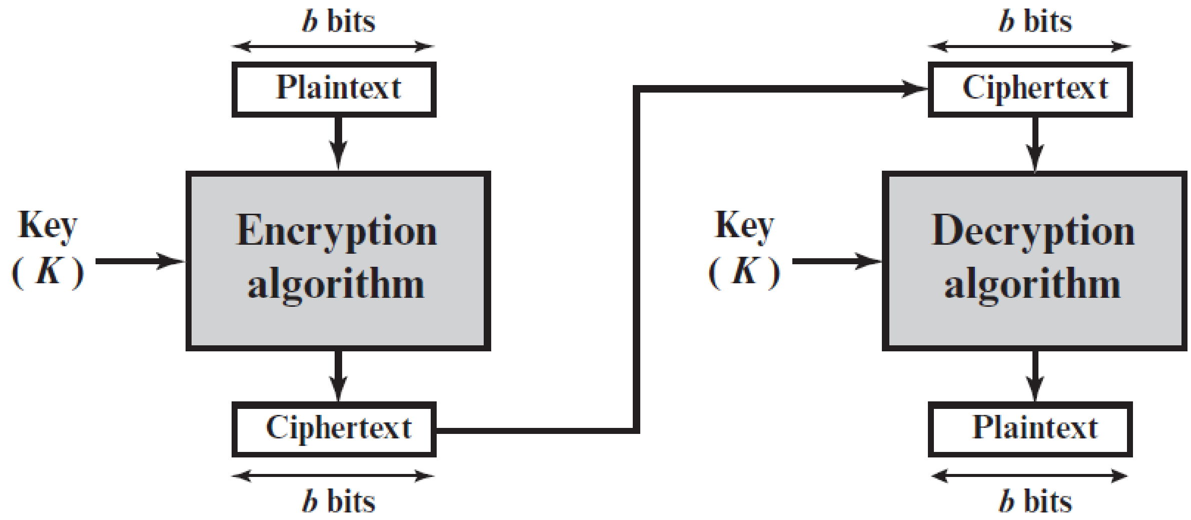
Stream Ciphers and Block Ciphers

Stream cipher: encrypts a digital data stream one bit or one byte at a time. Examples of classical stream ciphers are the autokey Vigenère cipher and the Vernam cipher. In the ideal case, a one-time pad version of the Vernam cipher would be used, in which the keystream (*ki*) is as long as the plaintext bit stream (*pi*).

A **block cipher** is one in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length. Typically, a block size of 64 or 128 bits is used. As with a stream cipher, the two users share a symmetric encryption key.



Block cipher

Stream Cipher and Block Cipher

Data Encryption Standard

DES is a symmetric-key algorithm, ensuring the same key encrypts and decrypts data.

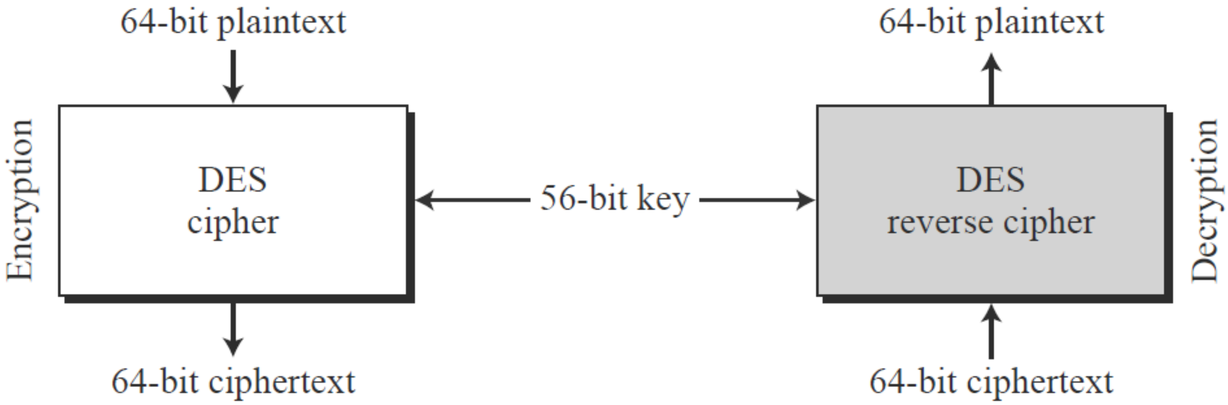
The symmetric systems provide a two-way channel to their users: A and B share a secret key, and they can both encrypt information to send to the other as well as decrypt information from the other. The symmetry of this situation is a major advantage.

