Block cipher modes of operation:

A block cipher is an encryption algorithm that processes data in fixed-size blocks rather than one bit at a time. Block cipher modes of operation define how to securely encrypt and decrypt large amounts of data using a block cipher.

Here are a few common modes:

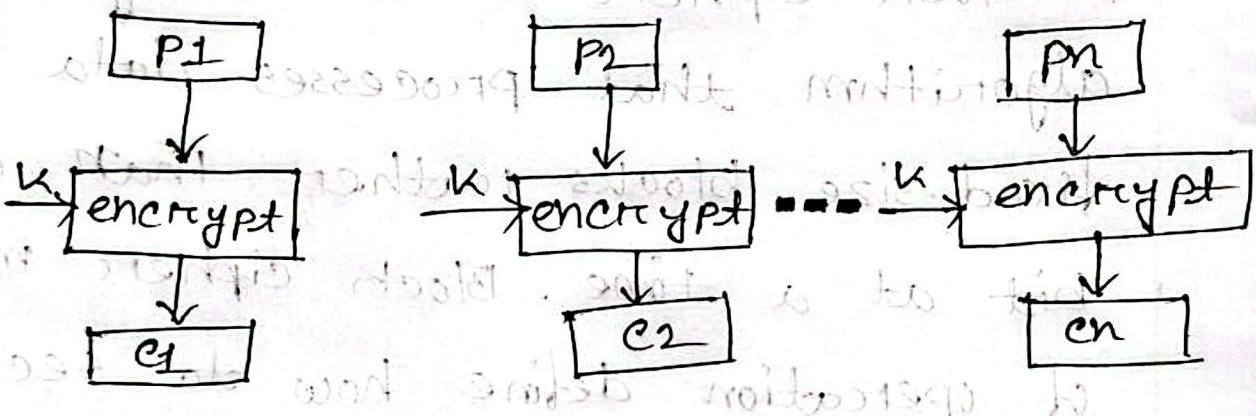
Electronic code Block (ECB):

The electronic codebook is the easiest block cipher mode of functioning. It is easier because of the direct

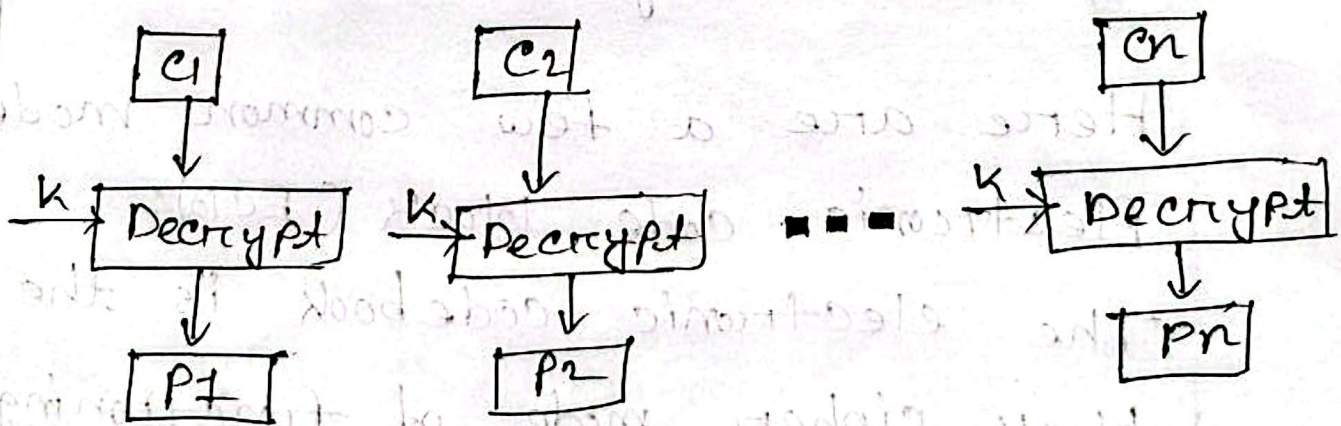
encryption of each block of input plaintext and output is in the form of blocks of encrypted ciphertext

Diagram:

Encryption:



Decryption:



Advantages of ECB:

- Parallel encryption of blocks of bits is possible, thus it is a faster way of encryption
- simple way of the block cipher

Disadvantages of ECB:

