UNIT-2

Symmetric Ciphers

Please study figure by comparing and reading with theoretical part then it will be easier to understand. If felt lengthy or 5 marks asked then shorten theory and mathematical part denoted by red can be escaped to make shorter and easier.

⊛ Feistel Cipher Structure:-

This structure is a particular form of substitution-permutation network (SPN) proposed by Shannon.

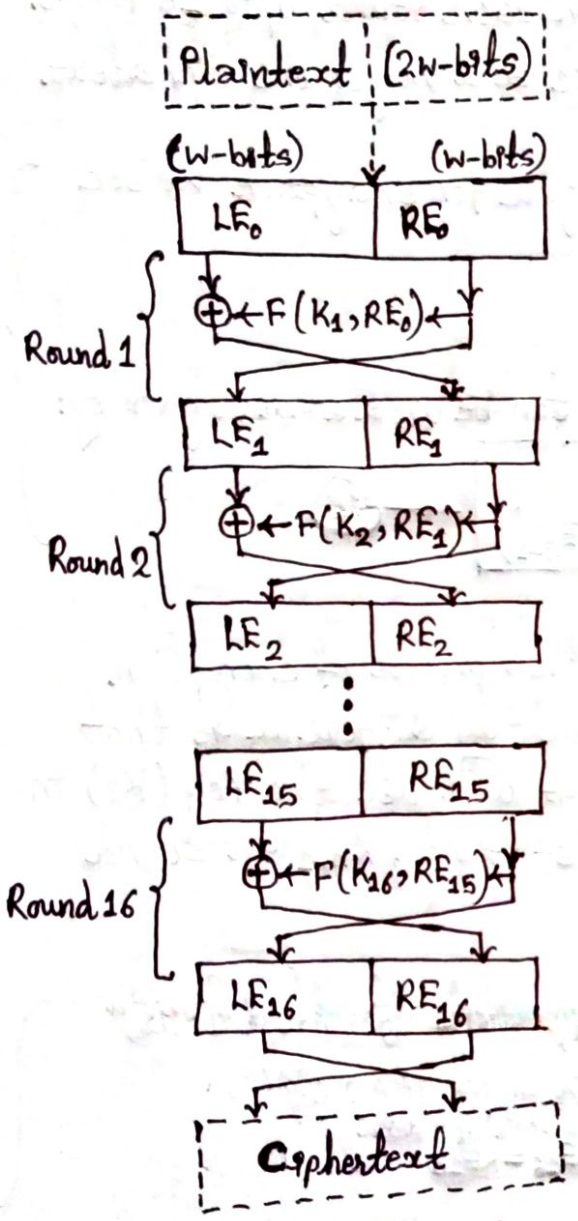


Fig: Feistel Encryption

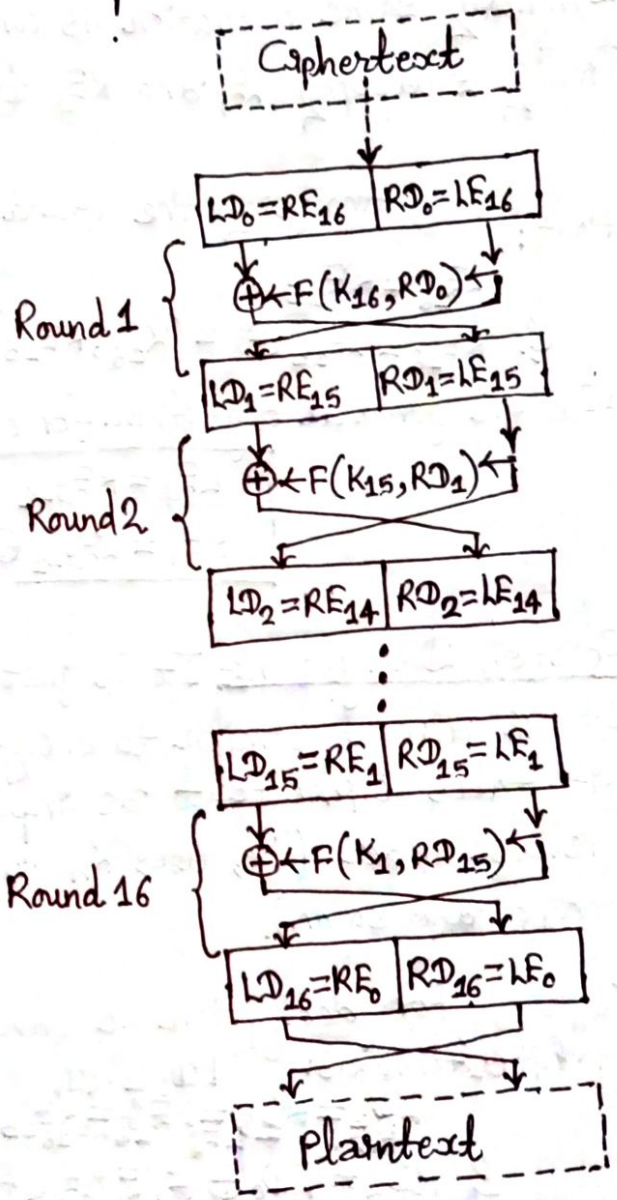


Fig: Feistel Decryption

Feistel Encryption: Firstly we divide the plaintext into two halves in which left half is denoted by LE_i (i.e, Left encryption for round i) and right half is denoted by RE_i (i.e, Right encryption for round i). In this method we have 16 rounds, each round having same structure and operation.

In each round substitution is performed on left half of data. It includes X-OR operation on left half with a round/encryption function (F) which takes two arguments Key (K_i) and right half (RE_i).

Following this substitution, a permutation is performed, that consists of interchange of data in two halves. The output generated by X-OR operation is applied to new RE_i (i.e, RE_{i+1}) and RE_i does not perform any operation and simply applied to new LE_i (i.e, LE_{i+1}). This substitution and permutation completes whole 1 round. Similarly same operation is continued in each round upto 16 rounds. Finally swapping LE_i and RE_i takes place and we get the ciphertext.

Analysing the round 16 in figure for encryption we see that:

$$LE_{16} = RE_{15}$$
$$\& \quad RE_{16} = LE_{15} \oplus F(K_{16}, RE_{15})$$

So in general algorithm for encryption for ith iteration is given by:

$$LE_i = RE_{i-1}$$
$$RE_i = LE_{i-1} \oplus F(K_i, RE_{i-1})$$

Feistel Decryption:- It is just the reverse process of encryption. It is almost similar to encryption, the only difference is that it takes ciphertext as input and it will use the key (K_i) in reverse order (i.e, uses K_n in the first round, K_{n-1} in second round and so on.).

For decryption we can write equations from figure for round 1 as;

$$LD_1 = RD_0 = LE_{16} = RE_{15}$$ ← In encryption for round 16 above we have $LE_{16} = RE_{15}$
$$\& \quad RD_1 = LD_0 \oplus F(K_{16}, RD_0)$$

So, in general algorithm for decryption for ith iteration is given by:

$$LD_i = RE_{16-i}$$
$$RD_i = LD_{i-1} \oplus F(K_{17-i}, RD_{i-1})$$

⊛ Data Encryption Standards (DES):-

The Data Encryption Standard (DES) is a method for encrypting information selected as an official by Federal Information Processing Standard (FIPS) for United States in 1976. DES is a block cipher and encrypts data in blocks of size of 64 bits each, which means 64 bits of plaintext goes as the input to DES, which produces 64 bits of ciphertext. The same algorithm and key are used for encryption and decryption, with minor differences. The key length is 56 bits.

Key Generation:-

The 64-bit key is used as input to the DES algorithm. The bits of key are number from 1 to 64. Every 8th bit is ignored, therefore total 8 bits are ignored for parity checking or discarded. The 64-bits are then divided into two 28-bits halves, let level C_0 and D_0. Each half is treated seperately in successive round and both half are rotated left by one or two bit specified by each round. This shifted value are taken as input to the next round.