As long as the key remains secret, the system also provides authentication, proof that a message received was not fabricated by someone other than the declared sender. Authenticity is ensured because only the legitimate sender can produce a message that will decrypt properly with the shared key.

Overview of the DES Algorithm

The DES algorithm is a careful and complex combination of two fundamental building blocks of encryption: **substitution and transposition**. The algorithm derives its strength from repeated application of these two techniques, one on top of the other, for a total of 16 cycles. The sheer complexity of tracing a single bit through 16 iterations of substitutions and transpositions has so far stopped researchers in the public from identifying more than a handful of general properties of the algorithm.

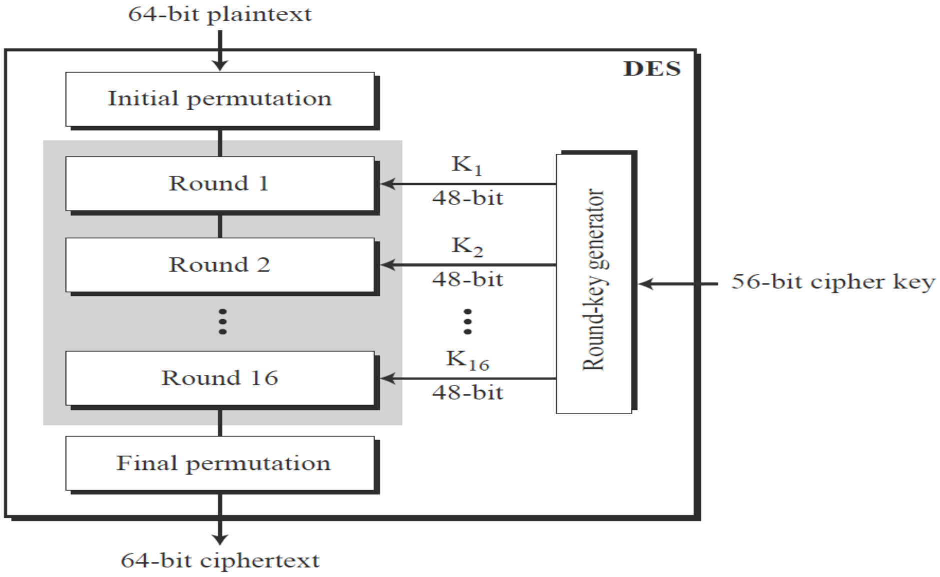
The algorithm begins by encrypting the plaintext as blocks of 64 bits. The key is 64 bits long, but it can be **any 56-bit number**. (**The extra 8 bits are often used as check digits and do not affect encryption in normal implementations**.) The user can change the key at will any time there is uncertainty about the security of the old key. DES is a block cipher, as shown in



Encryption and decryption with DES

At the encryption site, **DES takes a 64-bit plaintext and creates a 64-bit ciphertext**; at the decryption site, DES takes a 64-bit ciphertext and creates a 64-bit block of plaintext. **The same 56-bit cipher key is used for both encryption and decryption**.

The encryption process is made of two permutations (P-boxes), which we call initial and final permutations, and sixteen Feistel rounds. Each round uses a different 48-bit round key generated from the cipher key according to a predefined algorithm described later in the chapter.
The figure below shows the elements of the DES cipher at the encryption site.