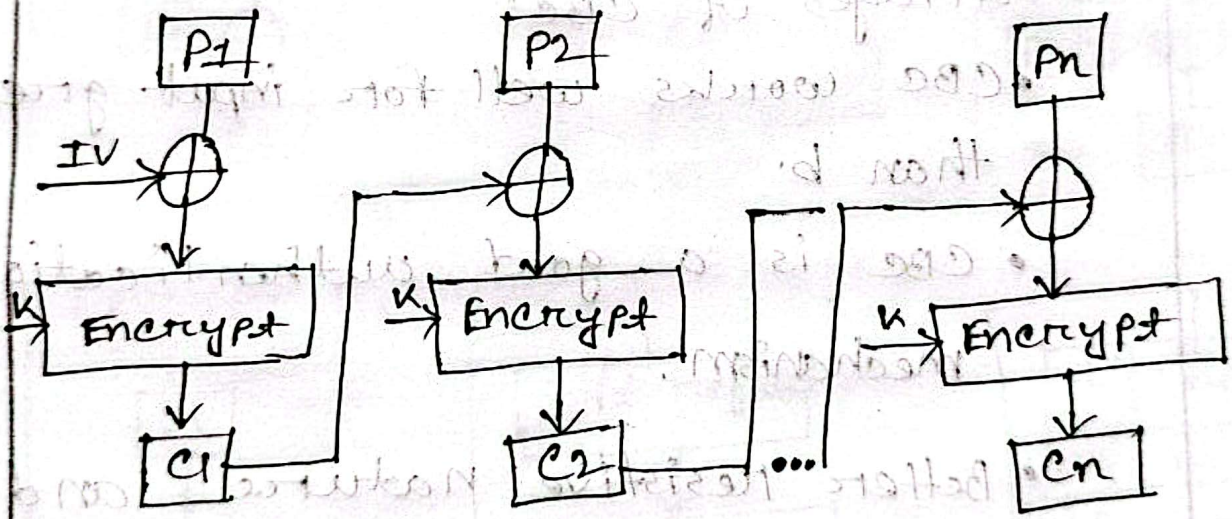
- Prone to cryption of blocks of bits is possible, thus it is a faster way of encryption.
- simple way of the block cipher

Cipher Block chaining (CBC)

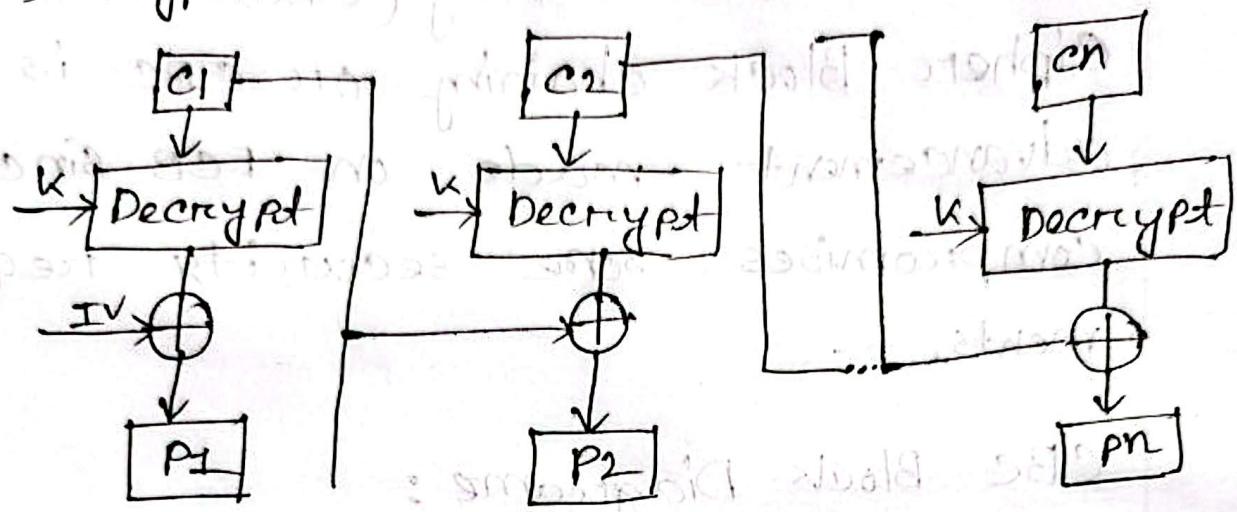
Cipher Block chaining or CBC is an advancement made on ECB since ECB compromises some security requirements.