DES Encryption:-

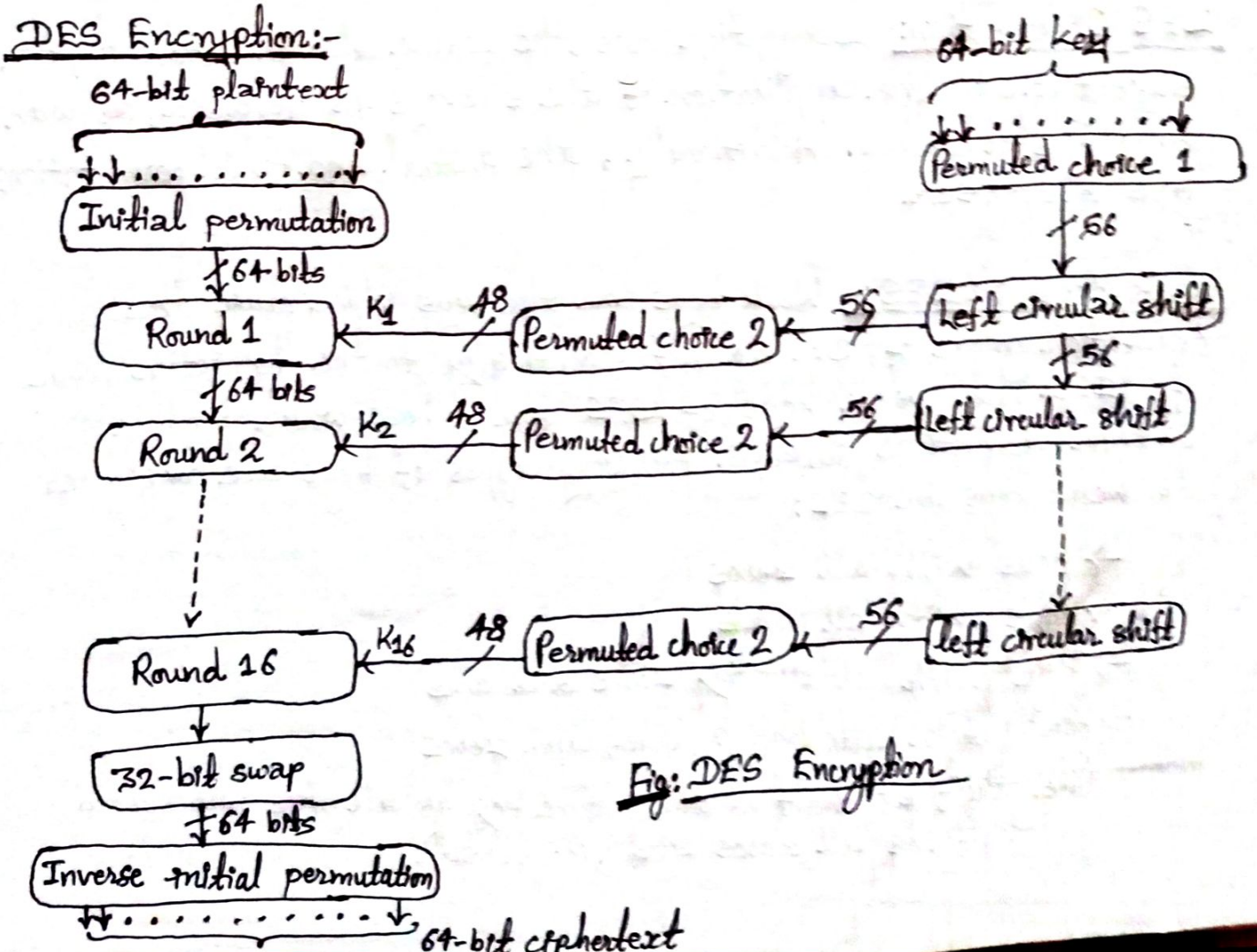


Fig: DES Encryption

The overall scheme for DES encryption is illustrated in the figure above. There are two inputs to the encryption function: the plaintext to be encrypted and the key. The plaintext must be 64-bits and in length and the key 56-bits. The processing of plaintext proceeds in three phases. First, the 64-bit plaintext passes through an initial permutation that rearranges the bits to produce the permuted input. This is followed by a phase consisting of 16 rounds of the same function, which involves both permutation and substitution functions. The left and right halves of the output are swapped to produce the preoutput. Finally the preoutput is passed through a inverse initial permutation to produce the 64-bit ciphertext.

In 64-bit key side initially, the key is passed through a permutation function. Then, for each of the sixteen rounds, a subkey (K_i) is produced by the combination of a left circular shift and a permutation. The permutation function is the same for each round, but a different subkey is produced because of repeated shifts of the key bits.

DES Decryption:- Decryption uses the same algorithm as encryption, except that the application of the subkeys is reversed, as with any Feistel cipher. Additionally, the initial and final permutations are reversed.

Weak Keys in DES:- Weak keys are the keys which cause the encryption mode of DES to act identically to the decryption mode of DES. The weak keys of DES are those which produce sixteen identical subkeys. Following types of keys are considered as weak keys in DES:

- → Keys with all zeros
- → Keys with all ones
- → Keys with alternating zeros and ones
- → Keys with alternating ones and zeros
  (i.e, the first half of the entire key is all ones and second half is all zeros and vice versa).

Double DES:-

The simplest form of multiple encryption has two encryption stages and two keys. Given a plaintext P and two encryption keys K_1 and K_2, ciphertext is generated as;

$$C = E(K_2, E(K_1, P))$$

Decryption requires that the keys be applied in reverse order:

$$P = D(K_1, D(K_2, C))$$

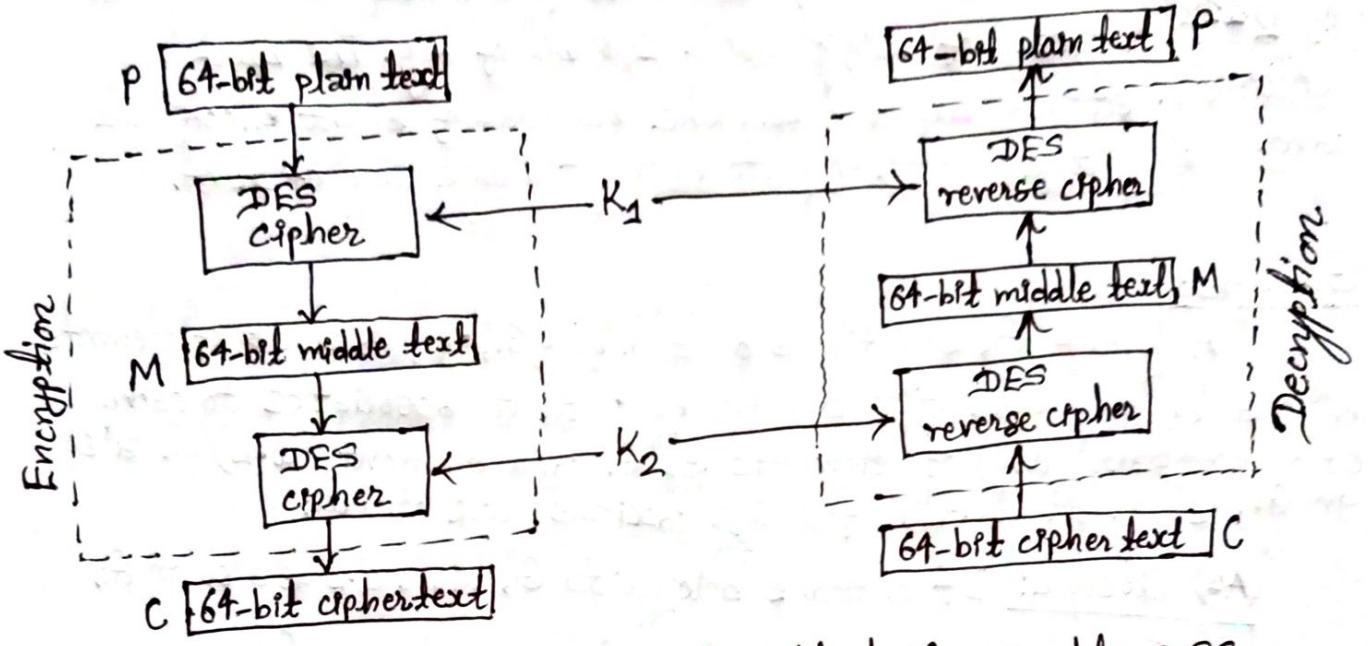


Fig:- Meet-in-the-Middle attack for Double DES.

Meet-in-the-Middle attack:

This algorithm is based on the observation that, if we have $C = E(K_2, E(K_1, P))$ then, $X = E(K_1, P) = D(K_2, C)$. Given a known pair (P,C), the attack proceeds as follows:

→ First, encrypt P for all 2^{56} possible values of K_1.
→ Store these results in a table and then sort the table by the values of X.
→ Decrypt C using all 2^{56} possible values of K_2, and check the result against the table for a match.
→ If a match occurs, then test the two resulting keys against a new known plaintext-ciphertext pair.
→ If the two keys produce the correct ciphertext, accept them as the correct keys.

Triple DES:

Triple DES is a encryption technique which uses three instances of DES on same plaintext. It uses three different types of key choosing technique in first, all used keys are different. In second, two keys are same and one is different. In third, all keys are same.

Triple DES is also vulnerable to meet-in-the middle attack because of which it give total security level of 2^{112} instead of using using 168 bit of key. The block collision attack can also be done because of short block size and using the same key to encrypt large size of text.

