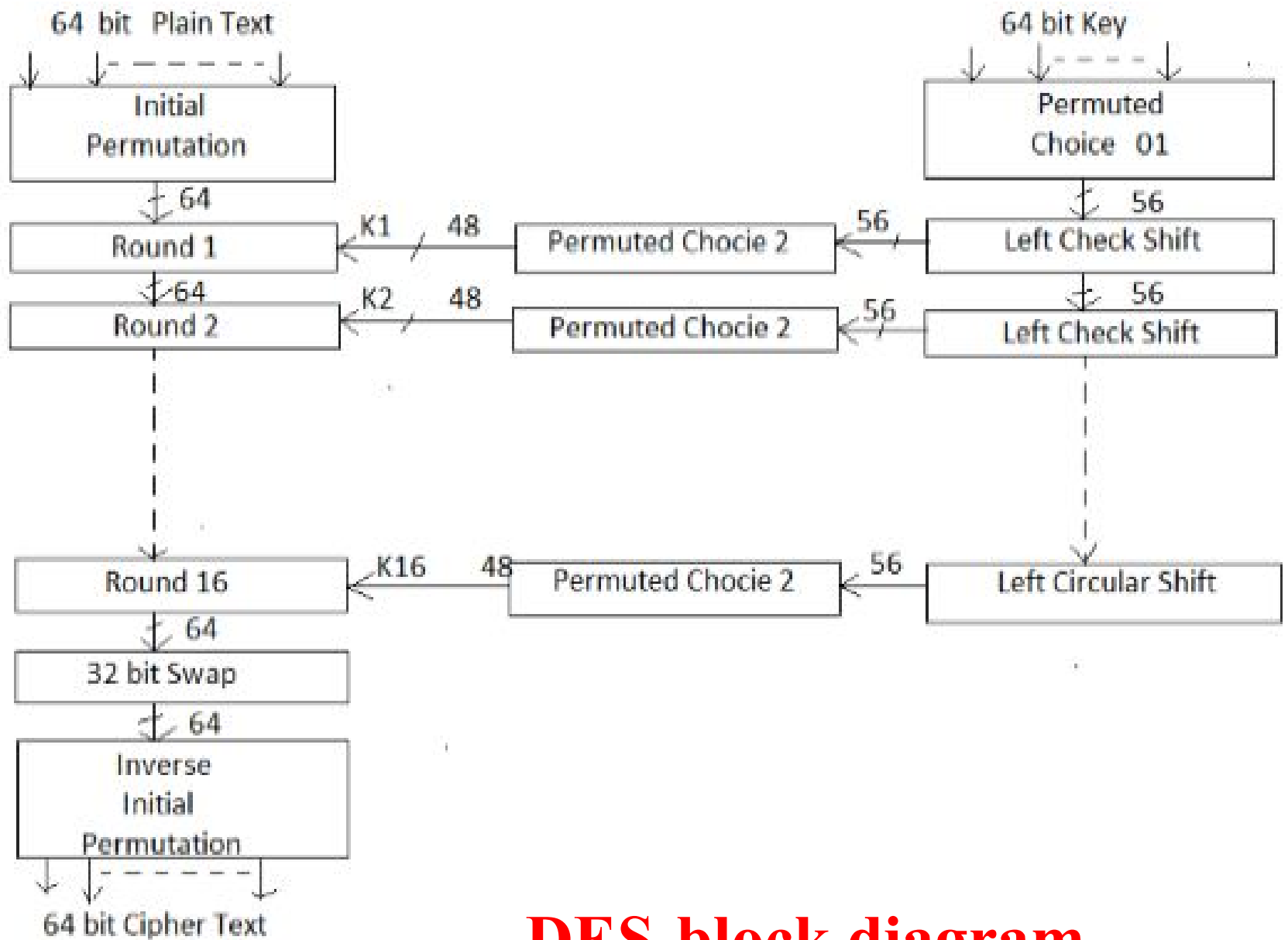
Decryption

Feistel Cipher Structure

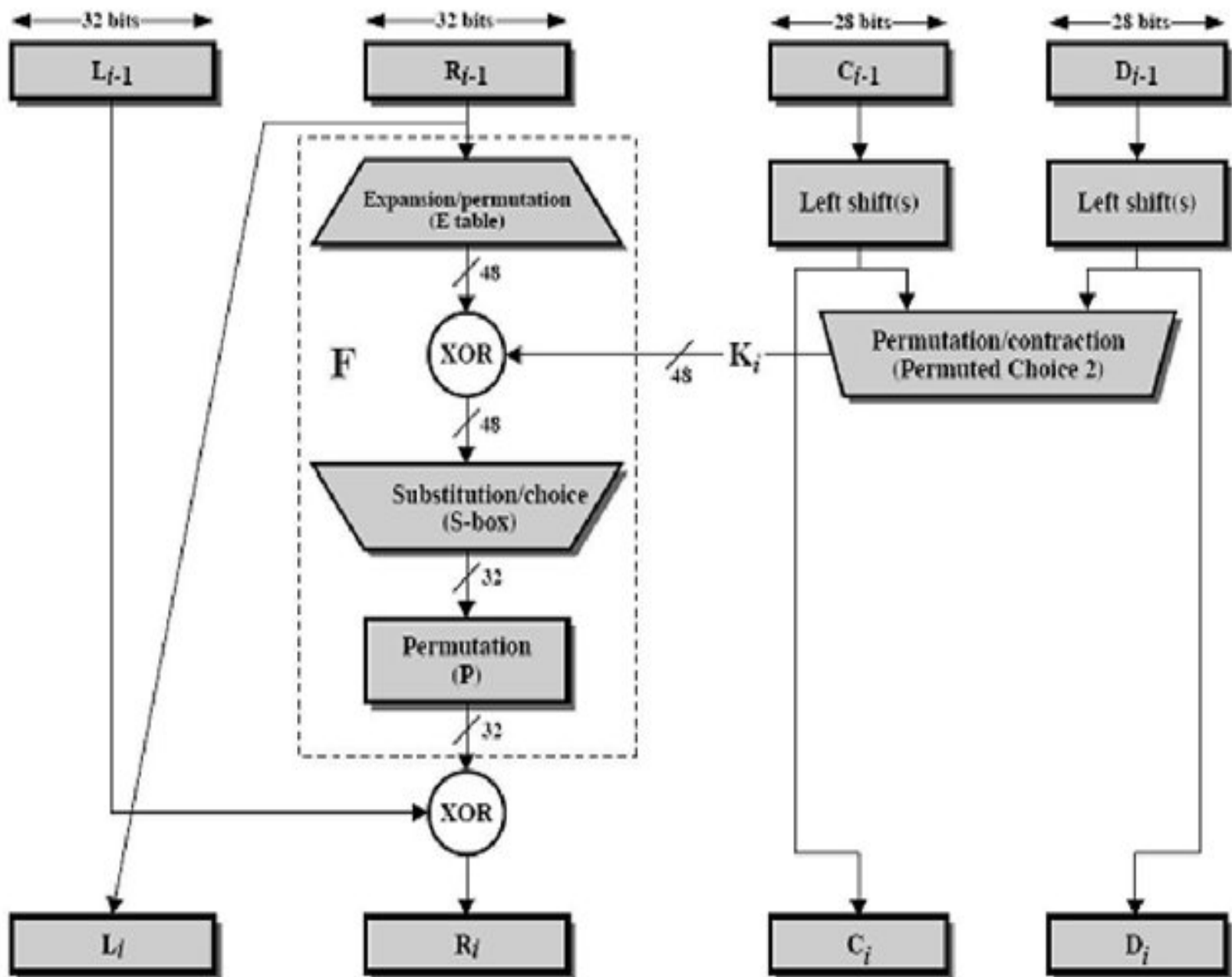


Feistel
Cipher
Structure





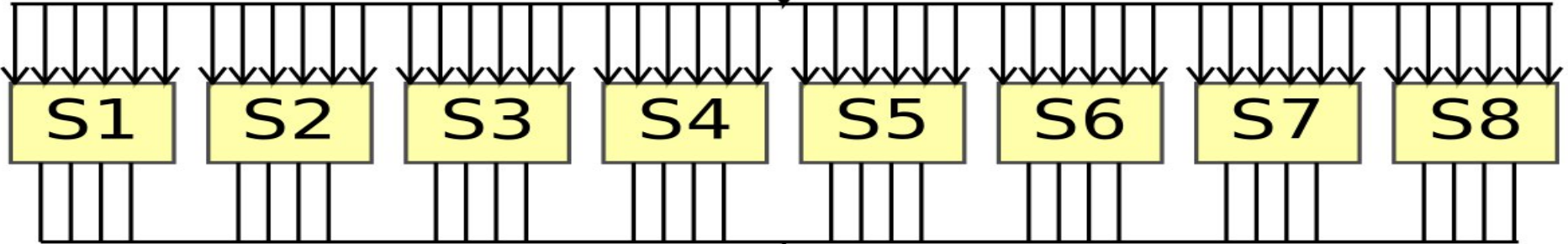
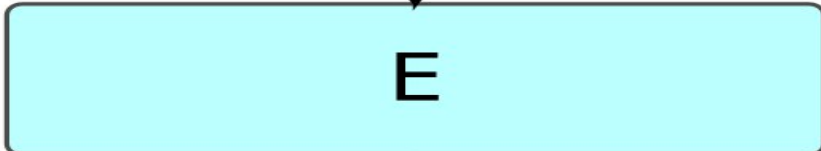
DES-block diagram