General structure of DES

In particular, Feistel proposed the use of a cipher that alternates substitutions and permutations, where these terms are defined as follows:

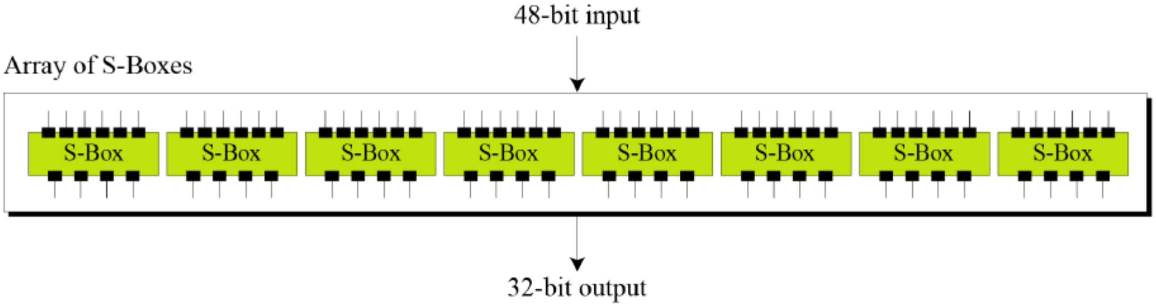
Substitution: Each plaintext element or group of elements is uniquely replaced by a corresponding ciphertext element or group of elements.

Permutation: A sequence of plaintext elements is replaced by a permutation of that sequence. That is, no elements are added or deleted, or replaced in the sequence, rather the order in which the elements appear in the sequence is changed.

