Block Cipher Modes of Operation:

A block cipher takes a fixed-length block of text of length ***b*** bits and a key as input and produces a ***b*-bit** block of ciphertext. If the amount of plaintext to be encrypted is greater than ***b* bits**, then the block cipher can still be used by breaking the plaintext up into ***b*-bit** blocks.

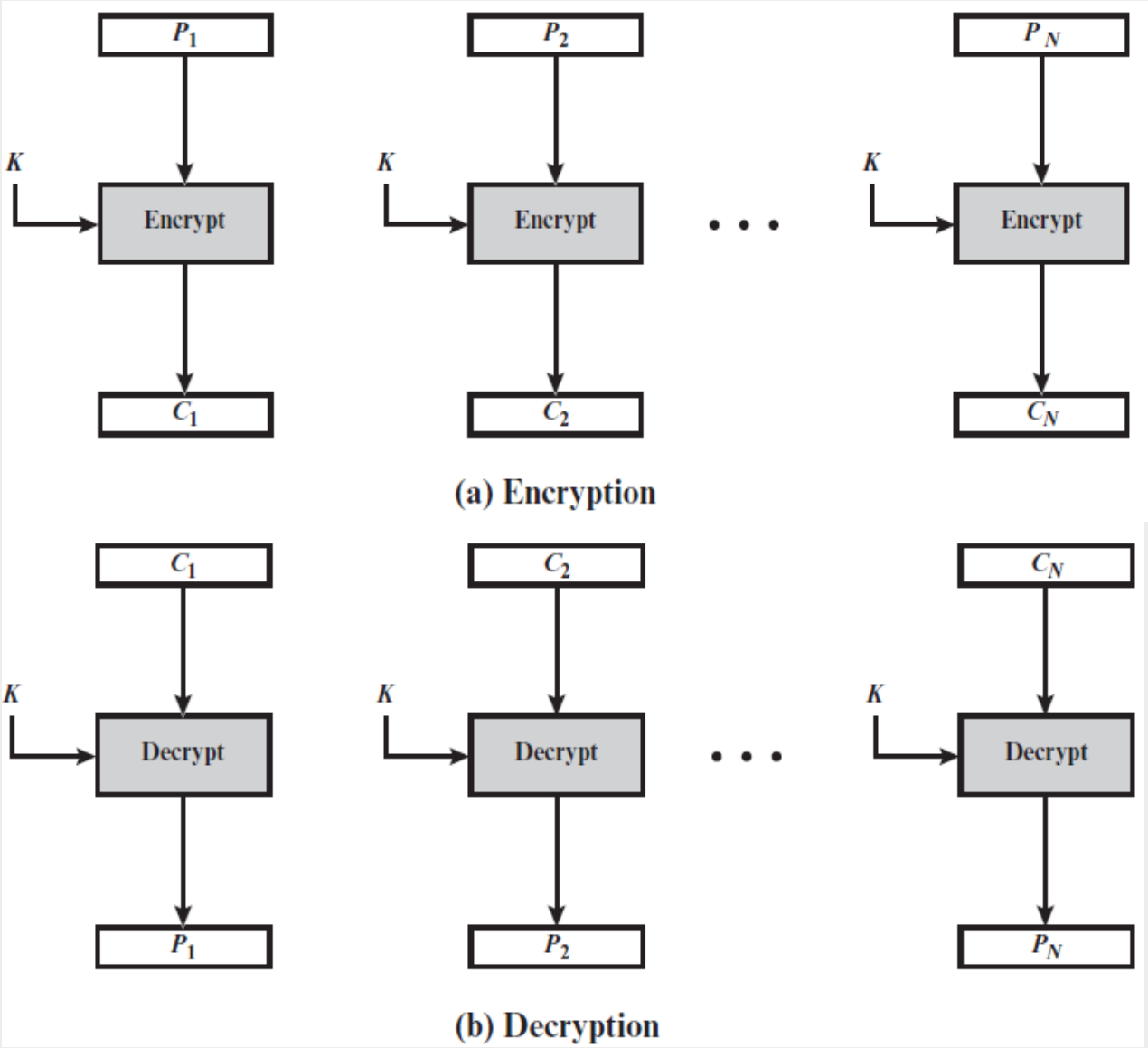
When multiple blocks of plaintext are encrypted using the same key, a number of security issues arise.

To apply a block cipher in a variety of applications, **five modes of operation** have been defined by NIST.

A mode of operation is a technique for enhancing the effect of a cryptographic algorithm or adapting the algorithm for an application, such as applying a block cipher to a sequence of data blocks or a data stream. These modes are intended for use with any symmetric block cipher, including **triple DES and AES**.