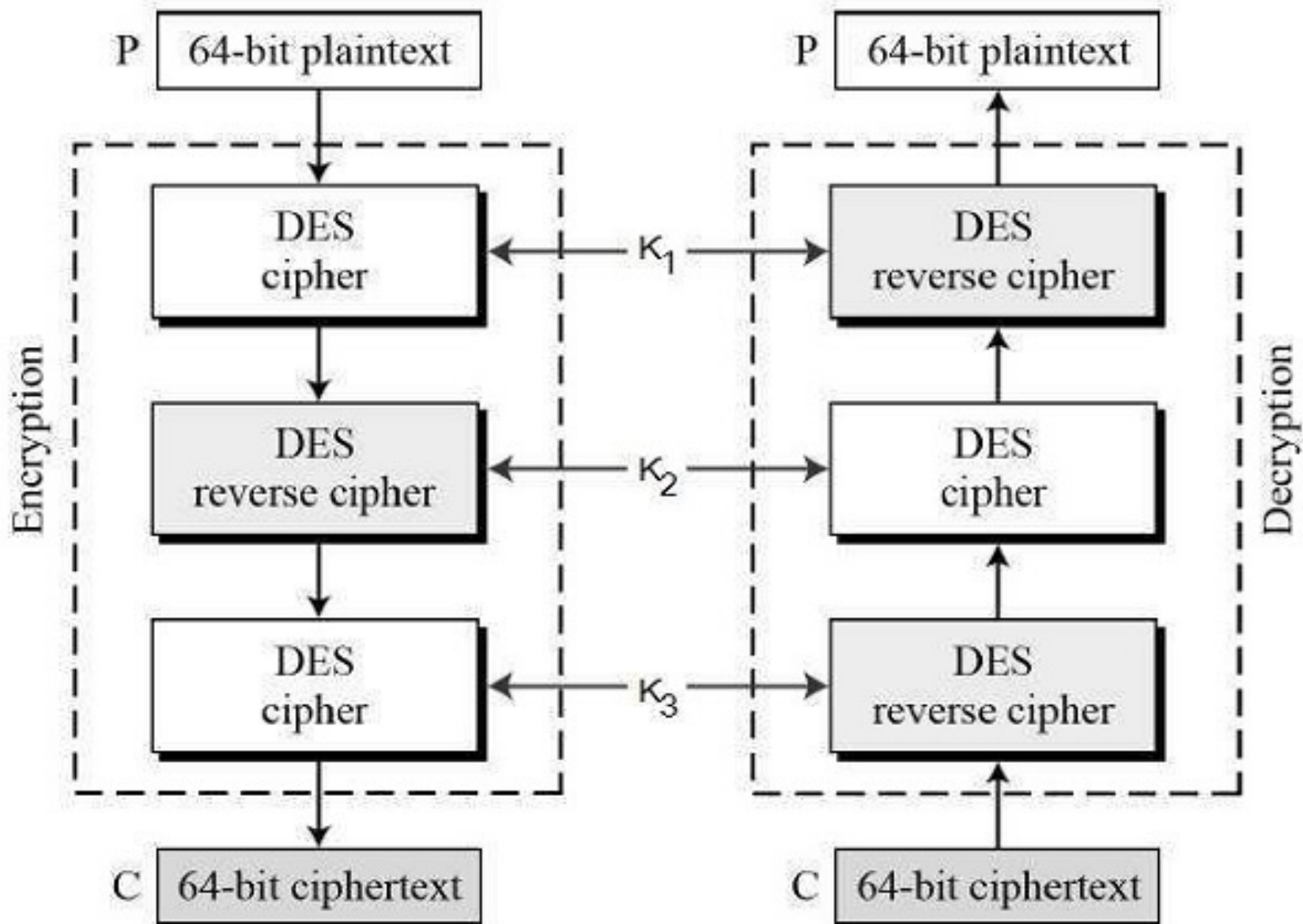


Single Round of DES Algorithm

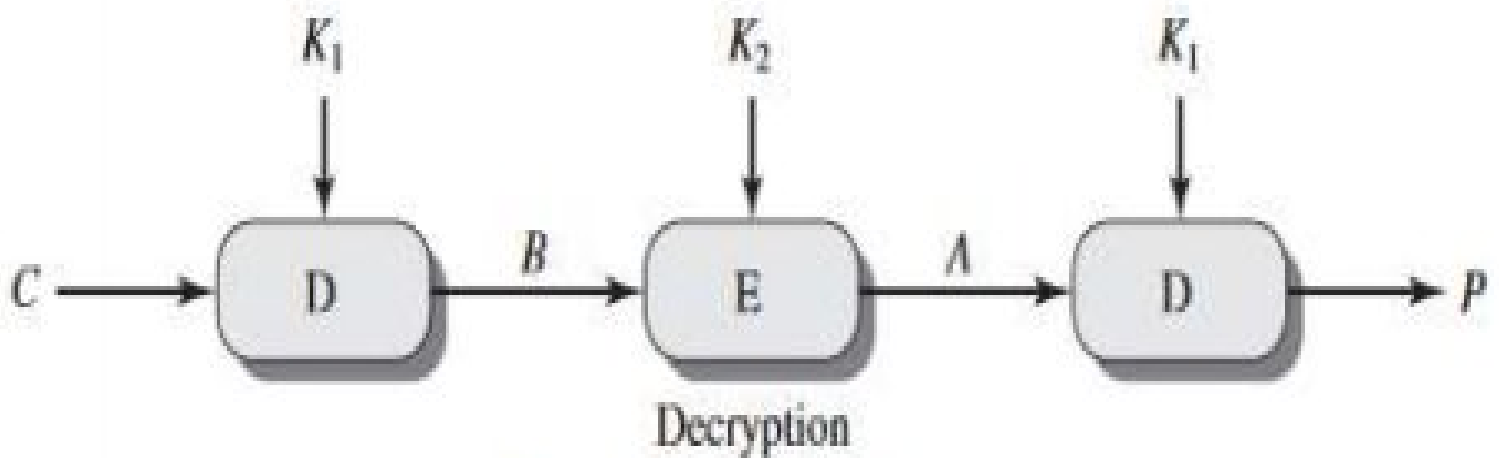
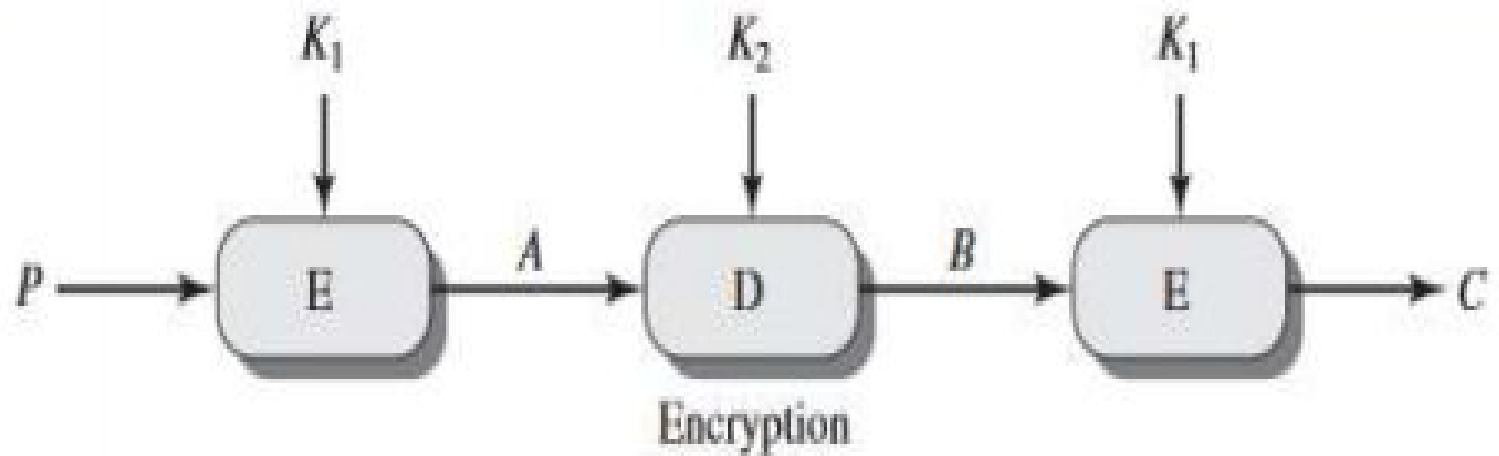
Half Block (32 bits)

Subkey (48 bits)