⊛ Groups:-

A group G, sometimes denoted by $\{G, \cdot\}$, is a set of elements with a binary operation denoted by $\cdot$ that associates to each ordered pair (a,b) of elements in G and element $(a \cdot b)$ is also in G, such that the following axioms are obeyed:

nothing just notation for axiom 1, axiom 2, ...

(A1) Closure: If a and b belong to G, then $a \cdot b$ is also in G.

(A2) Associative: $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ for all a, b, c in G.

(A3) Identity element: There is an element e in G such that $a \cdot e = e \cdot a = a$ for all a in G.

(A4) Inverse element: For each a in G, there is an element a' in G such that $a \cdot a' = a' \cdot a = e$.

Note: The operator $\cdot$ is generic and can refer to addition, multiplication, or some other mathematical operation. a' is an inverse of a.

Abelian Group: A group is said to be abelian if it satisfies the following additional condition:

(A5) Commutative: $a \cdot b = b \cdot a$ for all a, b in G.

Cyclic Group: A group G is cyclic if every element of G is a power a^k (k is an integer) of fixed element $a \in G$. A cyclic group is always abelian and may be finite or infinite.

⊛ Rings:

A ring R, sometimes denoted by $\{R, +, \times\}$, is a set of elements with two binary operations, called addition and multiplication, such that for all a,b,c in R the following axioms are obeyed:

(A1–A5) for the addition: i.e, R is an abelian group with respect to addition.

(M1) Closure under multiplication: If a and b belong to R, then ab is also in R.

(M2) Associativity of multiplication: $a(bc)=(ab)c$ for all a,b,c in R.

(M3) Distributive laws: $a(b+c)=ab+ac$ for all a,b,c in R.

$(a+b)c = ac+bc$ for all a,b,c in R.

⇒ A ring is said to be **commutative** if it satisfies the following additional condition:

(M4) Commutativity of multiplication: $ab=ba$ for all a,b in R.

⇒ A ring is said to be **integral domain** if it satisfies the following additional conditions:

(M5) Multiplicative identity: There is an element 1 in R such that $a1=1a=a$ for all a in R.

(M6) No zero divisors: If a,b in R and $ab=0$, then either $a=0$ or $b=0$.

⊛ Fields:

A field F, sometimes denoted by $\{F, +, \times\}$, is a set of elements with two binary operations, called addition and multiplication, such that for all a,b,c in F the following axioms are obeyed.

(A1–M6) F is an integral domain: i.e, F satisfies axioms A1 through A5 and M1 through M6.

(M7) Multiplicative inverse: For each a in F, except 0, there is an element $\bar{a}^{-1}$ in F such that $a\bar{a}^{-1} = (\bar{a}^{-1})a = 1$.

⊛ Greatest Common Divisor (GCD):

The polynomial c(x) is said to be the greatest common divisor of a(x) and b(x). if the following are true.

i) c(x) divides both a(x) and b(x).

ii) Any divisor of a(x) and b(x) is a divisor of c(x).

An Equivalent definition is the following:

gcd [a(x), b(x)] is the polynomial of maximum degree that divides both a(x) and b(x).

⊛ Euclidean Algorithm:-

Euclidean Algorithm is used to compute the greatest common divisor of two polynomials. The following equation can be repeatedly used to determine the gcd.

$$\boxed{\gcd[a(x), b(x)] = \gcd[b(x), a(x) \bmod b(x)]}$$

This algorithm assumes that the degree of a(x) is greater than degree of b(x).

Example: Find gcd [a(x), b(x)] for $a(x) = x^6+x^5+x^4+x^3+x^2+x+1$ and $b(x) = x^4+x^2+x+1$.

Solution: First we divide a(x) by b(x):

$$\begin{array}{r|l} & x^2+x \\ \hline x^4+x^2+x+1 & x^6+x^5+x^4+x^3+x^2+x+1 \\ & x^6 \quad +x^4+x^3+x^2 \\ \hline & x^5 \qquad\quad +x+1 \\ & x^5 \quad +x^3+x^2+x \\ \hline & x^3+x^2 \quad +1 \end{array}$$

This yeilds $r_1(x) = x^3+x^2+1$ and $q_1(x) = x^2+x$.

Now, we divide b(x) by $r_1(x)$.

$$\begin{array}{r|l} & x+1 \\ \hline x^3+x^2+1 & x^4 \quad +x^2+x+1 \\ & x^4+x^3 \quad +x \\ \hline & x^3+x^2 \quad +1 \\ & x^3+x^2 \quad +1 \\ \hline & \times \quad \times \end{array}$$

This yeilds $r_2(x) = 0$ and $q_2(x) = x+1$

Therefore, $\gcd[a(x), b(x)] = r_1(x) = x^3+x^2+1$.

⊛ Modular Arithmetic:

If a is an integer and n is a positive integer, we define a mod n to be the remainder when a is divided by n. The integer n is called the modulus.

For example:- 11 mod 7 = 4 and -11 mod 7 = 3.

> -ve [illegible] given [illegible]
> $a = q*n + r$
> or, $-11 = q*7 + r$
>
> now find q and r so that it will be -11. Here q can be -1, -2, -3, ... & r must be positive number between 0, 1, 2, ..., n-1.
> Hence $-11 = -2*7 + 3$ ∴ $r = 3$

Two integers a and b are said to be congruent modulo n, if (a mod n) = (b mod n). Symbolically, we express such a congruence by $a \equiv b \pmod{n}$.

⊛ Set of Residue (Z_n):

→ Define Z_n as a set of non-negative integers less than n. i.e, $Z_n = \{0, 1, 2, \ldots, n-1\}$.

→ This is required to as a set of residues or residue class m an integers between 0 to n-1.

→ The result of a (mod n) is always a non-negative integers less than n.

→ Actually, Z_n is a commutative ring.

→ Using extended euclidean algorithm to find multiplicative inverse.

⊛ Why is Z_n not an integral domain?

⇒ Even though Z_n possess a multiplicative identity, it does not satisfy the other condition of integral domain which says that if $a \times b = 0$ then either a or b must be zero. Consider modulo 8 arithmetic. We have $2 \times 4 = 0$, which is a clear violation of the second rule for integral domains.

Q. Find gcd (15, 26). [Using Extended Euclidean Algorithm].

Solution:

q	a	b	r
0	15	26	15
1	26	15	11
1	15	11	4
2	11	4	3
1	4	3	1
03	3	1	0

15 আছে 26 এ divide হচ্ছে না so, q=0 & r=a

a=b & b=r
26 আছে 15 এ divide হচ্ছে quotient 1 এ remainder and so on...

last non-zero value of r is gcd

∴ gcd (15, 26) = 1.

Residue classes:-

In modular arithmetic, a residue of an integer a in modulo n is the unique value of $0 \le r \le n-1$ such that $a = kn + r$. In the context of division, a residue is simply a remainder.

A residue class is a complete set of integers that are congruent modulo n for some positive integer n. In modulo n, there are exactly n different residue classes, corresponding to the n possible residues $\{0, 1, 2, 3, \ldots, n-1\}$. Each residue class contains all integers in the form kn+r where r is the corresponding residue.

Quadratic residues:-

An integer $x \in Z_N$ is a quadratic residue of N if x has a square root modulo N; that is, if there exists a value $y \equiv \sqrt{x} \pmod{n} \in Z_n$ that satisfies $y^2 = x \pmod{n}$.

where, N is the product of two prime numbers
i.e, $N = pq$.

Congruence:-

The mapping from Z to Z_n is not one-to-one infinite member of Z can map to a single member of Z_n.

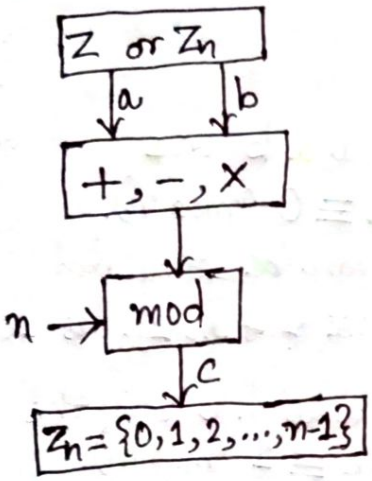
As Example: 12 mod 10 = 2,
22 mod 10 = 2,
132 mod 10 = 2 and so on

In the above example the value 10, 22 and 132 are called congruent of mod 10. The operator (≡) is used to represent congruence. The phrase (mod n) to the right side of the congruence is used to define the value of modulus for which the congruence relationship valid.

$$2 = 12 \pmod{10}$$
$$13 = 23 \pmod{10}$$
$$-8 = 12 \pmod{10}.$$

Operations on Z_n :-

The three binary operations (addition, subtraction, and multiplication) can be defined for the set Z_n. The result may need to be mapped to Z_n using the mod operator as shown in figure below:



Operations:

$(a+b) \bmod n = c$
$(a-b) \bmod n = c$
$(a \times b) \bmod n = c$.