1-Electronic codebook (ECB) mode:

The simplest mode is the electronic codebook **(ECB)** 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. Therefore, we can imagine a huge codebook in which there is an entry for every possible **b-bit** plaintext pattern showing its corresponding ciphertext. For a message longer than **b bits**, the procedure is simply to break the message into **b-bit** blocks, padding (اضافة لغرض اكمال العدد) the last block if necessary. Decryption is performed one block at a time, always using the same key.

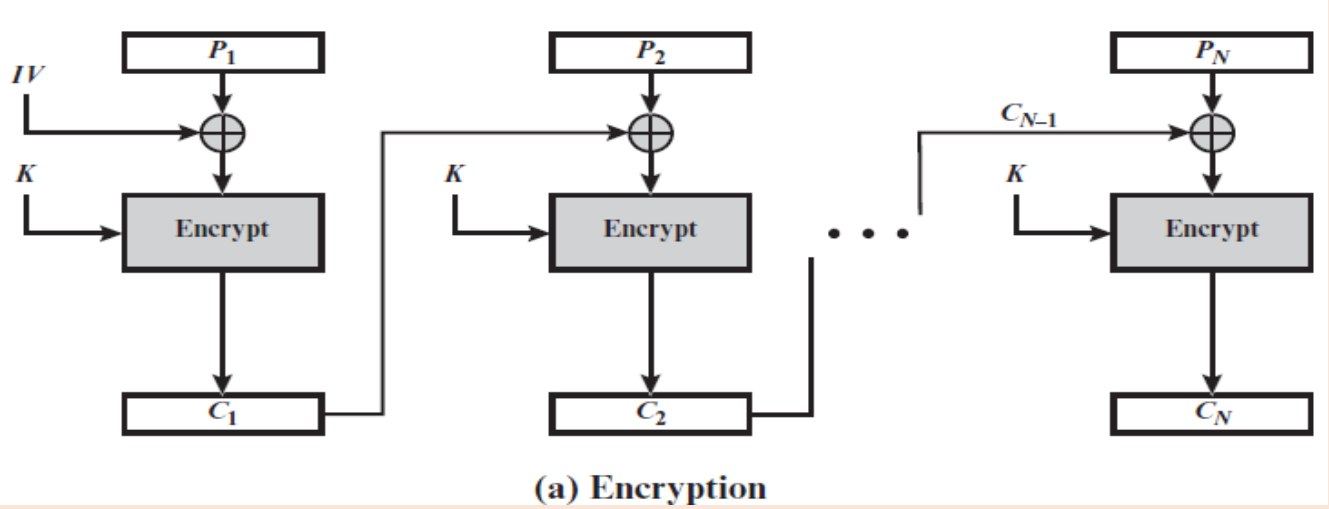


- This mode provide parallelism work because each block encrypted independently.
- The most significant characteristic of ECB is that if the same **b-bit** block of plaintext appears more than once in the message, it always produces the same ciphertext.

- ECB should be used only to secure messages shorter than a single block of underlying cipher (i.e., **64 bits** for 3DES and **128 bits** for AES), such as to encrypt a secret key.

Because in most of the cases messages are longer than the encryption block mode, this mode has a minimum practical value.

2- Cipher block chaining (CBC) mode: To overcome the security deficiencies of **ECB**, we would like a technique in which the same plaintext block, if repeated, produces different ciphertext blocks. A simple way to satisfy this requirement is the c**ipher block chaining (CBC) mode.**