CBC Block Diagrame:

Encryption



Decryption:



Advantages of CBC:

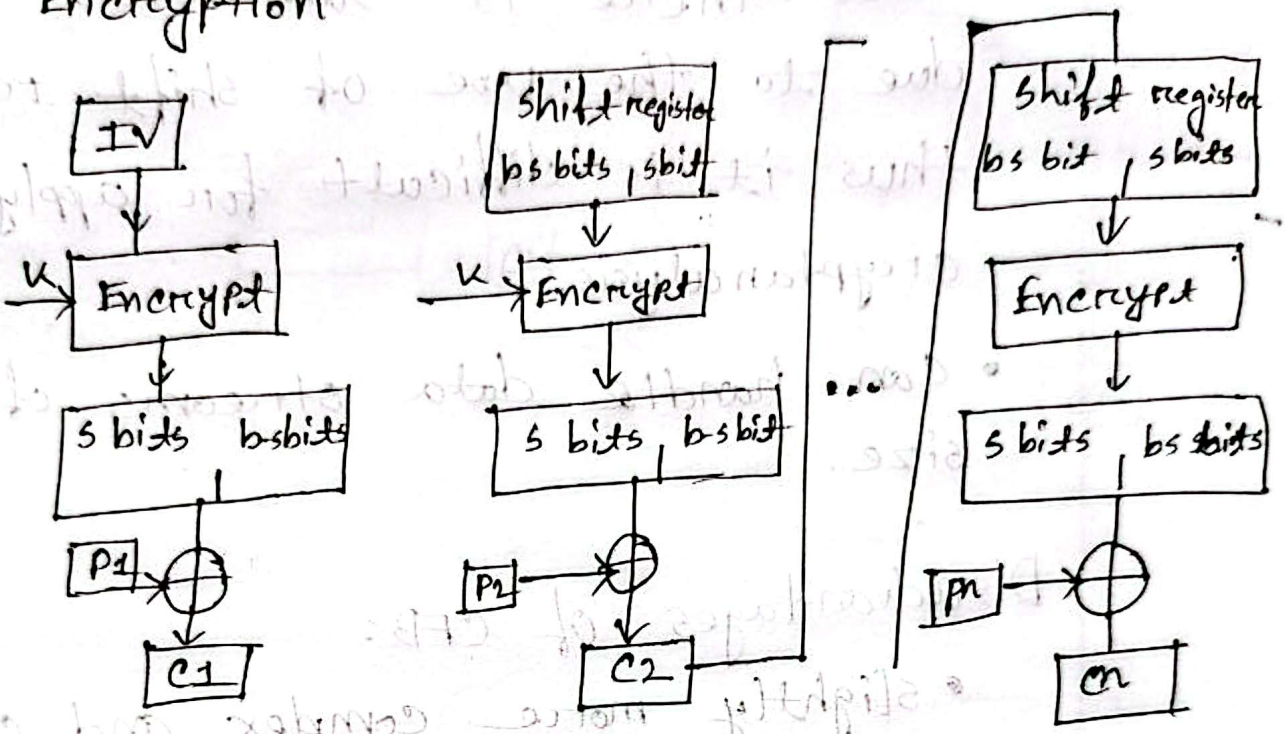
- CBC works well for input greater than b.
- CBC is a good authentication mechanism.
- Better resistive nature and more secure.

Disadvantages:

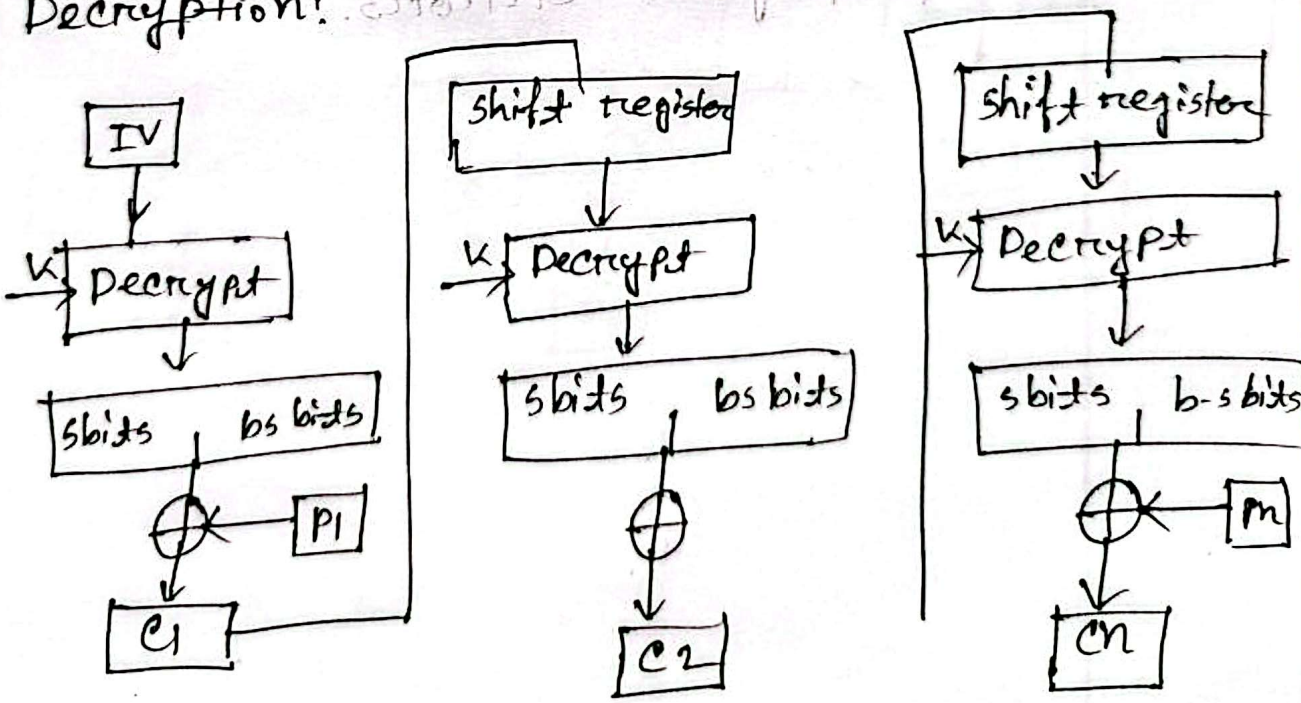
- Requires the previous ciphertext book for encryption and decryption.

CFB Block Diagram:

Encryption



Decryption:



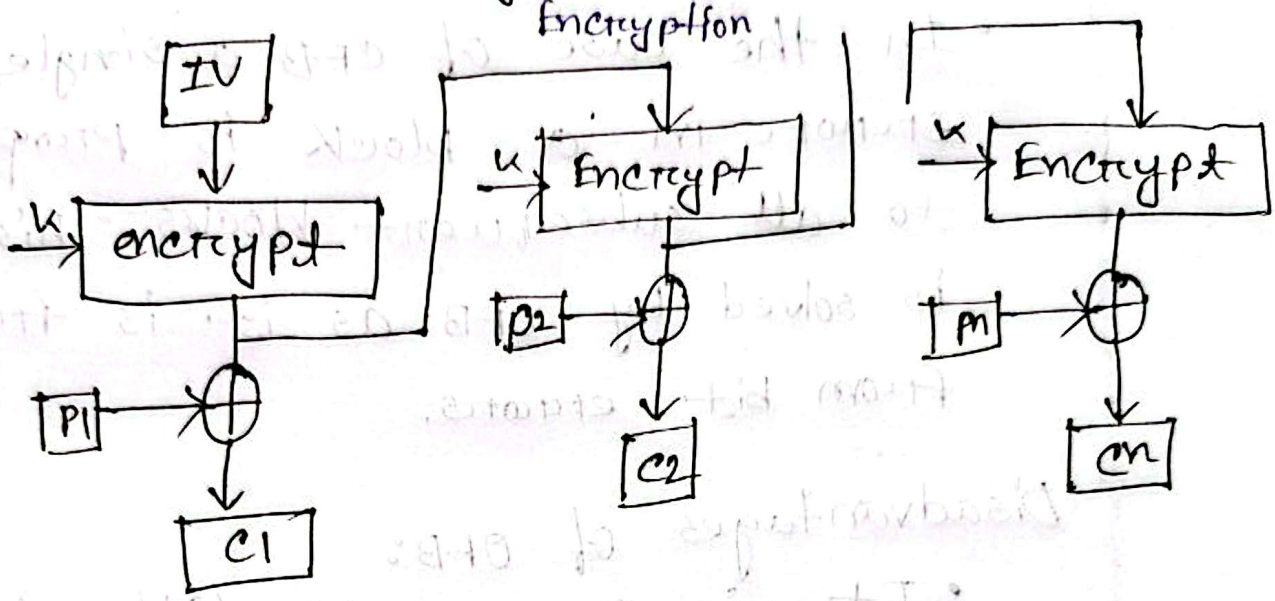
Advantages of CFB:

- since, there is some data loss due to the use of shift registers thus it is difficult for applying cryptanalysis.
- can handle data streams of any size.