Fig. Binary operations in Z_n

Actually two sets of operators are used here. The first set is one of binary operators (+, −, ×); the second is the mod operator. We need to use parentheses to emphasize the order of operations.

Example:- Perform following operations (the inputs come from Z or Z_n).

a) Add 7 to 14 in Z_{15}
b) Subtract 11 from 7 in Z_{13}
c) Multiply 11 by 7 in Z_{20}.

Solution:- The following shows the two steps involved in each case:

$(14+7) \bmod 15 \longrightarrow (21) \bmod 15 = 6$

$(7-11) \bmod 13 \longrightarrow (-4) \bmod 13 = 9$

$(7 \times 11) \bmod 20 \longrightarrow (77) \bmod 20 = 17$

⊛ Properties of Z_n:

First Property: $(a+b) \bmod n = [(a \bmod n) + (b \bmod n)] \bmod n$.

Second Property: $(a-b) \bmod n = [(a \bmod n) - (b \bmod n)] \bmod n$.

Third Property: $(a \times b) \bmod n = [(a \bmod n) \times (b \bmod n)] \bmod n$.

Example: The following shows the application of the above properties:

1. $(1,723,345 + 2,124,945) \bmod 11 = (8+9) \bmod 11 = 6$
2. $(1,723,345 - 2,124,945) \bmod 11 = (8-9) \bmod 11 = 10$
3. $(1,723,345 \times 2,124,945) \bmod 11 = (8 \times 9) \bmod 11 = 6$.

Since -ve
$-1 \bmod 11$
So, $a = q*n + r$
$-1 = q*11 + r$
or, $-1 = -1*11 + 10$
or, $-1 = -1$
$\therefore r = 10$

⊛ Inverses:-

Additive inverse:- In Z_n, two numbers a and b are additive inverses of each other if $a+b \equiv 0 \pmod n$.

In Z_n, the additive inverse of a can be calculated as $b = n-a$. For example, the additive inverse of 4 in Z_{10} is $10-4=6$.

Multiplicative inverse:- In Z_n, two numbers a and b are the multiplicative inverse of each other if $a \times b \equiv 1 \pmod n$. For example, if the modulus is 10, then the multiplicative inverse of 3 is 7. In other words, we have $(3 \times 7) \bmod 10 = 1$.

⊛ Extended Euclidean Algorithm:- [Imp]

The extended euclidean algorithm for any given integer a and b is not only calculate the gcd a,b = d but also additional two integers s and t such that following equation must satisfy:

$$sa + tb = d = \gcd(a,b).$$

Example: Express $\gcd(252,198) = 18$ as a linear combination of 252 and 198.

Solution:- Using Euclidean algorithm,

$$252 = 1 \times 198 + 54$$
$$198 = 3 \times 54 + 36$$
$$54 = 1 \times 36 + 18$$
$$36 = 2 \times 18 + 0$$

$\therefore \gcd(252,198) = 18$

Now, we have to find the two integers i.e, s & t which is a linear combination of 'a' & 'b' to find the gcd (a,b).

Now, from 3rd division,

$$18 = 54 - 1 \times 36 \quad \text{---(i)}$$

Again, from 2nd division,

$$36 = 198 - 3 \times 54 \quad \text{---(ii)}$$

Now, substitute the value of eqn (ii) on eqn (i).

$$18 = 54 - 1(198 - 3 \times 54)$$

$$\text{or, } 18 = 54 - 1 \times 198 + 3 \times 54$$

$$\text{or, } 18 = 4 \times 54 - 1 \times 198 \quad \text{---(iii)}$$

Again, from 1st division,

$$54 = 252 - 1 \times 198 \quad \text{---(iv)}$$

Substitute the value of eqn (iv) on eqn (iii).

$$18 = 4(252 - 1 \times 198) - 1 \times 198$$

$$\text{or, } 18 = 4 \times 252 - 4 \times 198 - 1 \times 198$$

$$\text{or, } 18 = 4 \times 252 - 5 \times 198 \quad \text{---(v)}$$

Now, compare the eqn (v) with extended euclidean algorithm

i.e, $sa + tb = d$

we get $s = 4$ & $t = -5$.

⊛ Finite Field / Galois Fields:-

A finite field or galois field contains a finite number of element. A finite field can be represented as p^n where n is positive integer and p is prime number. The number of element of a finite field is called its order. The finite field of order p^n is generally written as $GF(p^n)$.

Two special cases of $GF(p^n)$:

i) If $n = 1$ then $GF(p)$.

ii) If $n > 1$ then $GF(p^n)$.

If $p = 2$ then $GF(2^n)$.

Finite field of order p:

For the given prime number p, the finite field of order p, $GF(p)$ is defined as the set Z_p of integers $Z_p = \{0, 1, \ldots, (p-1)\}$ together

with the arithmetic operation modulo 'p'.

Example : GF(7)

[Imp] $2^7 = \{0,1,2,3,4,5,6\}$

Addition modulo 7

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

W → just a mark to understand better
• we have done mode 7 each time.

Multiplication modulo 7

X	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

Additive and multiplicative inverses modulo 7

Additive inverse ← Z_7 → $-W$; W^{-1} ← multiplicative inverse

W	−W	W^{-1}
0	0	—
1	6	1
2	5	4
3	4	5
4	3	2
5	2	3
6	1	6

Look at tables above i.e. Additive inverse of 0 is 0 and there is no multiplicative inverse of 0. Similarly, the additive inverse of 1 is 6 and multiplicative inverse is 1 and so on.

Q1. Find the multiplicative inverse of 11 in Z_{26}. [Imp]

[Imp] Solution :-

q	r_1	r_2	r	t_1	t_2	t
2	26	11	4	0	1	−2
2	11	4	3	1	−2	5
1	4	3	1	−2	5	−7
3	3	1	0	5	−7	26
X	1	0	X	−7	26	X

Always initially assume $t_1 = 0$ & $t_2 = 1$
q = quotient, r = remainder

$t = t_1 - t_2 \times q$
$= 0 - 1 \times 2$
$= -2$

Now shift r_2, r, t_2, t to one step left and continue process until $r_2 = 0$.

Hence, the gcd = 1 and multiplicative inverse is −7.
Since solution is in negative we can also get positive solution as; 26 + (−7) = 19.

[Imp]

Q2. What do you mean by multiplicative inverse? Find multiplicative inverse of each nonzero elements in Z_{11}.

Answer: In Z_n two numbers a and b are the multiplicative inverse of each other if $a \times b \equiv 1 \pmod{n}$. For example, if the modulus is 10, then the multiplicative inverse of 3 is 7. In other words, we have $(3 \times 7) \bmod 10 = 1$.

Finding multiplicative inverse of each nonzero elements in Z_{11}.

Since, 1, 2, 3, 4, 5, 6, 7, 8, 9 & 10 are the nonzero elements in Z_{11}.

So,

inverse of 1 = 1

i.e, a × multiplicative inverse mod n = 1

$1^{-1} = 1$

$2^{-1} = 6$, Since $(2 \times 6) \bmod 11 = 12 \bmod 11 = 1$

$3^{-1} = 4$, Since $(3 \times 4) \bmod 11 = 12 \bmod 11 = 1$

$4^{-1} = 3$, Since $(4 \times 3) \bmod 11 = 12 \bmod 11 = 1$

$5^{-1} = 9$, Since $(5 \times 9) \bmod 11 = 45 \bmod 11 = 1$

$6^{-1} = 2$, Since $(6 \times 2) \bmod 11 = 12 \bmod 11 = 1$

$7^{-1} = 8$, Since $(7 \times 8) \bmod 11 = 56 \bmod 11 = 1$

$8^{-1} = 7$, Since $(8 \times 7) \bmod 11 = 56 \bmod 11 = 1$

$9^{-1} = 5$, Since $(9 \times 5) \bmod 11 = 45 \bmod 11 = 1$

$10^{-1} = 10$, Since $(10 \times 10) \bmod 11 = 100 \bmod 11 = 1$

(*) Why do we need to study polynomial arithmetic?

Ans:- We study polynomial arithmetic so that we can represent a bit pattern by a polynomial in, say the variable x. Each power of x in the polynomial can stand for bit position in a bit pattern.

For Example:- We can represent the bit pattern 111 by the polynomial x^2+x+1. Similarly, the bit pattern 101 would be represented by the polynomial x^2+1 & the pattern 011 by the polynomial $x+1$.

⊛. Polynomial Arithmetic:-

In general, a polynomial is an expression of the form

$$a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0.$$

for some non-negative integer n and $a_0, a_1, \ldots, a_n$ are cofficients.

