Chapter 5

Introduction to Modern Symmetric-key Ciphers

Objectives

- To distinguish between traditional and modern symmetric-key ciphers.
- To introduce modern block ciphers and discuss their characteristics.
- To explain why modern block ciphers need to be designed as substitution ciphers.
- To introduce components of block ciphers such as P-boxes and S-boxes.

Objectives (Continued)

- To discuss product ciphers and distinguish between two classes of product ciphers: Feistel and non-Feistel ciphers.
- To discuss two kinds of attacks particularly designed for modern block ciphers: differential and linear cryptanalysis.
- To introduce stream ciphers and to distinguish between synchronous and nonsynchronous stream ciphers.
- To discuss linear and nonlinear feedback shift registers for implementing stream ciphers.

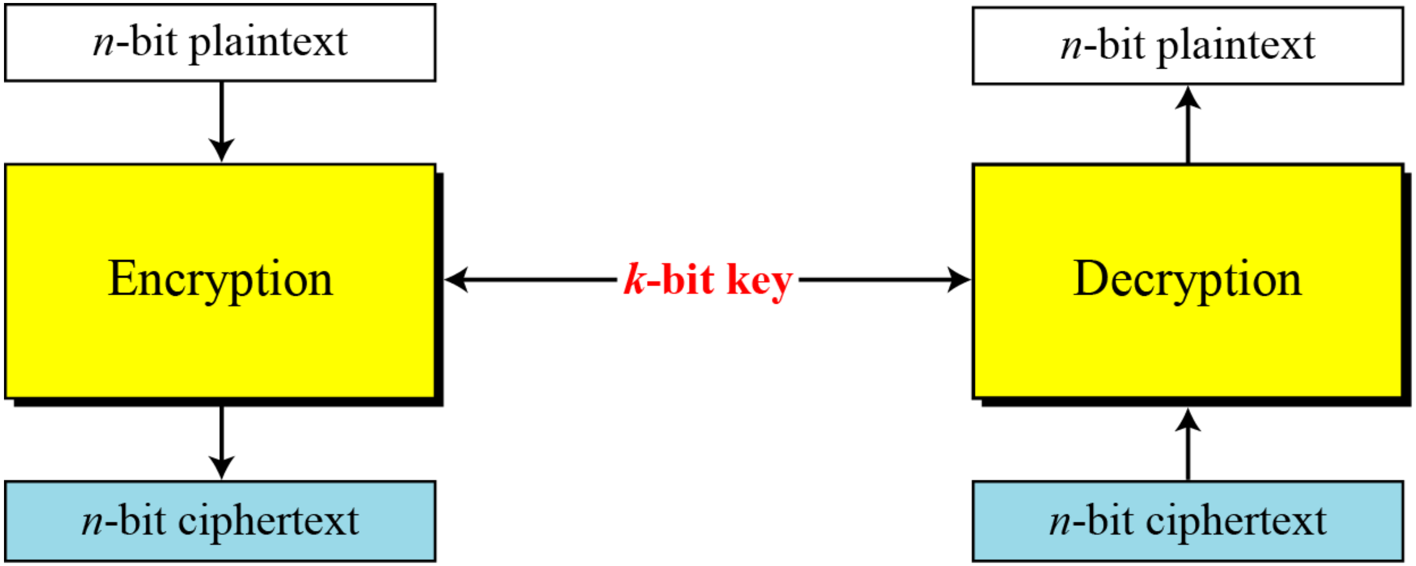
5-1 MODERN BLOCK CIPHERS

A symmetric-key modern block cipher encrypts an n-bit block of plaintext or decrypts an n-bit block of ciphertext. The encryption or decryption algorithm uses a k-bit key.

Topics discussed in this section:

5.1.1 Substitution or Transposition
5.1.2 Block Ciphers as Permutation Groups
5.1.3 Components of a Modern Block Cipher
5.1.4 Product Ciphers
5.1.5 Two Classes of Product Ciphers
5.1.6 Attacks on Block Ciphers

Figure 5.1 ***A modern block cipher***



5.1.3 Components of a Modern Block Cipher

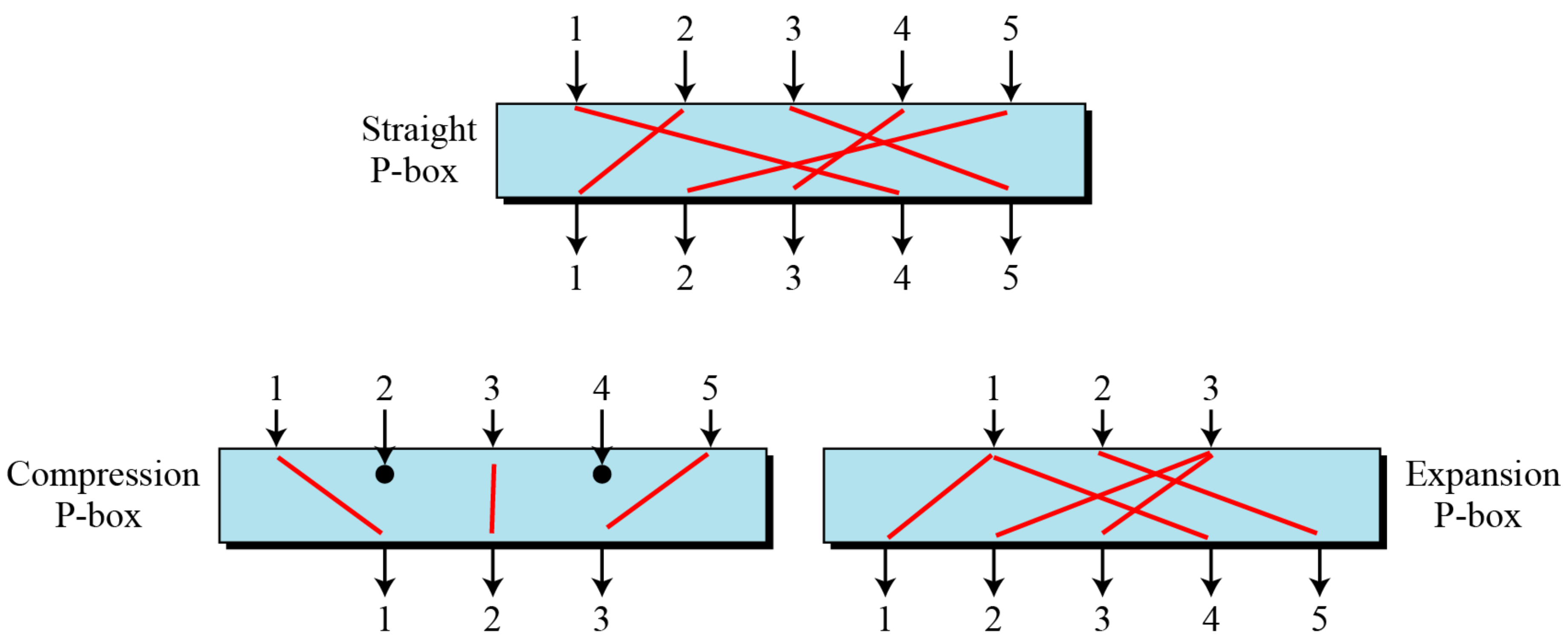
Modern block ciphers normally are keyed substitution ciphers in which the key allows only partial mappings from the possible inputs to the possible outputs.

P-Boxes

A P-box (permutation box) parallels the traditional transposition cipher for characters. It transposes bits.

5.1.3 Continued

Figure 5.4 *Three types of P-boxes*

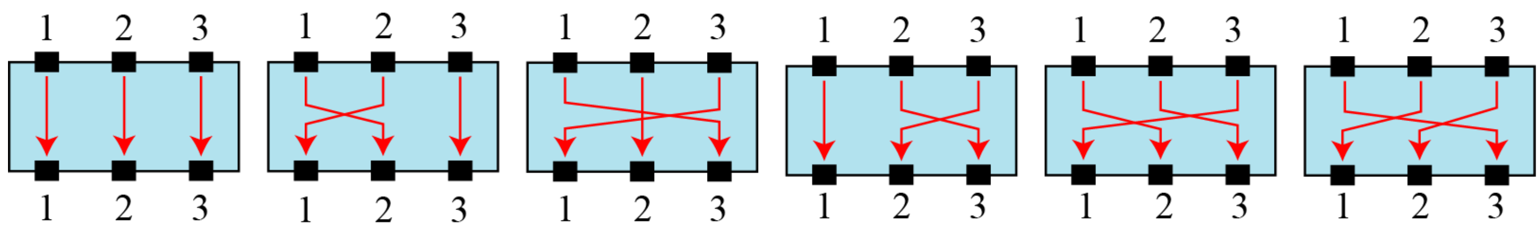


5.1.3 Continued

Example 5.5

Figure 5.5 shows all 6 possible mappings of a 3 × 3 P-box.

Figure 5.5 *The possible mappings of a 3 × 3 P-box*



5.1.3 Continued

Straight P-Boxes

Table 5.1 *Example of a permutation table for a straight P-box*

58	50	42	34	26	18	10	02	60	52	44	36	28	20	12	04
62	54	46	38	30	22	14	06	64	56	48	40	32	24	16	08
57	49	41	33	25	17	09	01	59	51	43	35	27	19	11	03
61	53	45	37	29	21	13	05	63	55	47	39	31	23	15	07

Example 5.6

Design an 8 × 8 permutation table for a straight P-box that moves the two middle bits (bits 4 and 5) in the input word to the two ends (bits 1 and 8) in the output words. Relative positions of other bits should not be changed.

Solution

We need a straight P-box with the table [4 1 2 3 6 7 8 5]. The relative positions of input bits 1, 2, 3, 6, 7, and 8 have not been changed, but the first output takes the fourth input and the eighth output takes the fifth input.

Compression P-Boxes

A compression P-box is a P-box with n inputs and m outputs where m < n.

Table 5.2 *Example of a 32 × 24 permutation table*

01	02	03	21	22	26	27	28	29	13	14	17
18	19	20	04	05	06	10	11	12	30	31	32

5.1.3 Continued

Compression P-Box

Table 5.2 *Example of a 32 × 24 permutation table*

01	02	03	21	22	26	27	28	29	13	14	17
18	19	20	04	05	06	10	11	12	30	31	32

5.1.3 Continued

Expansion P-Boxes

An expansion P-box is a P-box with n inputs and m outputs where m > n.