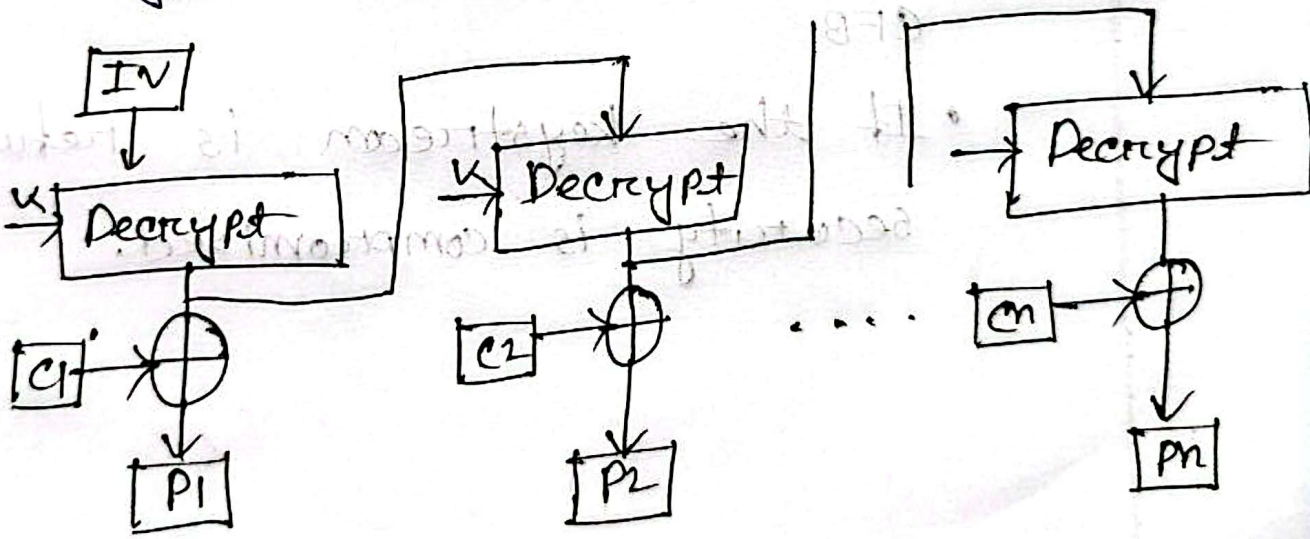
Disadvantages of CFB:

- slightly more complex and can propagates errors.

OFB Block Diagram:



Decryption:



Advantages of OFB

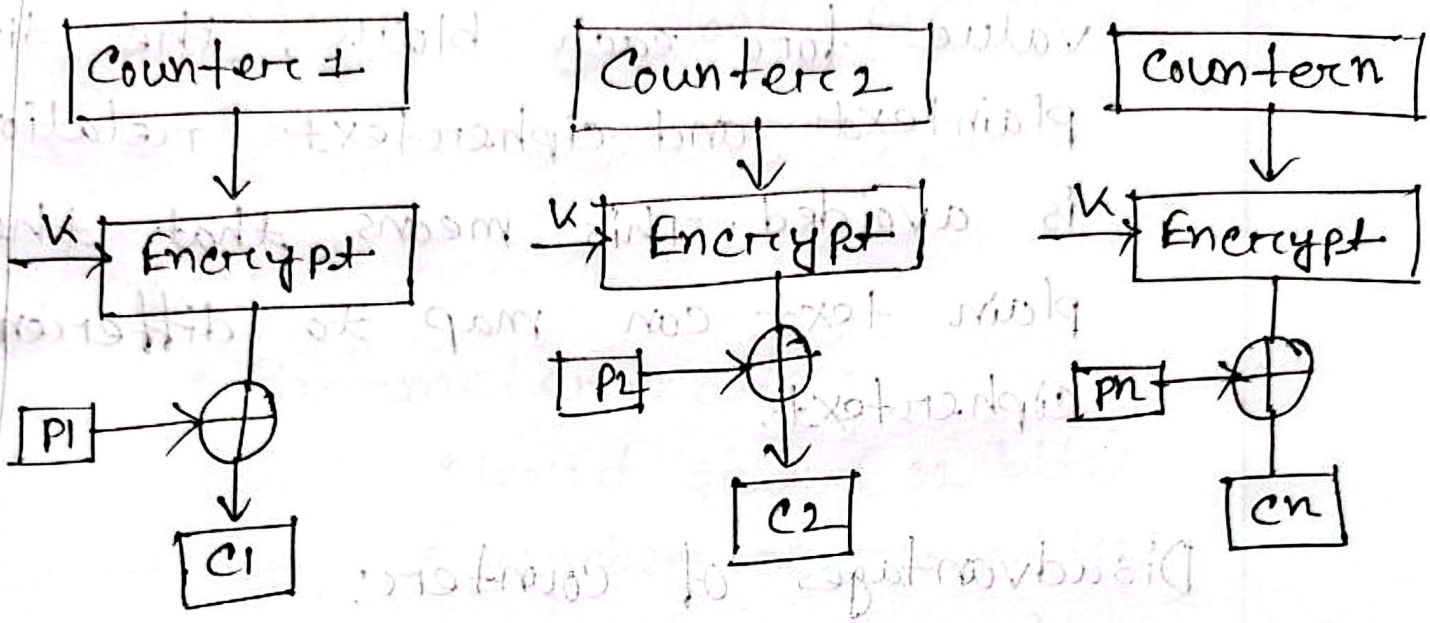
- In the case of CFB a single bit error in a block is propagated to all subsequent blocks. This problem is solved by OFB. as it is free from bit errors.

Disadvantages of OFB:

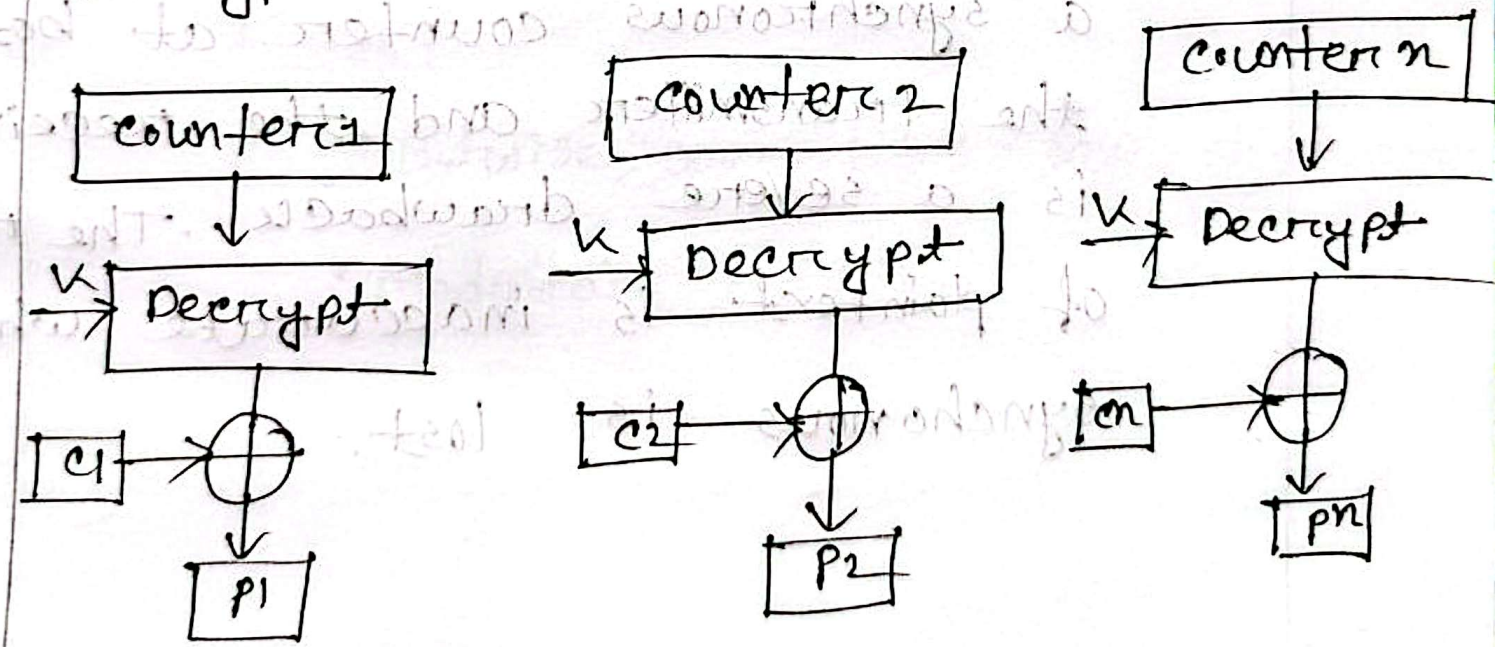
- It is more susceptible to message stream modification attack than CFB
- If the keystream is refused, security is compromised.

