Block cipher operation

Introduction

- 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
- 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.
- The five modes are intended to cover a wide variety of applications of encryption for which a block cipher could be used. These modes are intended for use with any symmetric block cipher, including triple DES and AES.

Electronic code book

- In **electronic codebook** (**ECB**) mode, plaintext is handled one block at a time and each block of plaintext is encrypted using the same key.
- For a message longer than *b* bits, the procedure is simply to break the message into *b*-bit blocks, padding the last block if necessary.
- Decryption is performed one block at a time, always using the same key.

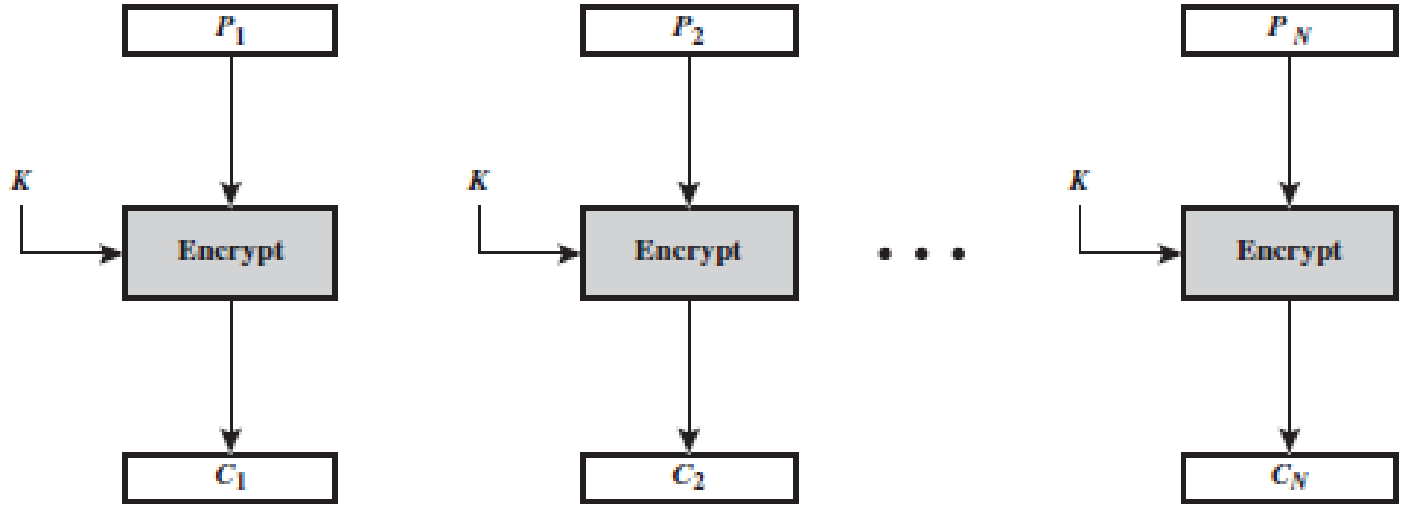
Block Cipher Modes of Operation

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

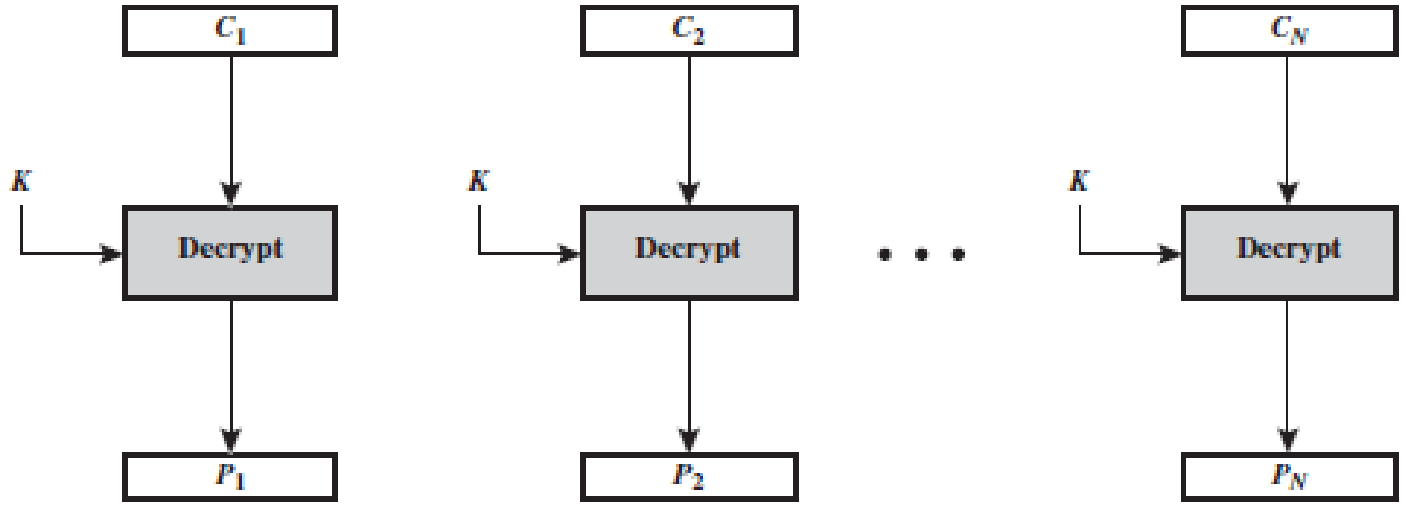
ECB	$C_j = \mathrm{E}(K, P_j) \qquad j = 1, \ldots, N$	$P_j = \mathrm{D}(K, C_j) \qquad j = 1, \ldots, N$