Table 5.3 *Example of a 12 × 16 permutation table*

01	09	10	11	12	01	02	03	03	04	05	06	07	08	09	12

5.1.3 *Continued*

P-Boxes: Invertibility

Note

A straight P-box is invertible, but compression and expansion P-boxes are not.

Example 5.7

Figure 5.6 shows how to invert a permutation table represented as a one-dimensional table.

Figure 5.6 ***Inverting a permutation table***

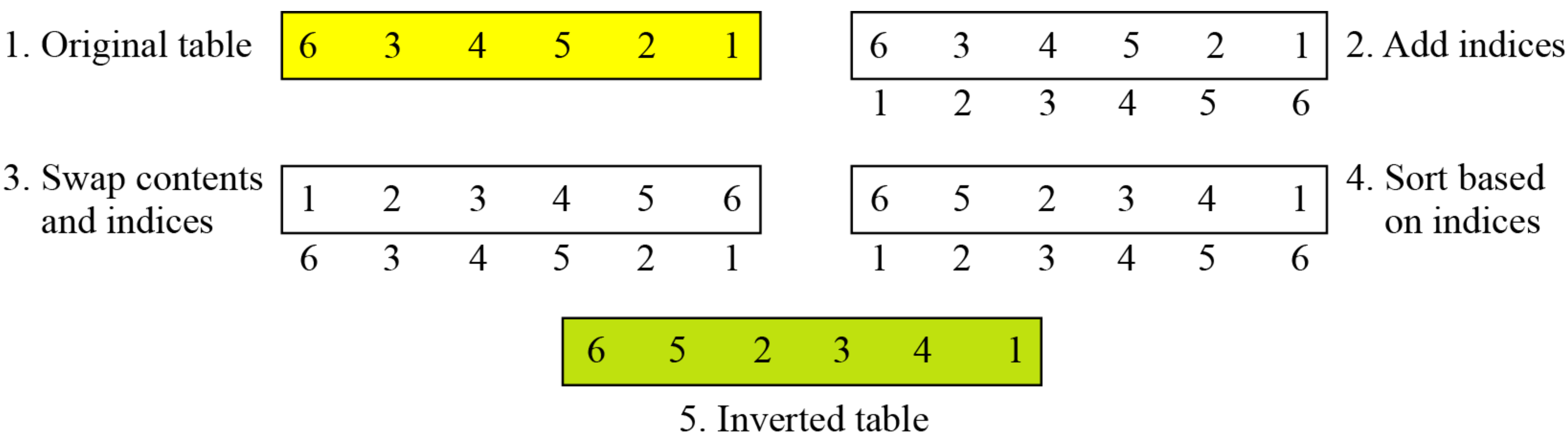
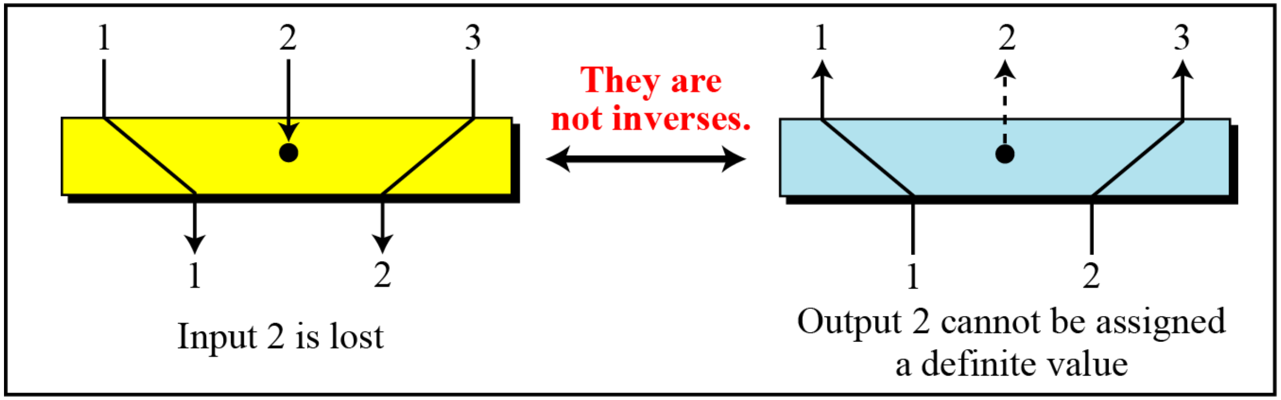
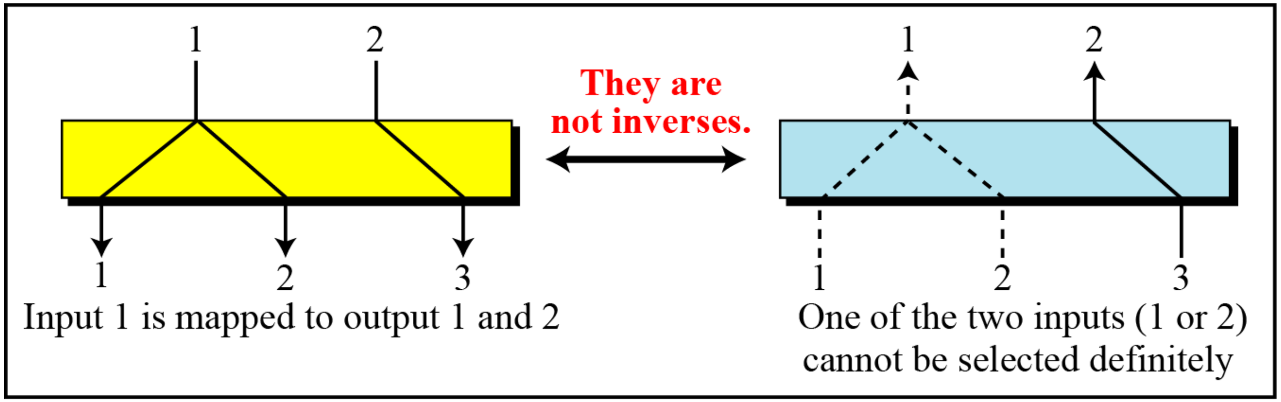


Figure 5.7 *Compression and expansion P-boxes are non-invertible*

Compression P-box





Expansion P-box

5.1.3 Continued

S-Box

An S-box (substitution box) can be thought of as a miniature substitution cipher.

Note

An S-box is an $m \times n$ substitution unit, where m and n are not necessarily the same.

Example 5.8

In an S-box with three inputs and two outputs, we have

$$y_1 = x_1 \oplus x_2 \oplus x_3 \qquad y_2 = x_1$$

The S-box is linear because $a_{1,1} = a_{1,2} = a_{1,3} = a_{2,1} = 1$ and $a_{2,2} = a_{2,3} = 0$. The relationship can be represented by matrices, as shown below:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

5.1.3 Continued

Example 5.9

In an S-box with three inputs and two outputs, we have

$$y_1 = (x_1)^3 + x_2 \qquad y_2 = (x_1)^2 + x_1x_2 + x_3$$

where multiplication and addition is in GF(2). The S-box is nonlinear because there is no linear relationship between the inputs and the outputs.

5.1.3 Continued

Example 5.10

The following table defines the input/output relationship for an S-box of size 3 × 2. The leftmost bit of the input defines the row; the two rightmost bits of the input define the column. The two output bits are values on the cross section of the selected row and column.

Leftmost bit ↓ / Rightmost bits ←

	00	01	10	11
0	00	10	01	11
1	10	00	11	01

Output bits

Based on the table, an input of 010 yields the output 01. An input of 101 yields the output of 00.

Example 5.11

Figure 5.8 shows an example of an invertible S-box. For example, if the input to the left box is 001, the output is 101. The input 101 in the right table creates the output 001, which shows that the two tables are inverses of each other.

Figure 5.8 *S-box tables for Example 5.11*

	00	01	10	11
0	011	101	111	100
1	000	010	001	110

Table used for encryption

	00	01	10	11
0	100	110	101	000
1	011	001	111	010

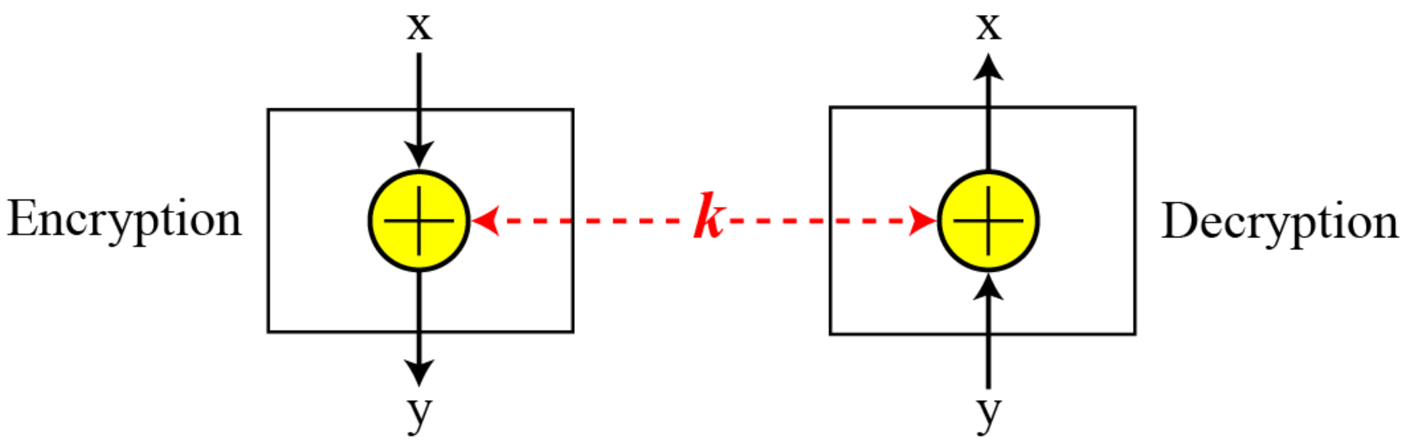
Table used for decryption

5.1.3 Continued

Exclusive-Or

An important component in most block ciphers is the exclusive-or operation.