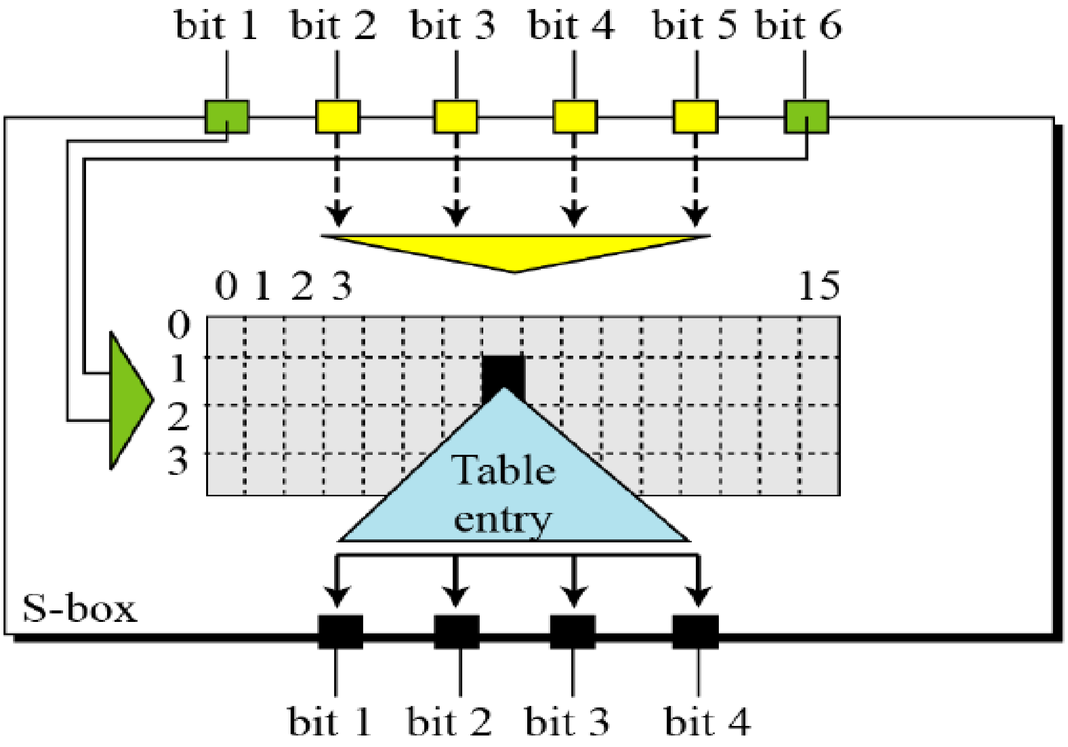
S-Boxes

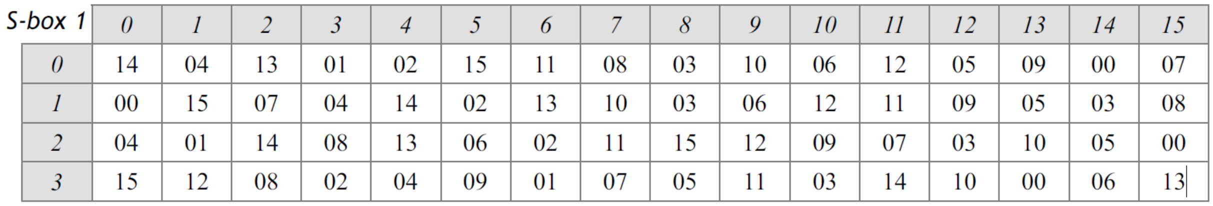
The S-boxes do the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output.

S-boxes

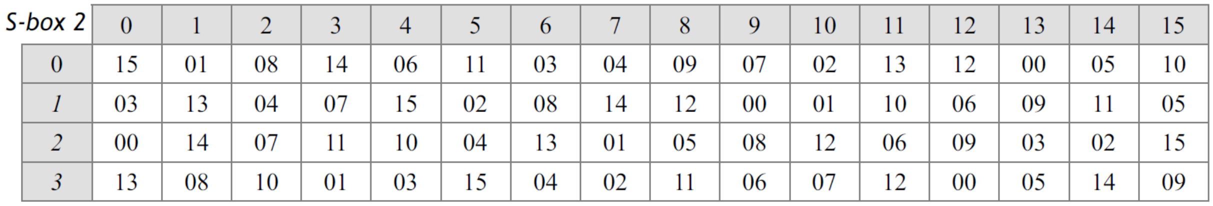


S-box rule

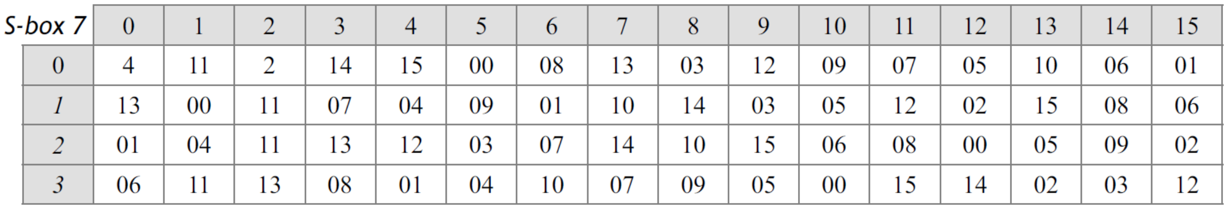




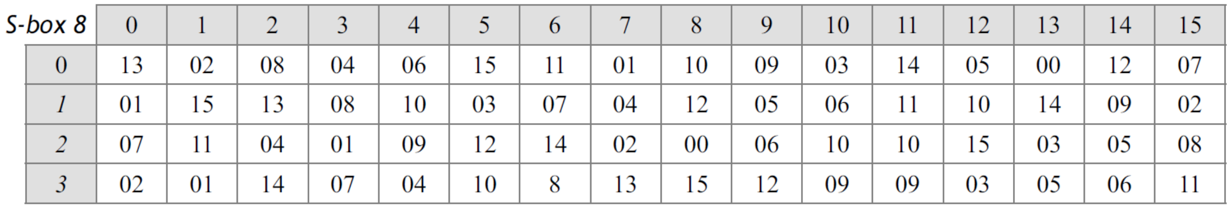
S-box 1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13