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

- For lengthy messages, the ECB mode may not be secure. If the message is highly structured, it may be possible for a cryptanalyst to exploit these regularities.
- For example, if it is known that the message always starts out with certain predefined fields, then the cryptanalyst may have a number of known plaintext–ciphertext pairs to work with. If the message has repetitive elements with a period of repetition a multiple of *b* bits, then these elements can be identified by the analyst. This may help in the analysis or may provide an opportunity for substituting or rearranging blocks.



(a) Encryption



(b) Decryption

Figure 6.3 Electronic Codebook (ECB) Mode

Following are criteria and properties for evaluating and constructing block cipher modes of operation that are superior to ECB:

- **Overhead:** The additional operations for the encryption and decryption operation when compared to encrypting and decrypting in the ECB mode.
- **Error recovery:** The property that an error in the *i*th ciphertext block is inherited by only a few plaintext blocks after which the mode resynchronizes.
- **Error propagation:** The property that an error in the *i*th ciphertext block is inherited by the *i*th and all subsequent plaintext blocks.
- **Diffusion:** How the plaintext statistics are reflected in the ciphertext. Low entropy plaintext blocks should not be reflected in the ciphertext blocks. Roughly, low entropy equates to predictability or lack of randomness.
- **Security:** Whether or not the ciphertext blocks leak information about the plaintext blocks.

Cipher block chaining 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 cipher block chaining (CBC) mode.
- In this scheme, 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. The input to the encryption function for each plaintext block bears no fixed relationship to the ciphertext block. Therefore, repeating patterns of b bits are not exposed. 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.

- For decryption, each cipher block is passed through the decryption algorithm. The result is XORed with the preceding ciphertext block to produce the plaintext block.