Figure 5.9 *Invertibility of the exclusive-or operation*



5.1.3 Continued

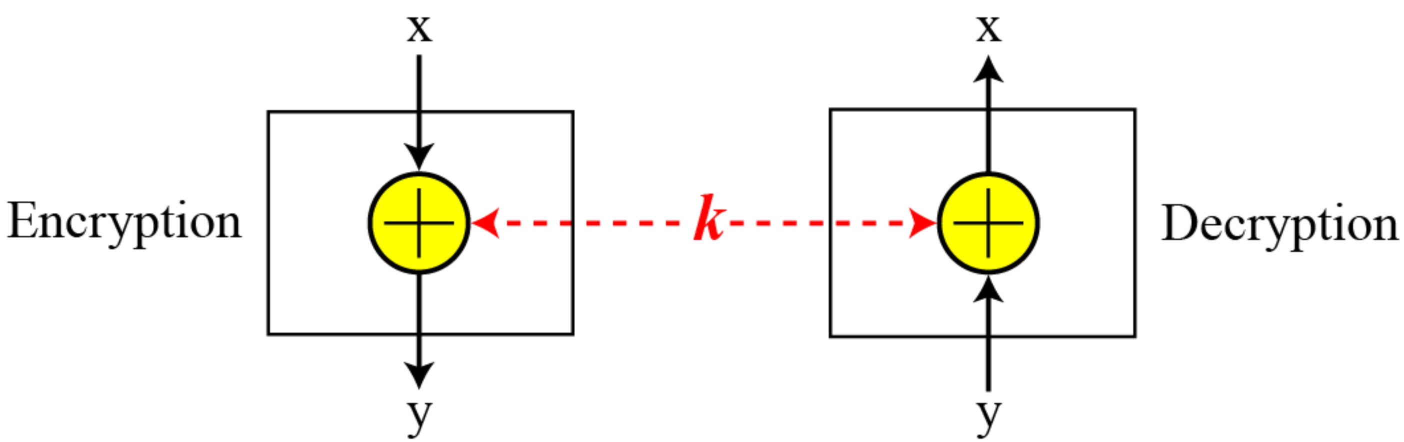
Exclusive-Or (Continued)

An important component in most block ciphers is the exclusive-or operation. As we discussed in Chapter 4, addition and subtraction operations in the GF(2^n) field are performed by a single operation called the exclusive-or (XOR).

The five properties of the exclusive-or operation in the GF(2n) field makes this operation a very interesting component for use in a block cipher: closure, associativity, commutativity, existence of identity, and existence of inverse.

5.1.1 Continued

Figure 5.9 *Invertibility of the exclusive-or operation*

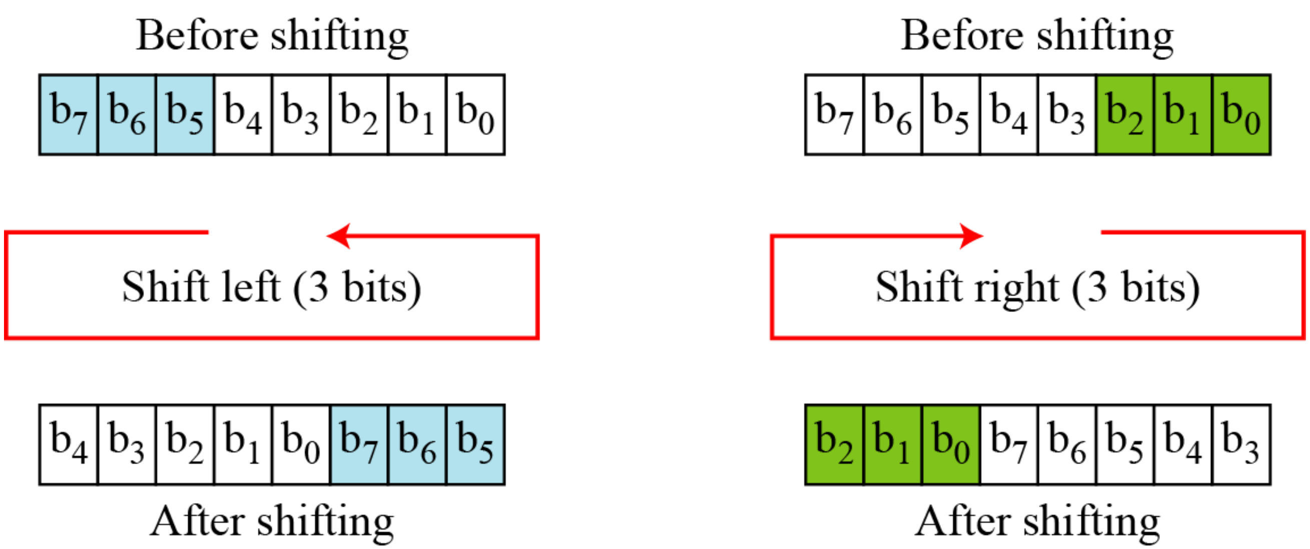


5.1.3 Continued

Circular Shift

Another component found in some modern block ciphers is the circular shift operation.

Figure 5.10 *Circular shifting an 8-bit word to the left or right*

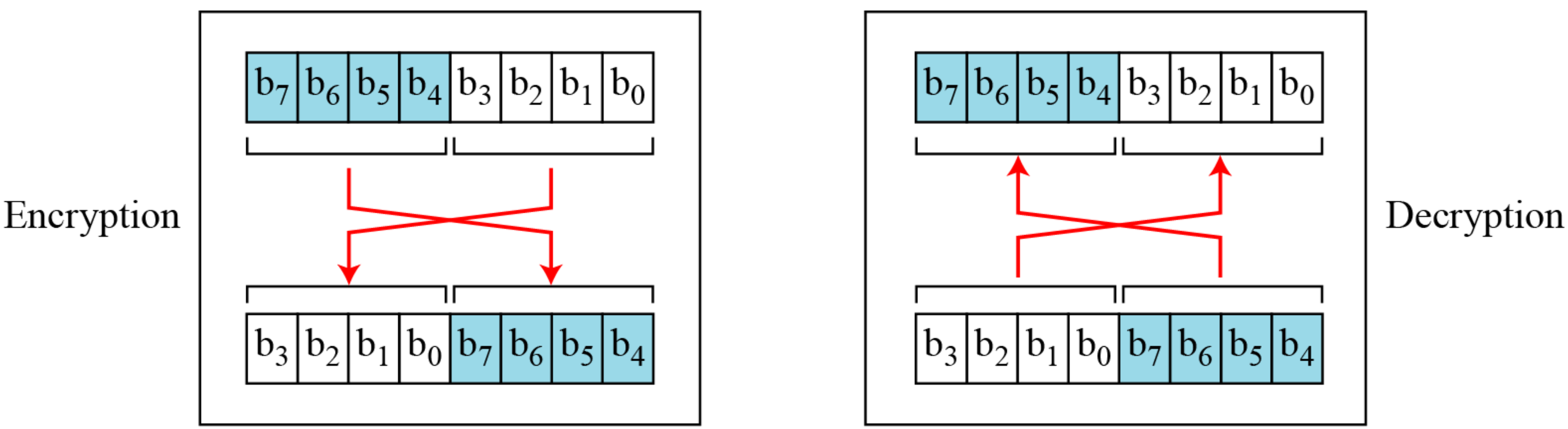


5.1.3 Continued

Swap

The swap operation is a special case of the circular shift operation where k = n/2.

Figure 5.11 ***Swap operation on an 8-bit word***



Split and Combine

Two other operations found in some block ciphers are split and combine.

Figure 5.12 ***Split and combine operations on an 8-bit word***

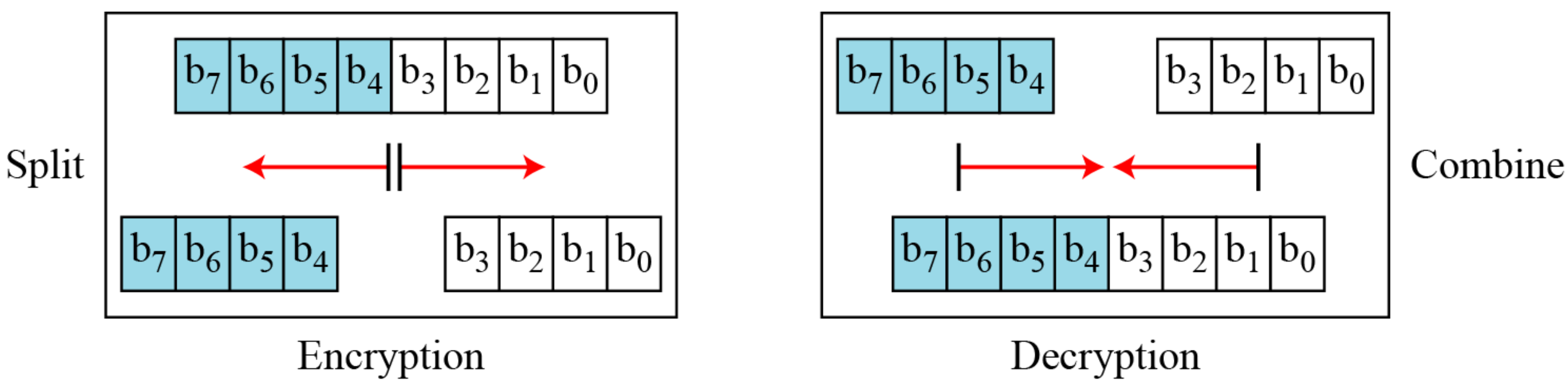
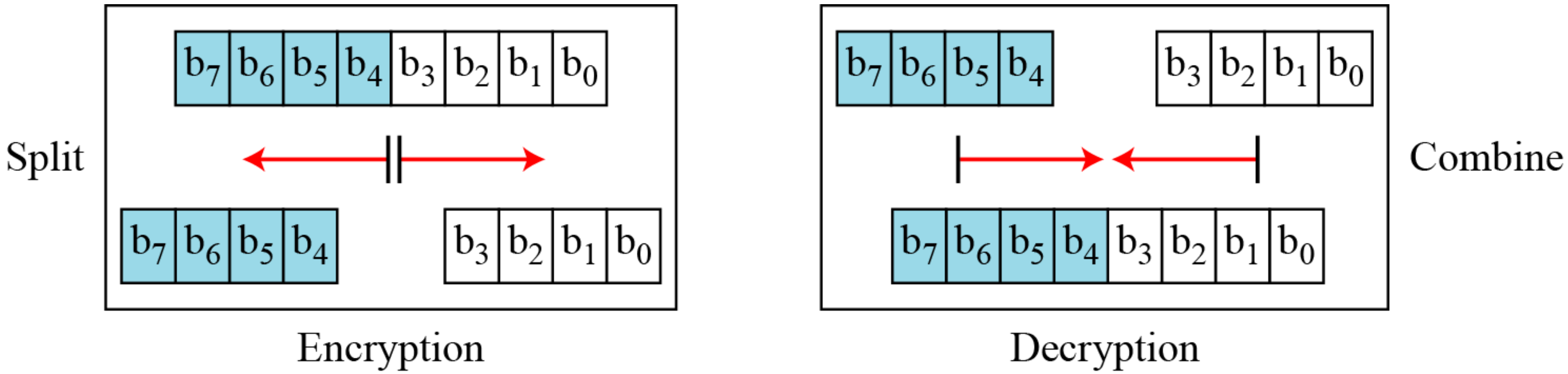


