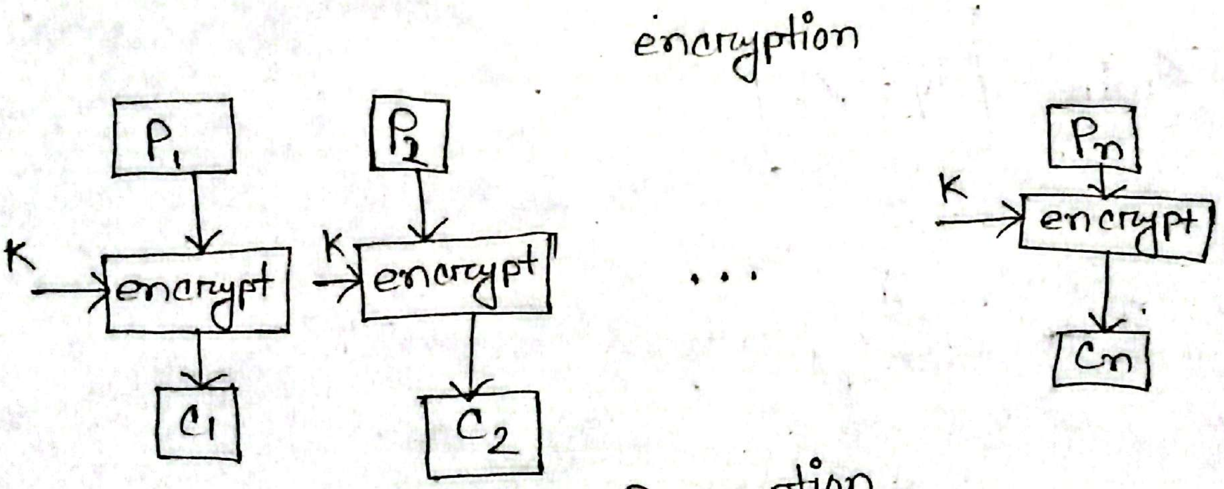
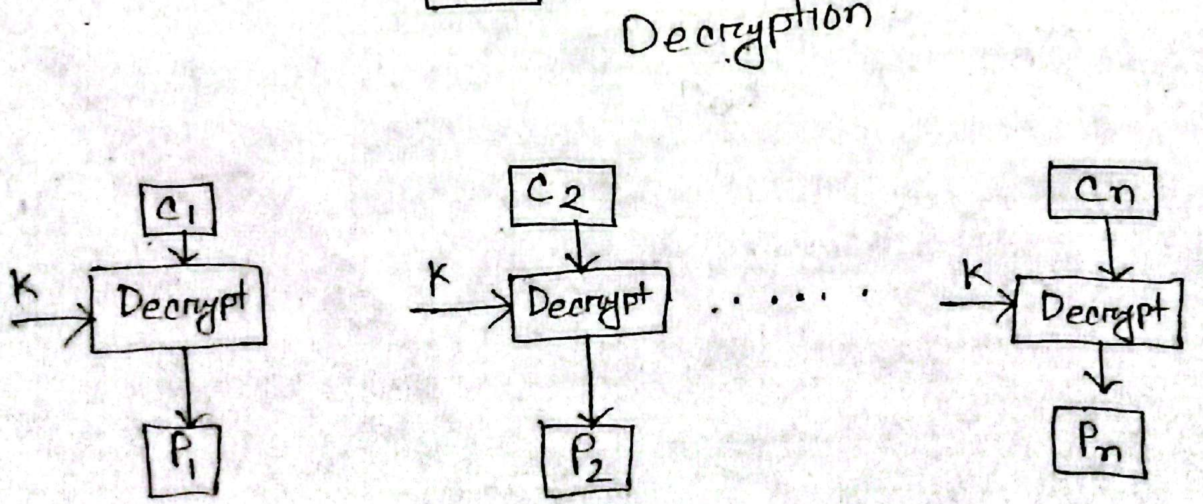
Electronic Code Book (ECB):

The electronic codebook is the easiest block cipher mode of functioning. It is easier because of the direct encryption of each block blockoof input plaintext and output is in the form of blocks of encrypted ciphertext.

Code Block:





Advantages of using ECB

→ Parallel encryption of blocks of bits is possib
thus it is a faster way of encryption.

→ Simple way of block cipher.

Disadvantages of using ECB:

→ Prone to cryptanalysis since there is a direct relationship between plaintext and ciphertex

→ Identical plaintext blocks produce identical ciphertext blocks, which can reveal patterns..