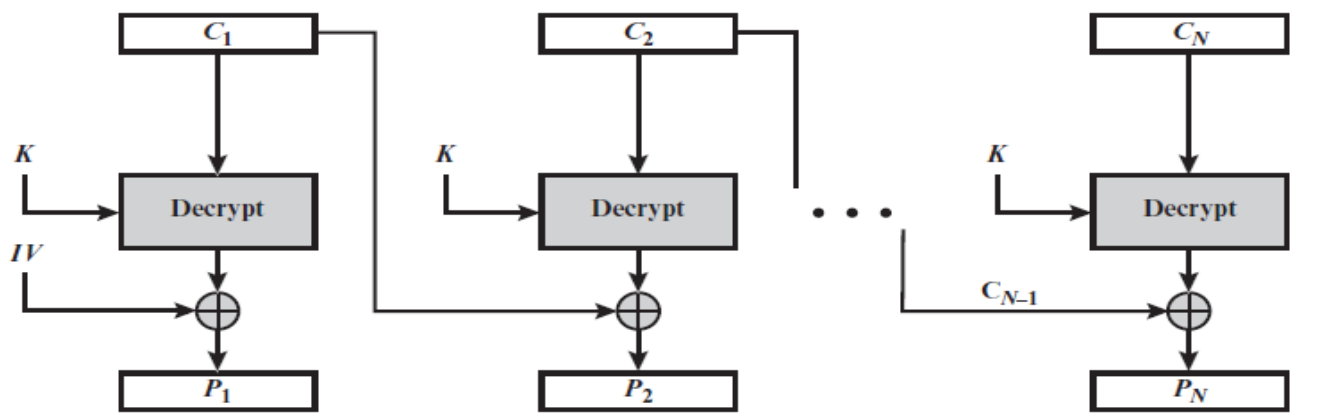


(a) Encryption

To produce the first block of ciphertext, an initialization vector **(IV)** is XORed with the first block of plaintext. On decryption, the **IV** is XORed with the output of the decryption algorithm to recover the first block of plaintext. The **IV** is a data block that is the same size as the cipher block. We can define **CBC** mode as:

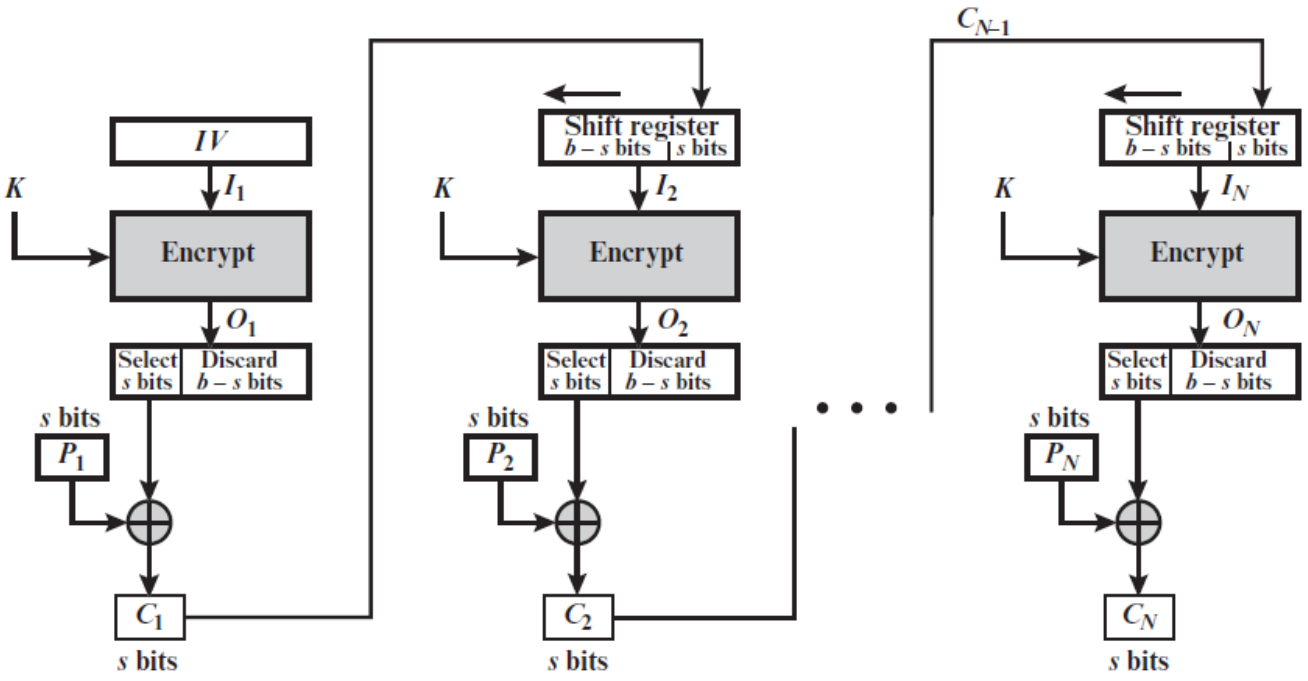
CBC	$C_1 = E(K, [P_j \oplus IV])$ $C_j = E(K, [P_j \oplus C_{j-1}])\ j = 2, \ldots, N$	$P_1 = D(K, C_1) \oplus IV$ $P_j = D(K, C_j) \oplus C_{j-1}, j = 2, \ldots, N$

The **IV** must be known to both the sender and receiver but be unpredictable by a third party. As with the **ECB** mode, the **CBC** mode requires that the last block be padded to a full **b bits** if it is a partial block. This mode appropriate mode for encrypting message of length greater than **b-bit** and provide confidentiality and authentication by using (IV). However, this mode slow and not parallelism.