S-box 2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	15	01	08	14	06	11	03	04	09	07	02	13	12	00	05	10
1	03	13	04	07	15	02	08	14	12	00	01	10	06	09	11	05
2	00	14	07	11	10	04	13	01	05	08	12	06	09	03	02	15
3	13	08	10	01	03	15	04	02	11	06	07	12	00	05	14	09



S-box 7	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	4	11	2	14	15	00	08	13	03	12	09	07	05	10	06	01
1	13	00	11	07	04	09	01	10	14	03	05	12	02	15	08	06
2	01	04	11	13	12	03	07	14	10	15	06	08	00	05	09	02
3	06	11	13	08	01	04	10	07	09	05	00	15	14	02	03	12



S-box 8	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	13	02	08	04	06	15	11	01	10	09	03	14	05	00	12	07
1	01	15	13	08	10	03	07	04	12	05	06	11	10	14	09	02
2	07	11	04	01	09	12	14	02	00	06	10	10	15	03	05	08
3	02	01	14	07	04	10	8	13	15	12	09	09	03	05	06	11

Example 1: The input to S-box 1 is **100011**. What is the output?

Solution If we write the first and the sixth bits together, we get 11 in binary, which is 3 in decimal.

The remaining bits are 0001 in binary, which is 1 in decimal. We look for the value in row 3, column 1,

in Table (S-box 1). The result is 12 in decimal, which in binary is 1100. So, the input 100011 yields the output 1100.

Example 2: The input to S-box 8 is **000000**. What is the output?

Solution If we write the first and the sixth bits together, we get 00 in binary, which is 0 in decimal.

The remaining bits are 0000 in binary, which is 0 in decimal. We look for the value in row 0, column 0,

in Table (S-box 8). The result is 13 in decimal, which is 1101 in binary. So, the input 000000 yields the output 1101.

S_1

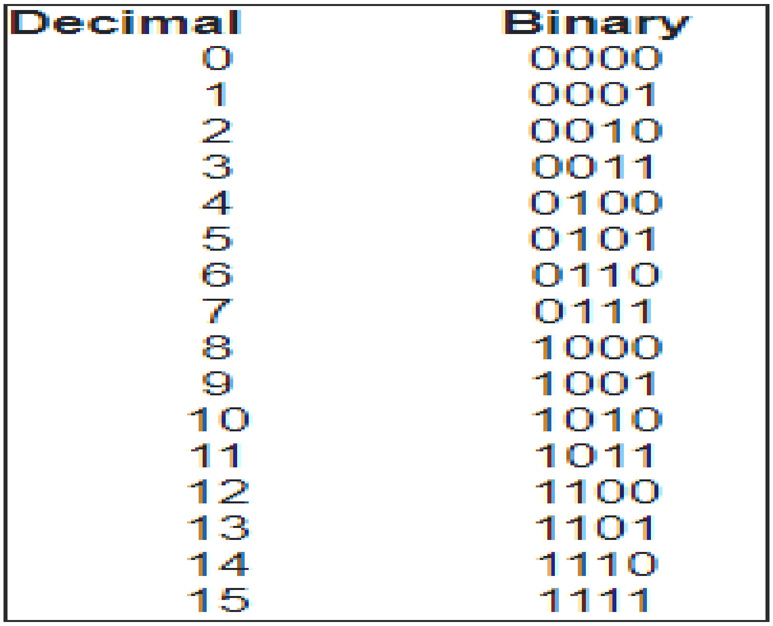
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S(i, j) <16, can be represented with 4 bits