Figure 5.12 *Split and combine operations on an 8-bit word*



5.1.4 Product Ciphers

Shannon introduced the concept of a product cipher. A product cipher is a complex cipher combining substitution, permutation, and other components discussed in previous sections.

5.1.4 Continued

Diffusion

The idea of diffusion is to hide the relationship between the ciphertext and the plaintext.

Note

Diffusion hides the relationship between the ciphertext and the plaintext.

5.1.4 Continued

Confusion

The idea of confusion is to hide the relationship between the ciphertext and the key.

Note

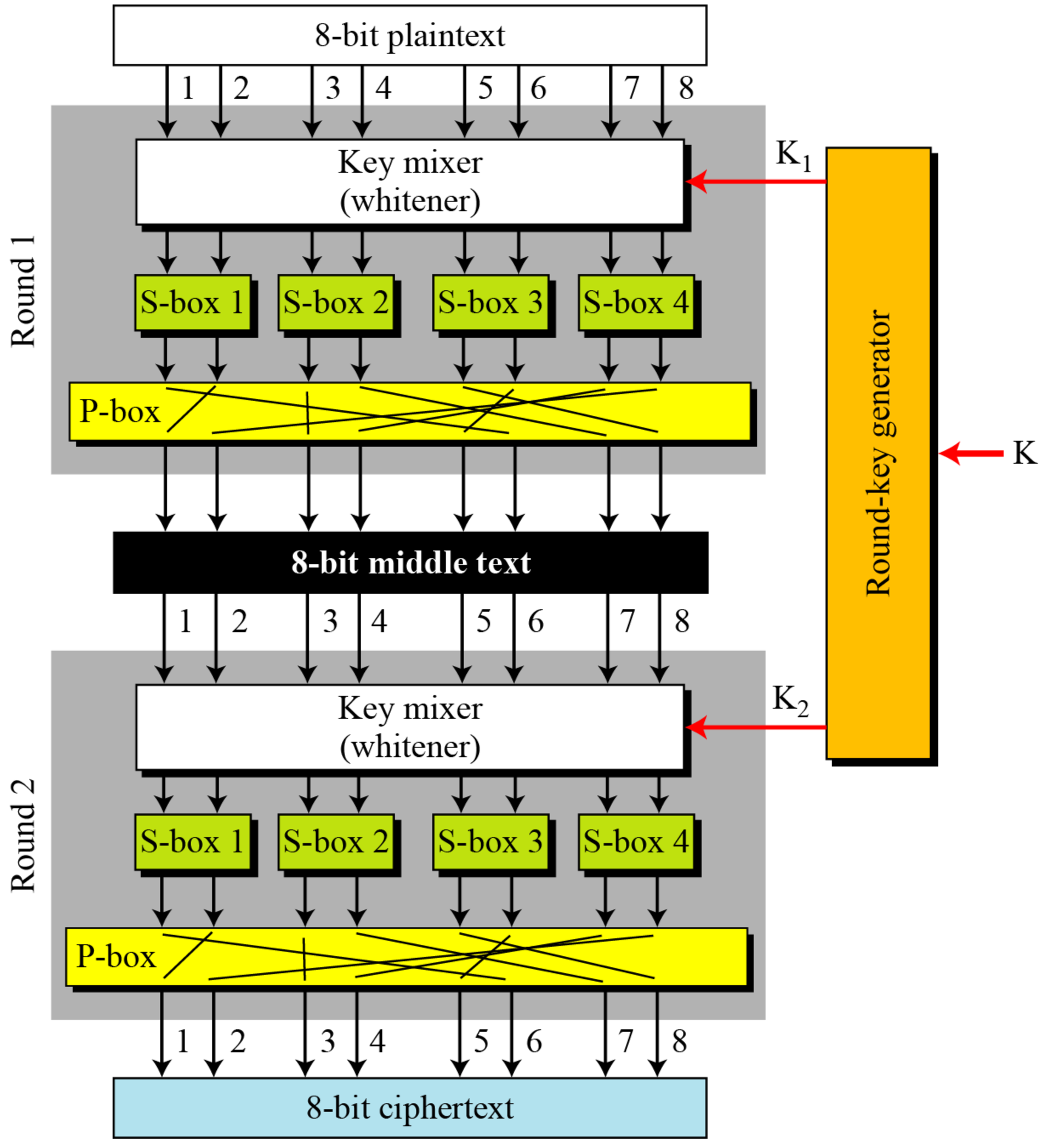
Confusion hides the relationship between the ciphertext and the key.

Rounds

Diffusion and confusion can be achieved using iterated product ciphers where each iteration is a combination of S-boxes, P-boxes, and other components.

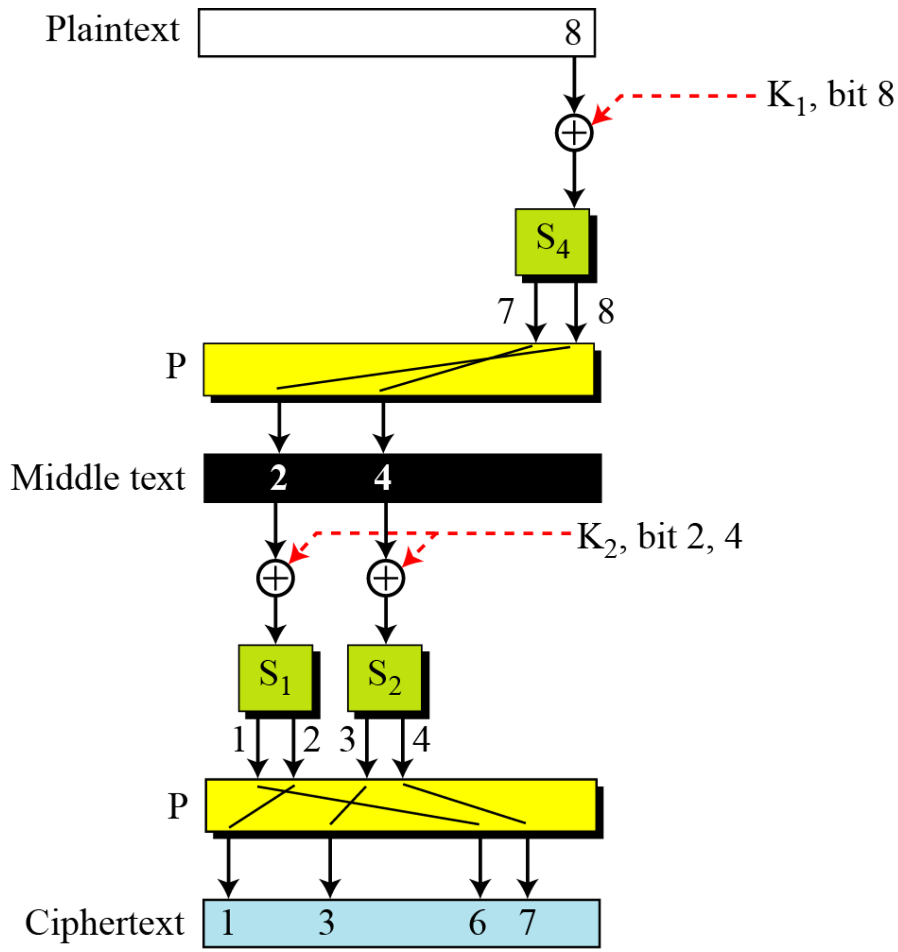
5.1.4 Continued

Figure 5.13 *A product cipher made of two rounds*



5.1.4 Continued

Figure 5.14 *Diffusion and confusion in a block cipher*



5.1.5 Two Classes of Product Ciphers

Modern block ciphers are all product ciphers, but they are divided into two classes.