Polynomial arithmetic deals with addition, subtraction, multiplication and division of polynomials. Ordinary polynomial arithmetic uses basic rules of algebra. Let we have, $f(x) = x^3 + x^2 + 2$ and $g(x) = x^2 - x + 1$ then,

$$f(x) + g(x) = x^3 + 2x^2 - x + 3$$
$$f(x) - g(x) = x^3 + x + 1$$
$$f(x) \times g(x) = x^5 + 3x^2 - 2x + 2$$

Polynomial arithmetic over GF(2):

Q. Calculate the result of the following polynomial if they are over GF(2). [Imp]

$$(x^4 + x^2 + x + 1) + (x^3 + 1)$$
$$(x^4 + x^2 + x + 1) - (x^3 + 1)$$
$$(x^4 + x^2 + x + 1) \times (x^3 + 1)$$
$$(x^4 + x^2 + x + 1) / (x^3 + 1)$$

Since GF(2) = {0,1} i.e, 0 to n-1

Add यदि coefficient ≥ 2 आए तब mod 2 गरेर लेख्ने / for eg. $2x \bmod 2 = 0$

Space को ठाउमा 0 राख्ने यदि छ भने for eg $0.x^3$ i.e, $x^4 + 0.x^3 + x^2 + 1$

Solution:-

For $(x^4 + x^2 + x + 1) + (x^3 + 1)$
(i.e, Addition)

$$\begin{array}{r} x^4 + x^2 + x + 1 \\ +(x^3 + x + 1) \\ \hline x^4 + x^3 + x^2 \end{array}$$

For $(x^4 + x^2 + x + 1) - (x^3 + 1)$
(i.e, Subtraction)

$$\begin{array}{r} x^4 + x^2 + x + 1 \\ -(x^3 + x + 1) \\ \hline x^4 - x^3 + x^2 \end{array}$$

For $(x^4 + x^2 + x + 1) \times (x^3 + 1)$
(i.e, Multiplication)

$$\begin{array}{r} x^4 + x^2 + x + 1 \\ \times (x^3 + x + 1) \\ \hline x^4 + x^2 + x + 1 \\ x^5 + x^3 + x^2 + x \\ \hline x^7 + x^5 + x^4 + x^3 \\ \hline x^7 + 1 \end{array}$$

x^2 दुईन माथि space छ so, x^2 को एक space थप हुन्छ।

For $(x^4 + x^2 + x + 1) / (x^3 + 1)$
(i.e, Division)

$$\begin{array}{r|l} & x \\ \hline x^3 + 1 & x^4 + x^2 + x + 1 \\ & x^4 + x \\ \hline & x^2 + 1 \end{array}$$

similarly over the field Z_7 आए value हरु add गर्दा (0,1,2,3,4,5,6) भन्दा सकदैन But if ≥ 7 आए mod 7 गरेर लेख्नि त्यहि हो अरु सबै normal basic algebra rules हुन्।

⊛ International Data Encryption Algorithm (IDEA):

Key Generation:- The 128-bit key is used to generate 52 sub keys of length 16 bits. 6 keys are used for each round and the remaining 4 keys are used in final half round. The figure below shows the key generation mechanism where we start getting the subkeys from the starting position of the 128 bits key, so in first round we get 8 subkeys. In each round of sub key generation 128 bits keys undergoes a 25-bit position right to generate next set of keys.

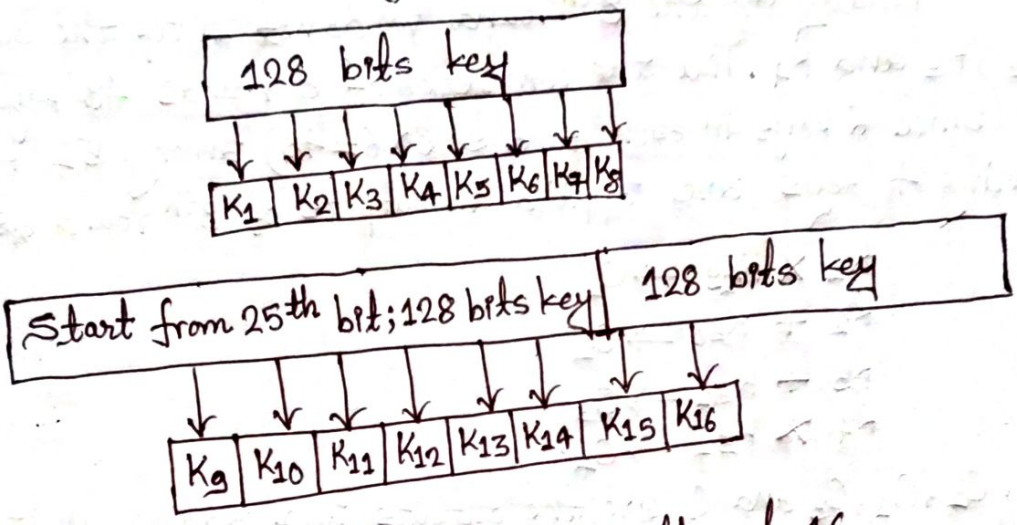


Fig: Generation of keys 1 through 16.

Encryption:- IDEA algorithm uses total of 8 rounds and in each round following steps are used:

[Imp] round operations in IDEA

1) $P_1 \times K_1$
2) $P_2 + K_2$
3) $P_3 + K_3$
4) $P_4 \times K_4$
5) Step1 ⊕ Step3
6) Step2 ⊕ Step4
7) Step5 × K_5
8) Step6 + Step7
9) Step8 × K_6
10) Step7 + Step9
11) Step1 ⊕ Step9 → R_1
12) Step3 ⊕ Step9 → R_2
13) Step2 ⊕ Step10 → R_3
14) Step4 ⊕ Step10 → R_4

Explanation: IDEA algorithm firstly divides the plaintext into 4 parts P_1, P_2, P_3 & P_4. Then, first part P_1 is multiplied with key 1, second part P_2 is added with key 2, third part P_3 is added with key 3 and fourth part P_4 is multiplied with key 4. Now XOR operation is performed between the odd steps: i.e, step1 and step3. Similarly, XOR operation is performed between the even steps: i.e, step 2 and step 4.

Now, the output of step5 is multiplied with key 5. Then, the output of step6 and step7 are added, Output of step8 is multiplied with key 6 and Output of step 7 and step 9 are added.

Now, From steps 1 to 4, the odd steps 1 and 3 are done XOR operation with step 9 and the even steps 2 and 4 are done XOR operation with step 10. From steps 11, 12, 13 and 14 we we get 4 encrypted outputs R_1, R_2, R_3 and R_4 which are the outputs of our first round. Now, R_2 and R_3 will get swapped and they will be applied as the input of second round P_1, P_2, P_3 and P_4 as; R_1, R_3, R_2 and R_4 respectively. Then, again those 14 steps will continue and the process will continue upto 8 rounds.

Finally in the 8th round we do not swap the outputs R_1, R_2, R_3 and R_4. Till the completion of 8 rounds 48 keys are used (since 6 keys in each round so, $8 \times 6 = 48$) among 52 keys and remaning 4 keys are applied to outputs of 8th round as follows:-

$$R_1 \times K_{49} \rightarrow C_1$$
$$R_2 + K_{50} \rightarrow C_2$$
$$R_3 + K_{51} \rightarrow C_3$$
$$R_4 \times K_{52} \rightarrow C_4$$

where, C_1, C_2, C_3 and C_4 are the outputs and adding all them we get the ciphertext 'C'.

⊛ Advanced Encryption Standard (AES):-

AES is a block cipher intended to replace DES for commercial applications. It uses a 128-bit block size and key size of 128, 192 or 256 bit. AES does not use Fiestel structure instead each full round consists of four seperate function. (Byte Substitution, Permutation, arithmetic operation over finite field and X-OR with key).