(b) Decryption

3-Cipher Feedback Mode (CFB): For AES, DES, or any block cipher, encryption is performed on a block of ***b* bits**. In the case of DES, $b = 64$ and in the case of AES, $b = 128$. However, it is possible to convert a block cipher into a stream cipher, using one of the three modes to be discussed: **cipher feedback (CFB)** mode, **output feedback (OFB)** mode, and **counter (CTR)** mode. A stream cipher eliminates the need to pad a message to be an integral number of blocks. It also can operate in real time. Thus, if a character stream is being transmitted, each character can be encrypted and transmitted immediately using a character-oriented stream cipher.



(a) Encryption

It is assumed that the unit of transmission is ***s* bits**, a common value is ***s* = 8**. As with CBC, the units of plaintext are chained together, so that the ciphertext of any plaintext unit is a

function of all the preceding plaintext. In this case, rather than blocks of ***b* bits**, the plaintext is divided into segments of ***s* bits**.

First, consider encryption. The input to the encryption function is a ***b*-bit** shift register that is initially set to some **initialization vector (IV)**. The leftmost (most significant) ***s* bits** of the output of the encryption function are XORed with the first segment of plaintext $\boldsymbol{P_1}$ to produce the first unit of ciphertext $\boldsymbol{C_1}$, which is then transmitted. In addition, the contents of the shift register are shifted left by ***s* bits**, and $\boldsymbol{C_1}$ is placed in the rightmost (least significant) ***s* bits** of the shift register. This process continues until all plaintext units have been encrypted.

For decryption, the same scheme is used, except that the received ciphertext unit is XORed with the output of the encryption function to produce the plaintext unit.

This is easily explained. Let MSB*s*(*X*) be defined as the most significant ***s* bits** of *X*.

Then

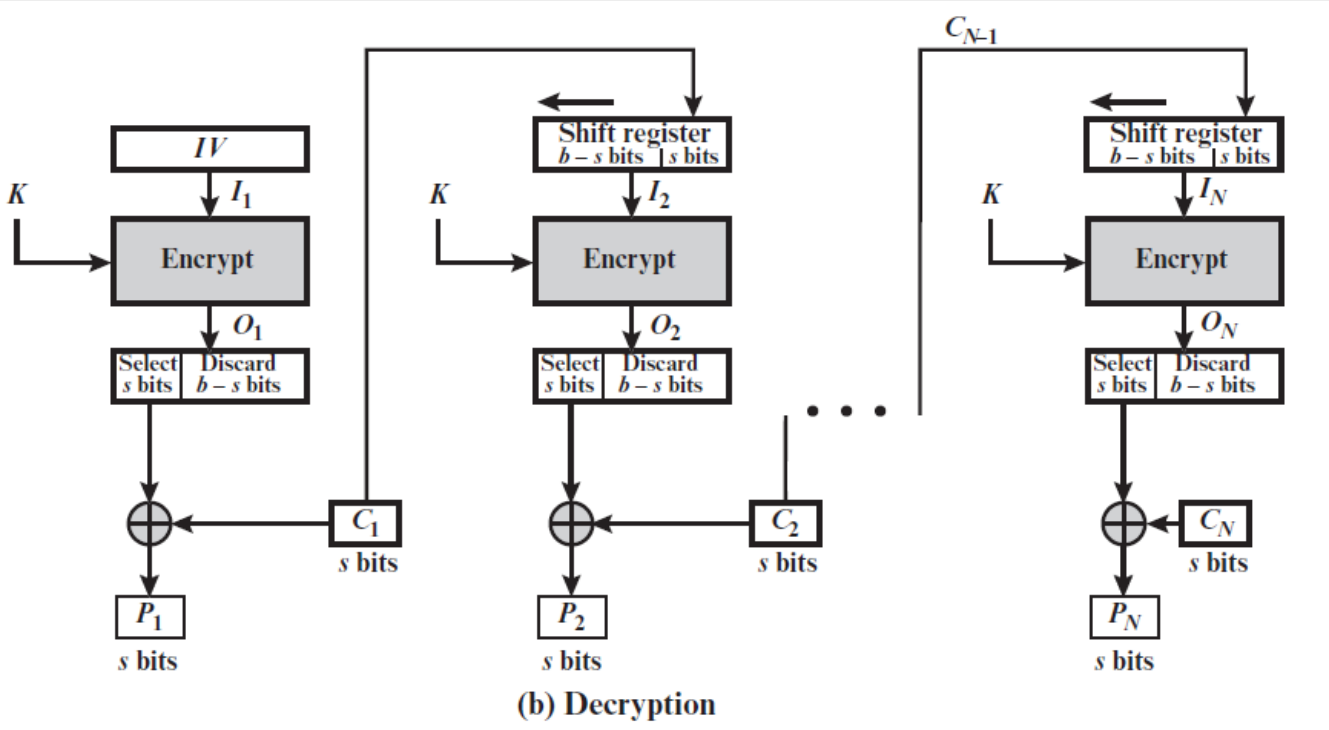
$$C_1 = P_1 \oplus MSB_s[E(K, IV)]$$

Therefore, by rearranging terms:

$$P_1 = C_1 \oplus MSBs[E(K, IV)]$$

The same reasoning holds for subsequent steps in the process.

- In CFB encryption, like CBC encryption, the input block to each forward cipher function (except the first) depends on the result of the previous forward cipher function.
- Multiple forward cipher operations cannot be performed in parallel. In CFB decryption, the required forward cipher operations can be performed in parallel if the input blocks are first constructed (in series) from the **IV** and the ciphertext.



(b) Decryption

4-Output Feedback Mode:

The **output feedback (OFB) mode** is similar in structure to that of CFB. For OFB, the output of the encryption function is feedback to become the input for encrypting the next block of plaintext. In CFB, the output of the XOR unit is feedback to become input for encrypting the next block. The other difference is that the OFB mode operates on full blocks of plaintext and ciphertext, whereas CFB operates on an **s-bit** subset. OFB encryption can be expressed as

$$C_j = P_j \oplus E(K, O_{j-1})$$

where

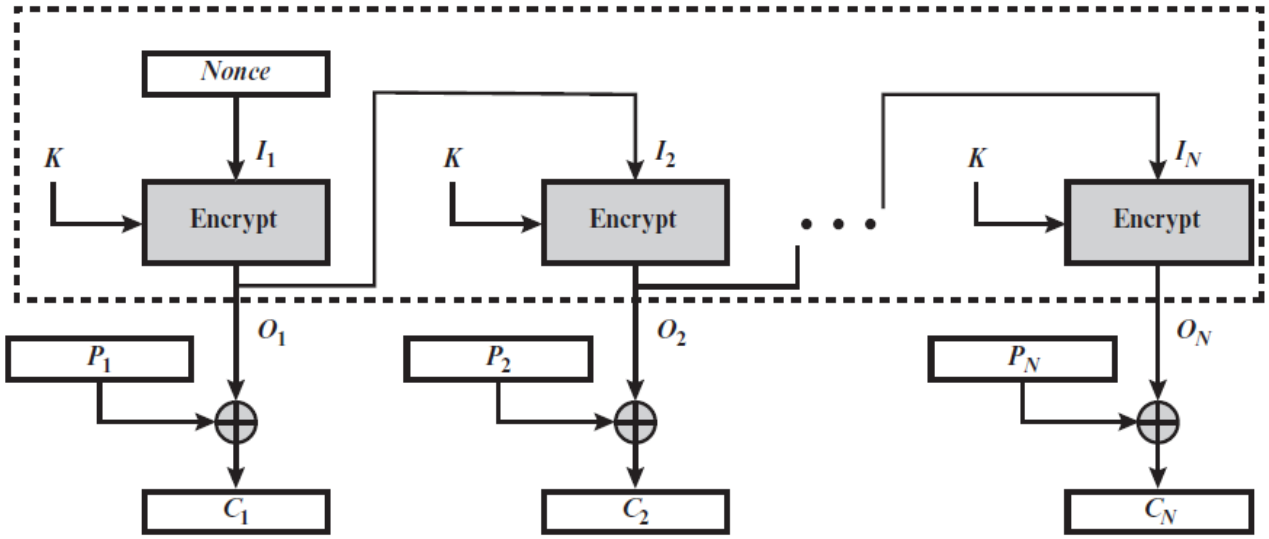
$$O_{j-1} = E(K, O_{j-2})$$

We can rewrite the encryption expression as:

$$C_j = P_j \oplus E(K, [C_{j-1} \oplus P_{j-1}])$$

By rearranging terms, we can demonstrate that decryption works.