1. ***Feistel ciphers***

2. ***Non-Feistel ciphers***

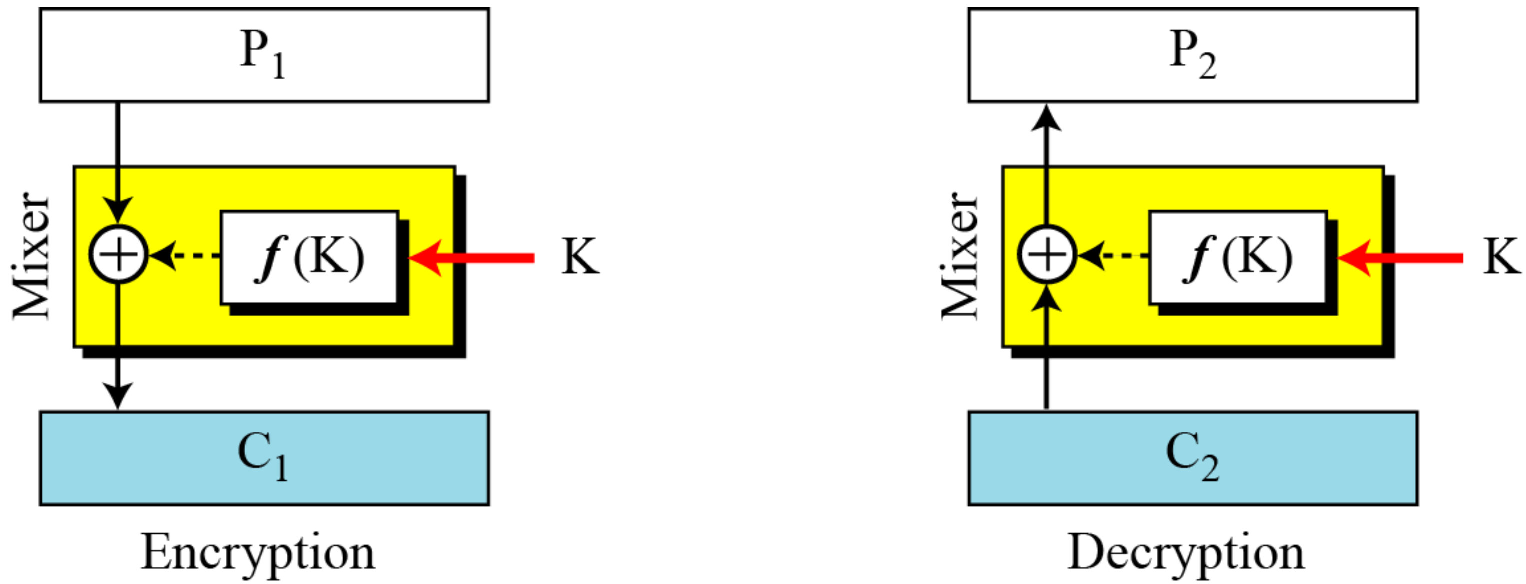
5.1.5 Continued

Feistel Ciphers

Feistel designed a very intelligent and interesting cipher that has been used for decades. A Feistel cipher can have three types of components: self-invertible, invertible, and noninvertible.

5.1.5 Continued

Figure 5.15 *The first thought in Feistel cipher design*



Note

Diffusion hides the relationship between the ciphertext and the plaintext.

5.1.3 Continued

Example 5.12

This is a trivial example. The plaintext and ciphertext are each 4 bits long and the key is 3 bits long. Assume that the function takes the first and third bits of the key, interprets these two bits as a decimal number, squares the number, and interprets the result as a 4-bit binary pattern. Show the results of encryption and decryption if the original plaintext is 0111 and the key is 101.

Solution

The function extracts the first and second bits to get 11 in binary or 3 in decimal. The result of squaring is 9, which is 1001 in binary.

Encryption: $C = P \oplus f\,(K) = 0111 \oplus 1001 = 1110$

Decryption: $P = C \oplus f\,(K) = 1110 \oplus 1001 = 0111$

Figure 5.16 *Improvement of the previous Feistel design*

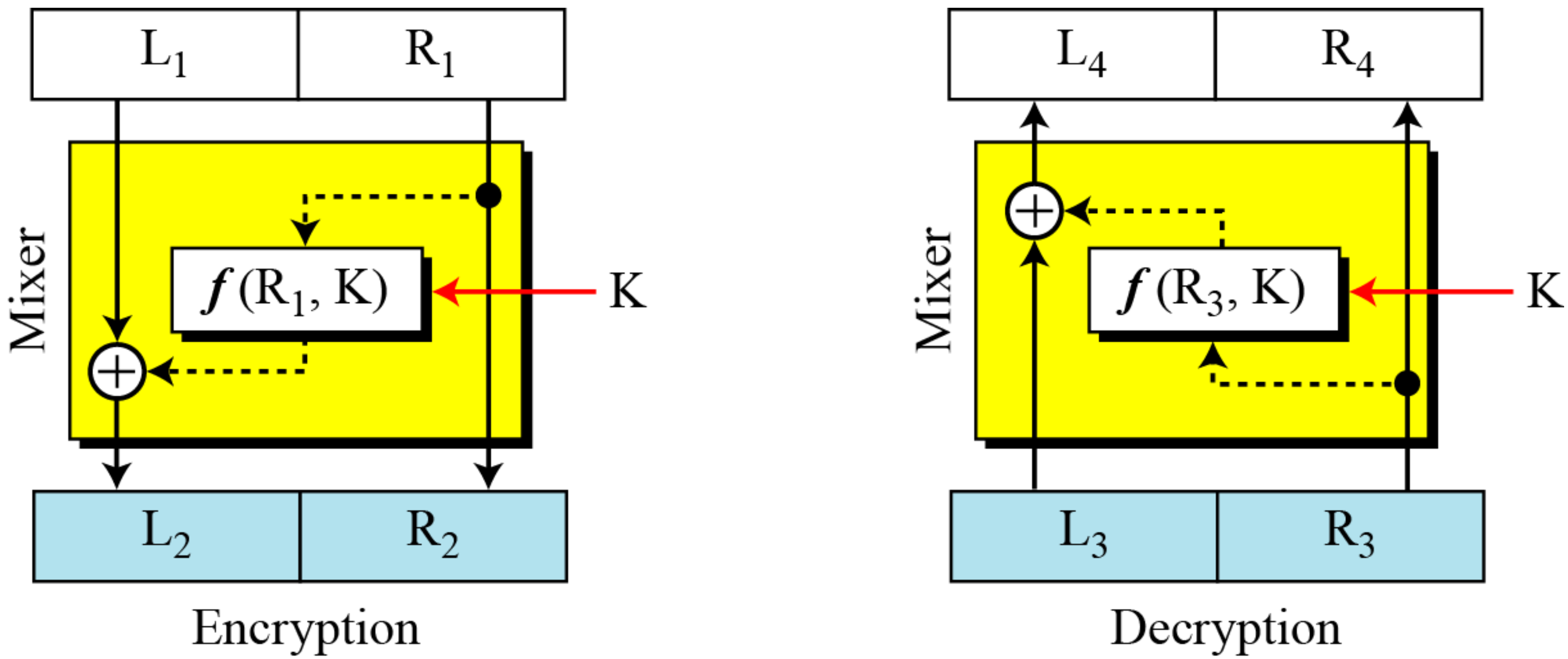
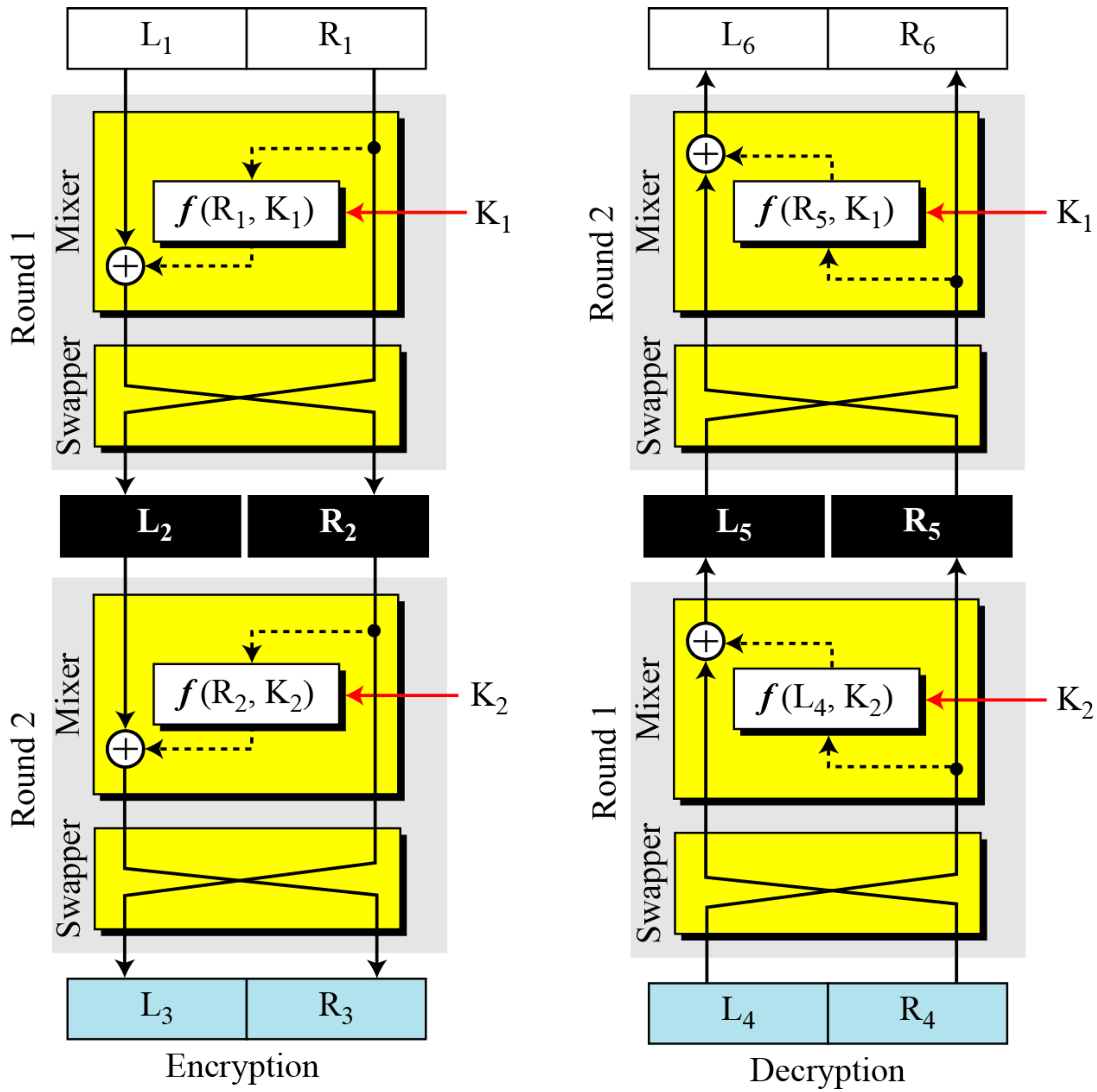


Figure 5.17 *Final design of a Feistel cipher with two rounds*



5.1.5 Continued

Non-Feistel Ciphers

A non-Feistel cipher uses only invertible components. A component in the encryption cipher has the corresponding component in the decryption cipher.