Block Diagram of CTR:

Encryption



Decryption



Advantages of counter (CTR)

- since there is a different counter value for each block, the direct plaintext and ciphertext relationship is avoided. This means that the same plain text can map to different ciphertext.

Disadvantages of counter:

- The fact that CTR mode requires a synchronous counter at both the transmitter and the receiver is a severe drawback. The recovery of plaintext is inaccurate when synchonous is lost.

Introduction of RC5:

RC5 is a fast, simple and secure symmetric key block cipher designed by Ron Rivest in 1994.

Key features:

- Parameterizable:
  - word size (32 bits)
  - Number of round (12)
  - key length (8 bytes)
- Uses:
  - Bitwise operations: XOR, shift, rotate
  - Modular addition

Java Implementation:

```
Public class RC5 {
    Private static final int WORD_size = 32;
    Private static final int R = 12;
    private static final int B = 8;
    private static final int C = 64;
    private static final int T = 2*(R+1);
    private int [] S = new int [T];
    private static final int P = 0xB7E15163;
    private static final int Q = 0x9E3779B9;
    public RC5 (byte [] key) {
        key Schedule (key);
        }
    private void keySchedule (byte [] key)
    {
        int [] = new int [C];
        for (int i = 0; i < B, i++) {
```

```
L[i/4] = (L[i/4] << B) + (key[i] & 0xFF);
}
S[0] = P;
for (int i=1; i<T; i++) {
    S[i] = S[i-1]+Q;
}
int A = 0; B=0, i = 0, j=0;
for (int k=0; k<3*T; k++) {
A = S[i] = Integer.rotateLeft((S[i]+
                              A+B), 3);
B = L[j] = Integer.rotateLeft((L[j]+
                              A+B), (A+B));
i = (i+1) % T;
j = (j+1) % C;
}
}
public int[] encrypt(int[] pt) {
    int A = pt[0] + S[0];
```

```
public int[] decrypt (int[] ct) {
    int B = ct[1];
    int A = ct[0];
    for (int i=R ; i>=1 ; i--) {
    A -= s[0];
    B -= s[1];
    return new int[] {A,B};
    }

    System.out.printf("Encrypted : %.08x
                %.08x \n", ct[0], ct[1]);
    int[] dt = rc5.decrypt (ct);
    System.out.println("Decrypted : %.08x
                %.08x \n", dt[0], dt[1]
    }
}
```

Output:. Encrypted : 7f93d8c2 1423ba29

Decrypted : 1234567& 9abcdef0

Code for ECB

```
import Javax.crypto.cipher;
import Javax.crypto:keyGenerator.;
import Java.io.fileInputstream;
import Java.io.fileOutputstream;
import Java.security.key;
public class ECBModeExample{
public.static void main(string[]
                        args)
throw exception{
key Generaton keyGeneration =
    key Generaton.get Instance("AES");
SecretKey secretkey = keyGeneraton.
                    generatekey();
Cipher cipher = cipher.get Instance
    ("AES/ECB/Pkcss padding");
```

```
cipher.init(cipher.ENCRYPT_MODE, secretkey
byte[] encrypted = cipher.doFinal
        ("This is a test", getbytes());
System.out.println("Encrypted: " + new
        String(encrypted)"
cipher.init(cipher.DECrypt_mode,
                secretkey);
byte[] decrypted = cipher.doFinal
        (encrypted);
System.out.println("Decrypted : "new
        String(decrypted));
}
}
```

Output: Encrypted: ??6.?? JD? Vx?·??

Decrypted : This is a test.