Java code:

```
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.CipherInputStream;
import javax.crypto.CipherOutputStream;
import javax. ~~crypto~~. io. FileInputStream;
import java.io.FileOutputStream;
import java.security..key;
```

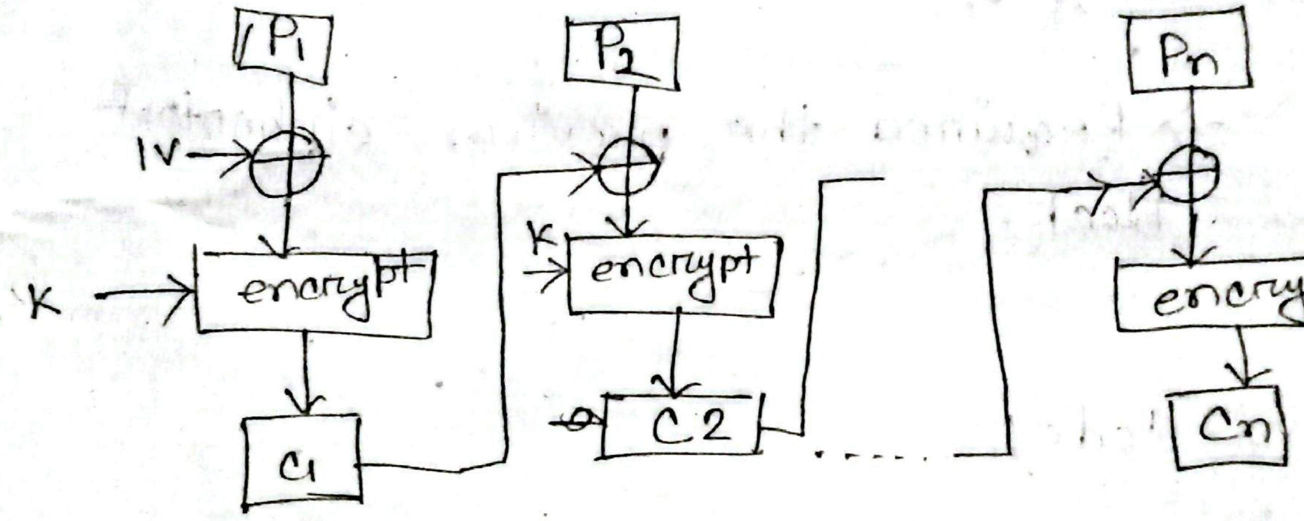
উদাঃ [illegible]

```
public class ECBModeExample {
        public static voide main (String[ ]args
            throws exception {
                Keygenerator keygenerator
                        = keygenerator .getInstance (AE
            Secret key secretkey= keygenerator.
                                    generatekey( );
        cipher cipher= Cipher.getInstance("AES/
                                    ECB/PKS55Padding
    cipher.init (Cipher.Encrypt-mode, secret).
   byte[ ] encrypted= cipher.dofinal(" This is atest
   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 ));
                           }
                             }.
```

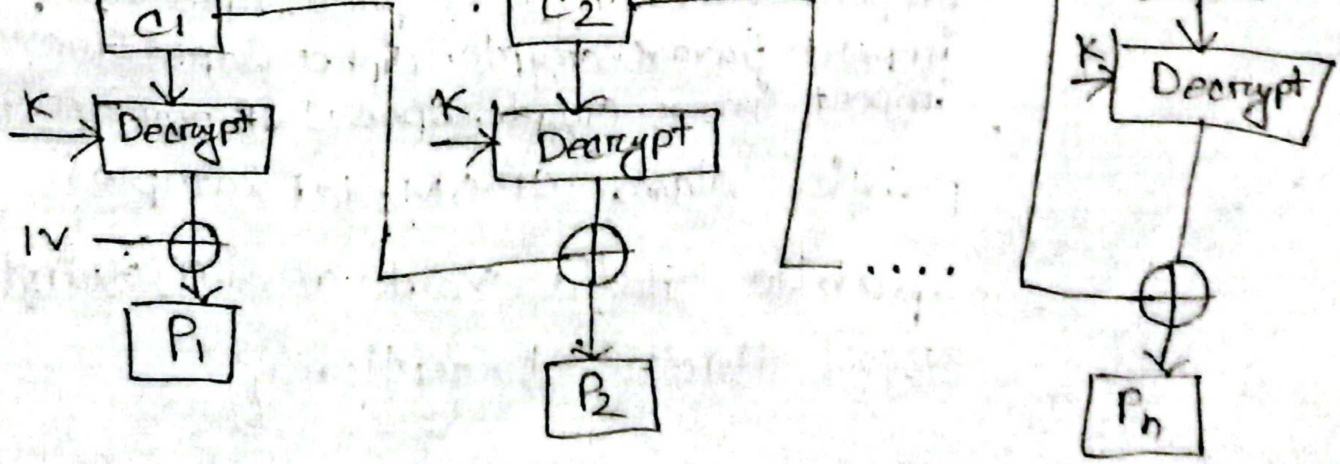
CBC: Cipher block chaining or CBC is an advancement made on ECB since ECB compromises some security requirements. In CBC.