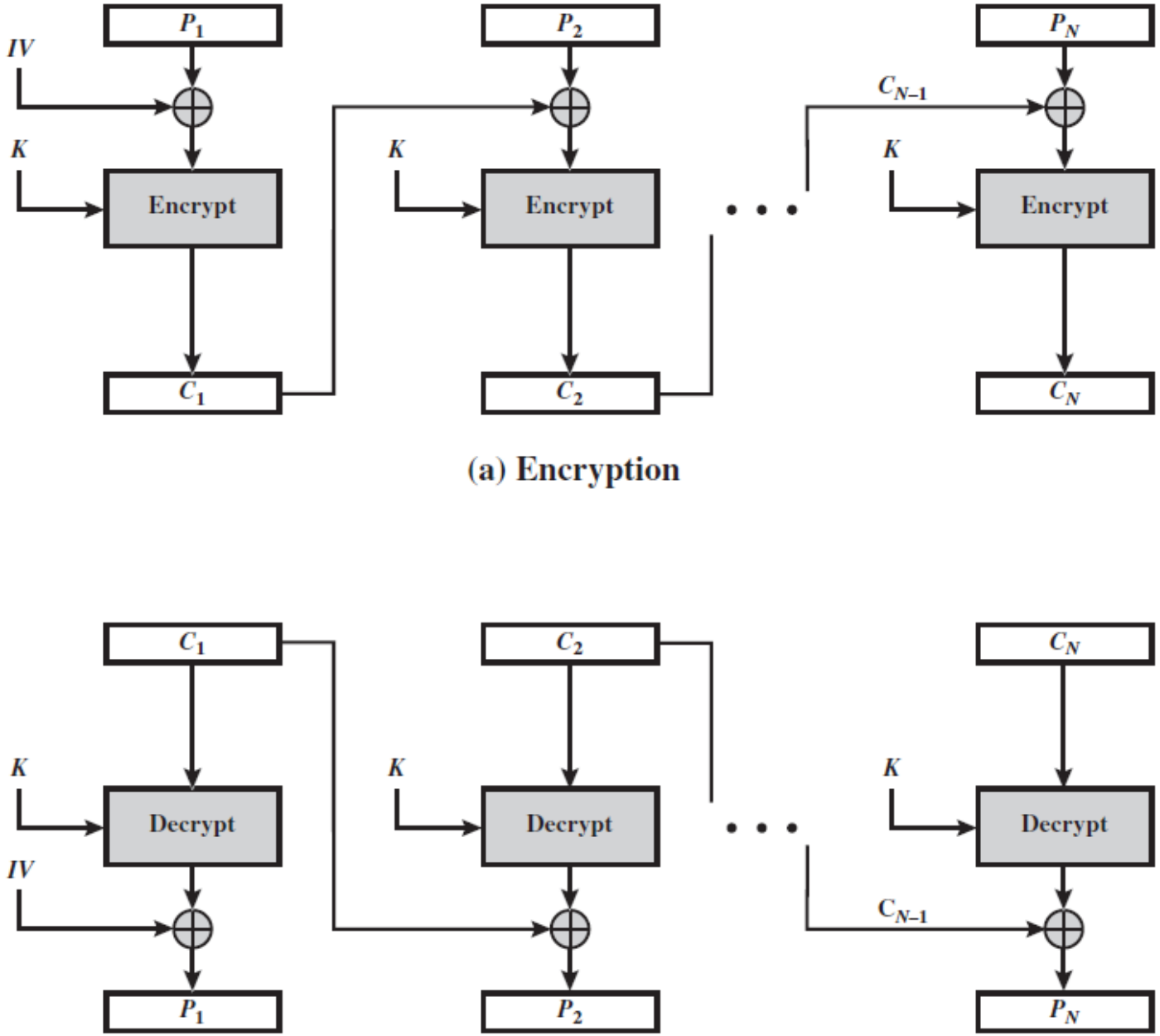


Figure 6.4 Cipher Block Chaining (CBC) Mode

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

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

$$\mathrm{D}(K, C_j) = C_{j-1} \oplus P_j$$

$$C_{j-1} \oplus \mathrm{D}(K, C_j) = C_{j-1} \oplus C_{j-1} \oplus P_j = P_j$$

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

- The IV must be known to both the sender and receiver but be unpredictable by a third party.
- For maximum security, the IV should be protected against unauthorized changes. This could be done by sending the IV using ECB encryption.
- One reason for protecting the IV is as follows: If an opponent is able to fool the receiver into using a different value for IV, then the opponent is able to invert selected bits in the first block of plaintext.