5.1.6 Attacks on Block Ciphers

Attacks on traditional ciphers can also be used on modern block ciphers, but today's block ciphers resist most of the attacks discussed in Chapter 3.

5.1.5 Continued

Differential Cryptanalysis

Eli Biham and Adi Shamir introduced the idea of differential cryptanalysis. This is a chosen-plaintext attack.

5.1.6 Continued

Example 5.13

Assume that the cipher is made only of one exclusive-or operation, as shown in Figure 5.18. Without knowing the value of the key, Eve can easily find the relationship between plaintext differences and ciphertext differences if by plaintext difference we mean P1 ⊕ P2 and by ciphertext difference, we mean C1⊕ C2. The following proves that C1 ⊕ C2 = P1 ⊕ P2:

$$C_1 = P_1 \oplus K \quad C_2 = P_2 \oplus K \quad \rightarrow \quad C_1 \oplus C_2 = P_1 \oplus K \oplus P_2 \oplus K = P_1 \oplus P_2$$

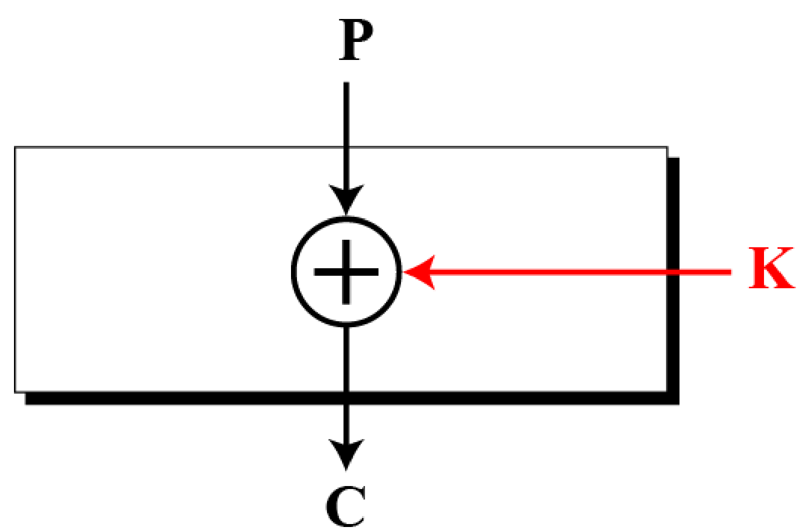


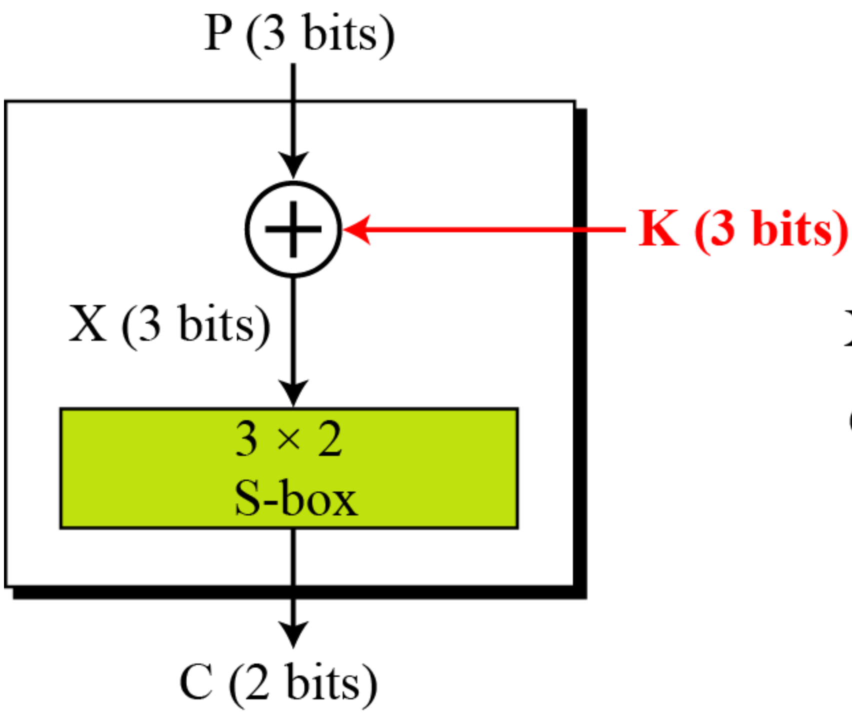
Figure 5.18 *Diagram for Example 5.13*

5.1.6 Continued

Example 5.14

We add one S-box to Example 5.13, as shown in Figure 5.19.

Figure 5.19 ***Diagram for Example 5.14***



X	000	001	010	011	100	101	110	111
C	11	00	10	10	01	00	11	00

S-box table

Example 5.14 Continued

Eve now can create a probabilistic relationship as shone in Table 5.4.

Table 5.4 *Differential input/output*

$P_1 \oplus P_2$ \ $C_1 \oplus C_2$	*00*	*01*	*10*	*11*
000	**8**			
001	**2**	**2**		**4**
010	**2**	**2**	**4**	
011		**4**	**2**	**2**
100	**2**	**2**	**4**	
101		**4**	**2**	**2**
110	**4**		**2**	**2**
111			**2**	**6**

Example 5.15

The heuristic result of Example 5.14 can create probabilistic information for Eve as shown in Table 5.5.

Table 5.5 *Differential distribution table*

$P_1 \oplus P_2$ \ $C_1 \oplus C_2$	*00*	*01*	*10*	*11*
000	**1**	**0**	**0**	**0**
001	**0.25**	**0.25**	**0**	**0.50**
010	**0.25**	**0.25**	**0.50**	**0**
011	**0**	**0.50**	**0.25**	**0.25**
100	**0.25**	**0.25**	**0.50**	**0**
101	**0**	**0.50**	**0.25**	**0.25**
110	**0.50**	**0**	**0.25**	**0.25**
111	**0**	**0**	**0.25**	**0.75**

Example 5.16

Looking at Table 5.5, Eve knows that if $P_1 \oplus P_2 = 001$, then $C_1 \oplus C_2 = 11$ with the probability of 0.50 (50 percent). She tries $C_1 = 00$ and gets $P_1 = 010$ (chosen-ciphertext attack). She also tries $C_2 = 11$ and gets $P_2 = 011$ (another chosen-ciphertext attack). Now she tries to work backward, based on the first pair, P_1 and C_1,

$$C_1 = 00 \rightarrow X_1 = 001 \text{ or } X_1 = 111$$
$$\text{If } X_1 = 001 \rightarrow K = X_1 \oplus P_1 = 011 \qquad \text{If } X_1 = 111 \rightarrow K = X_1 \oplus P_1 = 101$$

$$C_2 = 11 \rightarrow X_2 = 000 \text{ or } X_1 = 110$$
$$\text{If } X_2 = 000 \rightarrow K = X_2 \oplus P_2 = 011 \qquad \text{If } X_2 = 110 \rightarrow K = X_2 \oplus P_2 = 101$$

The two tests confirm that K = 011 or K =101.

5.1.6 Continued

Note

Differential cryptanalysis is based on a nonuniform differential distribution table of the S-boxes in a block cipher.

Note

A more detailed differential cryptanalysis is given in Appendix N.

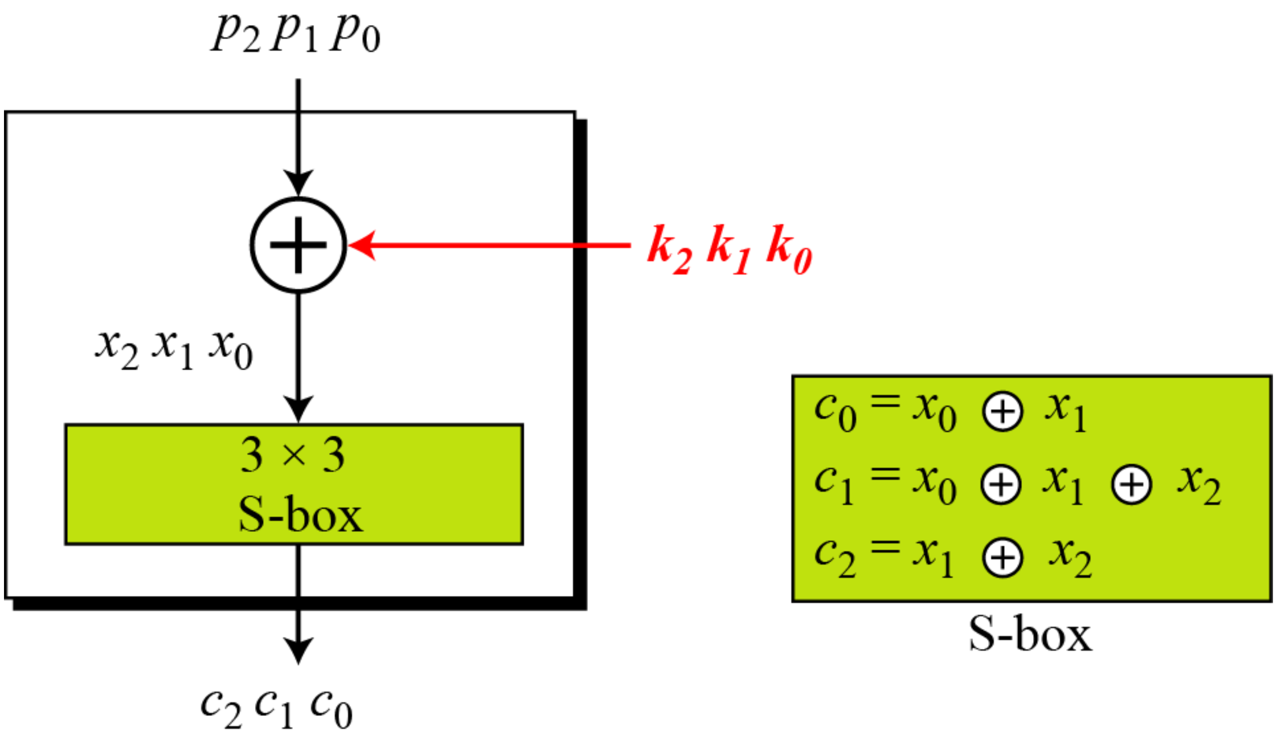
5.1.6 Continued

Linear Cryptanalysis

Linear cryptanalysis was presented by Mitsuru Matsui in 1993. The analysis uses known plaintext attacks.