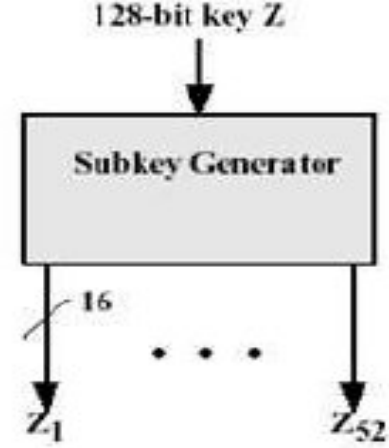
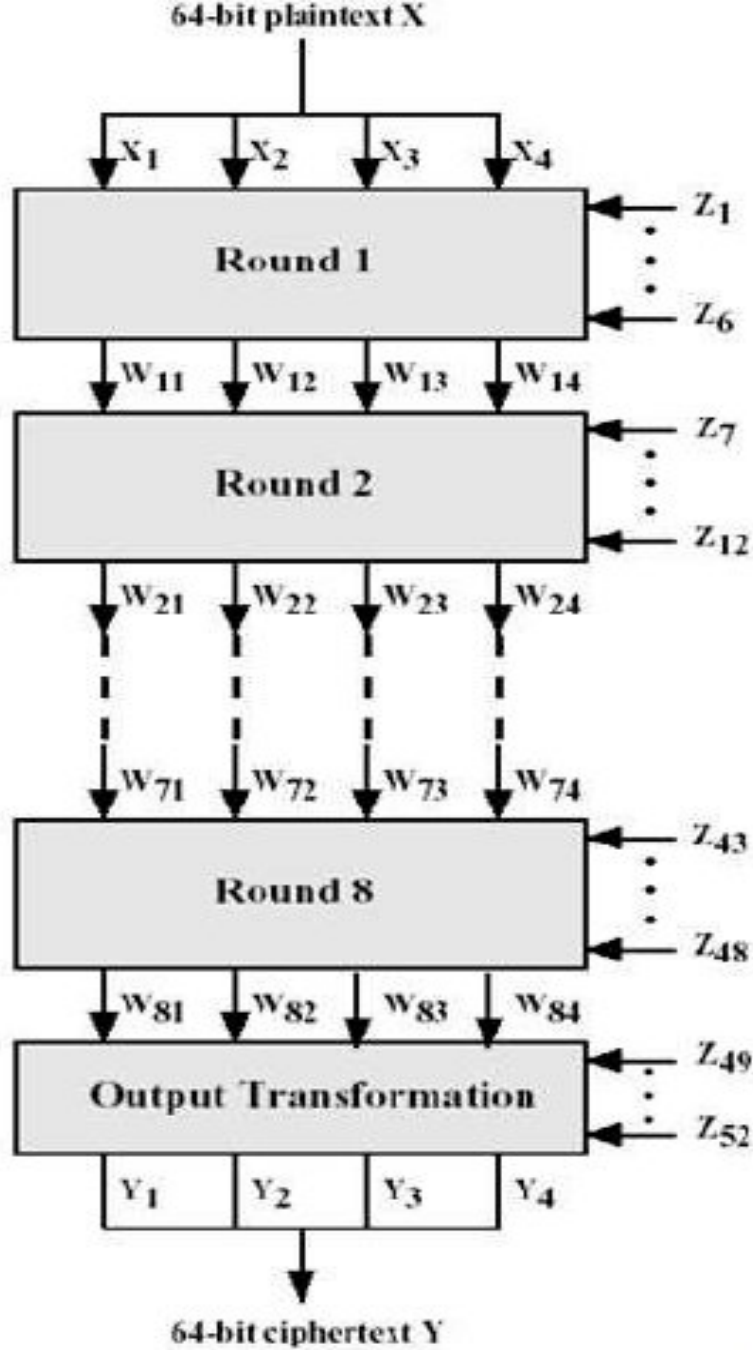




Triple DES



(b) Triple encryption



IDEA BLOCK DIAGRAM

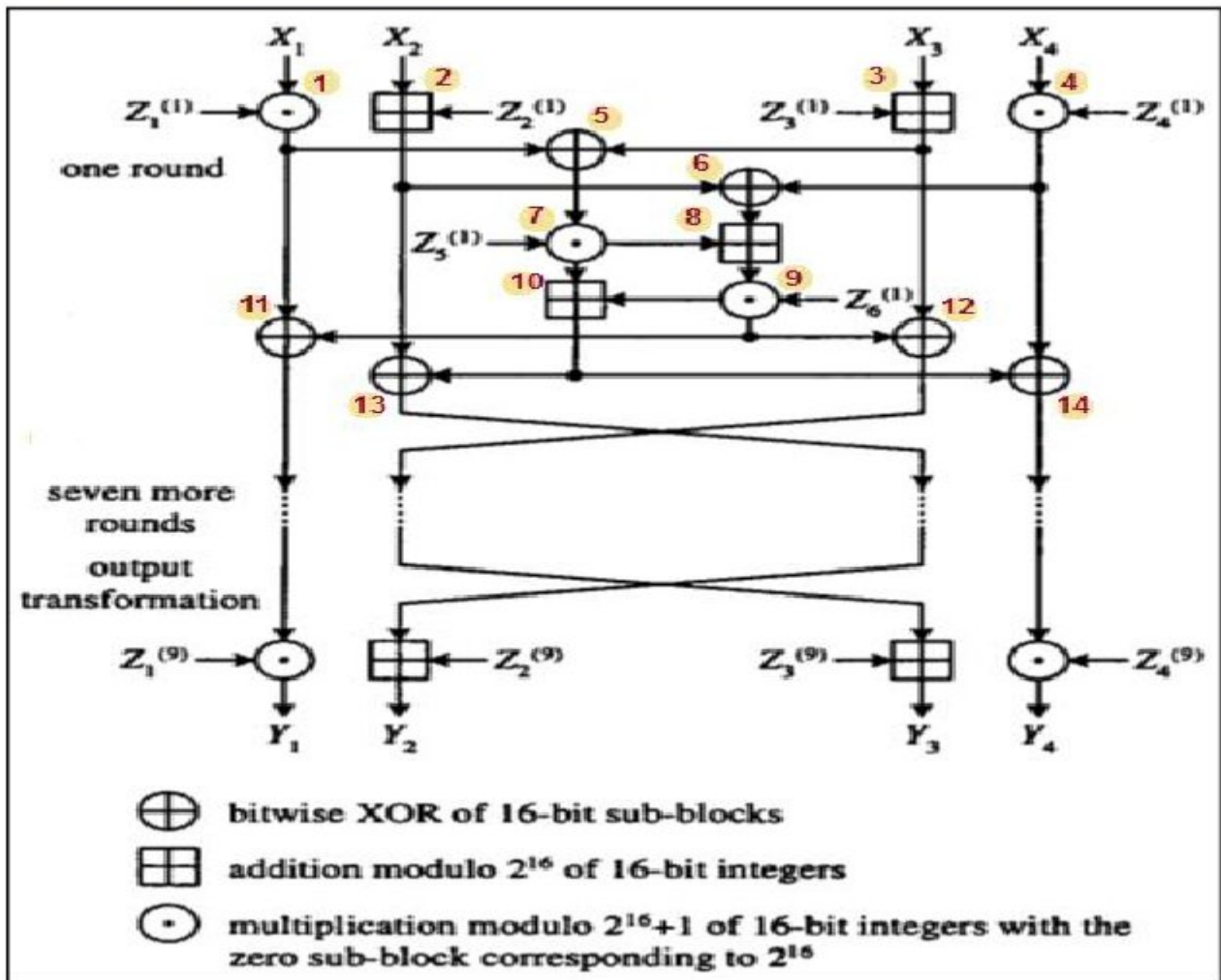
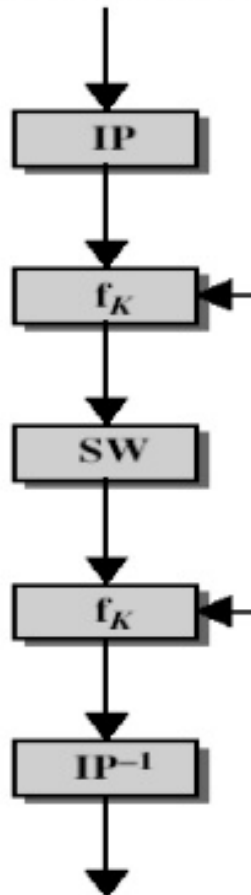


Figure 2: Structure of IDEA encryption process [3]