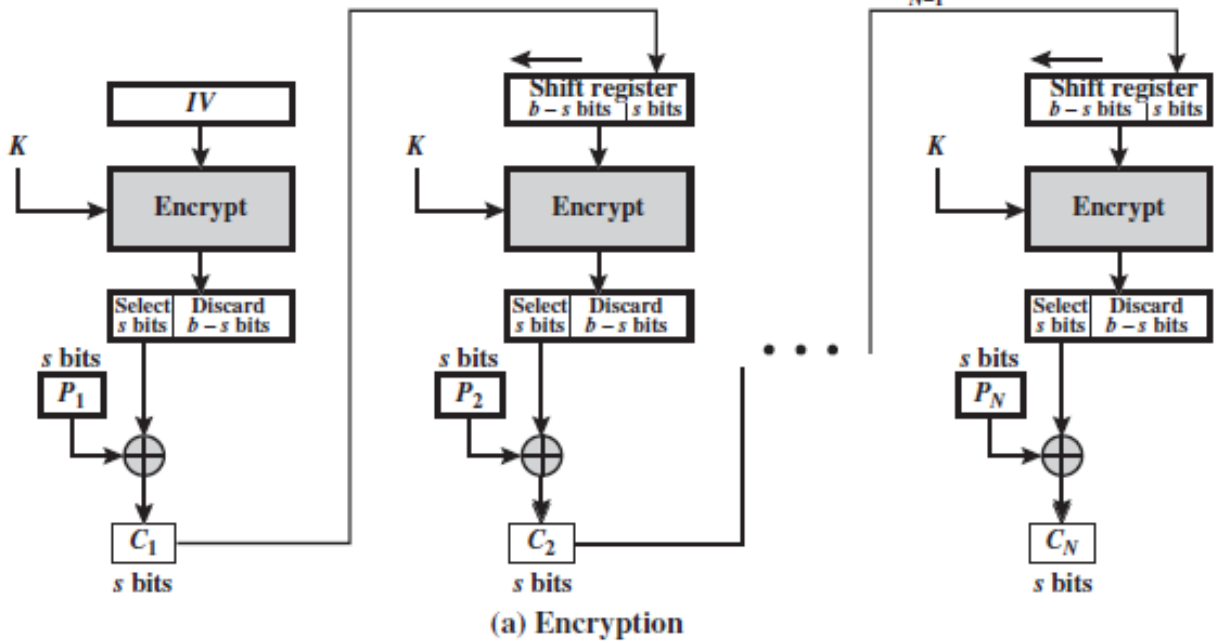
Cipher feedback mode

- 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.
- It is possible to convert a block cipher into a stream cipher, using: **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.
  The ciphertext be of the same length as the plaintext. Thus, if 8-bit characters are being transmitted, each character should be encrypted to produce a ciphertext output of 8 bits. If more than 8 bits are produced, transmission capacity is wasted.

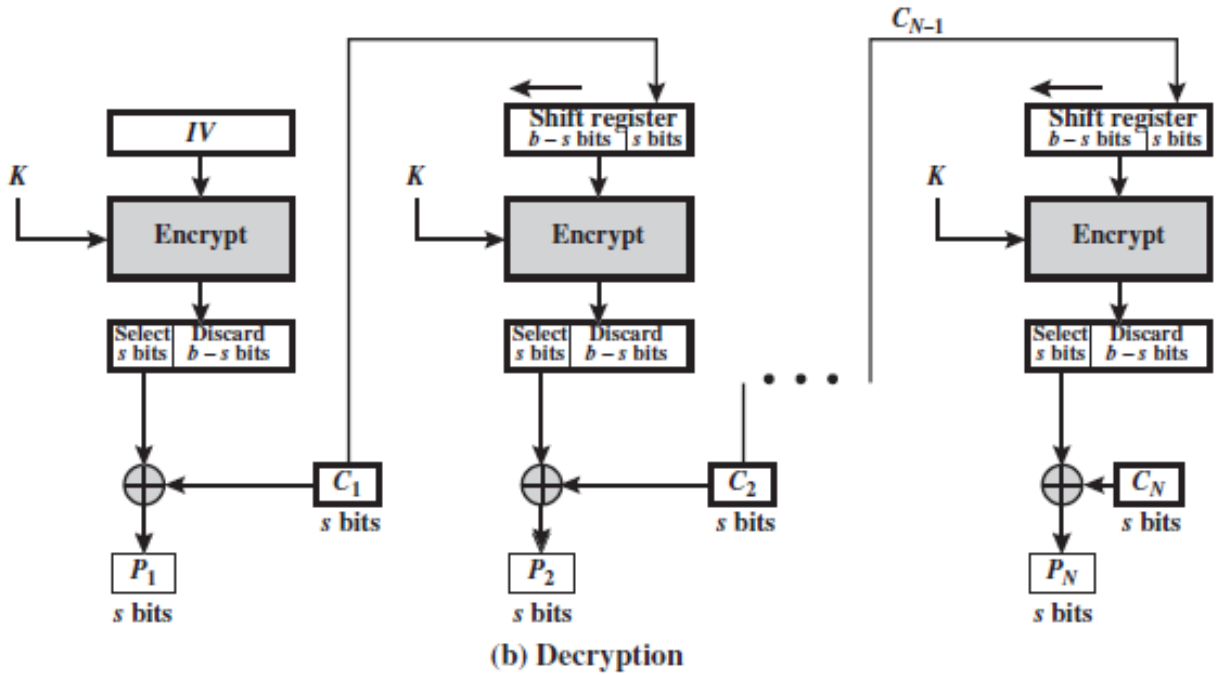
- It is assumed that the unit of transmission is *s* bits; a common value is *s* = 8. As with CBC, the units of plaintext are chained together, so that the ciphertext of any plaintext unit is a function of all the preceding plaintext.
- In this case, rather than blocks of *b* bits, the plaintext is divided 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 *P*1 to produce the first unit of ciphertext *C*1, which is then transmitted.
- In addition, the contents of the shift register are shifted left by *s* bits, and *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. Note that it is the *encryption* function that is used, not the decryption function.