Block Diagram;

Encryption



Decryption



Advantage:

- → CBC works well for input greater than b bits.
- → CBC is a good authentication mechanism.

Disadvantages:

- → Required the previous ciphertext block.

Java Code:

```
import javax.crypto.cipher;
import javax.crypto.keyGenerator;
import javax.crypto.Secrekey;
import jax.crypto.CipherInputStream;
import javax.crypto.Cipher OutputStream;
import javax.crypto.spec.IvParameterSpec;
public class CBCModeExample{
   public static void main(String[]
args] throws Exception {
```

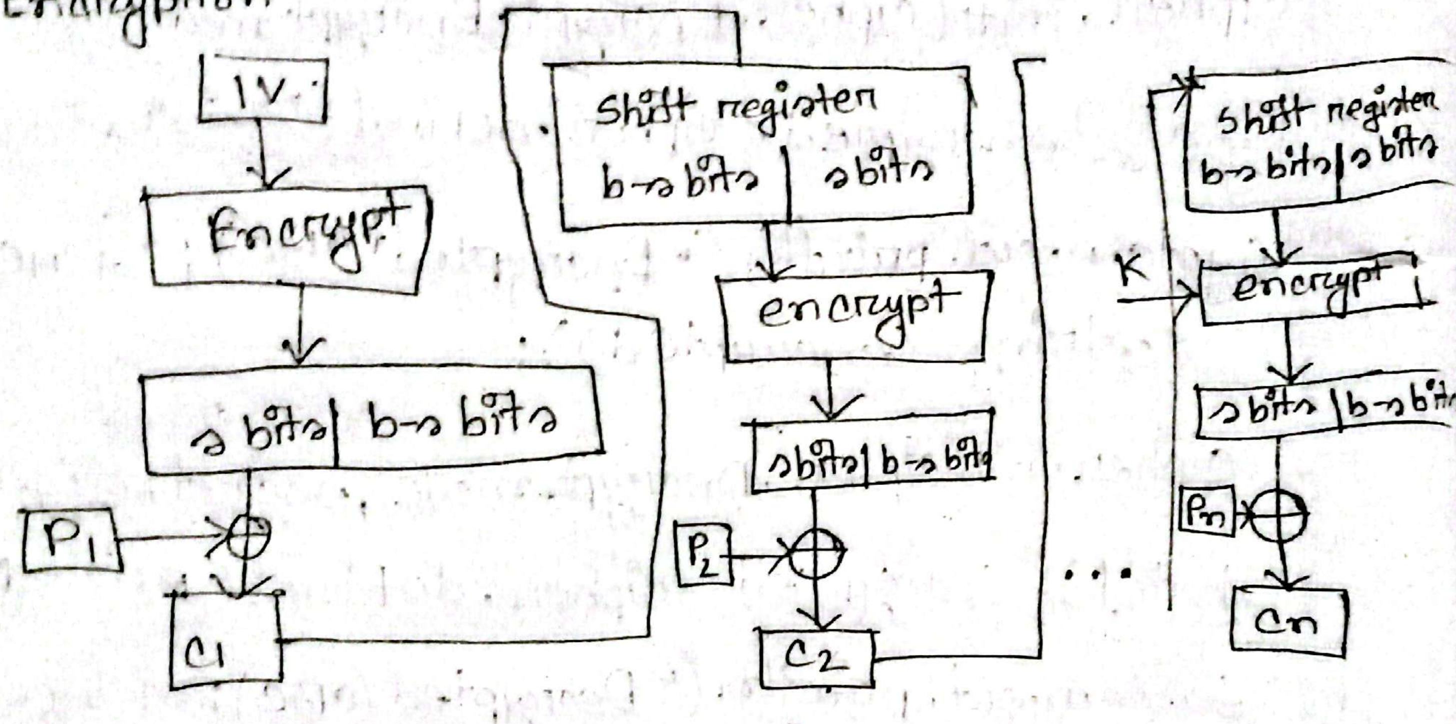
```
keygenerator keyGenerator = keyGenerator.getInstance

byte[] iv = new byte [16];
IVparameterSpec ivParameterSpec = new IvPara-
                        meterSpec (iv);

cipher cipher = cipher.getInstance ("AES");

cipher.init (cipher.Encrypt.Encrypt_mode);
byte[]encrypted = cipher.dofinal ("This is a test");
isystem.out.println ("Encrypted (CBC): " + new
      string (encrypted));
cipher.init(cipher.Decrypt_mode, secretkey);
byte[] decrypted = cipher.doFinal (Encrypte
System.out.println (" Decrypted (CBC): " + new
            String (decrypted));
                              }
                              };
```