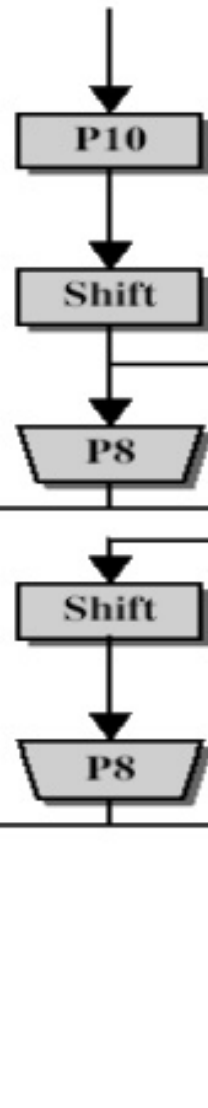
ENCRYPTION

8-bit plaintext



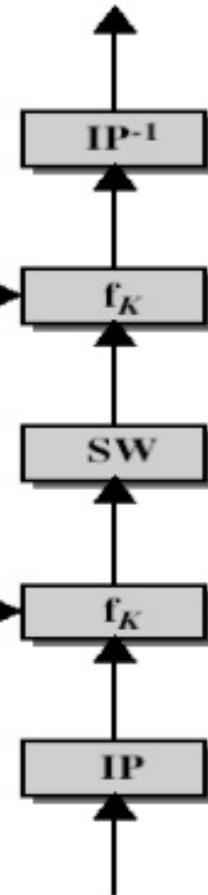
8-bit ciphertext

10-bit key



DECRYPTION

8-bit plaintext



8-bit ciphertext

S-DES scheme

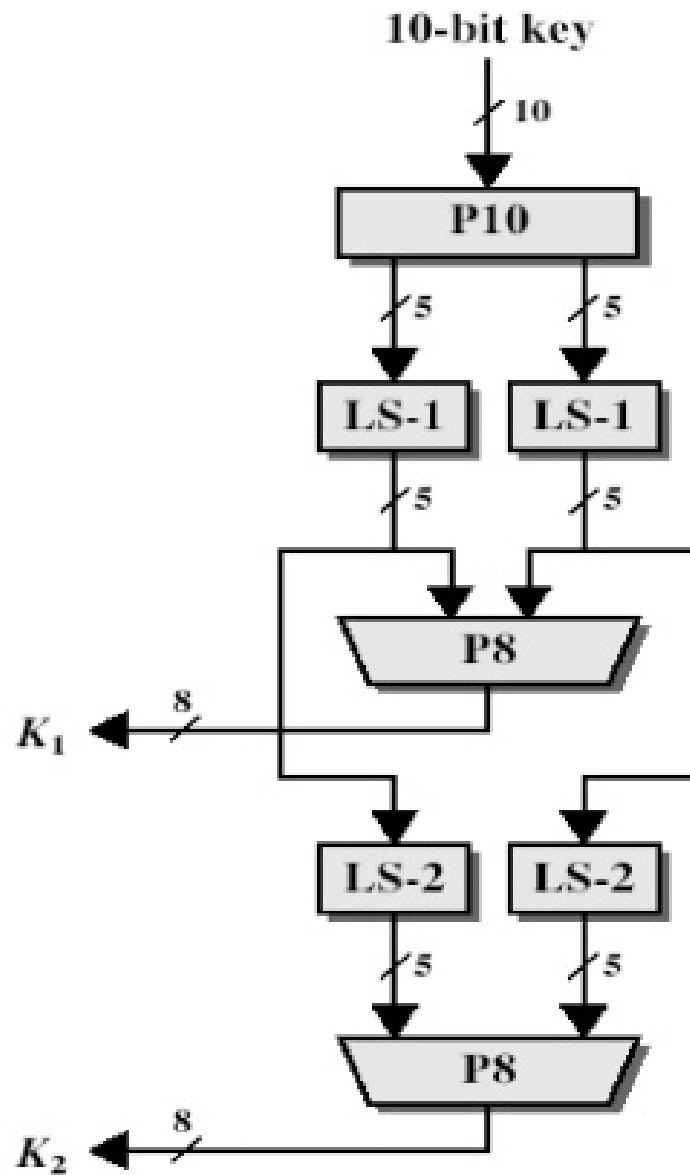
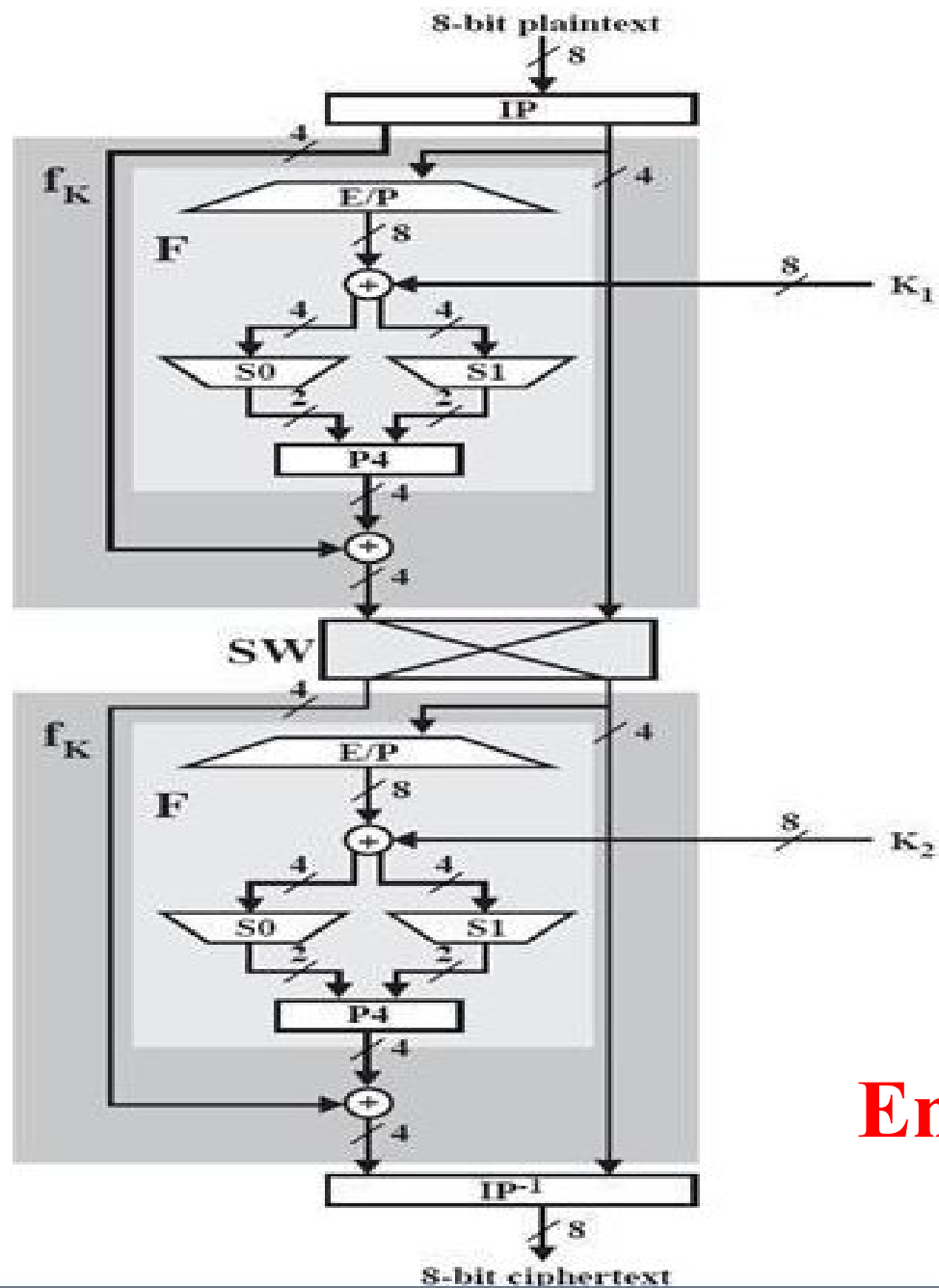
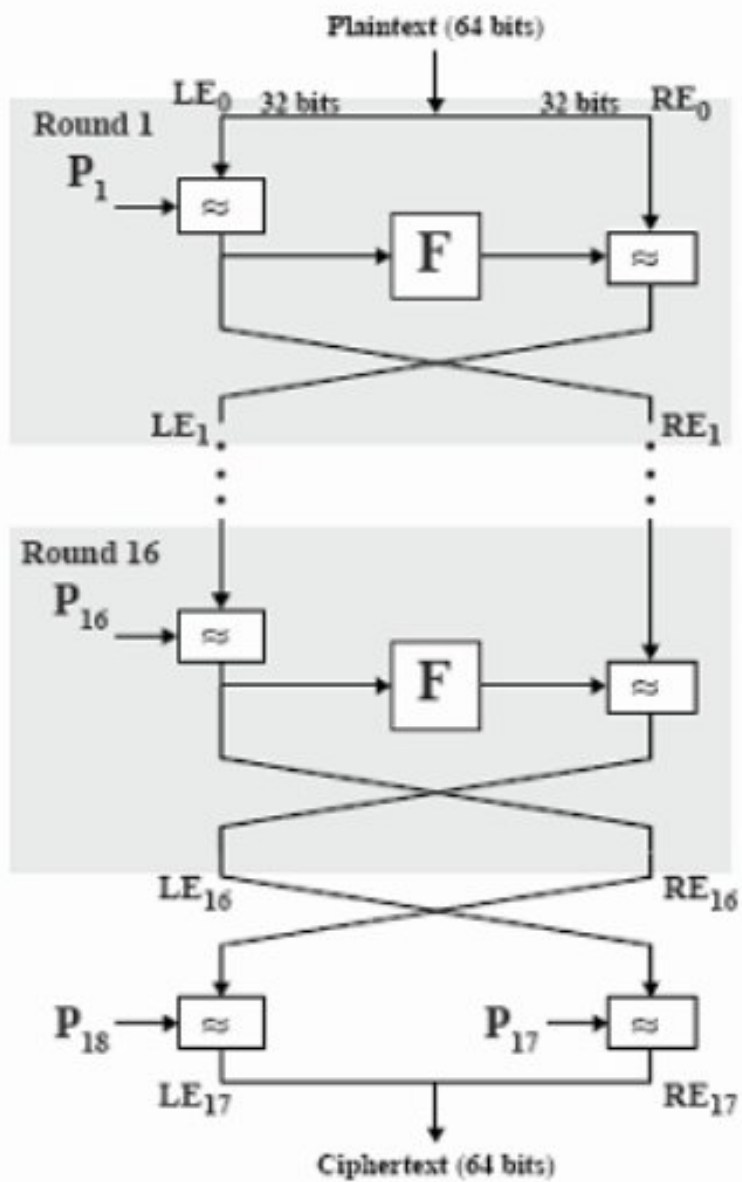