$$C_1 = P_1 \oplus \mathrm{MSB}_s[\mathrm{E}(K, \mathrm{IV})]$$

$$P_1 = C_1 \oplus \mathrm{MSB}_s[\mathrm{E}(K, \mathrm{IV})]$$



(a) Encryption



(b) Decryption

Figure 6.5 *s*-bit Cipher Feedback (CFB) Mode

CFB	$I_1 = IV$ $I_j = \text{LSB}_{b-s}(I_{j-1}) \| C_{j-1}$ $\quad j = 2, \ldots, N$ $O_j = \text{E}(K, I_j)$ $\quad j = 1, \ldots, N$ $C_j = P_j \oplus \text{MSB}_s(O_j)$ $\quad j = 1, \ldots, N$	$I_1 = IV$ $I_j = \text{LSB}_{b-s}(I_{j-1}) \| C_{j-1}$ $\quad j = 2, \ldots, N$ $O_j = \text{E}(K, I_j)$ $\quad j = 1, \ldots, N$ $P_j = C_j \oplus \text{MSB}_s(O_j)$ $\quad j = 1, \ldots, N$

- 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; therefore, 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.

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 fed back to become the input for encrypting the next block of plaintext. In CFB, the output of the XOR unit is fed back 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

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

where

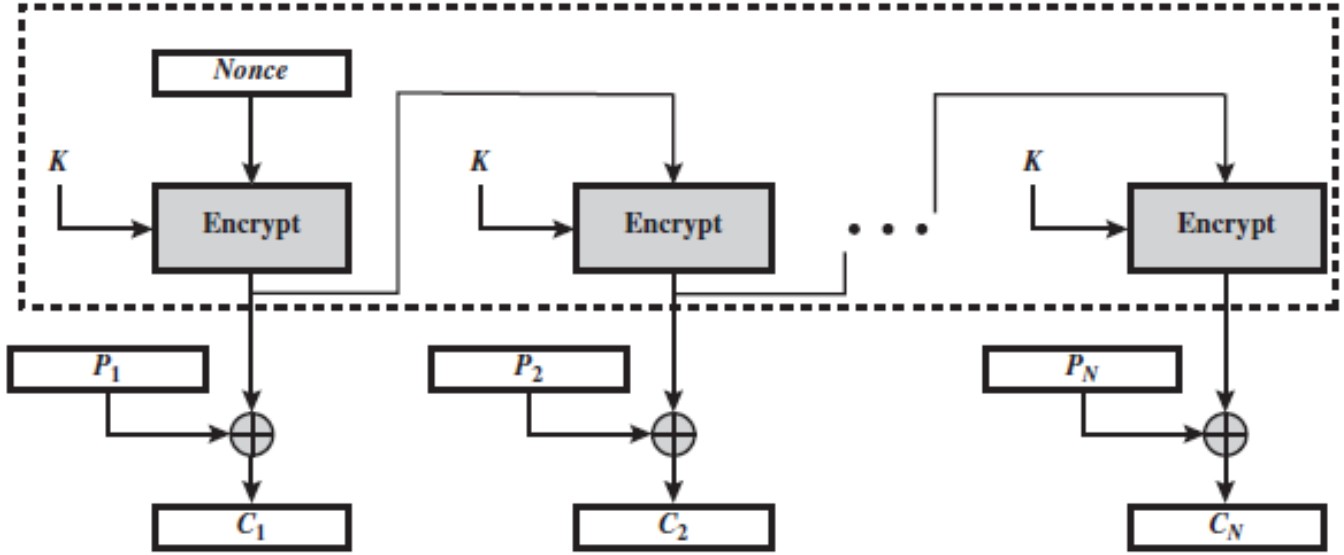
$$O_{j-1} = \mathrm{E}(K, O_{j-2})$$

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

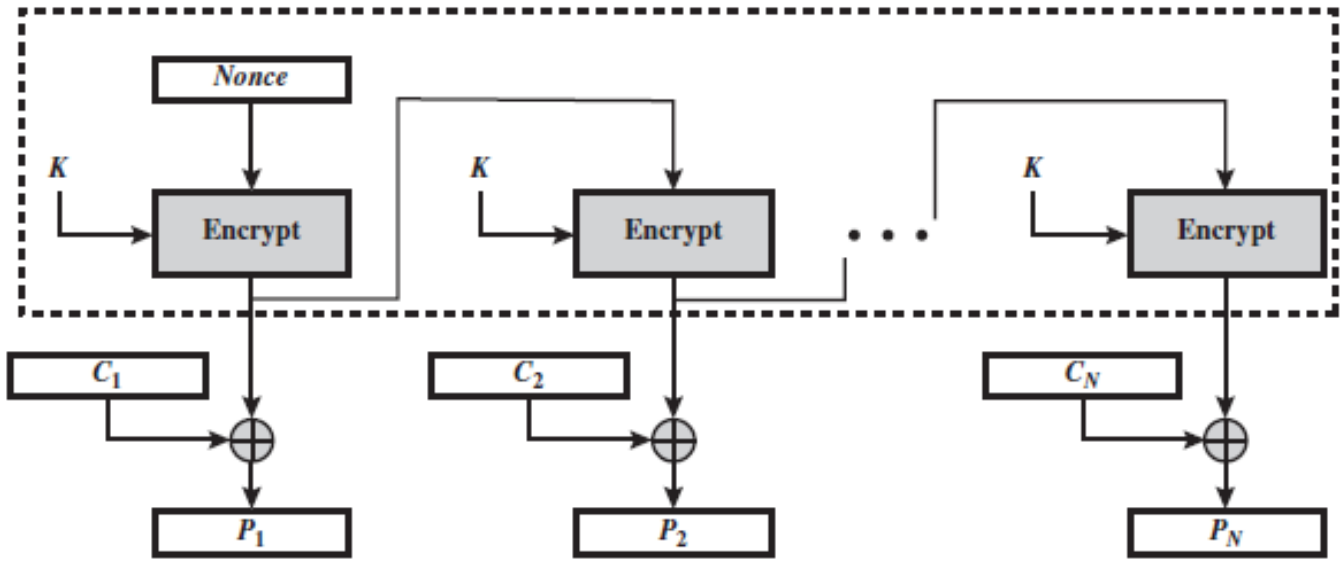
$$P_j = C_j \oplus \mathrm{E}(K, [C_{j-1} \oplus P_{j-1}])$$

OFB	$I_1 = Nonce$ $I_j = O_{j-1} \quad j = 2, \ldots, N$ $O_j = \mathrm{E}(K, I_j) \quad j = 1, \ldots, N$ $C_j = P_j \oplus O_j \quad j = 1, \ldots, N-1$ $C_N^* = P_N^* \oplus \mathrm{MSB}_u(O_N)$	$I_1 = Nonce$ $I_j = O_{j-1} \quad j = 2, \ldots, N$ $O_j = \mathrm{E}(K, I_j) \quad j = 1, \ldots, N$ $P_j = C_j \oplus O_j \quad j = 1, \ldots, N-1$ $P_N^* = C_N^* \oplus \mathrm{MSB}_u(O_N)$

- Let the size of a block be *b*. If the last block of plaintext contains *u* bits, 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; 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, *Oi*, 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 *Oi* stream.