AES is found at least six times faster than triple DES. A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback but it was found slow. Hence, even though we have strong algorithm like 3-DES, still AES is preferred as a resonable candidate for long term use. The figure below shows the overall structure of AES encryption and decryption:

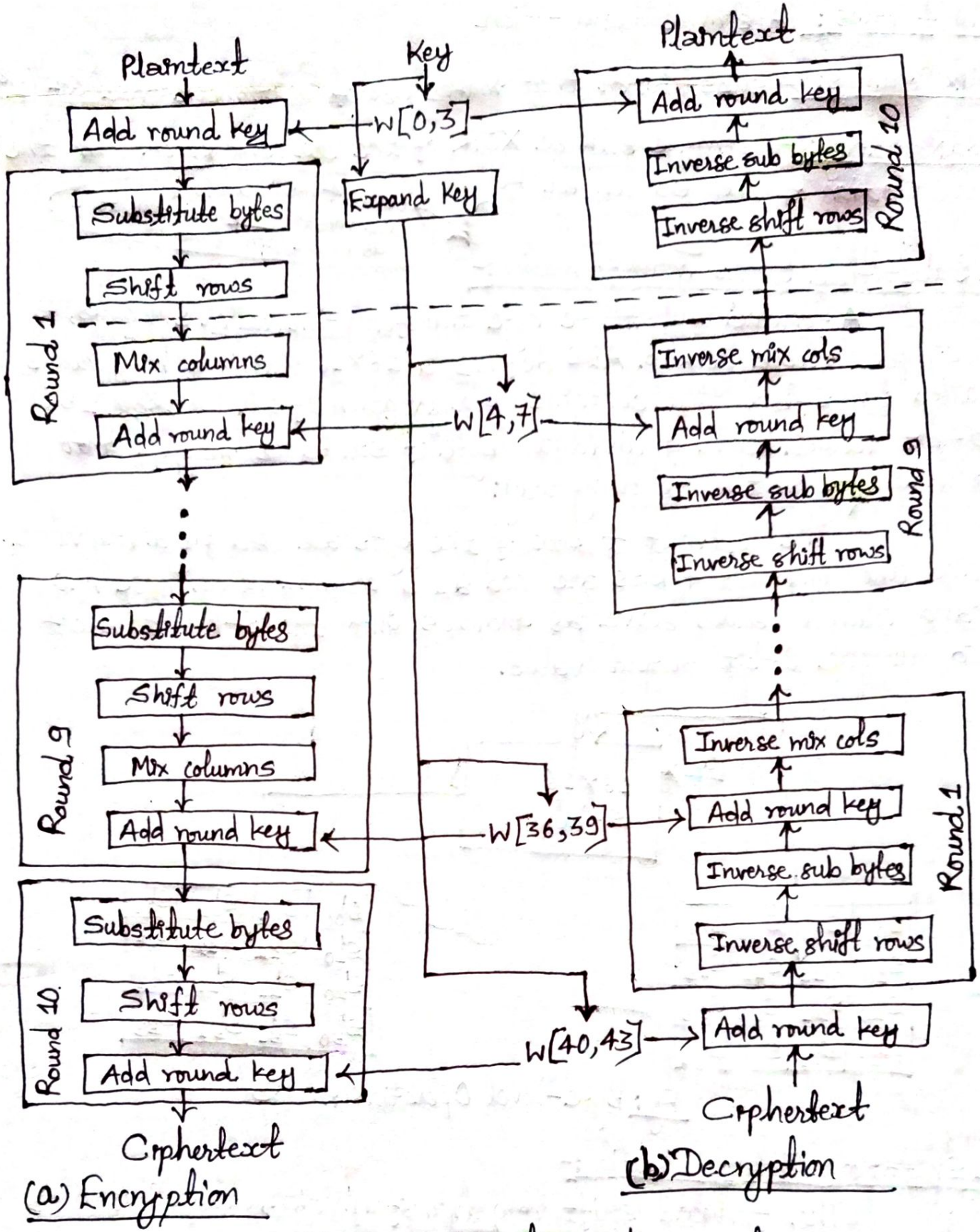


Fig: AES Encryption and Decryption

For both encryption and decryption, the cipher begins with an AddRoundKey stage, followed by nine rounds that each includes all four stages, followed by a tenth round of three stages as shown in figure above. The four stages are:

Substitute bytes: Uses an S-box to perform a byte-by-byte substitution of the block.

Shift rows: A simple permutation

Mix Columns: A substitution that makes use of arithmetic over GF (2^8).

AddRoundKey: A simple bitwise XOR of the current block with the portion of the expanded key.

below 4 points a,b,c,d are mostly imp from this topic as exam point of view

a) Substitute Bytes Transformation:-

A forward substitute byte transformation, called SubBytes, is a simple table lookup. AES defines a 16×16 matrix of byte values, called an S-box that contains permutation of all possible 256 8-bit values. Each individual byte of state is mapped into a new byte in the following way:

The leftmost 4 bits of the byte are used as a row value and the rightmost 4 bits are used as a column value. These row and column values serve as indexes into the S-box to select a unique 8-bit output value.

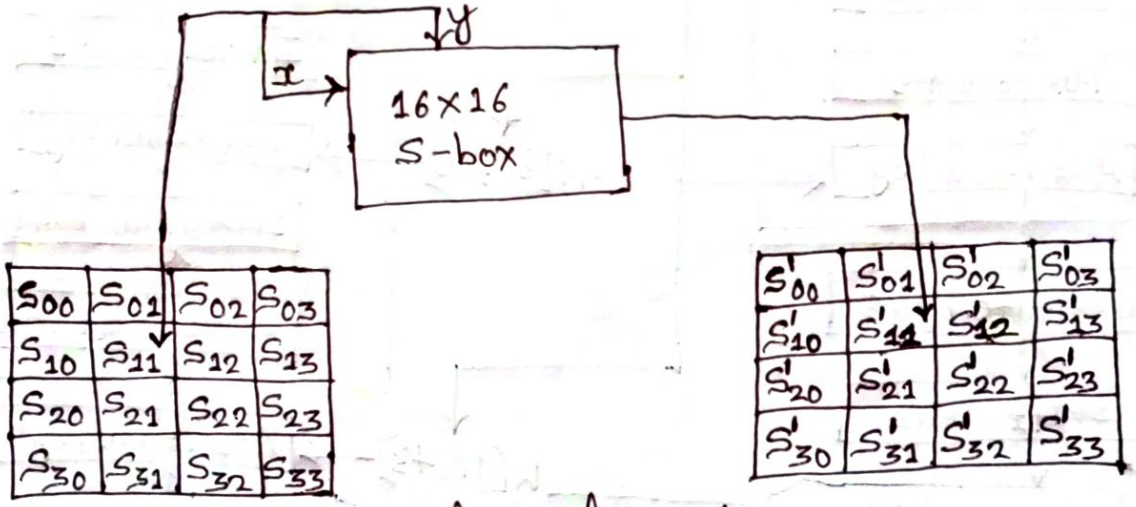


Fig: Byte-Level Operations in AES

b) Shift Rows Transformation:-

The forward shift row transformation, called ShiftRows, is illustrated in the figure below. The first row of state is not altered. For the second row, a 1-byte circular left shift is performed. For the third row, a 2-byte circular left shift is performed. For the fourth row, a 3-byte circular left shift is performed.

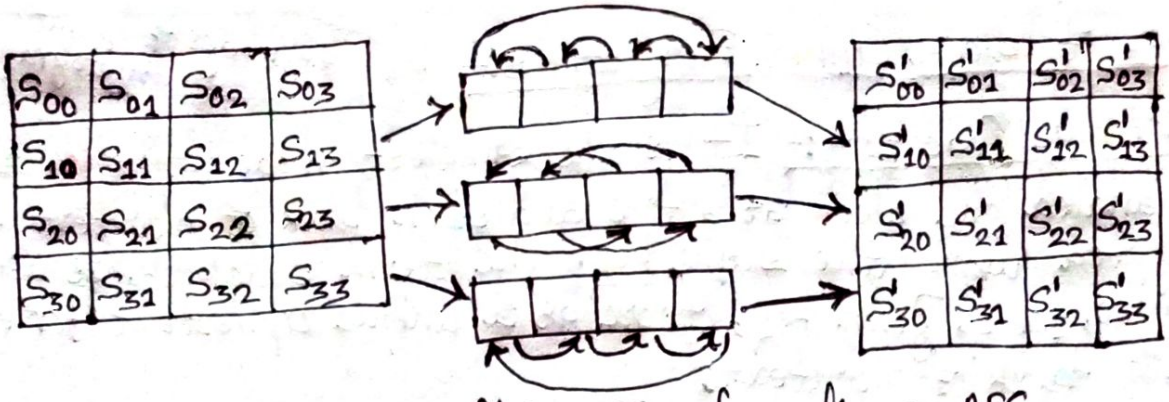


Fig: ShiftRow Transformation in AES.

c) Mix Columns Transformation:-

The forward mix column transformation, called MixColumns, operates on each column individually. Each byte of a column is mapped into a new value that is a function of all four bytes in that column.