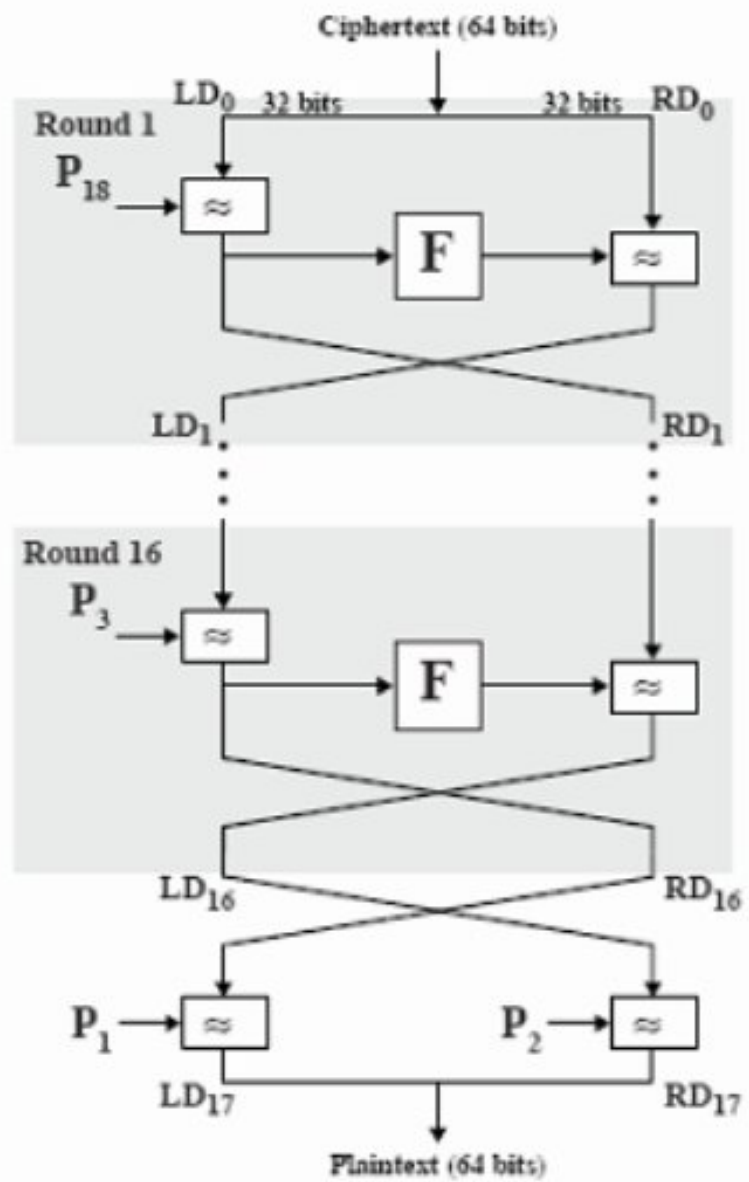
Figure: key generation for S-DES



S-DES Encryption



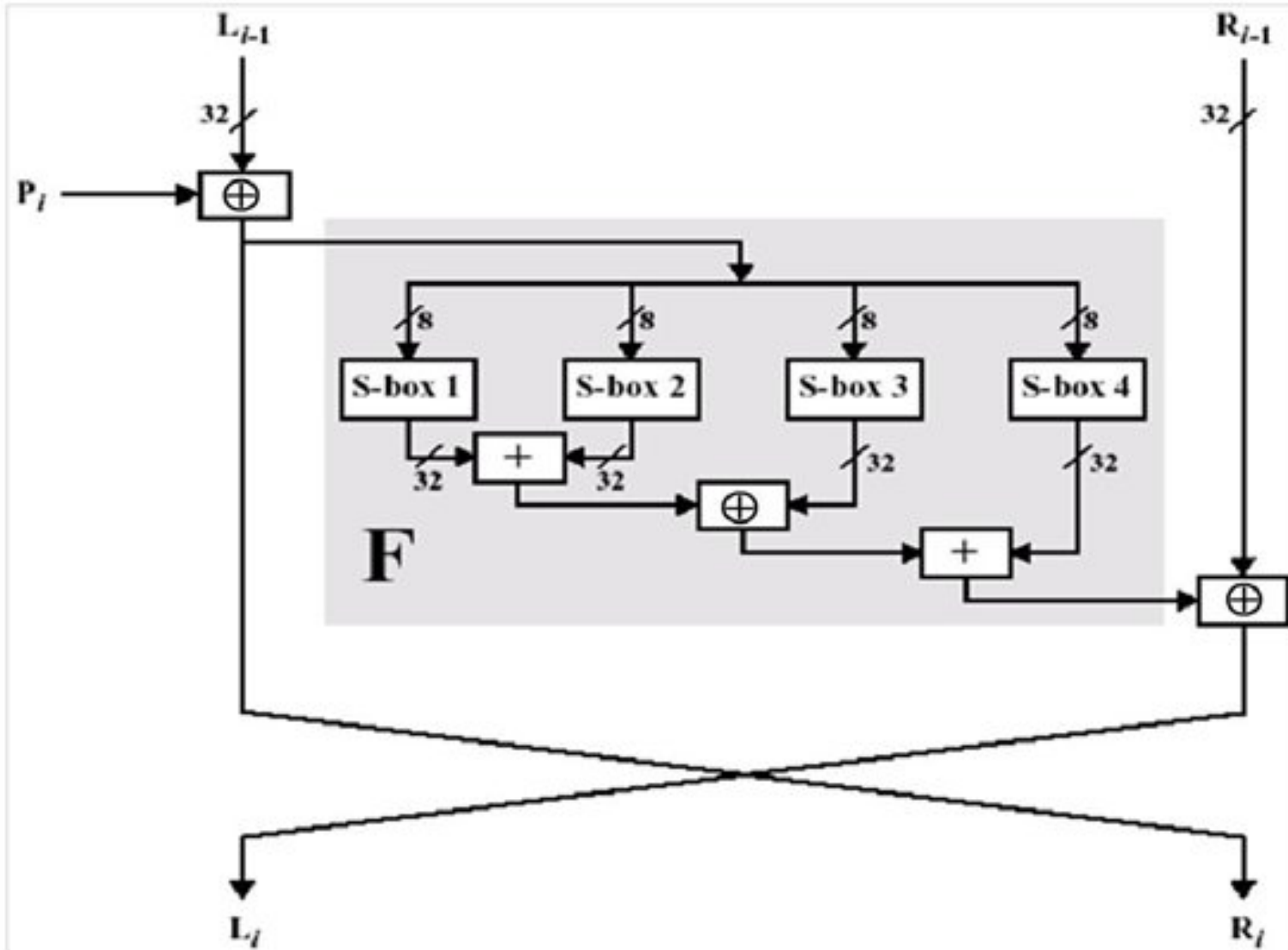
(a) Encryption



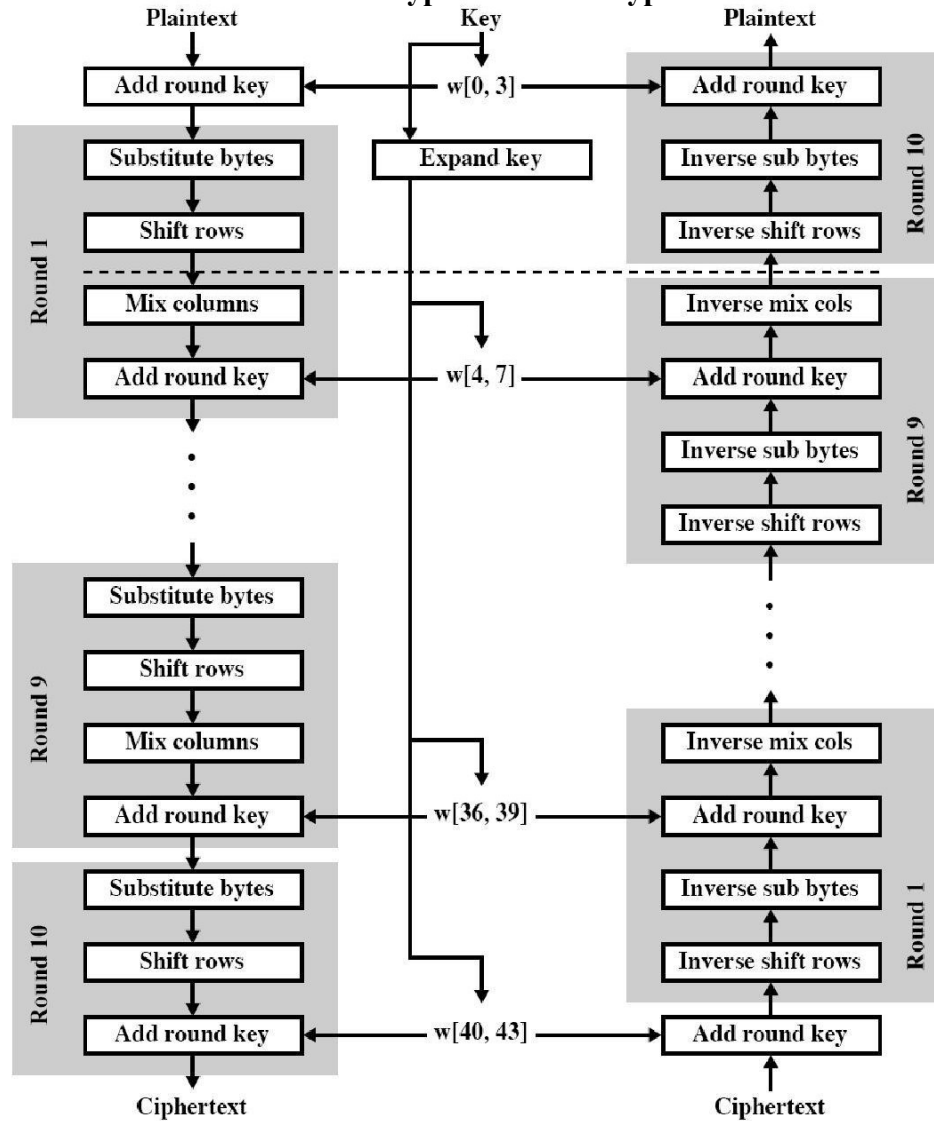
(b) Decryption

Blowfish Encryption and Decryption

Function

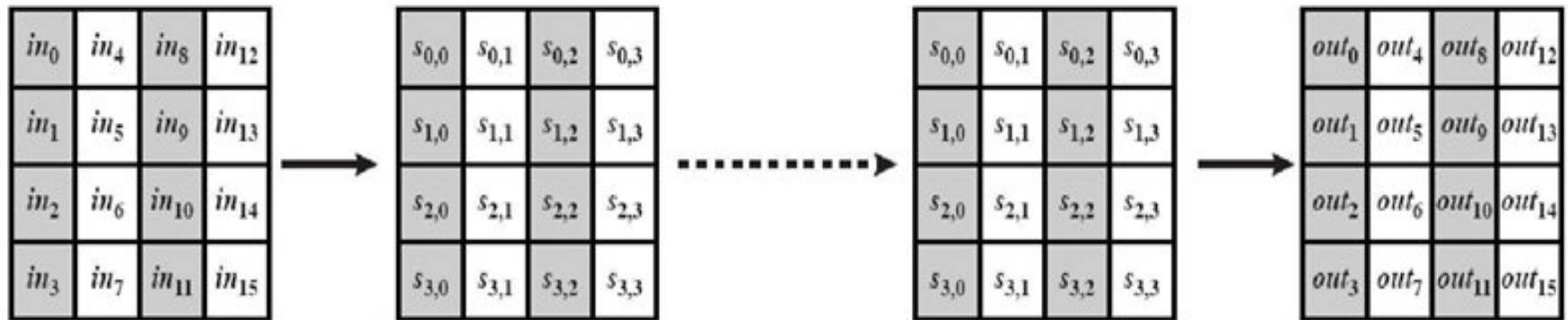


AES Encryption and Decryption



(a) Encryption

(b) Decryption

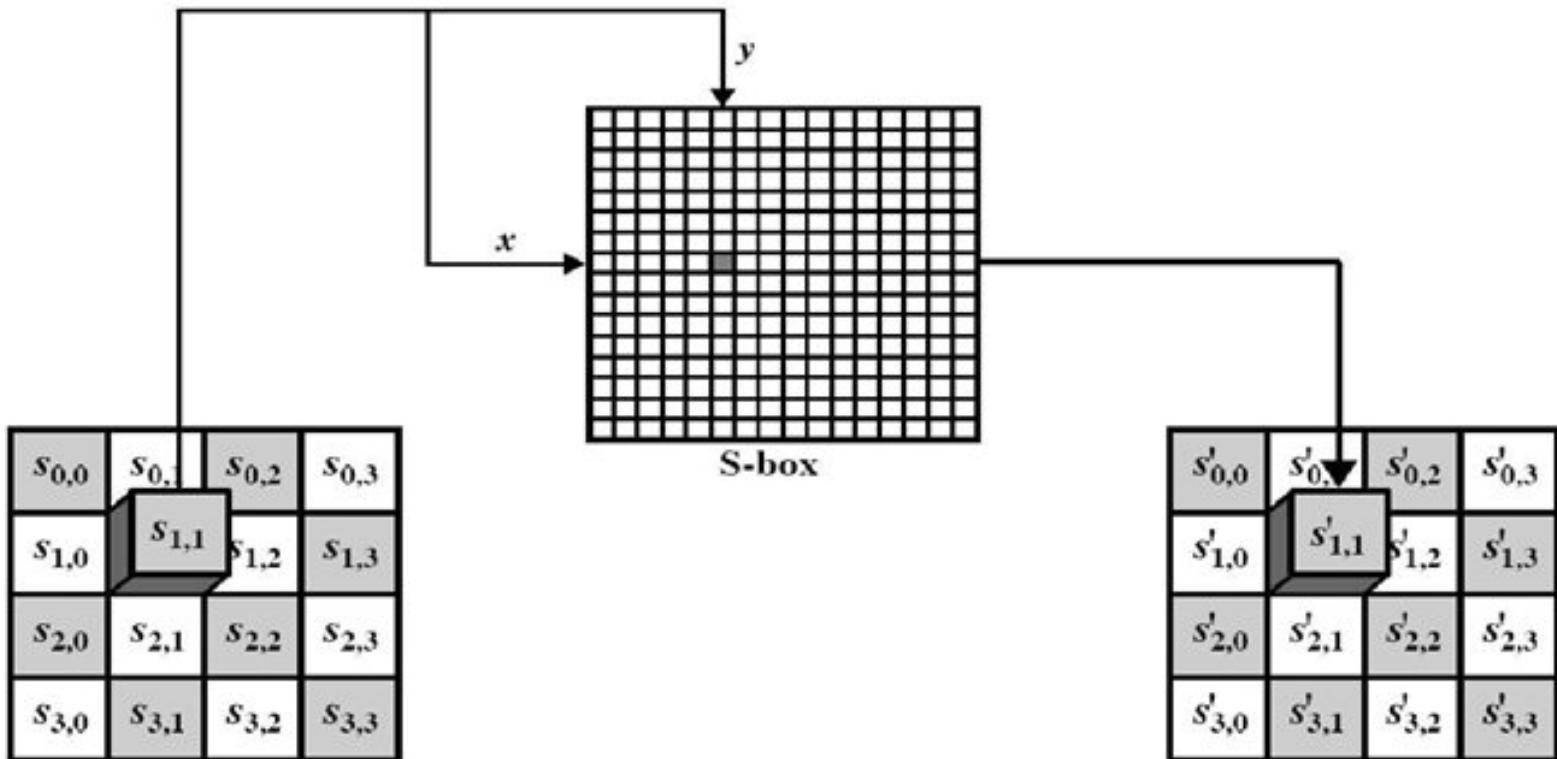