$$P_j = C_j \oplus E(K, [C_{j-1} \oplus P_{j-1}])$$

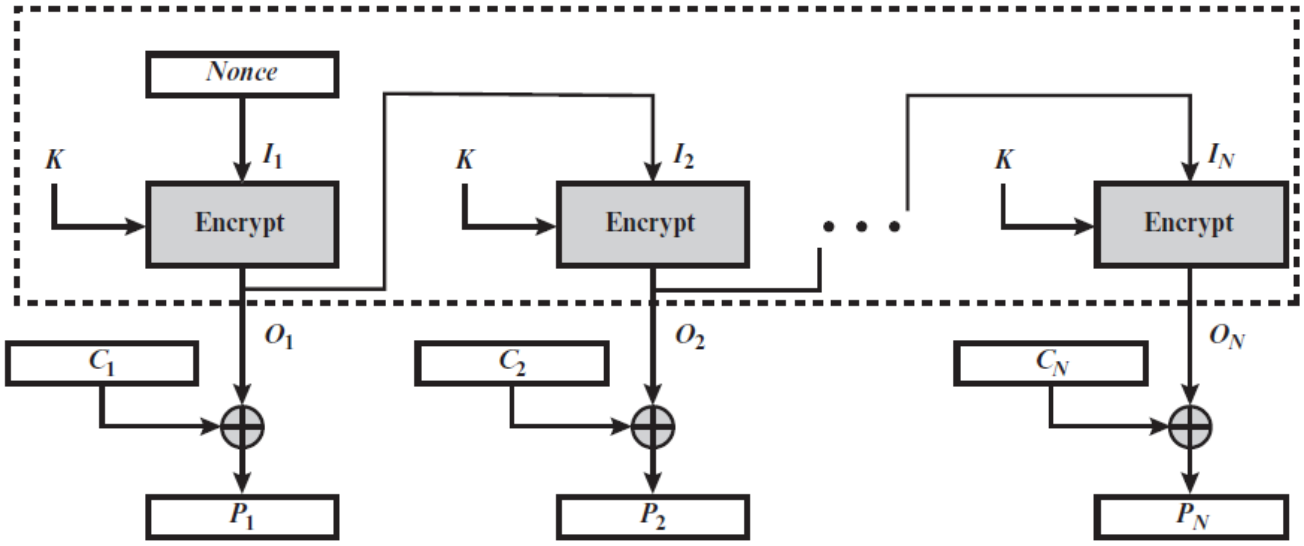


(a) Encryption

- Let the size of a block be **b**. If the last block of plaintext contains **u bits**, with $u < b$, the most significant **u bits** of the last output block O_N are used for the XOR operation; the remaining **b - u bits** of the last output block are discarded. As with CBC and CFB, the OFB mode requires an initialization vector.
- In the case of OFB, the **IV** must be a **nonce** (رقم عشوائي لكل رسالة), that is, the IV must be unique to each execution of the encryption operation. The reason for this is that the sequence of encryption output blocks, O_i, depends only on the key and the **IV** and does not depend on the plaintext. Therefore, for a given key and IV, the stream of output bits used to XOR with the stream of plaintext bits is fixed. If two different messages

had an identical block of plaintext in the identical position, then an attacker would be able to determine that portion of the O_i stream.

- One advantage of the **OFB** method is that bit errors in transmission do not propagate. For example, if a bit error occurs in C_1, only the recovered value of P_1 is affected; subsequent plaintext units are not corrupted.
- With CFB, C_1, also serves as input to the shift register and therefore causes additional corruption downstream. The disadvantage of OFB is that it is more vulnerable to a message stream modification attack than is CFB.



(b) Decryption

OFB has the structure of a typical stream cipher, because the cipher generates a stream of bits as a function of an initial value and a key, and that stream of bits is XORed with the plaintext bits. One distinction from the stream ciphers is that OFB encrypts plaintext a full

block at a time, where typically a block is 64 or 128 bits. Many stream ciphers encrypt one byte at a time.

5-Counter Mode:

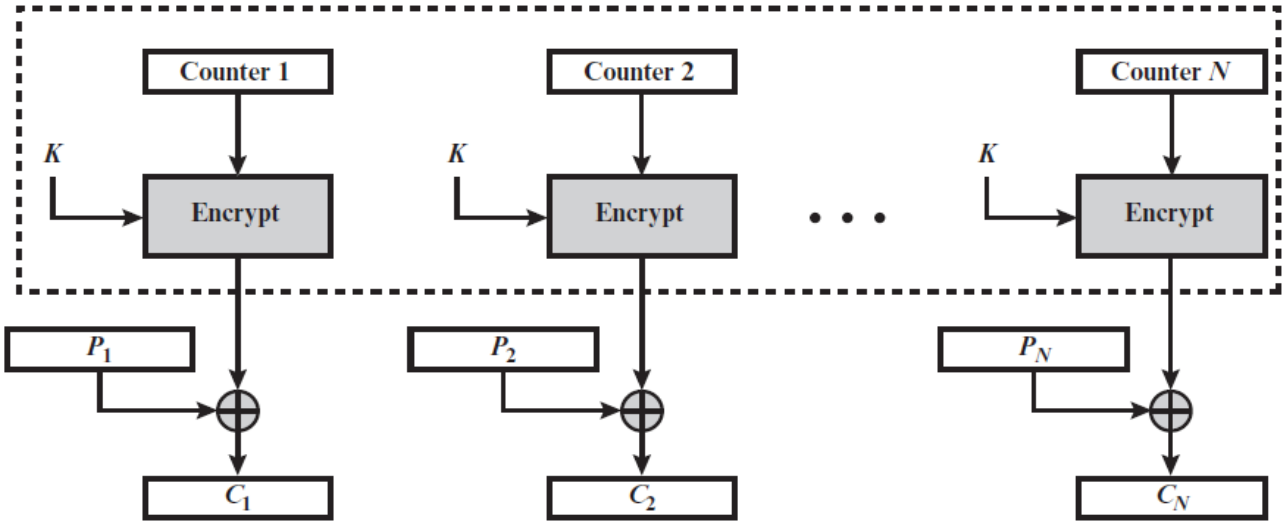
Although interest in the **counter (CTR) mode** has increased recently with applications to ATM (asynchronous transfer mode) network security and IPsec (IP security), this mode was proposed in 1979. A counter equal to the plaintext block size is used. The only requirement stated is that the counter value must be different for each plaintext block that is encrypted. Typically, the counter is initialized to some value and then incremented by **1** for each subsequent block (modulo $\mathbf{2^b}$, where b is the block size). For encryption, the counter is encrypted and then XORed with the plaintext block to produce the ciphertext block there is no chaining. For decryption, the same sequence of counter values is used, with each encrypted counter XORed with a ciphertext block to recover the corresponding plaintext block. Thus, the initial counter value must be made available for decryption. Given a sequence of counters $T_1, T_2, \ldots, T_N$, we can define CTR mode as follows.

$$C_j = P_j \oplus E(K, T_j) \quad j = 1 \ldots . N-1$$

$$C_N = P_N \oplus MSB[E(K, T_N])$$

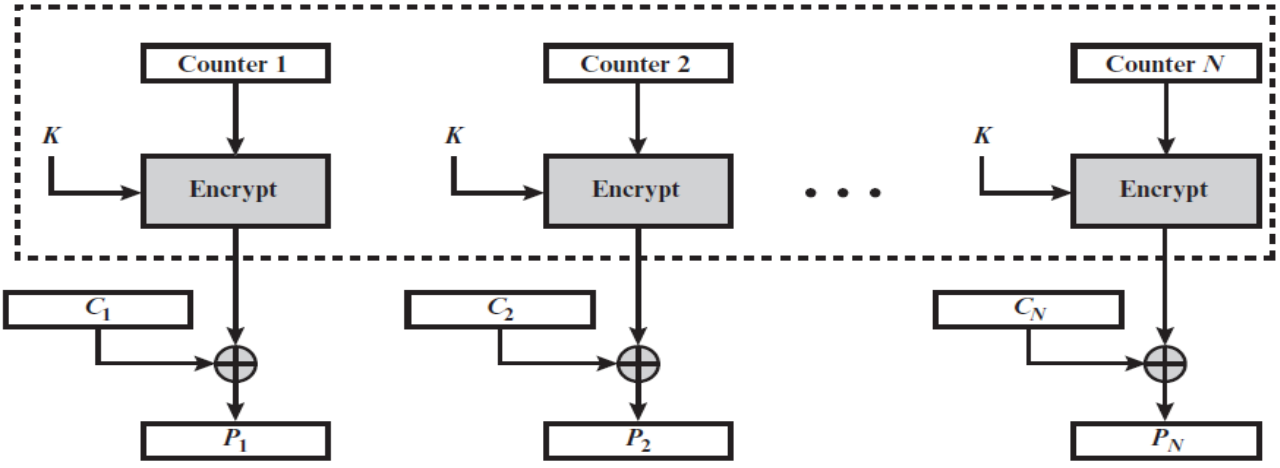
$$P_j = C_j \oplus E(K, T_j) \quad j = 1 \ldots . N-1$$

$$P_N = C_N \oplus MSB[E(K, T_N])$$



(a) Encryption

For the last plaintext block, which may be a partial block of ***u* bits**, the most significant ***u* bits** of the last output block are used for the XOR operation; the remaining ***b* - *u* bits** are discarded. Unlike the ECB, CBC, and CFB modes, we do not need to use padding because of the structure of the **CTR mode**. As with the OFB mode, the initial counter value must be a **nonce**; that is, T_1 must be different for all of the messages encrypted using the same key.