5.1.6 Continued

Figure 5.20 *A simple cipher with a linear S-box*



5.1.6 Continued

$$c_0 = p_0 \oplus k_0 \oplus p_1 \oplus k_1$$
$$c_1 = p_0 \oplus k_0 \oplus p_1 \oplus k_1 \oplus p_2 \oplus k_2$$
$$c_2 = p_1 \oplus k_1 \oplus p_2 \oplus k_2$$

Solving for three unknowns, we get.

$$k_1 = (p_1) \oplus (c_0 \oplus c_1 \oplus c_2)$$
$$k_2 = (p_2) \oplus (c_0 \oplus c_1)$$
$$k_0 = (p_0) \oplus (c_1 \oplus c_2)$$

This means that three known-plaintext attacks can find the values of k_0, k_1, and k_2.

5.1.6 Continued

In some modern block ciphers, it may happen that some S-boxes are not totally nonlinear; they can be approximated, probabilistically, by some linear functions.

$$(k_0 \oplus k_1 \oplus \cdots \oplus k_x) = (p_0 \oplus p_1 \oplus \cdots \oplus p_y) \oplus (c_0 \oplus c_1 \oplus \cdots \oplus c_z)$$

where $1 \leq x \leq m$, $1 \leq y \leq n$, ***and*** $1 \leq z \leq n$.

Note

A more detailed linear cryptanalysis is given in Appendix N.

5-2 MODERN STREAM CIPHERS

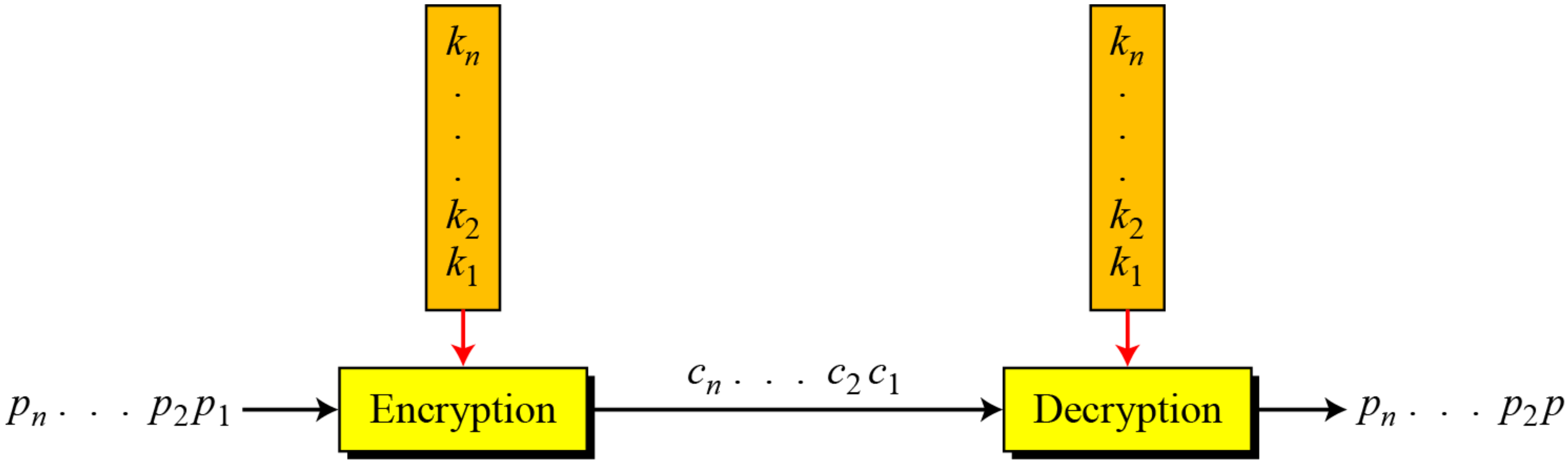
In a modern stream cipher, encryption and decryption are done r bits at a time. We have a plaintext bit stream $P = p_n \ldots p_2\ p_1$, ***a ciphertext bit stream*** $C = c_n \ldots c_2\ c_1$, ***and a key bit stream*** $K = k_n \ldots k_2\ k_1$, ***in which*** p_i , c_i , ***and*** k_i ***are r-bit words.***

Topics discussed in this section:

5.2.1 Synchronous Stream Ciphers
5.2.2 Nonsynchronous Stream Ciphers

5.2 Continued

Figure 5.20 ***Stream cipher***



Note

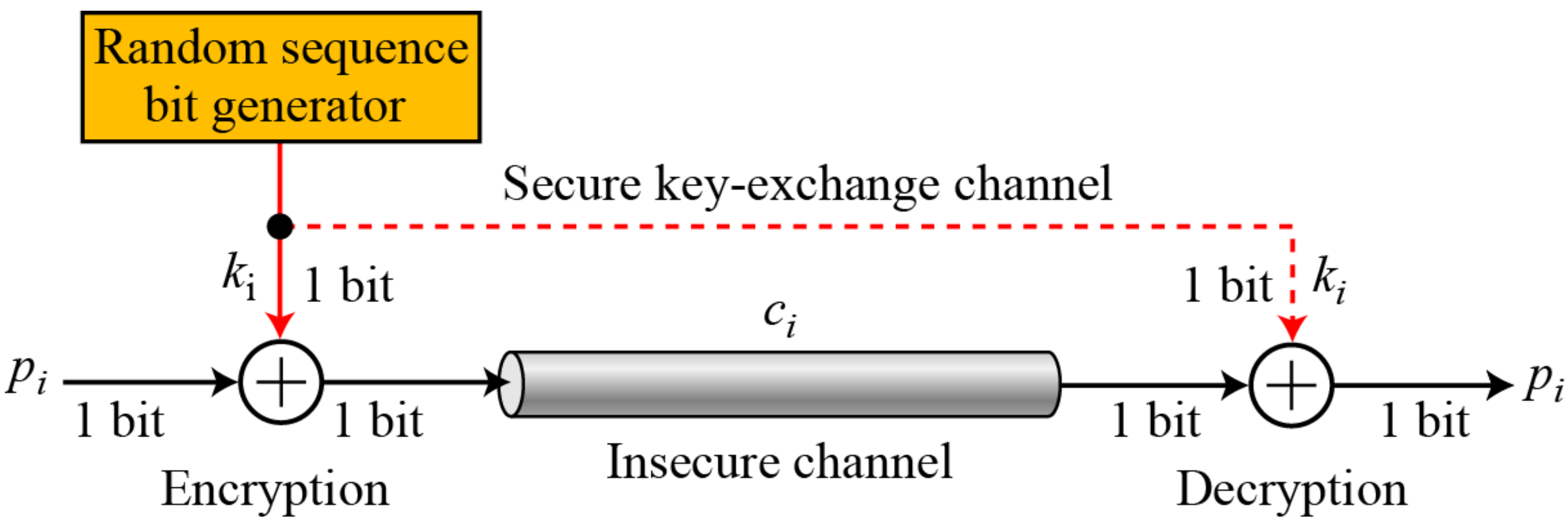
In a modern stream cipher, each *r*-bit word in the plaintext stream is enciphered using an *r*-bit word in the key stream to create the corresponding *r*-bit word in the ciphertext stream.

5.2.1 Synchronous Stream Ciphers

Note

In a synchronous stream cipher the key is independent of the plaintext or ciphertext.

Figure 5.22 ***One-time pad***



Example 5.17

What is the pattern in the ciphertext of a one-time pad cipher in each of the following cases?

a. The plaintext is made of *n* 0's.

b. The plaintext is made of *n* 1's.

c. The plaintext is made of alternating 0's and 1's.

d. The plaintext is a random string of bits.

Solution

a. Because $0 \oplus k_i = k_i$, the ciphertext stream is the same as the key stream. If the key stream is random, the ciphertext is also random. The patterns in the plaintext are not preserved in the ciphertext.