(a) Input, state array, and output

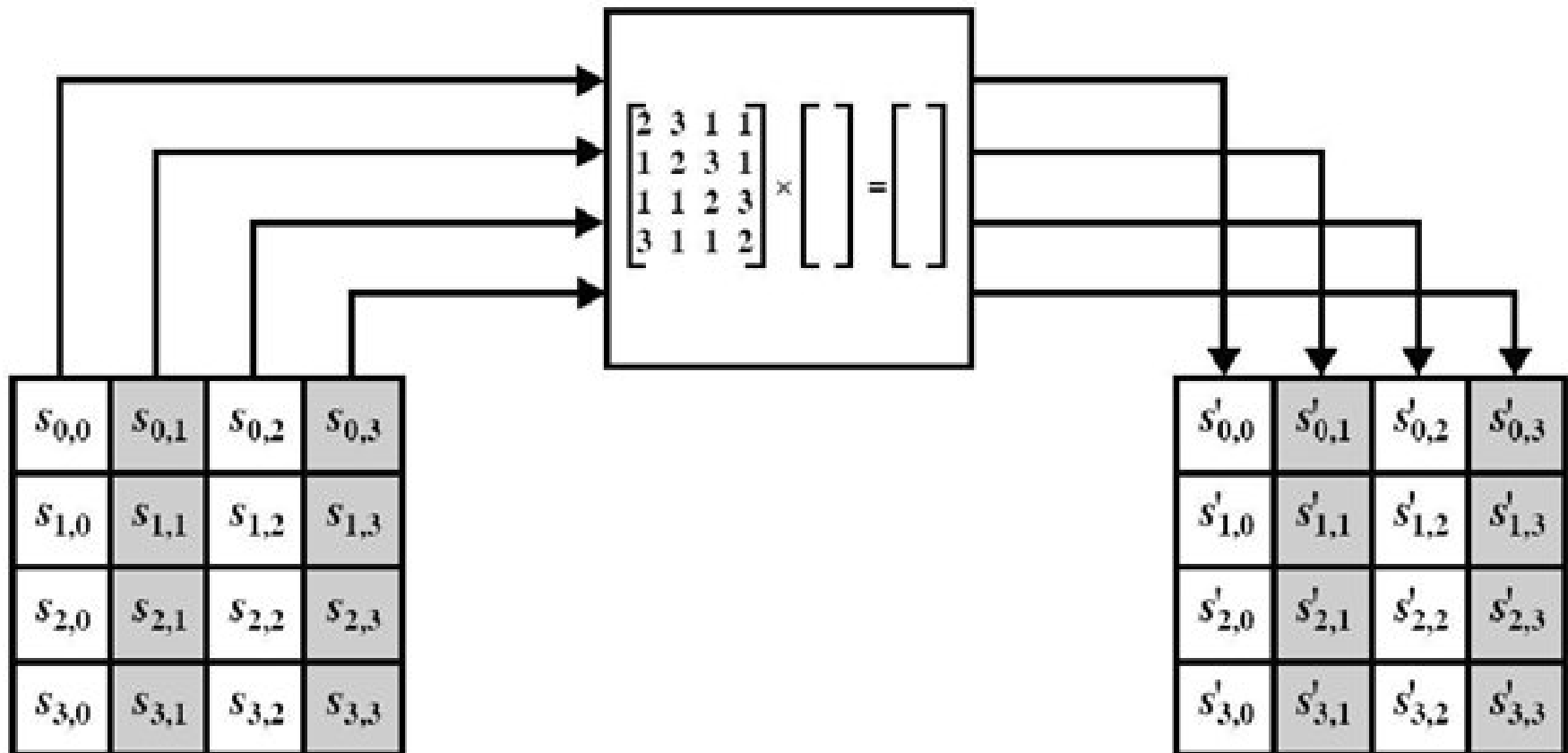


(b) Key and expanded key

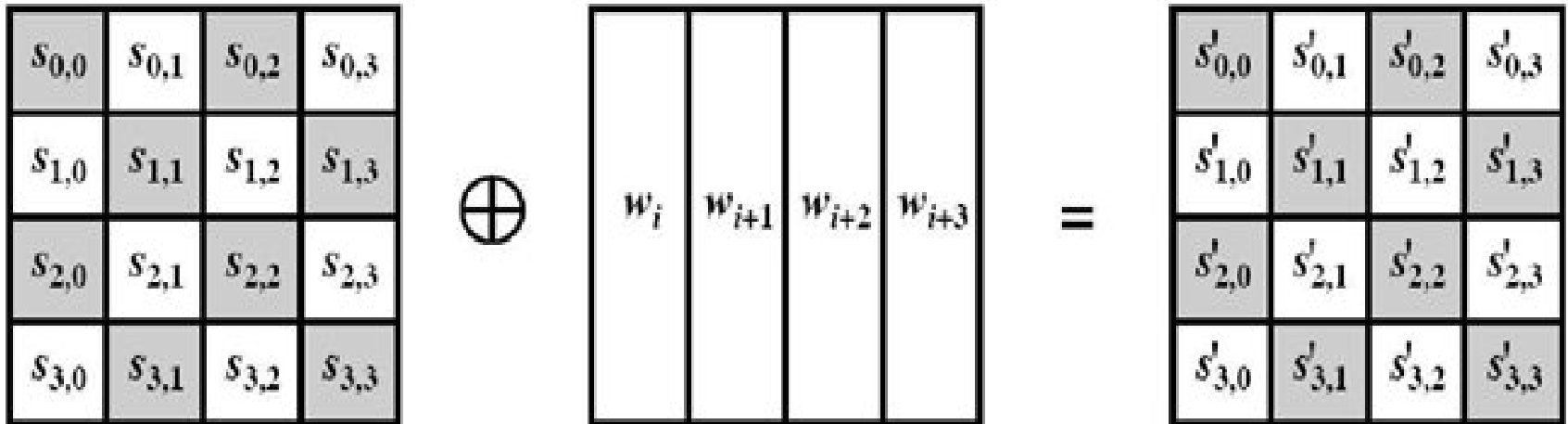
Substitute Bytes Transformation



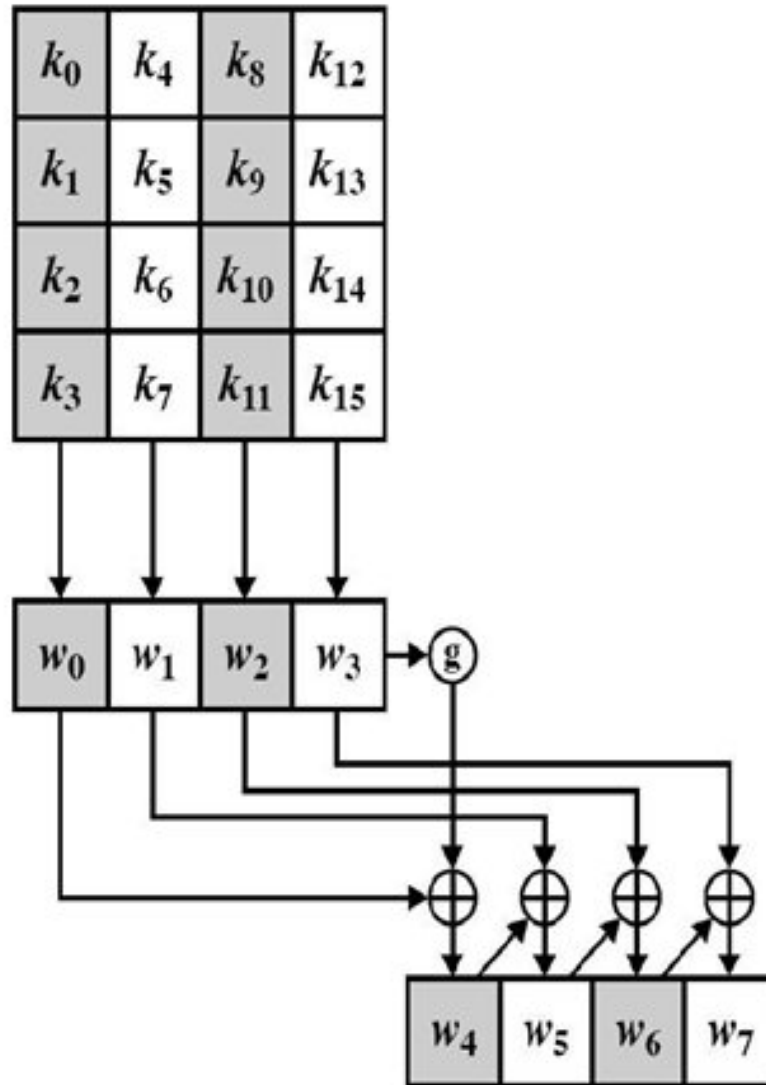
MixColumns Transformation



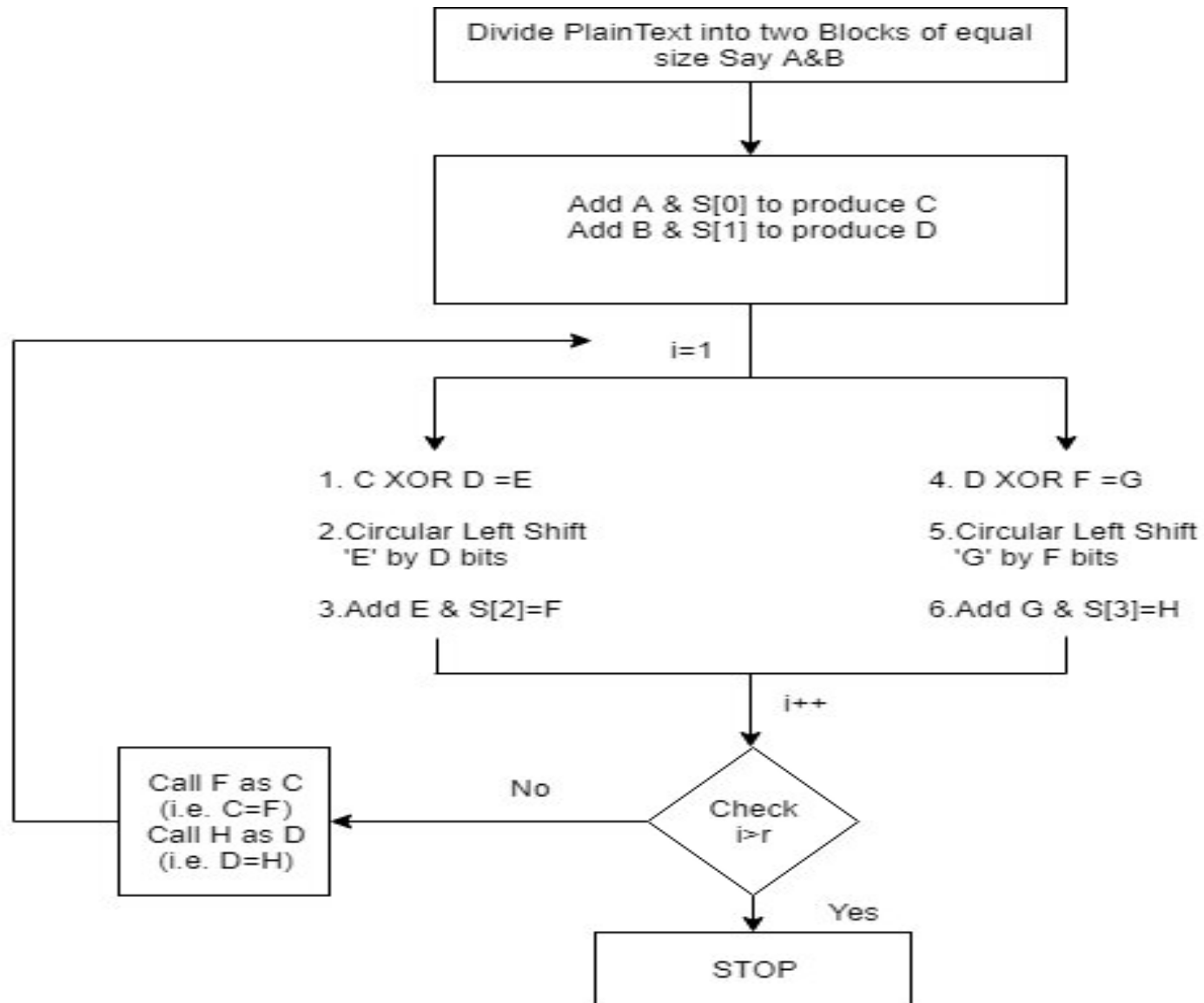
AddRoundKey Transformation



AES Key Expansion



RC-5 Algorithm