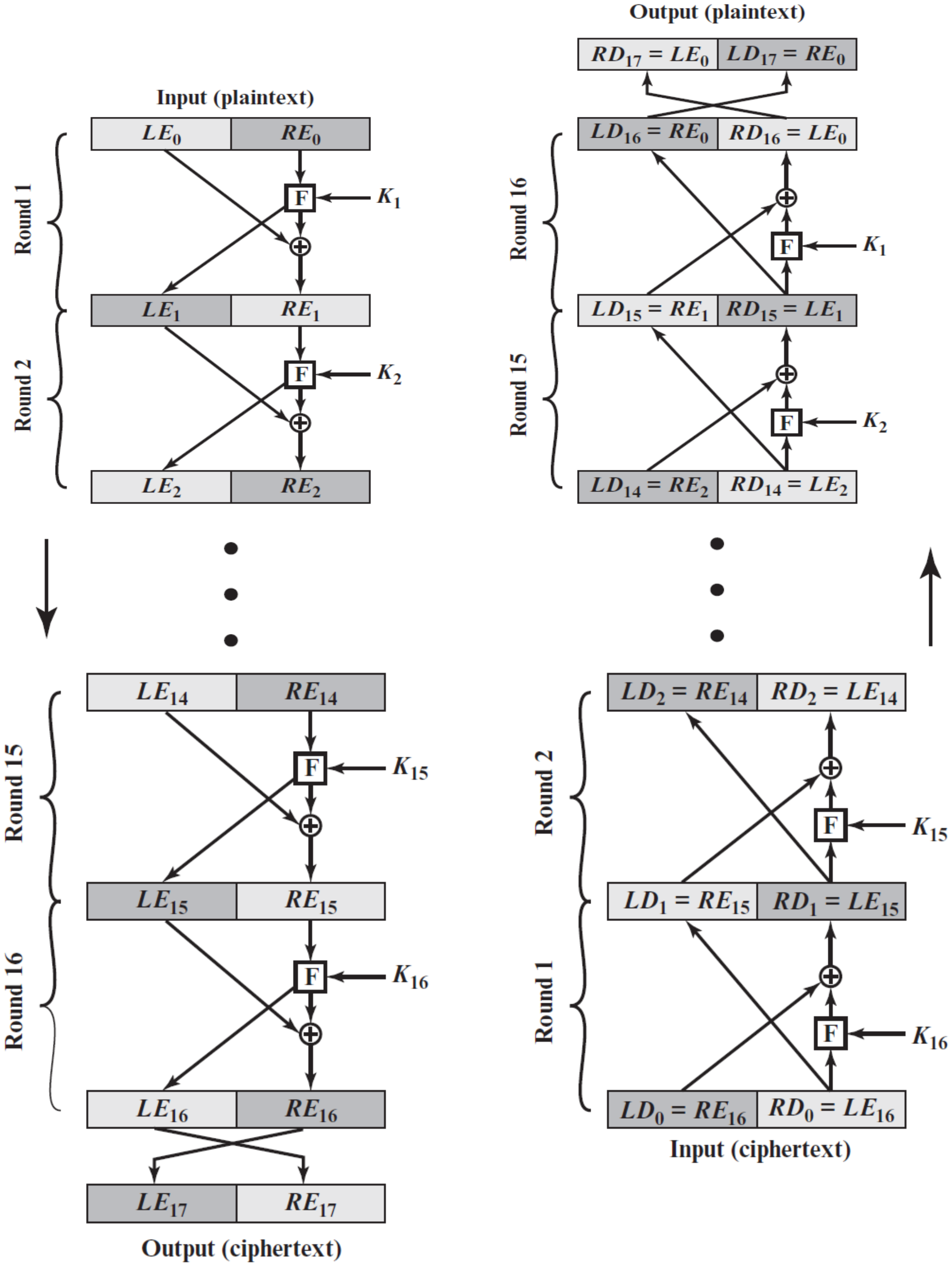
B =101111

b_1b_6 = 11 = row 3

$b_2b_3b_4b_5$= 0111 = column 7



Decimal	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111



Feistel Encryption and Decryption (16 rounds)