(a) Encryption



(b) Decryption

Figure 6.6 Output Feedback (OFB) Mode

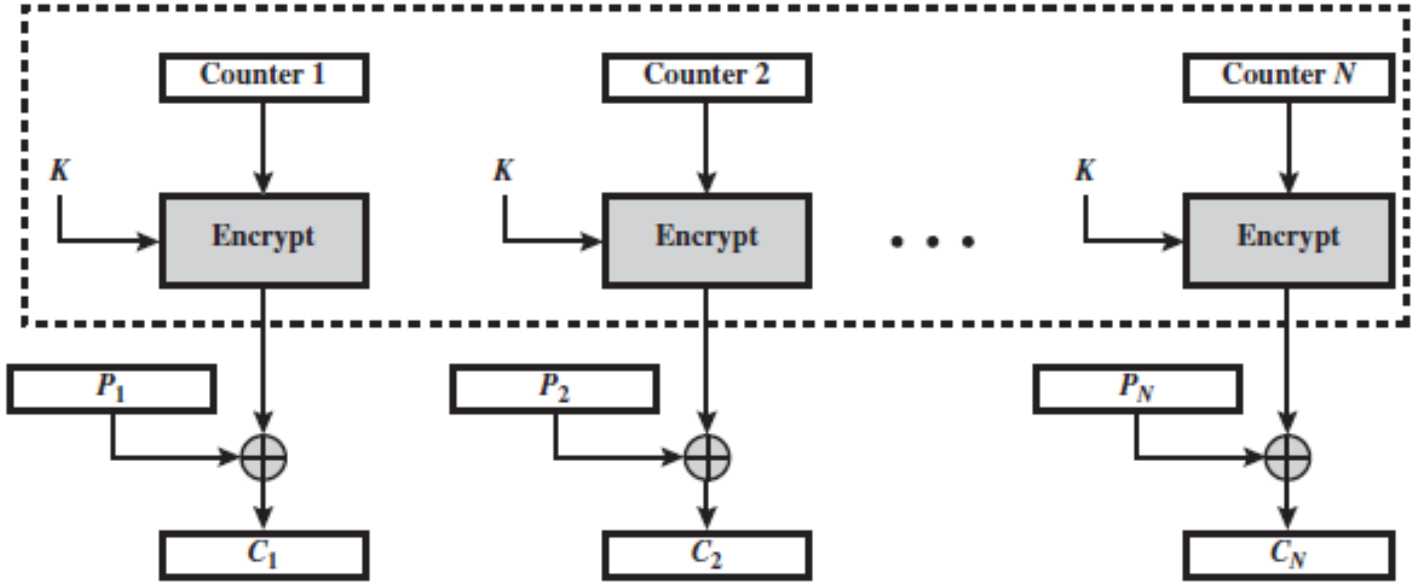
- One advantage of the OFB method is that bit errors in transmission do not propagate. For example, if a bit error occurs in $C1$, only the recovered value of $P1$ is affected; subsequent plaintext units are not corrupted.
- With CFB, $C1$ also serves as input to the shift register and therefore causes additional corruption downstream.
- The disadvantage of OFB is that it is more vulnerable to a message stream modification attack than is CFB. Consider that complementing a bit in the ciphertext complements the corresponding bit in the recovered plaintext. Thus, controlled changes to the recovered plaintext can be made.
- This may make it possible for an opponent, by making the necessary changes to the checksum portion of the message as well as to the data portion, to alter the ciphertext in such a way that it is not detected by an error-correcting code.
- 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.
- The generated stream that is XORed with the plaintext is itself independent of the plaintext.

Counter mode

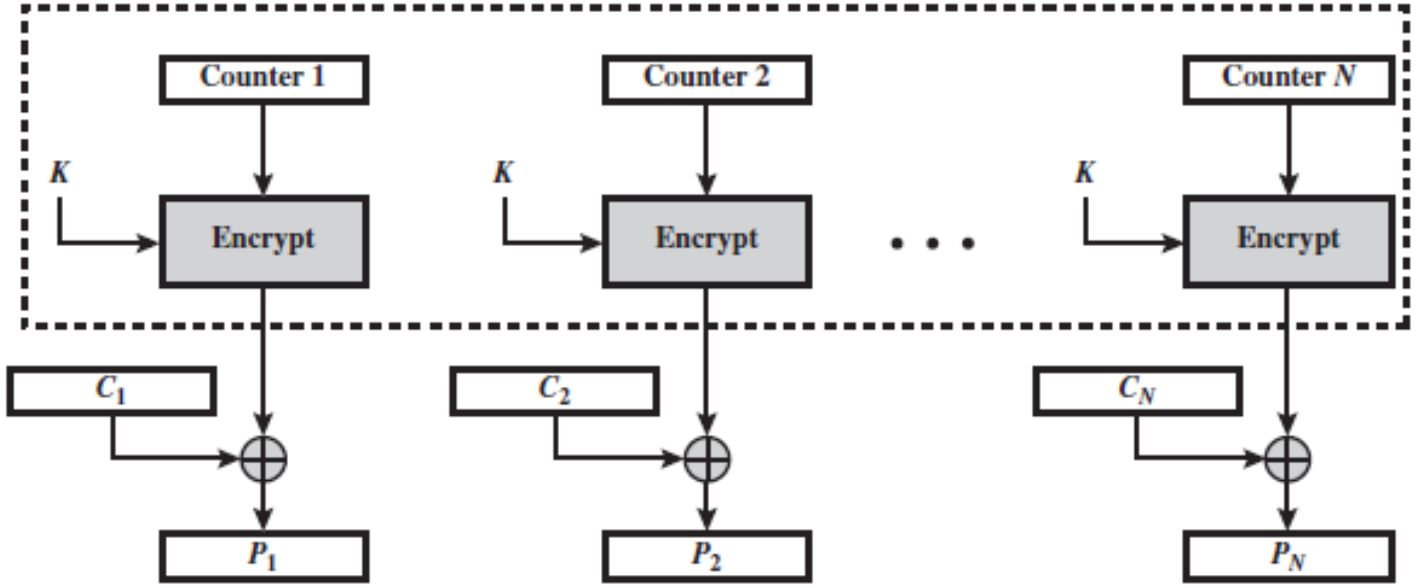
- A counter equal to the plaintext block size is used. The only requirement is that the counter value must be different for each plaintext block that is encrypted.
- Typically, the counter is initialized to some value and then incremented by 1 for each subsequent block (modulo 2^b, where b is the block 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 $T1$, $T2$, c, TN, we can define CTR mode as follows.