We divide the plaintext block into two equal parts A and B

Then they are XOR with two subkeys $S\{0\}$ and $S\{1\}$

$$C = A + S[0]$$

$$D = B + S[1]$$

for $i = 1$ to r do:

1. $C \oplus D = E$

2. perform circular left shift on E by D bits

3. add E and $S[2 * i]$ and store the result in F which is input for step 4

4. $D \oplus F = G$

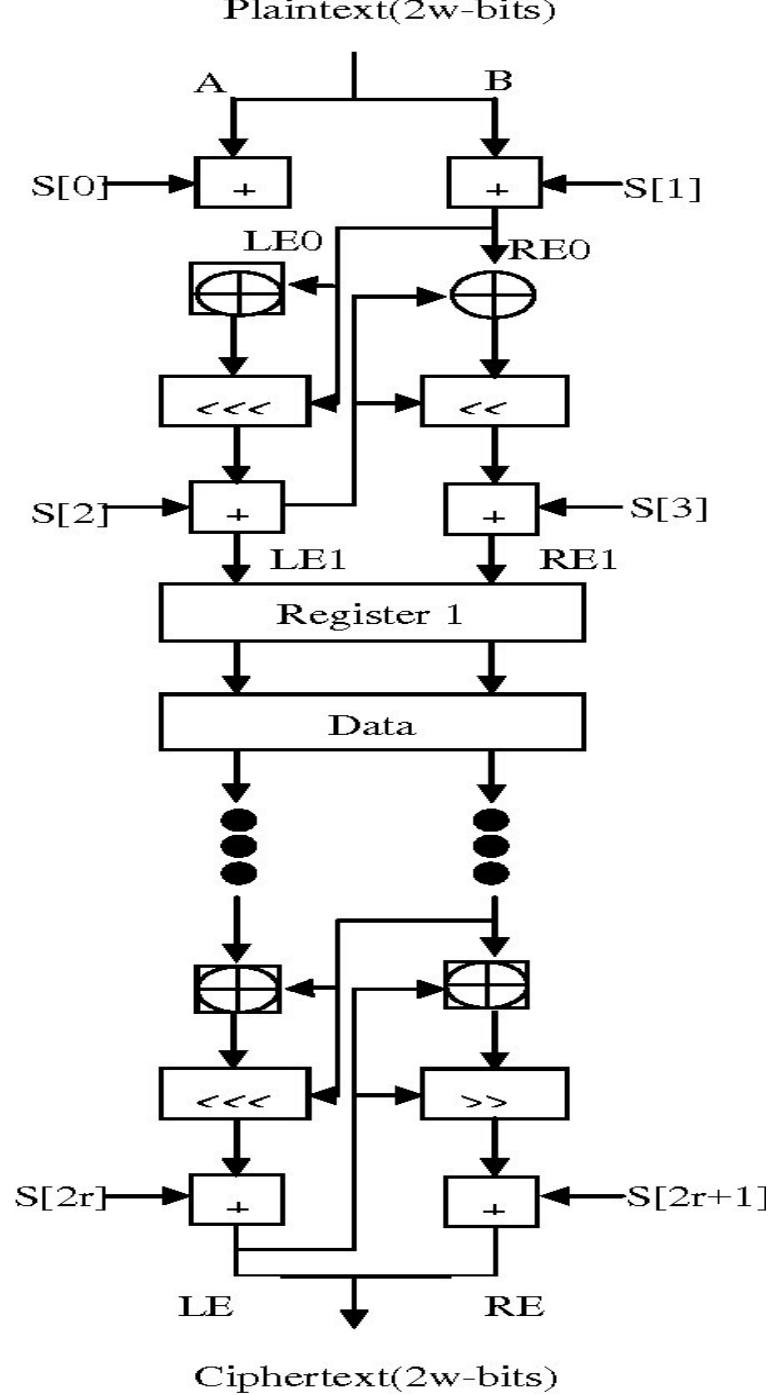
5. perform circular left shift on G by F bits