CFB:

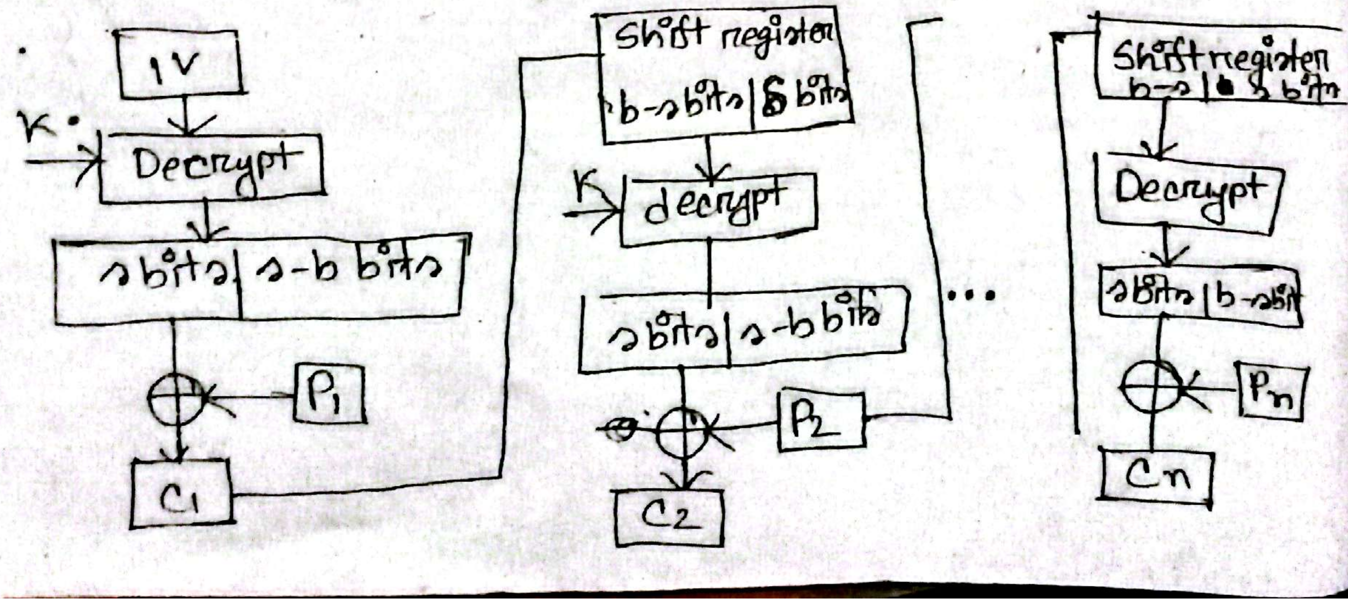
In this mode the cipher is given as feedback to the next block of encryption with some new sepecifications:

Block Diagram:

Encryption



Decryption



Advantages:

→ Since, there is some data loss due to the use of shift register, thus it is difficult

→ Can ~~do~~ handle data streams of any size

Disadvantages:

→ Slightly more complex and can propegate errors.

Java Code:

```
import. javax.crypto.cipher;
import. javax.cypto.Keygenerator;
import. javax. crypto. secret key;
import. javax. spec. Ivparameter;

Public Class CFBModeExample {
        public Static Void Main(String[] args)
    {
            Keygenerator keygenerator = keygenerator.getInstance("AES");
        Secretkey secretkey = kggenerator.generator();
byte[ ] iv = new byte[16];
Ivparameterspec= new Ivparapmeter(iv);
cipher cipher = cipher.getInstance ("AES ");

cipher.cipher = cipher .getInstance ("AES ");
byte[ ]encrypted=cipher.dofinal("This");
cipher. init (cipher.Decrypt-mode.);
system.out. println ("Descrypted( CFB):"
                    + new string (decrypte);
                                  }
                                  }
```