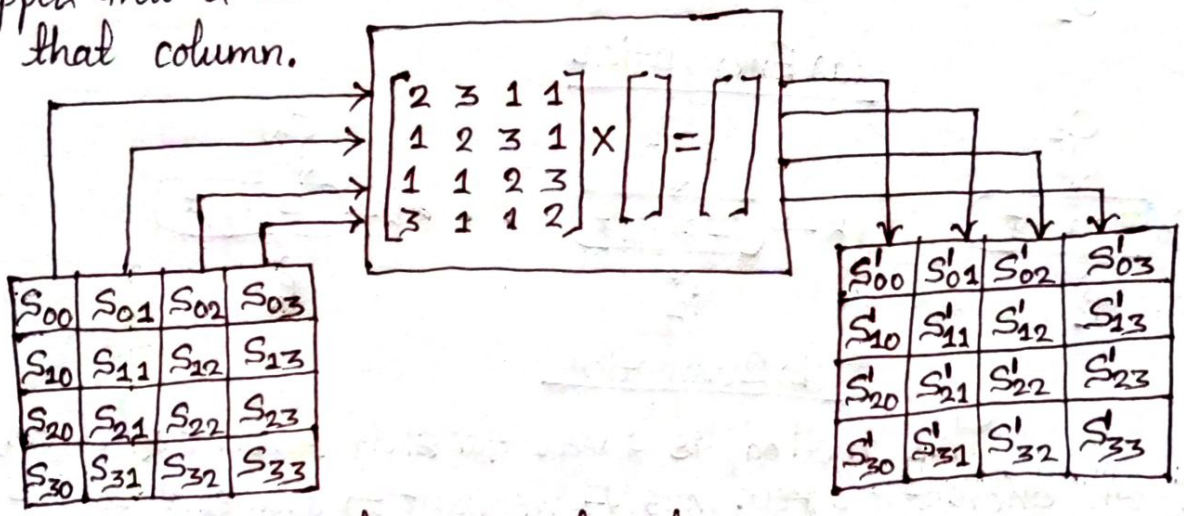


Fig: MixColumn Transformation in AES

d) AddRoundKey Transformation:-

In the forward add round key transformation, called AddRoundKey, the 128 bits of State are bitwise XORed with the 128 bits of the round key. As shown in figure below, the operation is viewed as a column wise operation between the 4 bytes of a State column and one word of the round key; it can also be viewed as a byte-level operation.

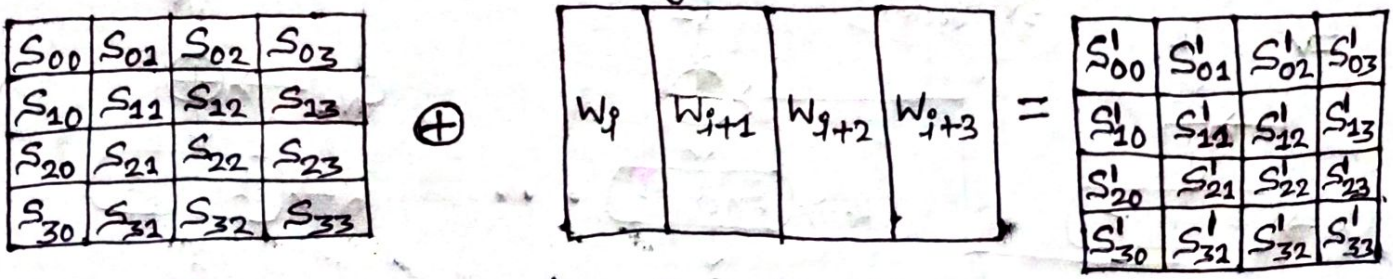
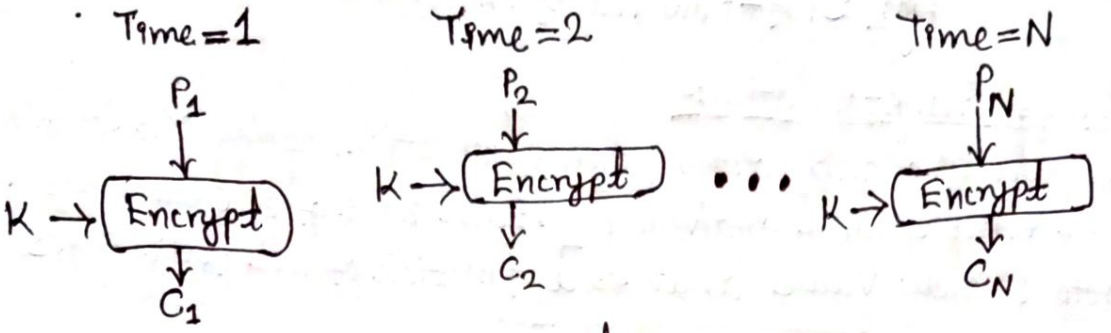


Fig: AddRoundKey transformation in AES.

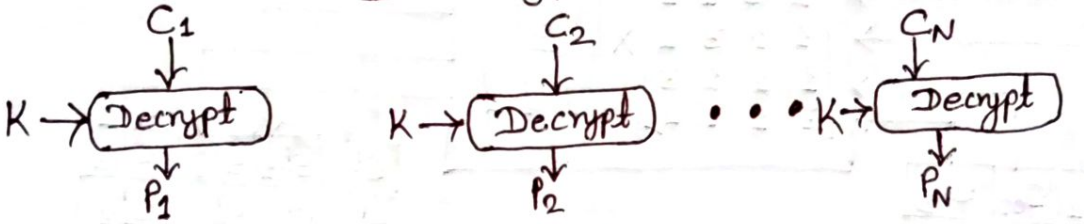
⊛ Modes of Block Cipher Encryptions:-

(i.e, modes of operations in cryptography)

⊛ Electronic Code Book:- It is the simplest mode in which plaintext is handled one block at a time and each block of plaintext is encrypted using the same key. The term codebook is used because, for a given key, there is a unique ciphertext for every b-bit block of plaintext.



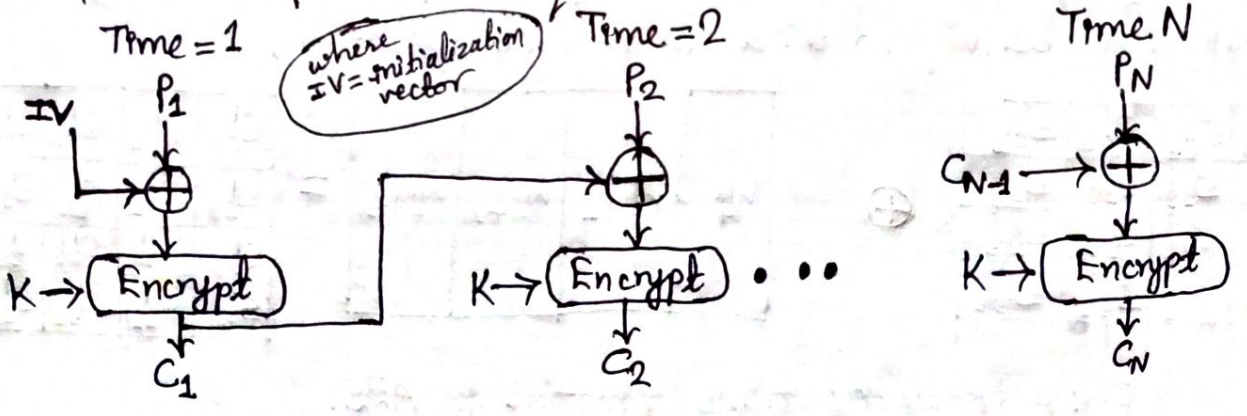
(a) Encryption



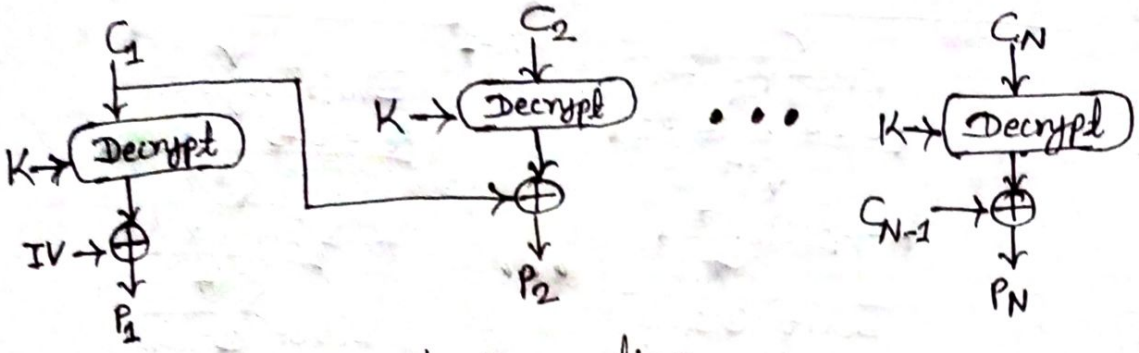
(b) Decryption

This method is ideal for short amount of data, such as an encryption key. Thus, if we want to transmit a DES key securely, ECB is the appropriate mode to use.

⊛ Cipher Block Chaining Mode:- In this technique, the input to the encryption algorithm is the XOR of the current plaintext block and the preceding ciphertext block; the same key is used for each block. In effect, we have chained together the processing of the sequence of plaintext blocks.