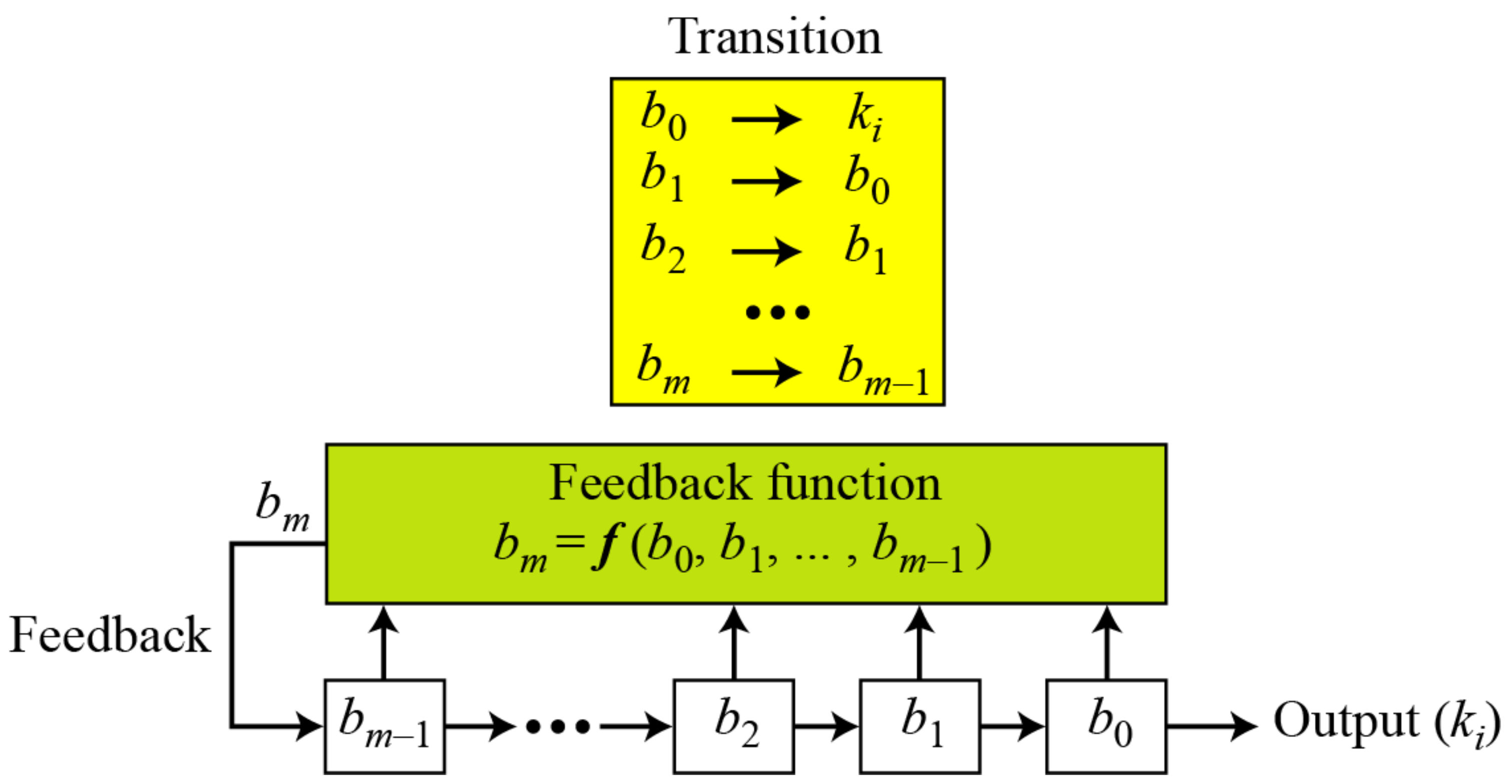
Example 5.7 (Continued)

b. Because $1 \oplus k_i = \overline{k_i}$ where $\overline{k_i}$ is the complement of k_i, the ciphertext stream is the complement of the key stream. If the key stream is random, the ciphertext is also random. Again the patterns in the plaintext are not preserved in the ciphertext.

c. In this case, each bit in the ciphertext stream is either the same as the corresponding bit in the key stream or the complement of it. Therefore, the result is also a random string if the key stream is random.

d. In this case, the ciphertext is definitely random because the exclusive-or of two random bits results in a random bit.

Figure 5.23 *Feedback shift register (FSR)*



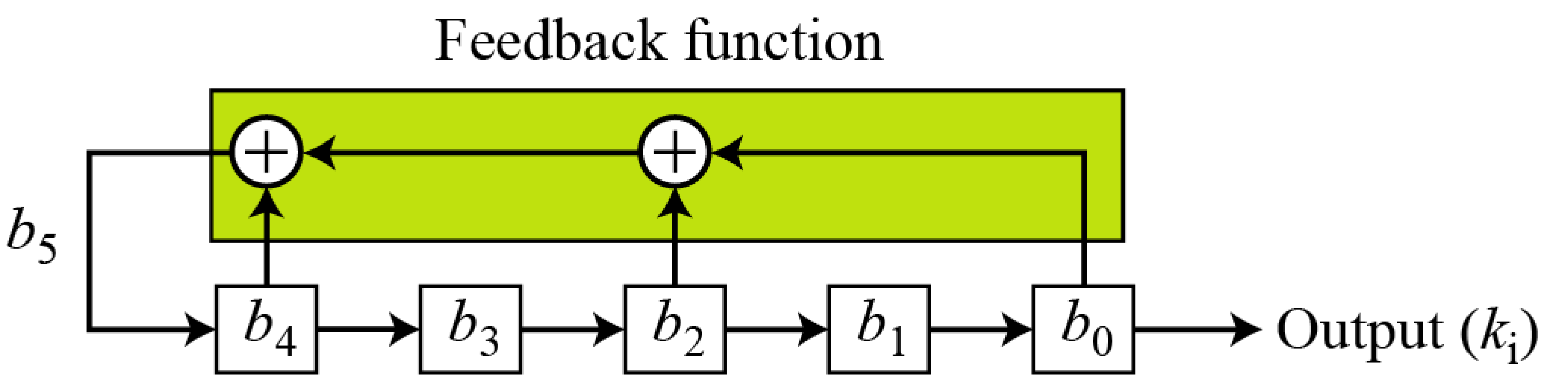
Example 5.18

Create a linear feedback shift register with 5 cells in which $b_5 = b_4 \oplus b_2 \oplus b_0$.

Solution

If $c_i = 0$, b_i has no role in calculation of b_m. This means that b_i is not connected to the feedback function. If $c_i = 1$, b_i is involved in calculation of bm. In this example, c1 and c3 are 0's, which means that we have only three connections. Figure 5.24 shows the design.

Figure 5.24 *LSFR for Example 5.18*

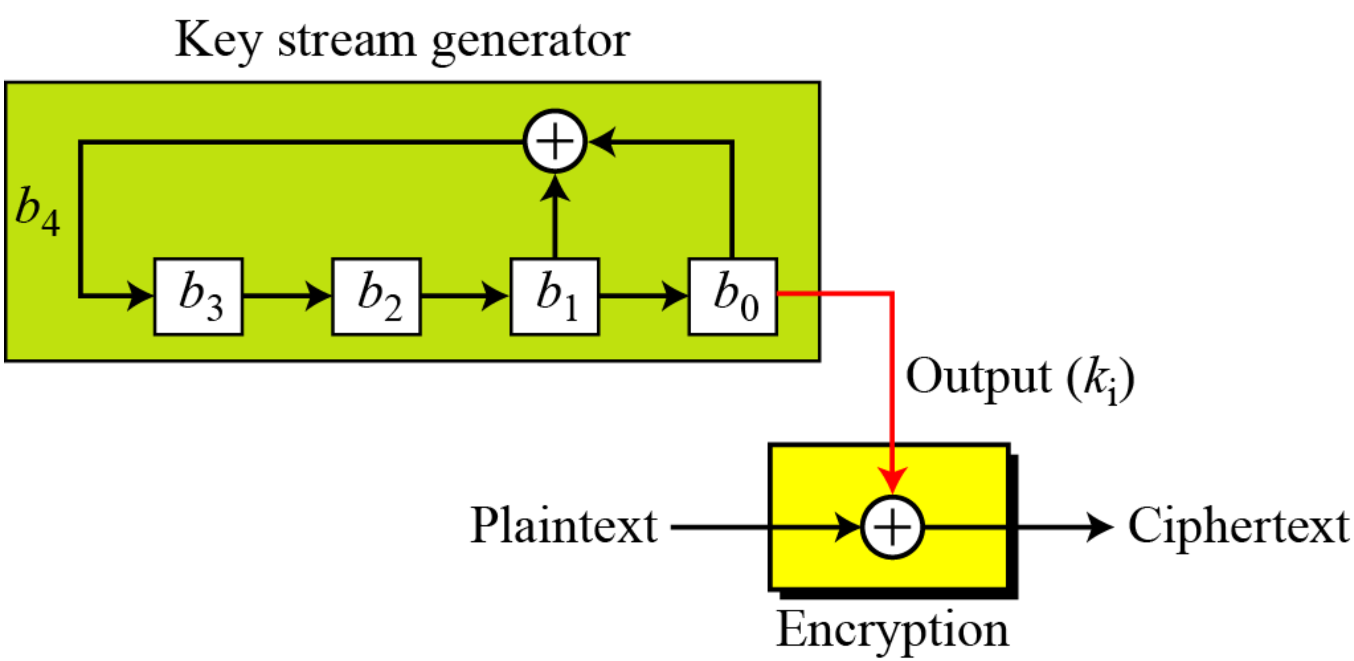


Example 5.19

Create a linear feedback shift register with 4 cells in which $b_4 = b_1 \oplus b_0$. Show the value of output for 20 transitions (shifts) if the seed is $(0001)_2$.

Solution

Figure 5.25 ***LFSR for Example 5.19***



5.2.1 Continued

Example 5.19 (Continued)

Table 4.6 *Cell values and key sequence for Example 5.19*

States	b_4	b_3	b_2	b_1	b_0	k_i
Initial	1	0	0	0	1	
1	0	1	0	0	0	1
2	0	0	1	0	0	0
3	1	0	0	1	0	0
4	1	1	0	0	1	0
5	0	1	1	0	0	1
6	1	0	1	1	0	0
7	0	1	0	1	1	0
8	1	0	1	0	1	1
9	1	1	0	1	0	1
10	1	1	1	0	1	0

5.2.1 Continued

Example 5.19 (Continued)

Table 4.6 Continued

11	1	1	1	1	0	1
12	0	1	1	1	1	0
13	0	0	1	1	1	1
14	0	0	0	1	1	1
15	1	0	0	0	1	1
16	0	1	0	0	0	1
17	0	0	1	0	0	0
18	1	0	0	1	0	0
19	1	1	0	0	1	0
20	1	1	1	0	0	1

5.2.1 Continued

Example 5.19 (Continued)

Note that the key stream is 100010011010111 10001…. This looks like a random sequence at first glance, but if we go through more transitions, we see that the sequence is periodic. It is a repetition of 15 bits as shown below:

100010011010111 **100010011010111** 100010011010111 **100010011010111** …

The key stream generated from a LFSR is a pseudorandom sequence in which the the sequence is repeated after *N* bits.

Note

The maximum period of an LFSR is to $2^m - 1$.

Example 5.20

The characteristic polynomial for the LFSR in Example 5.19 is ($x^4 + x + 1$), which is a primitive polynomial. Table 4.4 (Chapter 4) shows that it is an irreducible polynomial. This polynomial also divides ($x^7 + 1$) = ($x^4 + x + 1$) ($x^3 + 1$), which means $e = 2^3 - 1 = 7$.

5.2.2 Nonsynchronous Stream Ciphers

In a nonsynchronous stream cipher, each key in the key stream depends on previous plaintext or ciphertext.

Note

In a nonsynchronous stream cipher, the key depends on either the plaintext or ciphertext.