6. add G and $S[2 * i + 1]$ and store the result in H

7. If $i < r$

Call F as C and H as D and repeat the steps from 1 to 7

else stop



RC4 Algorithm

Key Scheduling:

```
j = 0;  
for i = 0 to 255 do  
{  
    j = (j + S[i] + T[i]) mod 256;  
    Swap(S[i], S[j]);  
}
```

Stream Generation:

- $i, j = 0;$
- while (true)
- $i = (i + 1) \bmod 256;$
- $j = (j + S[i]) \bmod 256;$
- $\text{Swap}(S[i], S[j]);$
- $t = (S[i] + S[j]) \bmod 256;$
- $k = S[t];$

RSA ALGORITHM

- Choose two large prime numbers (p and q)
- Calculate $n = p * q$ and $\Phi n = (p-1)(q-1)$
- Choose a number e where $1 < e < \Phi n$
- Calculate $d = e^{-1} \text{mod } \Phi n$
- You can bundle private key pair as (n, d)
- You can bundle public key pair as (n, e)

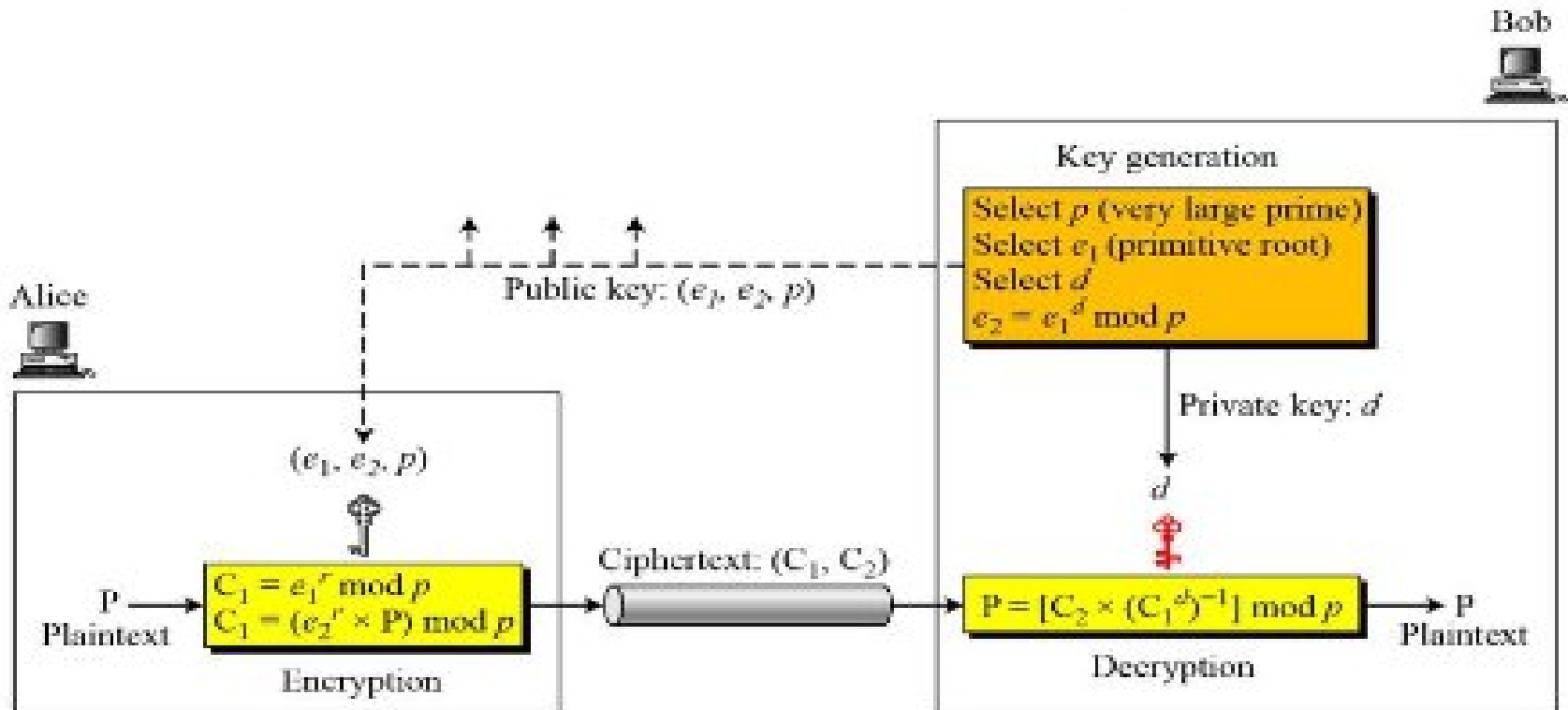
Encryption/Decryption Function

- If the plaintext is m , ciphertext = $m^e \text{ mod } n$.
- If the ciphertext is c , plaintext = $c^d \text{ mod } n$

Elgamal cryptography

El gamal cryptography works in 3 steps/stages

- Key generation
- Elgamal encryption
- Elgamal decryption



Elgamal cryptography Algorithm

A. Elgamal key generation:

- Select a large prime number 'p'
- Select encryption key 'e'1' to be primitive root of mod p
- Select decryption key 'd' such that $1 \leq d \leq p-2$
- Calculate encryption key 'e'2' such that
- $e_2 = e_1^d \text{ mod } p$
- Form the set i,e public key (e1,e2,p) to be announced publicly
- Private Key 'd' to be kept secret.

B. Elgamal key encryption:

- Select a random number 'r'
- Compute the first part of ciphertext 'c'1,c1=e1^r mod p'
- Compute the second part of ciphertext 'c'2,c2=(e2^r * PT) mod p

C. Elgamal key decryption:

- Calculate the PT
- $PT = (c_2 * (c_1^d)^{-1}) \text{ mod } p$

DIFFIE-HELLMAN KEY EXCHANGE

Global Public Elements

q	prime number
α	$\alpha < q$ and α a primitive root of q

User A Key Generation

Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{X_A} \text{ mod } q$

User B Key Generation

Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{X_B} \text{ mod } q$

Generation of Secret Key by User A

$$K = (Y_B)^{X_A} \text{ mod } q$$

Generation of Secret Key by User B

$$K = (Y_A)^{X_B} \text{ mod } q$$

KNAPSACK ALGORITHM

Example -

- $\{1, 2, 4, 10, 20, 40\}$ is a super increasing as
- $1 < 2$, $1+2 < 4$, $1+2+4 < 10$, $1+2+4+10 < 20$ and $1+2+4+10+20 < 40$.

Derive the Public key

- **Step-1:**
Choose a super increasing knapsack $\{1, 2, 4, 10, 20, 40\}$ as the private key.
- **Step-2:**
Choose two numbers n and m . Multiply all the values of private key by the number n and then find modulo m . The value of m must be greater than the sum of all values in private key, for example 110. And the number n should have no common factor with m , for example 31.
- **Step-3:**
Calculate the values of Public key using m and n .
 - $1 \times 31 \bmod(110) = 31$
 - $2 \times 31 \bmod(110) = 62$
 - $4 \times 31 \bmod(110) = 14$
 - $10 \times 31 \bmod(110) = 90$
 - $20 \times 31 \bmod(110) = 70$
 - $40 \times 31 \bmod(110) = 30$
- Thus, our public key is $\{31, 62, 14, 90, 70, 30\}$
And Private key is $\{1, 2, 4, 10, 20, 40\}$.

Now take an example for understanding the process of encryption and decryption.

Example -

Lets our plain text is **100100111100101110**.

1. Encryption :

As our knapsacks contain six values, so we will split our plain text in a groups of six:

- **100100 111100 101110**

Multiply each values of public key with the corresponding values of each group and take their sum.

- **100100 {31, 62, 14, 90, 70, 30}**
- **$1 \times 31 + 0 \times 62 + 0 \times 14 + 1 \times 90 + 0 \times 70 + 0 \times 30 = 121$**

- **111100 {31, 62, 14, 90, 70, 30}**
- **$1 \times 31 + 1 \times 62 + 1 \times 14 + 1 \times 90 + 0 \times 70 + 0 \times 30 = 197$**

- **101110 {31, 62, 14, 90, 70, 30}**
- **$1 \times 31 + 0 \times 62 + 1 \times 14 + 1 \times 90 + 1 \times 70 + 0 \times 30 = 205$**
- **So, our cipher text is 121 197 205.**

2. Decryption :

The receiver receive the cipher text which has to be decrypt. The receiver also knows the values of m and n .

So, first we need to find the x , which is multiplicative inverse of $n \bmod m$ i.e.,

- $n \times x \bmod(m) = 1$
- $31 \times x \bmod(110) = 1 \Rightarrow x = 71$

Now, we have to multiply 71 with each block of cipher text take modulo m .

- $121 \times 71 \bmod(110) = 11$

Then, we will have to make the sum of 11 from the values of private key $\{1, 2, 4, 10, 20, 40\}$ i.e.,

$$1+10=11$$

so make that corresponding bits 1 and others 0 which is **100100**.

Similarly,

- $197 \times 71 \bmod(110) = 17$
- $1+2+4+10=17 = 111100$

- And, $205 \times 71 \bmod(110) = 35$

- $1+4+10+20=35 = 101110$

- After combining them we get the decoded text. **100100111100101110** which is our plain text.