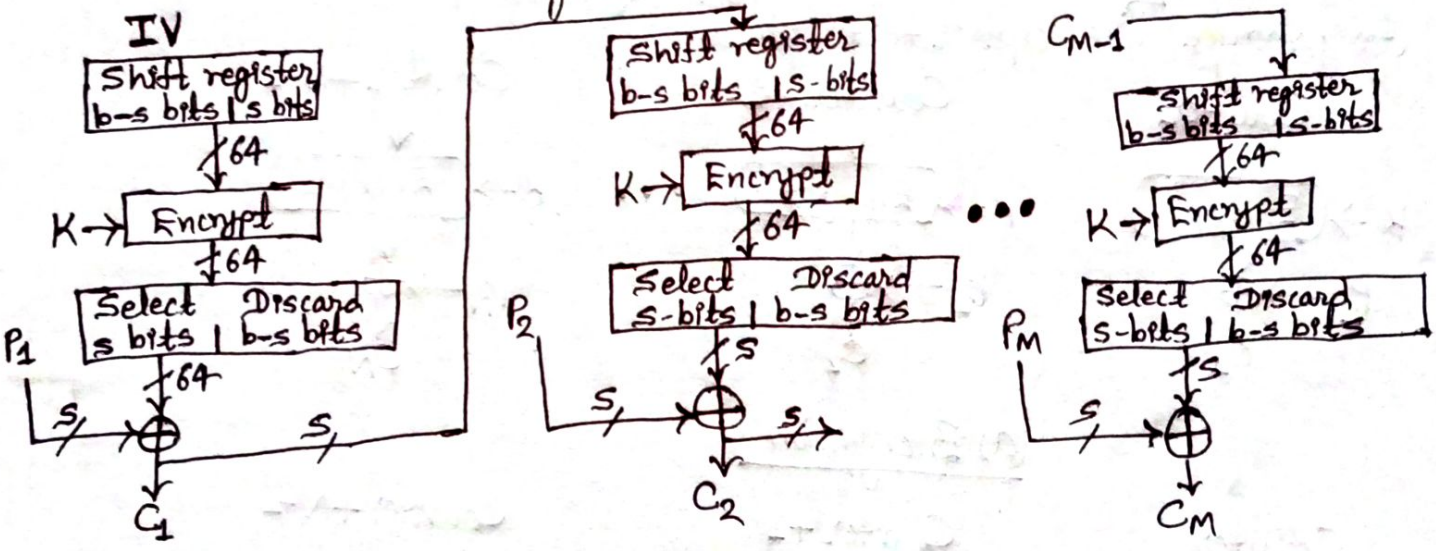
(a) Encryption



(b) Decryption

It is used to overcome the security deficiencies of ECB and is appropriate mode for encrypting messages of length greater than b bits.

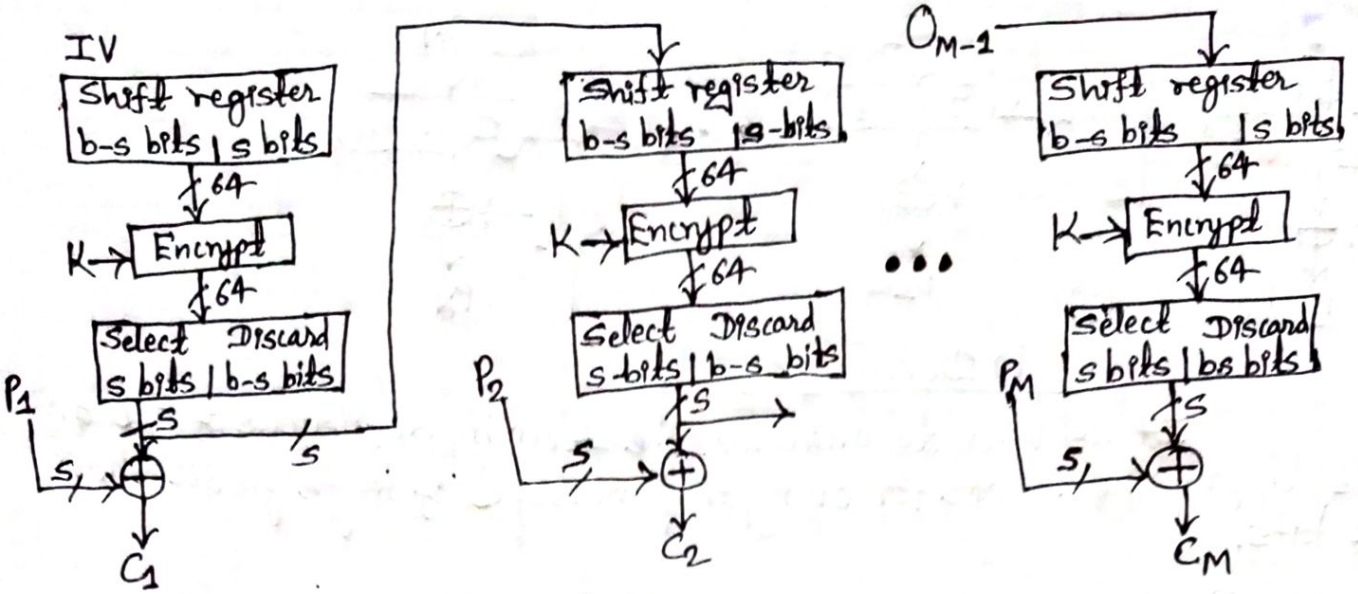
⊛ Cipher Feedback Mode:- In this mode, each cipher text block gets feedback into the encryption process, inorder to encrypt the next plain text block. The CFB mode requires an ~~init~~ initialization vector (IV) as the initial random b bit input block. Data is encrypted in the shift register with the key K, taking only S number of most significant bit.



Encryption

⊛ Output Feedback Mode:-

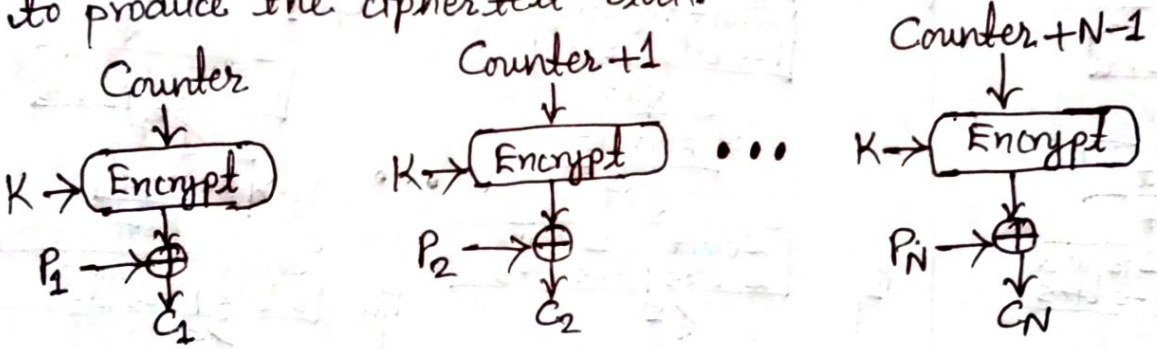
The output feedback mode (OFB) is similar in structure to that of CFB. The only difference is that the output of the encryption function that is fed back to the shift register in OFB, whereas in CFB the ciphertext unit is fed back to the shift register.



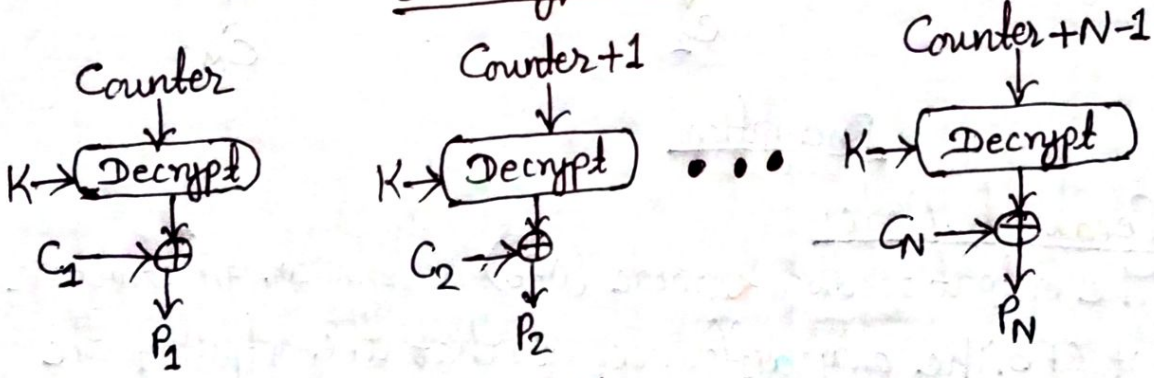
Encryption

⊛ Counter Mode:-

A counter equal to the plaintext block size is used in this mode. The counter is initialized to some value and then increment by 1 for each block. The counter is encrypted with the key K and XOR the output of the plaintext block to produce the ciphertext block.



(a) Encryption



(b) Decryption