CTR	$C_j = P_j \oplus \mathrm{E}(K, T_j) \qquad j = 1, \ldots, N-1$ $C_N^* = P_N^* \oplus \mathrm{MSB}_u[\mathrm{E}(K, T_N)]$	$P_j = C_j \oplus \mathrm{E}(K, T_j) \qquad j = 1, \ldots, N-1$ $P_N^* = C_N^* \oplus \mathrm{MSB}_u[\mathrm{E}(K, T_N)]$

- 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. Further, all *Ti* values across all messages must be unique. If, contrary to this requirement, a counter value is used multiple times, then the confidentiality of all of the plaintext blocks corresponding to that counter value may be compromised.
- In particular, if any plaintext block that is encrypted using a given counter value is known, then the output of the encryption function can be determined easily from the associated ciphertext block.
- This output allows any other plaintext blocks that are encrypted using the same counter value to be easily recovered from their associated ciphertext blocks.



(a) Encryption



(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.
- advantages of CTR mode.
- **Hardware efficiency:** Unlike the three chaining modes, encryption (or decryption)

in CTR mode can be done in parallel on multiple blocks of plaintext or

ciphertext. For the chaining modes, the algorithm must complete the computation

on one block before beginning on the next block. This limits the maximum

throughput of the algorithm to the reciprocal of the time for one execution of

block encryption or decryption. In CTR mode, the throughput is only limited

by the amount of parallelism that is achieved.

- **Software efficiency:** Similarly, because of the opportunities for parallel execution in CTR mode, processors that support parallel features, such as aggressive pipelining, multiple instruction dispatch per clock cycle, a large number of registers, and SIMD instructions, can be effectively utilized.
- **Preprocessing:** The execution of the underlying encryption algorithm does not depend on input of the plaintext or ciphertext. Therefore, if sufficient memory is available and security is maintained, preprocessing can be used to prepare the output of the encryption boxes that feed into the XOR functions, as in Figure 6.7. When the plaintext or ciphertext input is presented, then the only computation is a series of XORs. Such a strategy greatly enhances throughput.

• **Random access:** The *i*th block of plaintext or ciphertext can be processed in random-access fashion. With the chaining modes, block *Ci* cannot be computed until the *i* - 1 prior block are computed. There may be applications in which a ciphertext is stored and it is desired to decrypt just one block; for such applications, the random access feature is attractive.

• **Provable security:** It can be shown that CTR is at least as secure as the other modes discussed in this section.

• **Simplicity:** Unlike ECB and CBC modes, CTR mode requires only the implementation

of the encryption algorithm and not the decryption algorithm.

This matters most when the decryption algorithm differs substantially from the encryption algorithm, as it does for AES. In addition, the decryption key scheduling need not be implemented.

- With the exception of ECB, all of the block cipher modes of operation involve feedback.
- To highlight the feedback mechanism, it is useful to think of the encryption function as taking input from a input register whose length equals the encryption block length and with output stored in an output register.
- The input register is updated one block at a time by the feedback mechanism. After each update, the encryption algorithm is executed, producing a result in the output register.
- Meanwhile, a block of plaintext is accessed. Note that both OFB and CTR produce output that is independent of both the plaintext and the ciphertext. Thus, they are natural candidates for stream ciphers that encrypt plaintext by XOR one full block at a time.