(b) Decryption

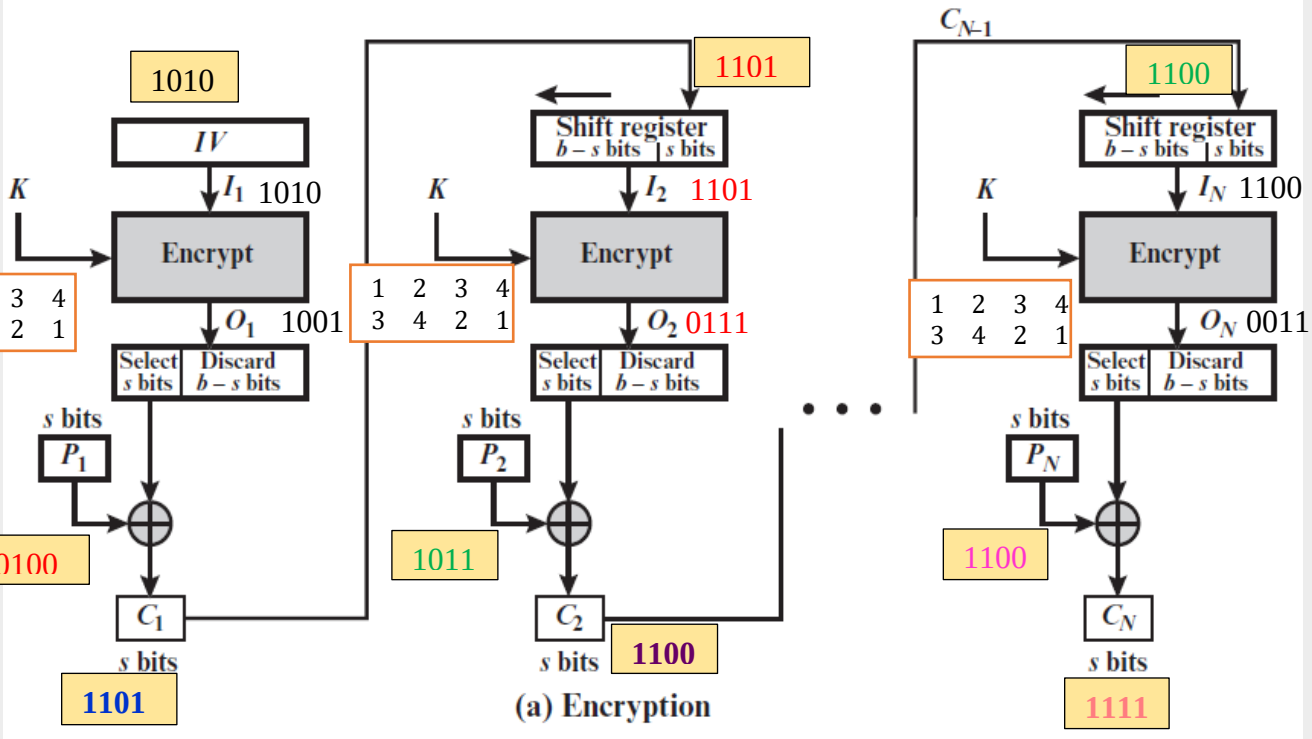
One way to ensure the uniqueness of counter values is to continue to increment the counter value by 1 across messages. That is, the first counter value of the each message is one more than the last counter value of the preceding message.

Mode	Description	Typical Application
Electronic Codebook (ECB)	Each block of plaintext bits is encoded independently using the same key.	• Secure transmission of single values (e.g., an encryption key)
Cipher Block Chaining (CBC)	The input to the encryption algorithm is the XOR of the next block of plaintext and the preceding block of ciphertext.	• General-purpose block-oriented transmission • Authentication
Cipher Feedback (CFB)	Input is processed *s* bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	• General-purpose stream-oriented transmission • Authentication
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding encryption output, and full blocks are used.	• Stream-oriented transmission over noisy channel (e.g., satellite communication)
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	• General-purpose block-oriented transmission • Useful for high-speed requirements

Example: Consider the CFB mode of operation where the block cipher is permutation cipher and key is a permutation $\begin{bmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 2 & 1 \end{bmatrix}$.

If the IV is taken as 1010 then what is the corresponding ciphertext corresponding for the plaintext 010010111100?

Solution: based on given IV the plain text are subdivided into 4 bits (010010111100)



The ciphertext is: 110111001111